Acknowledgements

Thanks are due to the members of Heriot-Watt University's SCHOLAR team who planned and created these materials, and to the many colleagues who reviewed the content.

We would like to acknowledge the assistance of the education authorities, colleges, teachers and students who contributed to the SCHOLAR programme and who evaluated these materials.

Grateful acknowledgement is made for permission to use the following material in the SCHOLAR programme:

The Scottish Qualifications Authority for permission to use Past Papers assessments.

The Scottish Government for financial support.

The content of this Study Guide is aligned to the Scottish Qualifications Authority (SQA) curriculum.

All brand names, product names, logos and related devices are used for identification purposes only and are trademarks, registered trademarks or service marks of their respective holders.
Contents

1 Project design process 1
  1.1 Revision ............................................. 3
  1.2 Software development versus Project management ................. 4
  1.3 Project initialisation ..................................... 9
  1.4 Research: feasibility study, user surveys .......................... 10
  1.5 Project initialisation documentation ............................ 15
  1.6 Analysis .............................................. 17
  1.7 Sprint phases .......................................... 20
  1.8 Evaluation ............................................. 25
  1.9 Learning points .......................................... 27
  1.10 End of topic test ....................................... 28

2 Project management 31
  2.1 Revision .............................................. 33
  2.2 Planning ............................................... 33
  2.3 Scheduling ............................................. 35
  2.4 Gantt chart ............................................ 40
  2.5 Evaluation ............................................. 44
  2.6 Learning points ......................................... 46
  2.7 End of topic test ....................................... 47

3 Information systems design techniques 51
  3.1 Revision .............................................. 53
  3.2 Human computer interaction (user-centred) ......................... 54
  3.3 User centred design ...................................... 54
  3.4 User interface design ..................................... 64
  3.5 Resolving user centred design and SCRUM ........................ 70
  3.6 Learning points ......................................... 71
  3.7 End of topic test ....................................... 72

4 Web development 73
  4.1 Revision .............................................. 75
  4.2 Web Server model ....................................... 76
  4.3 HTML form processing .................................... 77
  4.4 Learning points ......................................... 94
  4.5 End of topic test ....................................... 95

5 Database systems and SQL 97
  5.1 Revision .............................................. 99
  5.2 Database Server model ................................... 100
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.3 Database management tools</td>
<td>103</td>
</tr>
<tr>
<td>5.4 Structured Query Language (SQL)</td>
<td>103</td>
</tr>
<tr>
<td>5.5 Learning points</td>
<td>123</td>
</tr>
<tr>
<td>5.6 End of topic test</td>
<td>124</td>
</tr>
<tr>
<td>6 Web and database integration</td>
<td>127</td>
</tr>
<tr>
<td>6.1 Revision</td>
<td>129</td>
</tr>
<tr>
<td>6.2 Server side scripting requirements</td>
<td>130</td>
</tr>
<tr>
<td>6.3 Connecting to database</td>
<td>141</td>
</tr>
<tr>
<td>6.4 Executing query</td>
<td>144</td>
</tr>
<tr>
<td>6.5 Retrieving results</td>
<td>145</td>
</tr>
<tr>
<td>6.6 Editing/Amending data</td>
<td>148</td>
</tr>
<tr>
<td>6.7 Learning points</td>
<td>160</td>
</tr>
<tr>
<td>6.8 End of topic test</td>
<td>161</td>
</tr>
<tr>
<td>7 Implications: Legal, ethical, environmental, economic and social</td>
<td>163</td>
</tr>
<tr>
<td>7.1 Revision</td>
<td>165</td>
</tr>
<tr>
<td>7.2 Intellectual property rights</td>
<td>166</td>
</tr>
<tr>
<td>7.3 Storage of user data</td>
<td>168</td>
</tr>
<tr>
<td>7.4 Increasing use and power of intelligent systems</td>
<td>169</td>
</tr>
<tr>
<td>7.5 Energy (data centres, low carbon equipment)</td>
<td>169</td>
</tr>
<tr>
<td>7.6 Online marketing (web, e-mail, text)</td>
<td>171</td>
</tr>
<tr>
<td>7.7 Analytics</td>
<td>173</td>
</tr>
<tr>
<td>7.8 Cyber security risks and precautions</td>
<td>175</td>
</tr>
<tr>
<td>7.9 Tracking, privacy, online safety</td>
<td>177</td>
</tr>
<tr>
<td>7.10 Social media</td>
<td>178</td>
</tr>
<tr>
<td>7.11 Implications of “Big data”</td>
<td>179</td>
</tr>
<tr>
<td>7.12 Learning points</td>
<td>180</td>
</tr>
<tr>
<td>7.13 End of topic test</td>
<td>181</td>
</tr>
<tr>
<td>8 End of Unit 2 test</td>
<td>183</td>
</tr>
<tr>
<td>Glossary</td>
<td>190</td>
</tr>
<tr>
<td>Answers to questions and activities</td>
<td>201</td>
</tr>
<tr>
<td>1 Project design process</td>
<td>201</td>
</tr>
<tr>
<td>2 Project management</td>
<td>203</td>
</tr>
<tr>
<td>3 Information systems design techniques</td>
<td>204</td>
</tr>
<tr>
<td>4 Web development</td>
<td>206</td>
</tr>
<tr>
<td>5 Database systems and SQL</td>
<td>207</td>
</tr>
<tr>
<td>6 Web and database integration</td>
<td>210</td>
</tr>
<tr>
<td>7 Implications: Legal, ethical, environmental, economic and social</td>
<td>219</td>
</tr>
<tr>
<td>8 End of Unit 2 test</td>
<td>220</td>
</tr>
</tbody>
</table>
# Topic 1

## Project design process

## Contents

1.1 Revision .................................................. 3  
1.2 Software development versus Project management .......... 4  
1.2.1 Waterfall model .................................................. 5  
1.2.2 Agile model .................................................. 7  
1.3 Project initialisation .............................................. 9  
1.4 Research: feasibility study, user surveys ......................... 10  
1.4.1 Feasibility study .................................................. 10  
1.4.2 Types of feasibility ............................................... 10  
1.4.3 User surveys .................................................. 14  
1.5 Project initialisation documentation .............................. 15  
1.6 Analysis .......................................................... 17  
1.6.1 User and Business requirements ................................. 18  
1.6.2 Scope, constraints ............................................... 19  
1.6.3 Functional and Operational requirements ..................... 19  
1.7 Sprint phases .................................................... 20  
1.7.1 Design .......................................................... 21  
1.7.2 Build (Implementation) ......................................... 23  
1.7.3 Testing .......................................................... 24  
1.7.4 Accessibility ................................................... 25  
1.8 Evaluation ....................................................... 25  
1.8.1 Usability ........................................................ 26  
1.8.2 Efficiency ....................................................... 26  
1.8.3 Reliability, robustness ......................................... 26  
1.9 Learning points .................................................. 27  
1.10 End of topic test ................................................ 28

**Prerequisite knowledge**

*From your studies at Higher you should already:*

- *know the key features of the software development process including traditional (waterfall), RAD and Agile software development methodologies;*
• understand that software development is an iterative process requiring previous aspects of development to be revisited to make improvements;

• know that software development normally involves a number of stages including analysis, design, implementation, testing and evaluation.

Learning objectives

By the end of this topic you will be able to:

• describe the process of initialisation and subsequent research, feasibility study and development of a project;

• explain the importance of user and business requirements to determine the scope and constraints of a project and its functional and operational requirements;

• describe the processes undertaken by an agile software development team required to build, integrate and deploy a project;

• effectively test a software development project using a number of techniques including usability testing and accessibility;

• effectively evaluate a software development project by measuring and commenting on the usability, efficiency, reliability and robustness of the software.
1.1 Revision

Quiz: Revision

Q1: Using a paper prototype to collect feedback about a user interface is an example of?

a) Beta testing
b) Usability testing
c) Functional testing
d) Acceptance testing

Q2: A technique used to gather information before a project is designed is:

a) a paste bin
b) a developer resource
c) an interpreter
d) a questionnaire

Q3: A user interface is said to be robust if:

a) all inputs are validated and appropriate feedback given to the user.
b) it can be adapted for different platforms with a limited number of changes.
c) it is able to deal with all errors experienced during execution of the program.
d) support for a wide variety of input devices is planned in the interface design.

Q4: What common tool is used to create an initial interface design?

a) Structured design
b) Wire-framing
c) Agile development
d) Pseudocode

Q5: Select which types of feedback would be suitable for a user interface for visually impaired users of a smartphone:

a) sensory/vibration feedback when option selected
b) audio description
c) high contrast display
d) different button shapes
**Q6:** A user interface that is fit for purpose will:

a) be simple to change in the future, as the program requirements change.
b) be a model that can be adopted on a number of platforms such as mobile, desktop, tablet.
c) meet the functional needs of the program and be accessible for the target user.
d) be able used to operate the programs main functions.

**Q7:** An example of efficiency in the design of a user interface is:

a) that the interface keeps interactions with the user to a minimum to meet the functional requirements of the program.
b) that the interface uses fewest system resources such as processor time, memory, backing storage etc.
c) the use of software development tools to ensure the interface is correctly translated.
d) the use of cross compilers so an interface can be deployed on many devices.

### 1.2 Software development versus Project management

**Learning objective**

By the end of this section you will be able to describe and explain these models of software development:

- the waterfall model;
- the Agile model.

**Software development** is concerned with how a piece of software is designed, built and delivered. Software development is the process of developing software through successive phases in an orderly way. This process includes not only the actual writing of code but also the preparation of requirements and objectives, the design of what is to be coded, and confirmation that what is developed has met its objectives.

**Project management** is the process of controlling and managing a project from an initial idea through to completion and delivery. A project can be anything from designing a mobile phone app to building a bridge. Software development projects typically require a **project manager** and a team of software developers to work together to deliver a project on time, to budget and to an agreed scope. The scope of a project is the work that needs to be accomplished to deliver a product, service, or result with the specified features and functions.

**Software development methodologies**

There are two general approaches to software development that you should be aware of from Higher: the Waterfall model and the Agile model.
1.2.1 Waterfall model

The **Waterfall model** is a software development approach, introduced in the 1970s when software development was becoming commonplace. The process is very simple to understand and use. In the waterfall model, each phase must be completed before the next phase can begin. If the requirements for the software change in any way, or mistakes have been made at any point, then the early stages in the development have to be revisited and the new requirements included in the work for those stages. Because of this frequent returning to earlier stages in light of new information, the waterfall model is often referred to as an **iterative** process.

*Figure 1.1: Waterfall model - revisiting earlier phases can become difficult*

As with any approach to software development, the waterfall model has some advantages and some disadvantages.

**Advantages**

- The model is simple and easy to understand and use.
- Each phase of the waterfall model has a clear set of deliverables - things that have to be created.
- At the end of each phase there is a review process to examine the deliverables and decide to proceed to the next phase.
- Phases don’t overlap.
- Great for small projects where all of the requirements are known and fully understood.

**Disadvantages**

- Once development has reached the testing phase, it is very difficult to go back and change something which was not well-thought out or understood in the analysis or design phases.
- No working software is produced until relatively late in the project (analysis and design are completed prior to any implementation).
TOPIC 1. PROJECT DESIGN PROCESS

- High amounts of risk and uncertainty - the larger the project the more difficult it is to know everything about it before starting design work.
- Not suitable for projects where the requirements are at a high or moderate risk of changing.

The waterfall model is an unrealistic approach for many software development projects because it requires everything to be known about the project in advance of starting. This may work on very small scale projects where it is possible to understand all the requirements but for most projects there will be changes.

Returning to previous phases is time consuming, often requires additional resources to be used to deliver the project and leads to developments that have multiple changes and frequent errors/bugs in coding due to the late introduction of changes to the requirements.

There is very little client/user interaction in the development of software using the waterfall model: clients/users would be involved at the analysis phase and at the end of the testing phase but are excluded from the other phases of development.

The software can only be shown to the user when it has been completed and tested, after the initial four phases of development have been carried out. Once the product is developed, the cost of fixing any errors/bugs is very high, because changes would need to be made to every phase of the software design from documentation, to design, to program logic/coding.

Roles in the Waterfall model

In the Waterfall model there are a number of typical roles:

- the Systems analyst (who produces the requirements specification),
- the Programmer (who writes and tests the programme code),
- the Client (who the software is being produced for) and
- the Project manager (who is responsible for ensuring that the project is well managed, delivered on time and with the correct resourcing).

Deliverables

The waterfall model typically requires that a requirements specification be created at the end of the analysis phase. This forms the contractual arrangement for the development of the software.

The requirements specification details the scope and boundaries of the intended software, details the basis of payment for the work to be completed and sets out how the software will be designed, tested, documented and evaluated before hand over to the customer. It may also detail any longer term maintenance agreement between the customer and the developer.

The information in the requirements specification contributes to the project plan that the project manager creates to plan how the project is to be completed. During the documentation phase, user documentation will be produced to detail how the end user can make use of the software and technical documentation will be created to detail technical aspects of the software such as installation requirements.
1.2.2 Agile model

The **Agile model** of software development is designed to address the difficulties of the waterfall model. The Agile model of software development focuses on producing working software very quickly. This is an **iterative prototyping** approach, where each successive release of software builds on the previous one.

The Agile model recognises that the requirements for the software will change over time and that not all the requirements are known at the beginning of the development. The client will change his/her mind and an Agile approach allows changes to be incorporated into the software relatively easily and without significant impact on the project delivery. The most common implementation of the Agile model is an approach to software development called **SCRUM**.

This approach to software development is incremental. Software is developed in quick, incremental cycles. This produces small releases (working versions) of the software which start with only a few of the required features and gradually build into the complete program. The work to create these releases is called a **sprint** and each sprint consists of a design, build and test phase.

![Figure 1.2: Agile design sprints](image)

The client, users and other stakeholders are very closely involved in the development and the approach depends on frequent face-to-face meetings to capture feedback from the client/stakeholders/users and to update the list of required features for the software. Customers see working software very early in the development life-cycle and provide detailed feedback to improve the software as it is developed.

**Roles in the SCRUM model**

There are three main roles in the SCRUM approach to Agile software development.

The **Product owner** role is to reflect the views of the key stakeholders in the project. Part of the role of the Product owner is to have a vision of what he/she wishes to build, and to convey that vision to the scrum team. The product owner does this, partly, by prioritising the **product backlog** - the list of requirements for the product. The product owner is typically a lead user of the system and/or someone with a very clear understanding of the users, the intended market for the software and knowledge of other factors such as...
possible competing products and current trends. It is vital that the product owner is able to communicate with stakeholders effectively at any point in the project development.

The **Scrum team** is responsible for the development of the software. Together they select the requirements from the top of product backlog that they can commit to developing in any single **sprint**. Typical Scrum teams are between 5 and 9 people and there are no specific roles within the team - everyone involved in the project works together to complete the set of work they have committed to within a sprint. Because of this Scrum teams develop a feeling of "we're all in this together."

On larger projects, when more staff are required, Scrum teams don't scale up by adding more team members, they scale up by adding more teams - each team can then elect a representative to come to a "scrum of scrums" meeting to co-ordinate the work of the various teams.

The **ScrumMaster** is a coach for the Scrum team and is responsible for getting the best from the Scrum team. This involves removing any barriers to progress, facilitating meetings, and doing things like working with the product owner to make sure the product backlog is in good shape and ready for the next sprint. The ScrumMaster is there to ensure that the team make use of the Agile model for development. The ScrumMaster is in charge of the process but is not in charge of organising the work to be done.

The ScrumMaster is also often viewed as a protector of the team. The most common example is that the ScrumMaster protects the team by making sure they do not over-commit themselves to what they can achieve during a sprint due to pressure from an overly aggressive product owner. However, a good ScrumMaster also protects the team from complacency.

**Activity: Roles in the Agile model**

**Q8:**

Match the responsibilities to the roles:

**Roles:**

1. Product owner
2. ScrumMaster
3. Scrum team member

**Responsibility:**

a) Updates colleagues with his/her progress on a particular development.
b) Represents the views of the users to the developers.
c) Is part of a team of between 5 and 9 people.
d) Ensures the developers do not over commit to work during a sprint.
e) Presents a vision for the project.
f) Selects items from the product backlog for the sprint.
g) Encourages the use of Agile methods.

..........................................

© HERIOT-WATT UNIVERSITY
1.3 Project initialisation

Learning objective

By the end of this section you will be able to:

• describe the role of a project sponsor;
• describe the need for a project proposal document.

A project has to start somewhere. Project initialisation is the phase of a software development project where the idea for the project is examined and expanded on.

Typically, a project will be initiated to meet some organisational, business or commercial need. The idea for a software development project can come from any part of an organisation but will require some high level agreement, from senior managers for example, before the project can move to initialisation.

This high-level person is the representative of the client, often referred to as the project sponsor, since he or she will be the main point of contact during the life of the project.

The project sponsor, within an organisation, supports the project - they are the person or group of people to which the project team report. In most cases, the project sponsor represents who is paying for the development or, at least, is ultimately responsible for it.

The first goal of the project initialisation is to determine the feasibility of the project: is it possible (technically, legally, economically and within the available time) and report this to the project sponsor. This initial detail about the project and its feasibility is detailed in a project proposal document.

The project sponsor will then approve the project to move into development or decide that further work should not be carried out.

Figure 1.3: Approval of Development and Agile sprints
1.4 Research: feasibility study, user surveys

Learning objective

By the end of this section you will be able to:

- explain the different types of feasibility;
- explain the use of user surveys to gather information about a project.

Research is a vital part of the initial analysis of what the project sponsor is requesting.

1.4.1 Feasibility study

A key part of the project initialisation is undertaking research to ensure that the project is feasible. Software development projects, depending on their size, can be in development for many years and can have budgets worth millions of pounds. It would be foolish to proceed with any project without assessing if completing it is possible.

The basic purpose of a feasibility study is to work out if the proposed expenditure of time and money is likely to be worthwhile and whether the objectives of the project can be achieved: what some projects set out to do might be totally unrealistic.

The results of the feasibility study will determine which, if any, of a number of possible solutions can be further developed at the design phase.

One result of undertaking a feasibility study may indicate that only a simple solution is required to solve the problem:

- a software fault may be identified that is easily fixed;
- more staff training might be required if particular software is to be used.

The feasibility study is most often carried out by the project leader - an experienced member of staff who is able to understand the needs of the project and the various aspects of feasibility that apply to it.

1.4.2 Types of feasibility

A feasibility study will look at four main areas of feasibility: Technical (is it technically possible to create a solution with the available technology), Economic (is it possible to complete the project with the budget available or is the cost of creating the project justified by the financial reward of doing so), Legal (can the solution be created and adhere to existing laws) and Schedule (is there enough time to complete the project, are the right people and resources available when required to deliver the project on time).

1.4.2.1 Technical feasibility

The feasibility study must ascertain what technologies are necessary for the proposed system to work as it should. It may be the case that suitably advanced technology does not yet exist. Unless it is the object of the project to design a system to use such advanced technology, this would rule the project out as being a non-starter. It would
be a foolish move for a feasibility study to evaluate technologies which are either under
development or undergoing testing.

Given that suitable technology does exist, the study must establish if the organisation
already has the necessary resources. If not, the study must make clear what new
resources the organisation would have to acquire. This will also involve determining
whether the hardware and software recommended will operate effectively under the
proposed workload and in the proposed environmental conditions. The development of
a new system involves risks of one kind or another. Every understanding that might be
reached could carry the risk of some misunderstanding:

• software companies and their clients often have different vocabularies and
  consequently they appear to be in perfect agreement until the finished product
  is supplied;
• management may have unrealistic expectations of computer systems. The
  feasibility study is where idealism meets reality.

Further issues might include the training of personnel to use the new system,
consideration of service contracts, warranty conditions and the establishing of help desk
facilities for inexperienced users.

1.4.2.2 Economic feasibility

This deals with the cost implications involved. Management will want to know how much
each option will cost, what is affordable within the company's budget and what they get
for their money. A cost-benefit-analysis is part of the budgetary feasibility study. If the
project is not cost-effective then there is no point proceeding.

Setting up a new computer system is an investment and involves capital outlay. The
costs of a new system include the costs of acquiring it in the first place (consultancy fees,
program development, etc.); the costs of installing it (disruption of current operations,
cost of new equipment, alteration of workplace, etc.); and the costs of maintaining it
which also includes training.

In the long term management will also want to know the 'break-even point' when the
new system stops costing money and starts to make money. This is extremely difficult
to quantify. However an accurate estimate of a system's operational life span is a valid
option and will rely solely on the knowledge and experience of the systems analyst
involved.
Figure 1.4 depicts such a case where the break-even point is at the intersection of the graphs:

![Break-even point graph](image)

Figure 1.4: Break-even point

Tangible benefits that management would certainly be looking for in the new system would be:

- reduced running costs;
- increased operational speed;
- increased throughput of work;
- better reporting facilities.

Note that not all the costs and benefits lend themselves to direct measurement. For example, new systems generally affect the morale of the staff involved, for good or ill. This can only be resolved by competent personnel management practices.

### 1.4.2.3 Legal feasibility

This has to do with any conflicts that might arise between the proposed system and legal requirements: how would the new system affect contracts and liability, are health and safety issues in place and would the system be legal under such local laws as the UK Data Protection Act? What are the software licensing implications for the new system?

Software licensing can be quite a thorny problem. Licences can be purchased as: client licence (per seat), server licence, network licence or site licence and the period of operation may be annual or perpetual. Software vendors vary in their licensing regulations so this has to be fully investigated.
1.4.2.4 Schedule feasibility

Schedule feasibility may be assessed as part of technical feasibility. Most organisations have an annual schedule of events such as the AGM, end of financial year, main holiday period and so on. Obviously time is a main factor in the development of a new system.

Questions to be asked at this stage might include:

- how long will the proposed system take to develop?
- will it be ready within the specified time-frame?
- when is the best time to install?

For example, a project might have to start within six months; assuming it would take three months to purchase and install the required hardware and software and a further six months to train the end users. Such a project is not technically feasible because of shortage of time so it would not go ahead unless some of the time constraints were reviewed and changed.

In many cases of project management the scheduling component can be aided by means of a Gantt chart.

A Gantt chart is like a horizontal bar graph used to plan and schedule projects involving several concurrent tasks. The horizontal axis represents the time scale and start and finish times of component parts are graphically represented.

The advantage of a Gantt chart is that it shows, at a glance, the progress of a project as shown in Figure 1.5

![Example of a Gantt chart](https://example.com/ganttchart.png)

You will look more closely at the creation of a Gantt chart and the concepts of Project
Activity: Types of feasibility

Q9:

Match the parts together to generate true statements:

1. A project that cannot be delivered in the time available because of the complexity of the task
2. A project which has insufficient finance to hire the necessary staff to complete the project when required
3. A project that processes banking transactions to indicate spending habits in contravention of data protection laws
4. A project which cannot proceed because the software to link two data sources into the project is not available

a) has an issue with technical feasibility.
b) has an issue with legal feasibility.
c) has an economic feasibility issue.
d) has an issue with schedule feasibility.

1.4.3 User surveys

Often with the development of software, the experience of existing users is vital in shaping what the new software should do better than the old software. The users of a particular program are the best people to identify the existing problems with the software and how it could be improved.

Equally, if the software being developed is entirely new, the target user group should, generally, understand the process that the programme will carry out. For example, a new library system, which is to replace a manual card based system, would be developed following some time spent capturing the experience of existing librarians and how they operate.

When the opinion of a number of users is required, it is often easier to create a survey or questionnaire which will capture the information that the project team require.
1.5 Project initialisation documentation

**Learning objective**

By the end of this section you will be able to name and describe the purpose of the:

- project proposal;
- business case;
- product backlog.

The feasibility study is part of the **project proposal** and contributes to the **project initialisation documentation**. The feasibility study should be conducted relatively cheaply and within a fairly short time frame as part of the project initialisation.

There are no legal or contractual requirements at this stage and, because the feasibility study helps to determine if the project will be developed, there should also be no commitment to develop the project (don’t assume that just because someone has an idea for a new piece of software that it will be *automatically* designed and produced).

It is very possible that, on completion of the feasibility study, a project will be abandoned for any number of reasons including:

- too costly to develop;
- incompatible with data security laws;
- unable to be completed in the time available;
- the project is not aligned with the needs or current direction of the organisation.

Typically, before a project can move to a full-scale analysis of requirements and creation of a detailed project initialisation documentation, the feasibility is assessed. This is through the construction of a project proposal, which will outline the following areas:

- Why is this a project? What is it that needs to be done and why does the organisation need it (this is described as the **business need**).
- What should the output of the project be? What are the options for delivering the project.
- Is it feasible?
- Who are the possible/identified partners in the project?
A Template for an Initial Project Proposal is shown in Figure 1.6:

*Figure 1.6: Initial Business Case Template*

<table>
<thead>
<tr>
<th>Project Name</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Project Sponsor</td>
<td>Approval Date</td>
</tr>
<tr>
<td>Project Manager</td>
<td>Version</td>
</tr>
</tbody>
</table>

**1 Description**
Describe the idea and its objectives.

**2 Business Need**
Why is the project needed? What are the business needs and the current business situation?
What strategic objectives does it help deliver? What kind of tangible measurable benefits will it provide?
What are the risks to the organisation if the project does not go ahead?
What triggered this project?

**3 Options**
Describe any of the currently identified options of how the project will be delivered.

**4 Feasibility**
Outline the outcome of the initial feasibility study to support this project.
Is the project achievable in the time available, with the technology available. What are the funding options for this project and will they allow the project to be delivered effectively?

**5 Dependencies**
List those projects, initiatives, key decisions or other activities outside the control of the project that may influence the project or present a risk to its success.

**6 Next Stage – Requirements Specification and Business Case**
Define time, costs and resources to develop the Full Project Initialisation Documentaiton and Full Requirements Specification for the project.
Consider any knock-on effects of this new work on existing priorities.

**7 Partners Consulted/Involved**

<table>
<thead>
<tr>
<th>Partner</th>
<th>Who Consulted</th>
<th>Their Comments</th>
<th>Date</th>
</tr>
</thead>
</table>

**8 Document Revision History**

<table>
<thead>
<tr>
<th>Version</th>
<th>Reason</th>
<th>By</th>
<th>Date</th>
</tr>
</thead>
</table>

Following approval of the initial project proposal, the team will gather the requirements for the software development project into a **Product backlog** (also known as a **requirements specification**). This document details all the known requirements for the development and within it the requirements are placed in rank order.
The project initialisation documentation, before the start of the iterative sprints, will include:

- **Project proposal** (signed off and agreed before the product backlog is created).
- **Business case** (a justification of the need for the project and a cost/benefit analysis).
- **Product backlog** (a list of the current known requirements - the scope of the product - in priority order to maximise the value of delivery to the business and end users).

Again, once the full business case is available the project sponsor will consider the business case and decide if the project should proceed to development.

### 1.6 Analysis

**Learning objective**

By the end of this section you will be able to explain:

- the main principals of the Scrum methodology;
- key aspects of Scrum including: sprint planning, sprints, sprint backlog and stakeholder consultation;
- the features of user centred design used to capture user requirements.

Analysis is the process of understanding what it is that the client, users and stakeholders want the new software to do. The task will include discussions with each group and at the end of each period of analysis there will be an update to the **product backlog** - the detailed list of the requirements for the product.

Each period of analysis can be undertaken as either part of the **Weekly backlog review** of the product backlog or as a result of feedback from when the team demo the software they have been developing. Any changes and new requirements are recorded in the product backlog and success criteria for their implementation is agreed with the project manager.

At the start of each sprint, a **sprint planning meeting** is held, during which the product owner presents the top items on the product backlog to the team. The Scrum team selects the work they can complete during the coming sprint. That work is then moved from the product backlog to a **sprint backlog**, which is the list of tasks needed to complete the product backlog items the team has committed to complete in the sprint. During the period of the sprint there will be a daily stand up meeting, also called a daily scrum, where team members provide an update on progress and explain what they are doing next.
Once the sprint has started, a freeze is placed on the sprint backlog, and no further requirements can be added to it. The design team then focus on delivering working software with these features.

**Figure 1.7: Overview of the Scrum approach to Agile development**

---

### 1.6.1 User and Business requirements

To create a successful product, the design team need to understand the intended users of the product. In order to do this, the members of the team will spend a significant amount of time interviewing users, working with them in focus groups and engaging in workshops around the development of the product. The **Product owner** is the principal representative of the users and other stakeholders.

During this process of determining the user and business requirements, the design team will develop an understanding of what users need and the tasks that users will perform with the product. In order to do this a number of documents are created to help inform the design team about the users and their motivations.

#### 1.6.1.1 Personas

A first step for the design team is to develop Personas for each of the target user types. Personas are fictional characters created to represent the different user types that the design team have encountered or that the team believe are important to the project.
1.6.1.2 User stories

A user story is a brief statement that identifies the user and his/her need. It is a direct statement that relates to a specific persona. The personas created by the design team will have a general level of detail however the user story will provide specific details about a task that the user will be engaged in. One persona may have several user stories developed around it. User stories help to document the practical differences in need among those users.

More detail is in the *Information systems design techniques* topic later on this unit.

1.6.1.3 User scenarios

A user scenario expands upon already developed user stories by including details about how a system might be interpreted, experienced, and used. User scenarios provide detail about the user’s goal, detail any assumed knowledge the user has and the level of experience the user has. Just as with user stories, the design team may imagine several scenarios for each persona group.

1.6.1.4 Use case

A use case is a list of steps a user would take to perform an action. A typical use case will start by detailing how the user got to the current position and then goes through every step until the user completes the operation or fails. Use cases can be informed and/or verified by the results of usability testing.

1.6.2 Scope, constraints

The Scope of the project is held in the *product backlog* - it is the definition of the features that the product must have - also known as the requirements of the software. If a feature is no longer required then it is dropped from the product backlog because it is out of scope.

Constraints are limitations that affect the development of the product. A project may be constrained by time (so only a certain number of development sprints can be completed), or may be limited by budget or may be limited by legal position regarding the data that it processes.

1.6.3 Functional and Operational requirements

*Functional requirements* are defined as what the product should do and are contained in the scope within the *product backlog*.

*Operational requirements* refer to the operation of the product: logging in, start-up/shutdown controls, back up. These will also be recorded in the *product backlog*. 
1.7 Sprint phases

Learning objective

By the end of this section you will be able to:

- describe the operation of a Scrum sprint (design/build/test);
- describe the purpose of the daily scrum meeting;
- describe the design techniques used within the sprint;
- describe the implementation and testing techniques used within the sprint.

Each sprint consists of three phases: design, build, test. In the design phase, the team will use a variety of tools to model the data in use, the processes applied to that data and the user experience.

Figure 1.8: Phases within a sprint

Each day during the sprint, a brief meeting called the daily standup (sometimes called a daily scrum) is conducted. This meeting helps set the context for each day’s work and helps the team stay on track. All team members are required to attend the daily scrum. At this meeting each team member answers the following three questions:

1. What did you do yesterday?
2. What will you do today?
3. Are there any impediments in your way?

By focusing on what each person accomplished yesterday and will accomplish today, the team gains an excellent understanding of what work has been done and what work remains. The daily scrum meeting is not a status update meeting in which a boss is collecting information about who is behind schedule. Rather, it is a meeting in which team members make commitments to each other.

If a programmer stands up and says, “Today, I will finish the data storage module,” everyone knows that in tomorrow’s meeting, she will say whether or not she finished. This has the effect of helping a team realise the significance of these commitments, and
that their commitments are to one another, not to some far-off customer or salesman.

Any impediments that are raised in the scrum meeting become the ScrumMaster’s responsibility to resolve as quickly as possible. Typical impediments are:

- My hard drive broke and I need a new one today.
- I still haven’t got the software I ordered a month ago.
- I need help debugging a problem with a module.
- I’m struggling to learn an algorithm and would like to pair with someone on it.
- I can’t get the vendor’s tech support group to call me back.
- Our new contractor can’t start because no one is here to sign her contract.
- I can’t get the user interface group to give me any time and I need to meet with them.
- The department VP has asked me to work on something else “for a day or two.”

In cases where the ScrumMaster cannot remove these impediments directly himself or herself (e.g., usually the more technical issues), he/she still takes responsibility for making sure someone on the team does quickly resolve the issue.

1.7.1 Design

The emphasis in agile development is on working software over lots of detailed documentation therefore the formal documentation of the design is generally limited to that which is directly useful for building the software.

1.7.1.1 System modelling (Data, Process)

The system modelling is carried out by designing and modelling data structures to hold the information that will be processed as part of the requirements of the current sprint and by developing process models, again, to match the requirements of the current sprint.

Data modelling

Data modelling, within the sprint, is carried out by developing an entity relationship diagram and a data dictionary of the data objects necessary to meet the requirements.

Figure 1.9: An example Entity Relationship Diagram
Figure 1.10: An example data dictionary

<table>
<thead>
<tr>
<th>Entity</th>
<th>Name</th>
<th>Type</th>
<th>Size</th>
<th>Validation</th>
<th>Index/Key</th>
</tr>
</thead>
<tbody>
<tr>
<td>customer</td>
<td>customer_nbr</td>
<td>Integer</td>
<td></td>
<td>&gt; 0, required</td>
<td>Yes, Primary key</td>
</tr>
<tr>
<td>customer</td>
<td>customer_name</td>
<td>Text</td>
<td>50</td>
<td>required</td>
<td>No</td>
</tr>
<tr>
<td>customer</td>
<td>customer_address</td>
<td>Text</td>
<td>255</td>
<td>required</td>
<td>No</td>
</tr>
<tr>
<td>customer</td>
<td>customer_postcode</td>
<td>Text</td>
<td>8</td>
<td>any valid postcode format</td>
<td>No</td>
</tr>
<tr>
<td>customer</td>
<td>customer_tel_nbr</td>
<td>Text</td>
<td>22</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td>order</td>
<td>customer_nbr</td>
<td>Integer</td>
<td></td>
<td>required</td>
<td>Yes, Foreign Key</td>
</tr>
</tbody>
</table>

Process modelling

Processes can be modelled using UML Use Case diagrams - this allows the team to present the functional requirements within the design sprint. The UML Use Case diagram is useful to record system processes and to identify the actors involved.

Personas are "real" people and actors have a specific role in Use Case.

Figure 1.11: UML Use Case diagram for Airport check-in

In the diagram above the actors are the Passenger, Tour Guide, Minor Passenger (a Child), Passenger with Special Needs (e.g. with disabilities), all playing external roles in relation to the airport.

Use cases are Individual Check-In, Group Check-In (for groups of tourists), Security Screening, etc. - representing business functions or processes taking place in the airport.
and serving the needs of passengers.

Use cases Baggage Check-in and Baggage Handling extend Check-In use cases, because a passenger might have no luggage, so baggage check-in and handling are optional.

1.7.1.2 Human Computer Interaction (User-centred)

User centered design is used to ensure that the HCI and user experience designed for the product is simple and easy to use. The development team will typically create wireframes and low-fidelity prototypes of the user interface before creating a high fidelity prototype.

Often these will be created using interactive software which allows the HCI design to be moved to the build phase within the sprint.

Figure 1.12: Wireframe, low fidelity and high fidelity prototypes of HCI

1.7.2 Build (Implementation)

The build part of the sprint involves creating the software with all of the features that have been committed to in that sprint. The focus on an agile methodology is the generation of working software over documentation.

The team members will work on implementing the features they have committed to from the product backlog. Each team member will unit test and debug the code they are working on - team members may work together on certain sections of the program or if one team member requests assistance.
1.7.2.1 Integration

As separate modules or sub-programs pass unit testing, they then need to be integrated into a program. This integration is part of the build phase and brings the separate sections of code created by the team members into a program.

Integration testing is carried out to ensure that the units of code function correctly when brought together. This integration test would test aspects of the code such as parameter passing to ensure that each unit of code can correctly perform its role within the larger program.

1.7.2.2 Deployment

Once the release has been fully tested, it is ready for deployment. Deployment is the process of making the new software available to the customer.

1.7.3 Testing

Each release is through tested thoroughly before is made available to the customer. In addition to the unit and integration testing that is carried out during the sprint, validation testing is also carried out to check that each feature from the product backlog, which has been committed to, is successfully delivered by meeting its acceptance criteria.

Usability testing is also used to access the suitability of the product for real users.

1.7.3.1 Usability testing

Usability testing is technique used in user-centred design to evaluate a product by testing it on real users performing real tasks. Usability testing should be focused on the most important user goals (such as being able to change a password) and/or the most important organisation goals (for an ecommerce web site this would be making a purchase). These tasks will be detailed in the User Stories, User Scenarios and Use Cases.

1.7.3.2 Usability testing techniques

Usability testing involves watching people trying to use something for its intended purpose. Data can be gathered using a number of techniques.

Concurrent Think Aloud (CTA)

The user talks about his/her experience of using the software as he/she attempts to complete a task. The tester records this feedback. This is useful technique, as members of the design team would understand the user's thoughts as they occur and as they attempt to problem solve any issues as they happen.

However, this test can interfere with usability metrics, such as accuracy and time on tasks because the user is distracted from using the product while he/she is talking.
**TOPIC 1. PROJECT DESIGN PROCESS**

**Concurrent Probing (CP)**

Another technique is probing participants as they work on tasks—when they say something interesting or do something unique, the team member asks follow-up questions. Usability tests rarely include CP because it interferes with the natural thought process. It can also affect the progression participants would make on their own, without interruptions.

However, CP can be valuable if goals do not include strict user experience measurements. Team members can often use CP when pre-testing terms and wording to assess how well people understand the questions and concepts.

**Retrospective Probing (RP)**

Another technique is waiting until the session is complete and then asking questions about the participant’s thoughts and actions. Team members will often use RP in conjunction with other methods—as the user makes comments or actions, the team member takes notes and follows up with additional questions at the end of the session.

**1.7.4 Accessibility**

Accessibility is a subset of usability testing which focuses on ensuring access for users who have additional needs and access for users using other devices (such as smartphones and tablets).

There are legal requirements which developers are required to meet, to ensure that their products do not discriminate against particular groups of users. The Equality Act 2010 requires that web sites are accessible for all users.

**1.8 Evaluation**

**Learning objective**

By the end of this section you will be able to explain:

- usability, efficiency, reliability and robustness as it applies to the evaluation of software.

The evaluation of the release is carried out against the requirements using a number of techniques. Usability testing to determine if the requirements have been met is the focus of the testing and this will be carried out using real users and real tasks (these are detailed in the User Stories/User Scenarios and Use Cases).

At the end of each sprint, the release is evaluated against the agreed **Acceptance criteria** for each requirement in the **Sprint backlog**.
1.8.1 Usability

Follow usability testing the software can be evaluated against its usability. Usability is a combination of factors including:

- Intuitive design: a nearly effortless understanding of the architecture and navigation of the site.
- Ease of learning: how fast a user who has never seen the user interface before can accomplish basic tasks.
- Efficiency of use: How fast an experienced user can accomplish tasks.
- Memorability: after visiting the site, if a user can remember enough to use it effectively in future visits.
- Error frequency and severity: how often users make errors while using the system, how serious the errors are, and how users recover from the errors.
- Subjective satisfaction: If the user likes using the system.

1.8.2 Efficiency

The release should be efficient. It should not make unreasonable demands on the hardware on which it will run. But higher efficiency can lead to less maintainable software if programmers use shortcuts which are effective, but difficult to understand.

When evaluating the release for efficiency the team will use the acceptance criteria for the project.

1.8.3 Reliability, robustness

There is often confusion between the terms robustness and reliability.

A program is robust if it can cope with problems that come from outside and are not of its own making e.g. corrupt input data. Reliability is an internal matter. A program is reliable if it runs well, and is never brought to a halt by a design flaw.

When the program is complicated the distinction between the two terms is not always clear. When a machine hangs it is not always obvious whether this is due to a failure in robustness or reliability.

Robustness

The design team should try to ensure that the design is robust: the resulting software should be able to cope with mistakes that users might make or unexpected conditions that might occur. These should not lead to wrong results or cause the program to hang.

As examples of an unexpected condition, we could take something going wrong with a printer (it jams, or it runs out of paper) or a disc drive not being available for writing, because it simply isn't there (the user's forgotten to put in the flash drive), or the user entering a number when asked for a letter.

Reliability

A reliable program always produces the expected result when given the expected input. It is designed correctly to do the task specified. Software can be evaluated as being reliable and robust.
1.9 Learning points

Summary

You should now be able to:

- describe the initialisation of a project and the subsequent research, feasibility and development;
- explain the importance of user and business requirements to determine the scope and constraints of a project and the functional and operational requirements;
- describe the processes (within an agile software development team) required to build, integrate and deploy a project;
- effectively test a software development project using a number of techniques include usability testing and accessibility;
- effectively evaluate a software development project by measuring and commenting on the usability, efficiency, reliability and robustness of the software.
1.10 End of topic test

End of topic test

Q10: A major disadvantage of the Waterfall model is:

a) the deliverables are unclear.
b) that the phases overlap.
c) it is unsuitable for very large complex projects.
d) complex and difficult to understand.

Q11: Each iteration of an agile sprint produces:

a) a product backlog
b) a release
c) a design brief
d) a user guide.

Q12: Each sprint consists of three phases:

a) Analysis, Design, implementation
b) Version, Control, Evaluate
c) Design, Implement, Evaluate
d) Design, Build, Test.

Q13: ____________ is required before the product backlog and business case can be prepared.

a) Feasibility study
b) a project plan
c) a test plan
d) a sprint backlog.

Q14: There are four types of feasibility: legal, schedule, ____________ and ____________

a) Sequential, Iterative
b) Social, Economic
c) Economic, Technical
d) Economic, Logistical.
Q15: The role of a ScrumMaster is to:

a) manage the project and allocate tasks to specific team members.
b) reflect the views of the client during development.
c) maintain the product backlog and prioritise the requirements within it.
d) coach and encourage the scrum team to make use of agile.

Q16: A sprint planning meeting is used to allow the:

a) product owner to present the top items in the product backlog.
b) ScrumMaster to update the team on the current state of the project.
c) project manager time to adjust the project plan.
d) systems analyst to investigate the system.
Topic 2

Project management

Contents

2.1 Revision .................................................. 33
2.2 Planning .................................................. 33
  2.2.1 Relationship between time, cost, scope and quality .... 34
2.3 Scheduling ............................................... 35
  2.3.1 Sequence of tasks .................................. 39
  2.3.2 Concurrency of tasks .............................. 39
  2.3.3 Resources ........................................... 39
2.4 Gantt chart .............................................. 40
  2.4.1 Critical path ....................................... 43
  2.4.2 Float / calculating float .......................... 43
2.5 Evaluation .............................................. 44
  2.5.1 Use of project plan as project monitoring / management tool .. 44
  2.5.2 Adjustments to critical path ....................... 45
  2.5.3 Effectiveness (goal-tracking) ...................... 45
2.6 Learning points ....................................... 46
2.7 End of topic test ...................................... 47

Prerequisite knowledge

From your studies at Higher you should already know that:

- software development normally involves a number of stages including analysis, design, implementation, testing and evaluation;
- constructing a test plan is a key element of ensuring that a software development project is correctly tested and accepted as meeting the requirements of the client/user.

Learning objectives

By the end of this topic you will be able to:

- describe the relationship between time, cost, scope and quality in the project "triangle";
- understand the importance of scheduling and how this can be used to organise the work of a project;
• create a Gantt chart for a simple project, calculate the critical path and calculate float for tasks that are not part of that path;

• describe how a project plan can be used to monitor and evaluate progress within a project and the completion of specific project goals.
2.1 Revision

Quiz: Revision

Q1: A company wishes to add a network capability to their recently acquired computer program. In maintenance terms this would be an example of:

a) corrective maintenance  
b) routine maintenance  
c) perfective maintenance  
d) adaptive maintenance

Q2: Which one of these would NOT be found in the technical guide?

a) Operating system required.  
b) Hardware requirements  
c) Memory requirements  
d) Tutorial

Q3: During the software development process, which one of the following is responsible for converting the design into the actual program code?

a) Tester  
b) Programmer  
c) System analyst  
d) Client

Q4: Which one of the following terms is best described by the phrase below?
"How well a program operates without stopping due to design faults".

a) Robustness  
b) User friendly  
c) Reliability  
d) Efficiency

2.2 Planning

Project management is the process of planning and controlling the activities of a project to ensure that it is delivered on time, with the required features and within the budget available to the required quality.

The Association Of Project Management define project management as:

"Project management is the application of processes, methods, knowledge, skills and experience to achieve the project objectives."
Projects are different from the day-to-day business of an organisation - a project is a specific piece of work which is defined by what it attempts to achieve. A project is usually judged to be successful if it achieves the objectives (normally according to some acceptance criteria) within an agreed timescale and budget.

A project plan is more than just a list of tasks to be completed. It is a plan that details individual responsibilities, resources available to complete work, the order of work and the key milestones.

2.2.1 Relationship between time, cost, scope and quality

Learning objective

By the end of this section you will be able to:

- describe the relationship between time, cost, scope and quality in the project “triangle”.

Before looking more closely at the process of creating a simple project plan, it is important to understand the relationship between time, cost, scope and quality when planning a project.

These are the constraints of the project. Changes to any one of these constraints has an immediate impact on the others.

Figure 2.1: The Project management triangle

SCOPE is what needs to be done to deliver the project. The requirements for the software being developed. With the waterfall model of software development, the requirements of the project are completely known after the analysis phase. Any subsequent changes to the scope will add time and/or cost to the project because more time and/or more resources (people who have to be paid or over time hours for existing staff) will be required to deliver the project to the same required level of quality.

TIME is the amount of time it takes to complete the project, to the required scope with the available budget (cost) to the required quality.
**2.3 Scheduling**

The role of **project manager**, typically in projects which use the waterfall approach, is to use his/her skills to turn the project **requirements specification** into a **work breakdown structure** (WBS) and then to estimate the amount of **effort** required to complete each task.

These tasks can then be allocated as items of work to be completed by individuals and/or teams. The following example work breakdown schedule is from a project to produce a new web site to market a new children's toy. The site will have interactive elements and will provide detailed reports on usage.
Figure 2.2: Partial sample from an initial Work Breakdown Structure

1. Gather requirements
   1.1 Technical specifications
      1.1.1 Expected bandwidth
      1.1.2 User registration
      1.1.3 Restricted areas
   1.2 User requirements
      1.2.1 Menu navigation
      1.2.2 Interactive modules
      1.2.3 Static pages
      1.2.4 Flash elements
   1.3 Reporting requirements
      1.3.1 Bandwidth and usage
      1.3.2 Page views
      1.3.3 Session length

2. Establish design
   2.1 Design elements
      2.1.1 Banner
      2.1.2 Footer
      2.1.3 Logo
      2.1.4 Colour scheme
      2.1.5 Font usage
   2.2 Overall layout
      2.2.1 Column setup
      2.2.2 Optional modules
      2.2.3 Navigation layout

2.3 Content elements
   ...

A work breakdown structure is not simply a to-do list of tasks. It represents the hierarchy of tasks to be completed and the milestones within the project. A milestone is a specific point in the project timeline which has some level of importance. It will be a point at which the project progress is evaluated and reported to the client. It can also be a point where the client will be given a deliverable; where the client will receive some of the output from the project work. When the tasks within the milestone are completed then the milestone is achieved.
In the above example **Establish design** is a milestone and is a deliverable: on the completion of this milestone the client will receive the design documentation for the project.

The **effort** to complete each task in the work breakdown structure is normally estimated as person-hours/days or as a financial amount. If person hours/days are used then the resulting cost can be calculated using the pay rate(s) for the staff allocated to the task or the day rate charged by the developer.

This following WBS with Effort as person days shows some tasks which can be carried out one after the other and some tasks which can be done at the same time. This means that the total time to complete each milestone will vary.
<table>
<thead>
<tr>
<th></th>
<th>Task Description</th>
<th>Effort (days)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Gather Requirements</td>
<td>7</td>
</tr>
<tr>
<td>2</td>
<td>Technical specifications</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>Expected bandwidth</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>User registration</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>Restricted areas</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>User requirements</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>Menu navigation</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>Interactive modules</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>Static pages</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>Flash elements</td>
<td>4</td>
</tr>
<tr>
<td>11</td>
<td>Reporting requirements</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>Bandwidth &amp; usage</td>
<td>2</td>
</tr>
<tr>
<td>13</td>
<td>Page views</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>Session length</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>Establish Design</td>
<td>8</td>
</tr>
<tr>
<td>16</td>
<td>Design elements</td>
<td>4</td>
</tr>
<tr>
<td>17</td>
<td>Banner</td>
<td>1</td>
</tr>
<tr>
<td>18</td>
<td>Footer</td>
<td>0.5</td>
</tr>
<tr>
<td>19</td>
<td>Logo</td>
<td>0.5</td>
</tr>
<tr>
<td>20</td>
<td>Color scheme</td>
<td>3</td>
</tr>
<tr>
<td>21</td>
<td>Font usage</td>
<td>3</td>
</tr>
<tr>
<td>22</td>
<td>Overall layout</td>
<td>4</td>
</tr>
<tr>
<td>23</td>
<td>Column setup</td>
<td>2</td>
</tr>
<tr>
<td>24</td>
<td>Optional modules</td>
<td>2</td>
</tr>
<tr>
<td>25</td>
<td>Navigation layout</td>
<td>2</td>
</tr>
<tr>
<td>26</td>
<td>Content elements</td>
<td>3</td>
</tr>
<tr>
<td>27</td>
<td>About page</td>
<td>2</td>
</tr>
<tr>
<td>28</td>
<td>Contact page</td>
<td>2</td>
</tr>
<tr>
<td>29</td>
<td>Services page</td>
<td>2</td>
</tr>
<tr>
<td>30</td>
<td>FAQ page</td>
<td>1</td>
</tr>
<tr>
<td>31</td>
<td>Photo Gallery</td>
<td>3</td>
</tr>
<tr>
<td>32</td>
<td>Select Technical Framework</td>
<td>4</td>
</tr>
<tr>
<td>33</td>
<td>Evaluate options against requirements</td>
<td>1</td>
</tr>
<tr>
<td>34</td>
<td>Evaluate cost and time to develop</td>
<td>3</td>
</tr>
<tr>
<td>35</td>
<td>Make decision</td>
<td>1</td>
</tr>
<tr>
<td>36</td>
<td>Implement Technical Framework</td>
<td>11</td>
</tr>
<tr>
<td>37</td>
<td>Build or acquire back end</td>
<td>4</td>
</tr>
<tr>
<td>38</td>
<td>Build or acquire front end (user interface)</td>
<td>5</td>
</tr>
<tr>
<td>39</td>
<td>Integrate back end and front end</td>
<td>6</td>
</tr>
<tr>
<td>40</td>
<td>Create Content</td>
<td>14</td>
</tr>
<tr>
<td>41</td>
<td>Create content summary</td>
<td>3</td>
</tr>
<tr>
<td>42</td>
<td>Establish content details</td>
<td>3</td>
</tr>
<tr>
<td>43</td>
<td>Assign content creation</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 2.3: Sample WBS with Effort as person days
2.3.1 Sequence of tasks

There will be interdependencies between tasks that will dictate the order in which the tasks can be completed. For example: a sub-program to calculate the pay for workers cannot be tested until the sub-program to read the workers details from a database is completed. Another example would be that the installation of new hardware for a system cannot be carried out until the hardware has been purchased!

In any project, some tasks will have requirements that have to be completed before these tasks can be carried out and these requirements will determine where on the project timeline these tasks appear.

2.3.2 Concurrency of tasks

Tasks which do not depend on each other can be allocated to different team members to complete and can progress concurrently (at the same time as each other). For example: developing "Static Pages" and developing "Interactive modules" are not dependent on each other, so these tasks can be carried out at the same time by different team members.

2.3.3 Resources

Projects require resources in order to be completed. A significant amount of the resource required to complete a project will be the members of the project team. There will also be other resources that may be required to complete the project: specialist hardware/software, external consultants/designers, legal and other professionals.

When scheduling the work for the project it is important to consider how the work makes use of the resources available.

For example: developing "Static Pages" and developing "Interactive modules" are not dependent on each other, however, if these tasks were allocated to the same resource (team member) then that creates a issue in the scheduling of the project, because the same person cannot work on both tasks and complete them in the time available. In this case the resource for one of the tasks would have to be changed (give the work to someone else to do) or the tasks would have to be scheduled to create a dependency between them to reflect that the same resource will complete "Static Pages" and then "Interactive modules"- note: there is no change to the effort required to complete each task.

The amount of time it takes to complete an activity can be reduced by providing that activity with more resource. This means, generally, that two people working together can complete a task in less time than a single person would take to complete the same task. It would not normally be the case that doubling up on a resource halves the time to complete the activity. This is because two people working together will have to coordinate their work and this takes some time to do.
2.4 Gantt chart

Learning objective
By the end of this section you will be able to:

• create a Gantt chart for a simple project;
• calculate the critical path;
• calculate float for tasks that are not part of that path.

A Gantt chart is a way of showing the work breakdown schedule against a timeline.

Figure 2.4: A simple Gantt chart

The timeline is used to show:

• when the activity to complete tasks begins and ends,
• how long each activity is scheduled to last,
• where activities overlap with each other and
• the start and end dates for the whole project.

Gantt charts can also be used to show the dependencies between tasks on the timeline. Simple Gantt charts can be created on paper, however, more complex charts (consisting of more than 10 activities), should ideally be created using project management software such as the free ProjectLibre software or Microsoft Project.
The work breakdown structure for a simple project is shown below:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Predecessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Gather User Requirements</td>
<td>7 days</td>
<td></td>
</tr>
<tr>
<td>2. Establish Design</td>
<td>8 days</td>
<td>1</td>
</tr>
<tr>
<td>3. Select Technical Framework</td>
<td>4 days</td>
<td>2</td>
</tr>
<tr>
<td>4. Implement Technical Framework</td>
<td>11 days</td>
<td>3</td>
</tr>
<tr>
<td>5. Create Content</td>
<td>14 days</td>
<td>2</td>
</tr>
<tr>
<td>6. Load Content</td>
<td>2 days</td>
<td>4, 5</td>
</tr>
<tr>
<td>7. Test Site</td>
<td>2 days</td>
<td>6</td>
</tr>
<tr>
<td>8. Roll Out Site</td>
<td>3 days</td>
<td>7</td>
</tr>
</tbody>
</table>

The effort here is shown as the number of working days required to complete the task assuming that one person is allocated the work. The predecessor indicates which activities must be completed before an activity can be started. For example, the activity "Establish Design" cannot be started until the activity "Gather User Requirements" is completed.

A simple Gantt chart can be constructed from this information as follows:

Draw a bar to show the number of days allocated to the first activity. The length of the bar reflects the number of days.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td>■■■■■■■■■■</td>
</tr>
</tbody>
</table>

Activity number 2 depends on the completion of activity 1. This means that it starts when Activity 1 finishes. Now add a bar for activity number 2.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td>■■■■■■■■■■</td>
</tr>
<tr>
<td>2 Establish design</td>
<td>8 days</td>
<td>■■■■■■■■■■</td>
</tr>
</tbody>
</table>

Activity number 3 depends on the completion of activity number 2. We can now add the bar for activity number 3.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td>■■■■■■■■■■</td>
</tr>
<tr>
<td>2 Establish design</td>
<td>8 days</td>
<td>■■■■■■■■■■</td>
</tr>
<tr>
<td>3 Select technical framework</td>
<td>4 days</td>
<td>■■■■■■■■■■</td>
</tr>
</tbody>
</table>
Activity number 4 depends on the completion of activity number 3.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td></td>
</tr>
<tr>
<td>2 Establish design</td>
<td>8 days</td>
<td></td>
</tr>
<tr>
<td>3 Select technical framework</td>
<td>4 days</td>
<td></td>
</tr>
<tr>
<td>4 Implement technical framework</td>
<td>11 days</td>
<td></td>
</tr>
</tbody>
</table>

Activity number 5 depends on the completion of activity number 2. This bar can be added so that the activity runs concurrent with activities 3 and 4.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td></td>
</tr>
<tr>
<td>2 Establish design</td>
<td>8 days</td>
<td></td>
</tr>
<tr>
<td>3 Select technical framework</td>
<td>4 days</td>
<td></td>
</tr>
<tr>
<td>4 Implement technical framework</td>
<td>11 days</td>
<td></td>
</tr>
<tr>
<td>5 Create content</td>
<td>14 days</td>
<td></td>
</tr>
</tbody>
</table>

Activity number 6 depends on the completion of both activity 4 and activity 5 but activity 4 and activity 5 both finish on different days. When this happens, the new activity must start when both are complete, so the start of activity 6 is scheduled after activity 4 is completed.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Days</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather user requirements</td>
<td>7 days</td>
<td></td>
</tr>
<tr>
<td>2 Establish design</td>
<td>8 days</td>
<td></td>
</tr>
<tr>
<td>3 Select technical framework</td>
<td>4 days</td>
<td></td>
</tr>
<tr>
<td>4 Implement technical framework</td>
<td>11 days</td>
<td></td>
</tr>
<tr>
<td>5 Create content</td>
<td>14 days</td>
<td></td>
</tr>
<tr>
<td>6 Load content</td>
<td>2 days</td>
<td></td>
</tr>
<tr>
<td>7 Test site</td>
<td>2 days</td>
<td></td>
</tr>
<tr>
<td>8 Roll out site</td>
<td>3 days</td>
<td></td>
</tr>
</tbody>
</table>

The remaining activities are sequential (after each other) so the final Gantt chart looks like this:

![Figure 2.5: A complete simple Gantt chart](image)

Using project management software the same Gantt chart can be mapped to the working week and specific project dates can be calculated from an initial start date.

![Figure 2.6: A simple software generated Gantt chart](image)
2.4.1 Critical path

It is important that projects are delivered on time. The Critical path is the pathway through a project where a delay to the completion of an activity will extend the duration of a project.

It is calculated by first identifying the route through the project where each activity depends on the previous activity and there is no gap between the start and the end points. In our simple example this is now shown in red:

*Figure 2.7: Simple Gantt Chart with critical path shown in red*

Now that the critical path has been identified, it can now be calculated as a number of days. In this case, the critical path is 37 days of effort.

With project management software, the critical path is calculated automatically and the dependencies between activities are shown with arrows:

*Figure 2.8: A simple software generated Gantt Chart with critical path in red*

2.4.2 Float / calculating float

When an activity is not part of the critical path there is often some flexibility around how the activity can be scheduled. Because the activity is not "critical" there may be some flexibility around when the task is started or finished and around how long the task can take to complete. This flexibility can be used to schedule tasks so that they align with the availability of the resource or resources required to complete them. This flexibility is referred to as float.

In our simple example, activity 5 has a float of one day. This is because the task could be started one work day of effort late and still be completed without changing the critical path:
Activity: Gantt chart

Q5: The following work breakdown structure (WBS) is given: Complete the Gantt chart with the correct durations and with the correct dependencies.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Predecessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 System investigation</td>
<td>4 days</td>
<td></td>
</tr>
<tr>
<td>2 Design and consultation</td>
<td>10 days</td>
<td>1</td>
</tr>
<tr>
<td>3 Develop application code</td>
<td>4 days</td>
<td>2</td>
</tr>
<tr>
<td>4 Develop user experience</td>
<td>6 days</td>
<td>2</td>
</tr>
<tr>
<td>5 Create data sources</td>
<td>7 days</td>
<td>1</td>
</tr>
<tr>
<td>6 Test application</td>
<td>6 days</td>
<td>3, 4, 5</td>
</tr>
<tr>
<td>7 Submit application to app store</td>
<td>3 days</td>
<td>6</td>
</tr>
</tbody>
</table>

2.5 Evaluation

The reason for creating a Gantt chart and for planning the activity of a project is to ensure that the project is delivered on time and to budget.

2.5.1 Use of project plan as project monitoring / management tool

The project manager uses these documents as a base line plan and manages the real progress of the project against this base line. He/She gathers reports on progress from the individuals/teams concerned and updates the plan accordingly.

These updates will report the level of completion of an activity. The project plan can then be updated to show the level of completion and this can be compared to expected progress.

Figure 2.10: Project with completion shown

In this chart the amount of an activity which is completed is shown by the dark bar within the activity bar. In this case, activity 3 is 75% complete and activity 5 is 35% complete.
If the date of this progress update was April 23rd 2015 then activity 3 is behind schedule and activity 5 is ahead of schedule.

When tasks fall behind schedule this is a problem for the project and it is the responsibility of the project manager to take action to support the project and bring it back on schedule or to seek agreement about its delay with the client.

To bring a project back on schedule, it may be possible to reduce the scope of the work to be carried out, allowing activities to be completed in less time. This will allow the project to be completed on time but without all of the original requirements being met.

Alternatively, the project manager may allocate additional resources to the activities of the project to ensure that the work is brought back on schedule however this will result in additional costs which the developer and/or the client may have to be responsible for.

2.5.2 Adjustments to critical path

As the project manager monitors and controls the project, some tasks on the critical path may be completed ahead of schedule and/or other tasks may take longer than planned and become part of the critical path. During the execution of the project the critical path has to be closely monitored and adjusted to reflect the true position of the project and the expected completion date.

2.5.3 Effectiveness (goal-tracking)

Successful completion of each activity in the project plan will mean that an element of the requirements specification has been achieved within a waterfall model project.

Similarly, within an agile project, each item in the product backlog is a goal that is to be achieved - something that the software is to be able to do.

It is important, as the project progresses, that these goals are tracked and monitored to ensure they are completed effectively. This will be done using either the project plan or the product backlog to identify features of the software and to sign them off as completed.
2.6 Learning points

You should now know:

- how to describe the relationship between time, cost, scope and quality in the project “triangle”;
- the importance of scheduling and how this can be used to organise the work of a project;
- how to create a Gantt chart for a simple project, calculate the critical path and calculate float for tasks that are not part of that path;
- how a project plan can be used to monitor and evaluate progress within a project and the completion of specific project goals.
2.7 End of topic test

End of topic test

Q6: Project management is the application of processes, methods, knowledge, skills and experience to achieve the __________

a) project objectives.
b) requirements definition.
c) project plan.
d) business case.

Q7: There are four constraints that are applied to any project. These are:

a) developer resource, compiler time, rentals and management.
b) cost, time, scope and milestones.
c) cost, time, scope and quality.
d) cost, time, product backlog and sequence.

Q8: If time and cost allocated to project are fixed, how can it still be delivered if items are added to the scope?

a) By asking staff to work for free on the project in their own time.
b) By changing the development methodology to agile.
c) By asking staff to find similar open source projects and copy the code.
d) By reducing the quality of the final product or removing items from the scope.

Q9: A project manager defines the tasks to be completed in the work breakdown structure and estimates the __________ required to complete each task.

a) time (or cost)
b) effort (or cost)
c) resources (or cost)
d) quality

Q10: Activities within a project plan can be carried out:

a) in order on the timeline.
b) in priority order.
c) sequentially or concurrently.
d) according to developer preferences.
TOPIC 2. PROJECT MANAGEMENT

Q11: Calculate the duration of the critical path for a project with the following work breakdown structure:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Predecessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather User Requirements</td>
<td>6 days</td>
<td></td>
</tr>
<tr>
<td>2 Design work</td>
<td>8 days</td>
<td>1</td>
</tr>
<tr>
<td>3 Develop server code</td>
<td>4 days</td>
<td>2</td>
</tr>
<tr>
<td>4 Develop application code</td>
<td>5 days</td>
<td>2</td>
</tr>
<tr>
<td>5 User Interface Development</td>
<td>6 days</td>
<td>2</td>
</tr>
<tr>
<td>6 Test server code</td>
<td>2 days</td>
<td>3</td>
</tr>
<tr>
<td>7 Test application</td>
<td>2 days</td>
<td>4,6</td>
</tr>
<tr>
<td>8 Test application/server integration</td>
<td>5 days</td>
<td>7,8</td>
</tr>
<tr>
<td>9 Roll out mobile application</td>
<td>3 days</td>
<td>9</td>
</tr>
</tbody>
</table>

a) 28 person days  
b) 29 person days  
c) 30 person days  
d) 31 person days

Q12: Calculate the float for activity 7, given a project with this work breakdown structure:

<table>
<thead>
<tr>
<th>Activity</th>
<th>Effort</th>
<th>Predecessor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Gather User Requirements</td>
<td>6 days</td>
<td></td>
</tr>
<tr>
<td>2 Design work</td>
<td>8 days</td>
<td>1</td>
</tr>
<tr>
<td>3 Develop server code</td>
<td>4 days</td>
<td>2</td>
</tr>
<tr>
<td>4 Develop application code</td>
<td>5 days</td>
<td>2</td>
</tr>
<tr>
<td>5 User Interface Development</td>
<td>6 days</td>
<td>2</td>
</tr>
<tr>
<td>6 Test server code</td>
<td>2 days</td>
<td>3</td>
</tr>
<tr>
<td>7 Test application</td>
<td>2 days</td>
<td>4,6</td>
</tr>
<tr>
<td>8 Test application/server integration</td>
<td>5 days</td>
<td>7,8</td>
</tr>
<tr>
<td>9 Roll out mobile application</td>
<td>3 days</td>
<td>9</td>
</tr>
</tbody>
</table>

a) 0 person days  
b) 1 person days  
c) 2 person days  
d) 3 person days
Q13: A project manager uses the project plan as a __________ to control and manage the project.

a) baseline  
b) benchmark  
c) budget tool  
d) specification

..........................................

© HERIOT-WATT UNIVERSITY
Topic 3
Information systems design techniques

Contents

3.1 Revision ................................................. 53
3.2 Human computer interaction (user-centred) ................. 54
3.3 User centred design ...................................... 54
  3.3.1 Principles of User centred design .................. 55
  3.3.2 Stages of User centred design ...................... 57
  3.3.3 Capturing User requirements ....................... 59
  3.3.4 User stories ........................................ 61
  3.3.5 User scenarios ...................................... 61
  3.3.6 Use case ........................................... 63
3.4 User interface design .................................... 64
  3.4.1 Wireframe ........................................... 65
  3.4.2 Low fidelity prototype .............................. 66
  3.4.3 High fidelity prototype ............................. 67
  3.4.4 Application style guide ............................ 68
3.5 Resolving user centred design and SCRUM .................. 70
3.6 Learning points ......................................... 71
3.7 End of topic test ........................................ 72

Prerequisite knowledge

From your studies at Higher you should already know:

- the key features of usability. That well designed user interface should be fit for purpose, efficient, robust and maintainable;

- that Accessibility is a major feature of modern software. User interfaces are designed to be accessible from a wide range of devices and by a diverse group of users (including those with additional needs such as auditory or visual needs);

- that interfaces are optimised for specific platforms and for specific delivery media.

Learning objectives

By the end of this topic you will be able to:
• design, describe and apply the key aspects of user-centred design including using analysis tools to capture user requirements;

• develop user profiles, user personas, user stories, user scenarios and use cases;

• describe, explain and apply the development of user interfaces using prototypes (from low to high fidelity including wireframes);

• describe how an application style guide is used to ensure a consistent look and feel to an application;

• create an initial application style guide for an application;

• how User Centred Design can be applied within the Scrum methodology.
3.1 Revision

Quiz: Revision

Q1: Designers use this technique to generate an outline design of the user interface of an application/website:

a) pseudocode
b) RAD
c) iterative prototype
d) wireframe

Q2: Two key features of any user interface are:

a) contrast and definition
b) modularity and security
c) usability and accessibility
d) robustness and reliability

Q3: Responsive design aims to ensure that an interface is:

a) viewable and appropriate on any device type.
b) activated by touch input.
c) adapts to a range of inputs.
d) constructed using CSS.

Q4: Meta-data is one method of optimising web pages for:

a) pre-loading of links.
b) faster download of page elements.
c) bookmarking
d) search engines

Q5: A user interface for expert users would feature:

a) icons and pull down menus.
b) shortcut keys and text-based commands.
c) voice activated commands.
d) tactile input
3.2 Human computer interaction (user-centred)

**Learning objective**
By the end of this section you will be able to:

- define the term "User Experience (UX)".

Human Computer Interaction (HCI) is a major factor in the design and implementation of the hardware and software that we use every day. As computer technology has moved, from limited use to become an every day part of our lives, the interfaces used to control the software we use have expanded to meet the requirements of a diverse range of users and computing platforms.

The design of Human Computer Interaction is a specialised area of software development and the development of the user experience (UX): the blend of physical interaction, feedback to users and the interface through which that interaction occurs; has become one of the key factors in the success or otherwise of new technologies.

In this topic you will look at techniques for designing, implementing and testing user interfaces and capturing the needs of specific users.

3.3 User centred design

**Learning objective**
By the end of this section you will be able to:

- design, describe and apply the key aspects of user-centred design including using analysis tools to capture user requirements;
- develop user profiles, user personas, user stories, user scenarios and use cases.

User-centred design (UCD) is a method that ensures that a program, product, app or website will be easy to use. This approach to the design of a product focuses on the user: their needs, their wants, and their limitations and these are considered at each and every stage of development.

User-centred design is a multi-stage problem solving process. At regular points during the development of a product, designers must analyse and imagine how users are likely to use the product. User-centred design also requires that designers carry out usability testing, with real world users, to ensure that the product meets the needs of the users and functions in the way that users would expect it to function.

This kind of testing is necessary as it is often very difficult for the designers of a product to understand intuitively what a first-time user of their design experiences, and what each user's learning curve may look like.

The key strength of user-centred design is that it focuses on how users can, want or need to use the product rather than forcing the users to change their behaviour to
accommodate the product.

### 3.3.1 Principles of User centred design

The model for user-centred design is an agreed international standard (ISO 9241-210 *Human-centred design for interactive systems*) which describes the approach to design as "an approach to systems design and development that aims to make interactive systems more usable by focusing on the use of the system and applying human factors/ergonomics and usability knowledge and techniques."

This standard approach to user-centred design defines the principles of the approach as:

1. The design is based upon an explicit understanding of users, tasks and environments.
2. Users are involved throughout design and development.
3. The design is driven and refined by user-centered evaluation.
4. The process is iterative.
5. The design addresses the whole user experience.
6. The design team includes multi-disciplinary skills and perspectives.

This approach ensures that users are always considered during the development of the product.

**The design is based upon an explicit understanding of users, tasks and environments**

It is key that the design team understand how the intended users will use the product: the context of use. To achieve this the design team need to consider the three main features of user experience.

To understand:

- the user;
- what the user wants to do with the product;
- understand the environment in which the product is used.

As an example, consider the interface aimed at a teenager downloading music on a mobile phone with a businessperson accessing corporate data on a device. What makes a positive user experience in one situation may not be acceptable in another so understanding the context of use is vital to developing a successful product.

**Users are involved throughout design and development**

Users should be actively involved in all the design phases. These means that user involvement should go beyond initial focus groups and interviews (such as at the beginning of the project) or user surveys (at the end of the project).

This "active involvement" is normally achieved through significant engagement with the
user, not just demonstrating a product but also actively involving the user in the design process.

**The design is driven and refined by user-centred evaluation**

Usability testing is used throughout the development to shape how the product works with the user. Usability testing consists of a number of formal procedures to test a product on real users doing real tasks. This regular testing of working prototypes is one example of how users can be involved throughout the design and development of the product.

Because issues relating to the usability of the product can occur at any point in the product development, it is important to regularly test the working elements of the product with users. This approach fits well with the Agile Model and Agile's focus on working code, short development times and with fewer functions delivered early in the project.

**The process is iterative**

To be effective this process must be iterative. Users are very poor at explaining what they require from a system. In most cases, users have to be shown an initial design and then work out what they do and don't like. In this way, the user then influences the next version of the product and so on. This iterative approach again matches closely with the approach of Agile development.

**The design addresses the whole user experience**

User experience is more complex that just making something easy to use. It also includes how the user sees and feels about the product. Making a product easy to use is a useful starting position but usability, and a positive user experience, is about more than simplifying the user interface.

If the design team are engaged in iterative user-centred evaluation and frequently involving the user in the development of the product then the team will learn what is important to the user and can then attempt to incorporate these features.

**The design team includes multi-disciplinary skills and perspectives**

If the product is to be successful, the design team needs a range of experience and to drawn on ideas and inspiration from across the team. Too often software development teams are too specialised, often predominately composed of programmers or system analysts.

An overly specialised team is often the wrong way to approach user centred design. To be successful the design team should draw views from a range of personnel: graphics designers, programmers, user experience experts, end users, project managers and many others.
3.3.2 Stages of User centred design

In common with many design methodologies, UCD is iterative and consists of four stages.

**Understand and specify context of use**

At this initial stage it important to understand the user, the tasks the user will carry out and the environment in which this will happen. A teenager, downloading and managing a music library on his laptop is a significantly different context of use from a nuclear scientist, controlling a maintenance robot in a nuclear reactor.

At this initial stage, the design team would look to describe three main areas: the environment, the user and the user tasks (and associated risks).

To assist with understanding users it is common to develop User profiles and from these profiles various User personas. These documents describe the user, his/her wants, needs and expectations from the product.

To develop a better understand on the tasks related to the product, the design team will develop User stories/User scenarios and Use cases. These documents describe the tasks and the motivation to complete them.

The process will also record the technical and environmental constraints on development (for example: what types of hardware will be used, in which organisation, technical and physical environments?)
Specify the User and organisational requirements

Now that the design team have a good understanding of the user, the tasks and the environment in which the product will operate they can produce a full specification for the product. This specification will include the documentation generated when developing an understanding of the "context of use" but will also include performance and satisfaction criteria (measures of usability for the product) and test methods (the means of determining whether the usability requirements have been met).

To ensure that the requirements are correct, other user centred design activities (such as interviews, surveys, focus groups, field studies, task analysis, benchmark usability tests, or paper prototyping) can be used early in the development process to obtain feedback from users to iteratively refine requirements.
Produce design solutions

The production of the solution will generally start with simple wireframes such as initial pencil sketches and move through formal wireframes to low-fidelity prototypes and then to high-fidelity prototypes and then the fully implemented and functional solution.

![Different iterations of solution prototype](image)

Typically, at each iteration of the cycle, some complexity/refinement is added to the solution, based on the updates to the requirements (as the result of the previous user testing).

Evaluate design against requirements

The evaluation of the solution is carried out against the requirements using a number of techniques. Usability testing to determine if the requirements have been met is the focus of the testing and this will be carried out using real users and real tasks (these are detailed in the User stories/User scenarios and Use cases).

While the solution is in development the feedback from testing can be documented in a Prototyping session report. This document records the results of the testing and includes recommendations to make changes to the "Context of Use" documentation, to the performance and satisfaction criteria or to the redefine the test methods.

3.3.3 Capturing User requirements

The requirements of the users of an application are captured using a number of tools. Because end users are not always available (there may be many different end users - a single product owner cannot directly represent all the views of these users). The team developing the application will start by understanding each user type and then develop what each user is looking for from the application.
**Personas**

A first step for the design team is to develop personas for each of the target user types. Personas are fictional characters created to represent the different user types that the design team have encountered or that the team believe are important to the project.

Personas can be developed iteratively and their development can involve users and the design team members. Developing the personas with a number of people increases the accuracy of the personas and creates a level of awareness about the users that helps the design team. As people become familiar with the personas, they start talking about them as if they were actual people.

Personas do not need to be complex to be useful. The design team can begin by creating brief outlines of personas based on initial conversations with those involved in the project.

An example persona would be:

**Figure 3.4: An example persona**

<table>
<thead>
<tr>
<th>Nikki Taylor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>PROFILE:</strong></td>
</tr>
<tr>
<td><strong>GENDER:</strong> Female</td>
</tr>
<tr>
<td><strong>AGE:</strong> 26</td>
</tr>
<tr>
<td><strong>LOCATION:</strong> Central London</td>
</tr>
<tr>
<td><strong>OCCUPATION:</strong> Lawyer</td>
</tr>
</tbody>
</table>

**Motivations**
- Likes to meet friends after work but often organises this at the last minute.
- Nikki has started to use her mobile phone to organise her social life and wants to be able to book a restaurant table easily.
- She also wants to know which places are the best to eat in locally without asking her colleagues or friends.

**Goals**
- Needs to organise a meal out for herself and four friends.
- Would like a choice of restaurants to match her budget.

**Frustrations**
- Not being able to search for restaurants locally online.
- Doesn’t find the reviews she reads online match the places she eats in.
- Hates making calls. She’s too busy!
**Activity: Personas**

Consider a simple application that you wish to develop.

Create two personas that describe two typical users of the application. Use the same format as the sample persona above.

..........................................

### 3.3.4 User stories

A user story is a brief statement that identifies the user and his/her need. It is a direct statement that relates to a specific persona. The personas created by the design team will have a general level of detail however the user story will provide specific details about a task that the user will be engaged in. One persona may have several user stories developed around it.

Here's an example user story for a restaurant finder app:

"As a busy lawyer, Nikki wants to arrange a quick lunchtime meeting with colleagues. She needs quick access to restaurant information in the local area and a quick way to see which have tables available because she is making arrangements between meetings."

Notice that the user story identifies who the user is, what she needs, and why she needs it.

Here’s another example of a user story that might apply to the same restaurant finder app designed to meet our first user’s need:

"Jack is a single parent and is meeting his children after school. He wants to find a local family friendly restaurant for a surprise birthday party for one of his children."

This second example maintains the same structure, but represents a very different user. A restaurant finder app is likely to have a wide array of users with different needs, big and small. User stories help to document the practical differences in need among those users.

**Activity: User stories**

For each user develop a small user story that defines what drives a user to make use of your application.

..........................................

### 3.3.5 User scenarios

A user scenario expands upon already developed user stories by including details about how a system might be interpreted, experienced, and used. User scenarios provide detail about the user's goal, detail any assumed knowledge the user has and the level of experience the user has.
Just as with user stories, the design team may imagine several scenarios for each persona group.

Here’s an example of a user scenario, again from our hypothetical restaurant finder app:

Jerry is a vegan and is arranging a night out with two friends: one is a vegetarian and the other eats meat. Jerry finds it difficult to find restaurants that meet all their needs and, in the past, nights out haven’t been very good because there is often a problem with the food order. He has tried searching online but, when he does find somewhere, frequently he rings them to find out they have no tables available. He now has only one day left to make arrangements for the night out and is getting worried. He still has to book a restaurant and then call his friends to make arrangements to meet.

Notice how this scenario gives the user some backstory, gives some context about his needs, and tries to specify the gaps in his knowledge that might lead to interaction difficulties.

**Activity: User scenarios**

Extend your user stories from the previous section, development them to become user scenarios.
### 3.3.6 Use case

A use case is a list of steps a user would take to perform an action. A typical use case will start by detailing how the user got to the current position and then goes through every step until the user completes the operation or fails.

Here is an example based on Jerry from the scenario previously.

Jerry is booking a restaurant for himself and 2 friends. He:

<table>
<thead>
<tr>
<th>Step</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Opens the Restaurant finder app.</td>
</tr>
<tr>
<td>2.</td>
<td>Selects the search options from the app.</td>
</tr>
<tr>
<td>3.</td>
<td>Taps the Vegan and Vegetarian options.</td>
</tr>
<tr>
<td>4.</td>
<td>Taps the local (less than 5 miles) option.</td>
</tr>
<tr>
<td>5.</td>
<td>Taps the 3 stars or more option.</td>
</tr>
<tr>
<td>6.</td>
<td>Selects the Search button.</td>
</tr>
<tr>
<td>7.</td>
<td>Search results appear.</td>
</tr>
<tr>
<td>8.</td>
<td>Taps Stars and the results are sorted with the best rating at the top.</td>
</tr>
<tr>
<td>9.</td>
<td>Selects the restaurant at the top of the list.</td>
</tr>
<tr>
<td>10.</td>
<td>When the restaurant is displayed there are no tables available.</td>
</tr>
<tr>
<td>11.</td>
<td>Taps the back button to return to the search results.</td>
</tr>
<tr>
<td>12.</td>
<td>Taps the Stars again to sort the list.</td>
</tr>
<tr>
<td>13.</td>
<td>Selects the restaurant second on the list.</td>
</tr>
<tr>
<td>14.</td>
<td>The restaurant is displayed and it has a table for three available.</td>
</tr>
<tr>
<td>15.</td>
<td>Selects 8pm from the book field and taps the Book Now button.</td>
</tr>
<tr>
<td>16.</td>
<td>The booking is made.</td>
</tr>
<tr>
<td>17.</td>
<td>A confirmation text message is sent to his phone.</td>
</tr>
</tbody>
</table>

This detailed case highlights several points at which Jerry’s experience could be improved. With this case as a guide, improvements could be made to the listing of the restaurants (so that only those with availability are shown) and to the order of the list (so that it remembers how it was sorted when the user returns to it). Use cases can be informed and/or verified by the results of usability testing.

**Activity: Use case**

For each user story that you have created, develop a Use case which details what each user will do with the application.
3.4 User interface design

Learning objective

By the end of this section you will be able to:

• describe, explain and apply the development of user interfaces using prototypes (from low to high fidelity including wireframes);
• describe how an application style guide is used to ensure a consistent look and feel to an application;
• create an initial application style guide for an application.

The solutions designed as part of user-centred design, become increasingly more sophisticated as the design team develop their solution and test it on users. Early versions of the solution may be simple pencil drawings/wireframes. Towards the end of the development process the solutions will be sophisticated computer based products that users can interact with fully.

These increasingly complex solutions are developed using a number of prototyping and design methods.

Figure 3.5: Development of prototypes

- Sketch: Communicates ideas, possibilities to pursue
- Wireframe: Structure, overall flow
- Prototype: Structure, flow, and details (without final visual design)
- Design: Final branded and fully developed product

© HERTIOT-WATT UNIVERSITY
3.4.1 Wireframe

A wireframe can start out as a simple pencil sketch of a user interface. The wireframe can be developed from a low level of detail and functionality up to a high level of detail and functionality.

Figure 3.6: A simple sketch wireframe
3.4.2 Low fidelity prototype

Low-fidelity prototypes may be paper-based (and without user interactions) or computer based with little or no interaction. They range from a series of hand-drawn mock-ups to printouts to simple models of the user interface with limited interaction.

In theory, low-fidelity sketches are quick to create. Low-fidelity prototypes are helpful in enabling early development of alternative designs, which helps provoke innovation and improvement. An additional advantage to this approach is that when using rough sketches or very simple models, users may feel more comfortable suggesting changes.

Figure 3.7: A low-fidelity prototype
3.4.3 High fidelity prototype

High-fidelity prototypes are computer-based, and usually allow realistic (mouse-keyboard) user interactions. High-fidelity prototypes take you as close as possible to a true representation of the user interface.

High-fidelity prototypes are assumed to be much more effective in collecting true human performance data (e.g., time to complete a task), and in demonstrating actual products to users, clients, management, and others.

Figure 3.8: A high fidelity prototype
Activity: Development of prototypes

**Q6:** On the below development of prototypes diagram, identify the correct labels (A to H) from the following:

1. wireframe
2. Usability
3. Ideation
4. Design
5. Sketch
6. Refinement
7. Concept Validation
8. Prototype

---

**3.4.4 Application style guide**

An application style guide is a detailed definition of the visual presentation of an application. It is a consistent set of rules that must be applied to the visual elements of the application.

The application style guide is a living document which saves development time and
makes communicating the user interface of an application to programmers, project sponsors, product owners, clients and end-users straight-forward. The application style guide presents a visual and consistent representation of the elements of the application for everyone involved in the project.

Many companies make their application style guides available online. Some examples are:

- BBC (Global Experience Language): [http://www.bbc.co.uk/gel](http://www.bbc.co.uk/gel) and [mobile guidelines: http://downloads.bbc.co.uk/guidelines/mobile_guide_v1.1_compressed.pdf](http://downloads.bbc.co.uk/guidelines/mobile_guide_v1.1_compressed.pdf)
- Yelp: [http://www.yelp.co.uk/styleguide](http://www.yelp.co.uk/styleguide)

As an application is developed the application style guide will evolve but the key design elements will remain consistent throughout the application. The application style guide details the visual language of the application: the palette of colours, the size of headings, the size of buttons, text blocks, popups, the fonts to be used, etc.

The application style guide needs to be started early in the application’s development. The generation of a single document detailing the visual elements within the application ensures that each developer is aware of and can use the defined styles - without these definitions each programmer would write his/her own styles resulting in an application without a consistent look and feel which also has an excessive amount of additional code (each style requires separate coding). The application style guide allows programmers to reuse the defined resulting in a leaner application with fewer lines of code.

**Creating a simple Application Style Guide**

An application style guide will detail the foundations of the application: colour pallet, any grid layout system and the basic headers and body text. For example the style guide for Yelp initially defines some simple typography, grid layout and a colour palette.
The application style guide is best kept as an HTML/CSS document which links to the original styles used by the application - as the guide is updated so will the underlying styles used in the application.

3.5 Resolving user centred design and SCRUM

Learning objective

By the end of this section you will understand:

- how User Centred Design can be applied within the Scrum methodology.

The elements of user centred design provide key documents for a team developing a project based on Scrum principals. As the product backlog evolves and each sprint is
carried out, the principals of user-centred design will be applied.

Each sprint will execute the user-centred design cycle:

- **Understand and specify context of use**
  Developing the personas and other elements of user requirements are carried out at an early stage of the development. The product owner represents the views of these users.

- **Specify the user and organisational requirements**
  These requirements are taken from the sprint backlog (the work to be completed in the current sprint).

- **Produce design solutions**
  The prototyping and development of the application UX (User experience) and UI (User interface) during the sprint.

- **Evaluate designs against requirements**
  Carried out as the application is delivered during the sprint. The application delivered is evaluated against the **sprint backlog**.

### 3.6 Learning points

**Summary**

You should now know how:

- to design, describe and apply the key aspects of user-centred design including using analysis tools to capture user requirements;
- to describe and develop user profiles, user personas, user stories, user scenarios and use cases;
- to describe, explain and apply the development of user interfaces using prototypes (from low to high fidelity including wireframes);
- an application style guide is used to ensure a consistent look and feel to an application;
- to create an initial application style guide for an application;
- User Centred Design can be applied within the Scrum methodology.
3.7 End of topic test

End of topic test

Q7: Designers use this technique to generate an outline design of the user interface of an application/website:
   a) persona
   b) wireframe
   c) pseudocode
   d) use case

Q8: The team creating an application write the following text:
   "Sally has family around the world and shares photos and messages with them throughout her day. She needs quick access to her photos and her contacts and the ability to send messages."
   This is an example of:
   a) a use case
   b) a user scenario
   c) a user story
   d) system requirements

Q9: A paper prototype is a:
   a) version of the application drawn on paper with which users can interact.
   b) sketch of the user interface which users can comment on.
   c) pseudocode design.
   d) template of buttons, menus, and text boxes that users can discuss.

Q10: Two activities that can be used to ensure user requirements are correctly understood are:
   a) data dictionary and environment analysis.
   b) performance and satisfaction criteria.
   c) interviews and focus groups.
   d) independent test groups and data flow.

Q11: A design team consists of multi-disciplinary skills and perspectives because the:
   a) team should only be focused on coding rather than user experience design.
   b) team is directed and managed by the product owner to complete the application.
   c) team needs to draw on the views of number of areas: graphic designers, programmers, users experience experts, project managers and others to be successful.
   d) needs of the project require only user experience experts to design the application.
Topic 4

Web development

Contents

4.1 Revision ................................................. 75
4.2 Web Server model ....................................... 76
4.3 HTML form processing ................................. 77
  4.3.1 Form structure ................................... 78
  4.3.2 Form submission .................................. 86
  4.3.3 Methods .......................................... 87
  4.3.4 Form security .............................. 89
  4.3.5 Validation ................................. 90
4.4 Learning points .......................................... 94
4.5 End of topic test ....................................... 95

Prerequisite knowledge

From your studies at Higher you should already know:

- that web sites can be designed in a multi-level structure;
- that a HTML web page consists of <head> and <body> sections and that a <title> is set within the <head> section;
- how to design and build simple CSS rules and the purpose of cascading style sheets to format a document;
- how to craft meta tags to generate meta-data about a web page in the <head> section;
- that server-side and client-side scripting together with the use of online databases can be used to generate dynamic web sites;
- how to code simple client and server side scripts as part of creating a webpage.

Learning objectives

By the end of this topic you will be able to:

- describe the client-server model as it applies to web services and a number of server types used to host a web site;
• create an HTML form to submit data for processing using both the GET and POST methods. Describe the purpose of each method;

• use client-side and server-side methods to validate data and implement data security to protect against attack.
4.1 Revision

Quiz: Revision

Q1: The following CSS code sets style rules for:
.paragraph { color: red; margin-top: 20px; border: 1px #000; }

a) a HTML tag.
b) a unique ID.
c) a class.
d) a paragraph.

Q2: The head of an HTML document cannot contain?

a) Links to external stylesheets.
b) The title of the HTML document.
c) A DIV grouping element.
d) JavaScript code.

Q3: The correct metadata for keywords for a page relating to snowboard in Scotland's mountains would be:

a) <meta keywords="snowboarding, Scotland, mountains"/>

Q4: A web page URL is changed from http://riskcoding.com/index.php?page=23 to http://riskcoding.com/summer/projects/build-a-robot. This is an example of?

a) Optimization for web search.
b) URL encoding.
c) Server-side scripting.
d) Client-side scripting.

Q5: A dynamic web page is:

a) composed in interactive elements which can be dynamically expanded by the user.
b) produced when an HTML file is opened by a web browser.
c) a page which is constructed by a server script or application.
d) contains JavaScript to provide user interactivity.
4.2 Web Server model

Learning objective
By the end of this section you will be able to:

- describe the client-server model as it applies to web services and a number of server types used to host a web site;

The web services we use on the Internet are based on the **client-server model** - the **client**: your web browser or application requests data from the **server**. The server retrieves that data and sends it to the client.

The client has some power to process and display the data; it receives and it can send data with the requests it makes to the server. The server can process these requests and the data it receives.

Don't think of a server as a big computer. A server can refer to either the hardware/computer that is running server software or the server software itself. A single computer is capable of running several servers at the same time.

A typical set up for web development has a web server, one or more server-side scripting languages and a database server for data storage. The database server could be running on the same computer as the web server or could run on a computer in another location.

Figure 4.1: A web server model with MySQL database and PHP

For the following practical work you will require the following:

- a **web server** (such as Apache or Nginx) operating the **PHP** language;
- a **MySQL** database server;
- a text editor or **Integrated Development Environment (IDE)** for code editing.

Web server packages such as EasyPHP (http://www.easyphp.org), XAMPP (https://www.apachefriends.org), MAMP (https://www.mamp.info) and WampServer (http://www.wampserver.com/en/) and many other applications, provide access to a suitable server environment for the web development and testing.
4.3 HTML form processing

**Learning objective**

By the end of this section you will be able to:

- create an HTML form to submit data for processing using both the GET and POST methods. Describe the purpose of each method;
- use client-side and server-side methods to validate data and implement data security to protect against attack.

Forms in web development are typically used to submit data for processing. A web form has two parts: the **HTML** “front end” and a “back end” form processor. The front end operation occurs in the client (the browser) and the back end processing occurs on the server side.

*Figure 4.2: HTML form processing*

The interactions with a web form are typically:

1. A user visits a web page that contains a form.
2. The web browser displays the HTML form.
3. The user completes the form and submits it.
4. The browser sends the submitted form data to the web server.
5. A form processor script running on the web server processes the form data.
6. A response page is sent back to the browser.
In the following tutorial you will create a simple HTML form, process the form data, explore issues of form security and how hackers exploit poorly designed web forms, introduce client side and server side validation of the form using PHP.

### 4.3.1 Form structure

Forms are created using the `<form>` HTML element. This element is used to open a collection of form associated elements (the things that form is made up of) and the close tag `</form>` is used to end the collection.

**Activity: Signup form - Stage 1**

Create the following HTML code and save in a file called `signup_form.php` in the root folder of your testing web server e.g. htdocs or httpdocs.

```html
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Sign-up form</title>
  <link rel="stylesheet" media="screen" href="styles.css">
</head>
<body>
  <!-- start form for signup -->
  <form class="signup_form" action="" method="" name="signup_form">
    <!-- collection of form associated elements goes here -->
  </form>
</body>
</html>
```

This simple HTML document contains a `<meta>` element to force it to use Unicode encoding, a title for the document and a link to cascading style sheet that does not yet exist (we will create the HTML elements first then add the CSS and JavaScript later). The `<form>` element has a number of attributes:

- class (the CSS class applied);
- action (which will be the URL of the backend processing script);
- method (which will be how the data will be transmitted to the processing script);
- and a unique name to identify the form - "signup_form".

**Form Input elements**

Forms may use a number of input elements. These are can be text boxes, check boxes, radio buttons, password fields, buttons and many more.
A single line text box can be used to collect name, email and other details that normally occupy a single line.

The `<input>` element is used to create the text box using the text type.

```html
<input type="text" name="firstname" />
```

As is the convention, because the `<input>` element doesn't have a closing tag, we use `/` to close it.

Common attributes used with the `<input>` element:

- `type="text"` specifies that the browser should display a single line text input box;
- `name="firstname"` means that when the form is submitted the contents of this input will be referred to as firstname;
- `value="Please enter"` Value specifies a value to place in the text box when the form is created;
- `maxlength="60"` gives the text box a maximum number of characters that it can hold;
- `size="40"` the size of the text box as it appears in the web page;
- `placeholder="Jenny Smith"` The placeholder attribute specifies a short hint that describes the expected value of the input and displays this hint in the input area before the user enters a value.

A complete list can be found at w3schools.com - [http://www.w3schools.com/tags/tag_input.asp](http://www.w3schools.com/tags/tag_input.asp)

**Activity: Signup form - Stage 2**

Add the following code to the signup_form.php under the comment line `<!- collection of form associated elements goes here ->`

```
<ul>
  <li>
    <h2>Sign Up</h2>
    <span class="required_notification">* Denotes Required Field</span>
  </li>
  <li>
    <label for="username">Username:</label>
    <input id="username" type="text" name="username" value="" size="40" maxlength="40" placeholder="jensmith72" />
  </li>
  <li>
    <label for="realname">Real Name:</label>
    <input id="realname" type="text" name="realname" value="" size="40" maxlength="40" placeholder="Jenny Smith" />
  </li>
</ul>
```

© Heriot-Watt University
This code produces a form which looks like this:

- **Sign Up**

* Denotes Required Field

- Username:
- Real Name:
- Email:

Hints can be added to the form to assist users when completing it. e.g.

```
<li>
<label for="email">Email:</label>
<input type="email" name="email" value="" size="40" maxlength="60" placeholder="jenny.smith@example.com"/>
</li>

<!-- end of user ID sign up fields -->
```

**Activity: Signup form - Stage 3**

Add suitable hints for the three single line text inputs in the form using the code `<span class="form_hint">Use the format "name@domain.com"</span>` above.

Change the contents of the hint to be appropriate to each input.

- A username should contain only alphabetic characters and numbers.
- A real name should be at least a firstname and lastname.
- An email address must be in a valid format.

To complete the form, we will add a small text area for multi-line text input, a check box to agree to some terms and conditions and a submit button.

<table>
<thead>
<tr>
<th>Text area</th>
<th>A multi line text area can be used to collect an extended about of text that would occupy multiple lines.</th>
</tr>
</thead>
</table>

The `<textarea>` element is used to create the text area.
<textarea name="message" cols="40" rows="6">

Common attributes used with the **textarea** element:

- **name="message"** means that when the form is submitted the contents of this textarea will be referred to as message;
- **cols="40"** Sets how wide the textarea will be in terms of number of characters;
- **rows="6"** Sets the number of visible lines of text in the text area.

A complete list can be found at w3schools.com - [http://www.w3schools.com/tags/tag_textarea.asp](http://www.w3schools.com/tags/tag_textarea.asp)

<table>
<thead>
<tr>
<th>Button</th>
<th>A clickable button used to perform an action of some sort.</th>
</tr>
</thead>
</table>

The `<button>` element is used to create a button. Unlike the button which can be created with the `<input>` element, the `<button>` element can hold content, like text or images.

```
<button class="submit" type="submit">Sign Up</button>
```

Common attributes used with the **button** element:

- **type="<value>"** details the type of button. A value of button creates a "button" without a form action which can be used with JavaScript and an OnClick function. A value of "reset" creates a reset button that clears the values entered in the form. A value of "submit" creates a button that sends the forms contents for processing.

A complete list can be found at w3schools.com - [http://www.w3schools.com/tags/tag_button.asp](http://www.w3schools.com/tags/tag_button.asp)

**Activity: Signup form - Stage 4**

Amend the signup_form.php file to include the following lines of code after the HTML comment `<!- end of user ID sign up fields ->` and before the closing tag.

```html
<li>
  <label for="message">Message:</label>
  <textarea id="message" name="message" cols="40" rows="6"></textarea>
  <span class="form_hint">A brief message, why you want to sign up.</span>
</li>
<li>
  <label for="terms_and_conditions">Agree to terms and conditions</label>
  <input id=" terms_and_conditions" type="checkbox" name="terms_and_conditions">
</li>
<li>
  <button class="submit" type="submit">Submit Form</button>
</li>

<!-- end of additional fields -->
```

.................................
Preview the form in your web browser using the server url e.g. http://localhost/signup_form.php.

The form will now appear as:

- **Sign Up**

  * Denotes Required Field

  - **Username:** jensmith72
  - **Real Name:** Jenny Smith
  - **Email:** jenny.smith@example.com

  - **Message:** A brief message, why you want to sign up.
  - **Agree to terms and conditions**
  - [Submit Form]

**Applying CSS rules to the form**

Create a stylesheet file called styles.css and save this in the same location as your signup_form.php file. Enter the following rules into the file.

```css
@charset "UTF-8";
/* CSS Document */

/* set default fonts and styles for type */
body {
}

.signup_form h2, .signup_form label {
    font-family:Georgia, Times, "Times New Roman", serif;
}

.required_notification {
    font-size: 11px;
}

.form_hint {
    font-size: 11px; vertical-align:top;
}

/* remove the focus style which looks odd */
*:focus {
    outline: none;
}

/*make the form fields more attractive by redefining the list style*/
```
.signup_form ul {
    width:950px;
    list-style-type:none;
    list-style-position:outside;
    margin:0px;
    padding:0px;
}

.signup_form li{
    padding:12px;
    border-bottom:1px solid #eee;
    position:relative;
}

/*add some visual style to the top and bottom of the form*/
    border-bottom:1px solid #777;
}

/*add a better header style and put the "required" field on the right*/
.signup_form h2 {
    margin:0;
    display: inline;
}

.required_notification {
    color:#d45252;
    margin:5px 0 0 0;
    display:inline;
    float:right;
}

/*space out the form input elements to make them more attractive*/
.signup_form label {
    width:200px;
    margin-top: 3px;
    display:inline-block;
    float:left;
    padding:3px;
}

.signup_form input {
    height:20px;
    width:220px;
    padding:5px 8px;
}

.signup_form textarea {
    padding:8px; width:300px;
}

.signup_form button {
margin-left:156px;
}

/* add some enhanced visual styles */
.signup_form input, .signup_form textarea {
  border:1px solid #aaa;
  box-shadow: 0px 0px 3px #ccc, 0 10px 15px #eee inset;
  border-radius:2px;
}

.signup_form input:-focus, .signup_form textarea:focus {
  background: #fff;
  border:1px solid #555;
  box-shadow: 0 0 3px #aaa;
}

/* Button Style */
button.submit {
  background-color: #68b12f;
  background: -webkit-gradient(linear, left top, left bottom, from(#68b12f), to(#50911e));
  background: -webkit-linear-gradient(top, #68b12f, #50911e);
  background: -moz-linear-gradient(top, #68b12f, #50911e);
  background: -ms-linear-gradient(top, #68b12f, #50911e);
  background: -o-linear-gradient(top, #68b12f, #50911e);
  background: linear-gradient(top, #68b12f, #50911e);
  border: 1px solid #50911e;
  border-bottom: 1px solid #5b992b;
  border-radius: 3px;
  -webkit-border-radius: 3px;
  -moz-border-radius: 3px;
  -ms-border-radius: 3px;
  -o-border-radius: 3px;
  box-shadow: inset 0 1px 0 0 #9fd574;
  -webkit-box-shadow: 0 1px 0 0 #9fd574 inset;
  -moz-box-shadow: 0 1px 0 0 #9fd574 inset;
  -ms-box-shadow: 0 1px 0 0 #9fd574 inset;
  -o-box-shadow: 0 1px 0 0 #9fd574 inset;
  color: white;
  font-weight: bold;
  padding: 6px 20px;
  text-align: center;
  text-shadow: 0 -1px 0 #396715;
}

button.submit:hover {
  opacity:.85;
  cursor: pointer;
}

button.submit:active {
  border: 1px solid #20911e;
  box-shadow: 0 0 10px 5px #356b0b inset;

© HERIOT-WATT UNIVERSITY
This CSS code significantly enhances the visual appearance and behavior of the form. Finally, before we deal with the submission and processing of the form, we can use the HTML5 required attribute to tell the browser that an input/textarea element must have a value before the form can be submitted e.g.

```html
<input type="text" name="username" value="" size="40" maxlength="40" placeholder="jensmith72" required />
```

**Activity: Signup form - Stage 5**

Add the **required** attribute to the inputs for username, realname and email. Leave the message textarea as an optional item.

```
.........................
```

Some additional styling can be added to these fields. These rules will add a red asterisk to the background of each required field.

**Activity: Signup form - Stage 6**

Add the following rules to your *styles.css* file.

```
/*add red asterisk for the required elements*/
.signup_form input, .signup_form text {padding-right:30px;
}

input:required, textarea:required {
  background: #fff url(images/red-asterisk.png) no-repeat 98% top;
}
```
Download this image of a red asterisk and place it in a subfolder called "images" inside your web folder. Name the downloaded file "red-asterisk.png" to match the CSS rule.

![red-asterisk.png]

Please go online to download this file.

The page should now look something like this:

![Signup form](image)

### 4.3.2 Form submission

Forms are submitted when a submit event occurs, either triggered via JavaScript or when a Submit button is activated. A form is submitted to a specific script on the server. This script is identified in the action attribute of the form element. The action specifies the URL of the processing script.

**Activity: Signup form - Stage 7**

Amend the `action` attribute of the `<form>` element so that it points to a script called `process_signup.php` in the same location as the `signup_form.php` file.

```
action="process_signup.php"
```
4.3.3 Methods

The `method` attribute of the `form` element specifies how the form-data - the data entered in the form collection and sent to the processing script specified by the `action` attribute - is sent. The form-data is sent either as a URL containing the values (with the method="get") or as an HTTP post transaction (with the method="post").

The GET method

The GET method adds the list of values from the form to the URL of the processing script. The GET method sends the data after the URL of the processing script (if it was processing only two fields) would be:

```
```

The values from the form are visible in the URL so are insecure. The string on values starts with ? (question mark) and each name/value pair is joined using a & (ampersand). Notice the %20 in the string where the space would have been. Non-text characters are converted to unicode Hexadecimal values. You can review a list of Unicode UTF-8 values here (http://www.utf8-chartable.de/)

Key points about the GET method:

- appends form-data into the URL in name/value pairs (delimited by &);
- the length of a URL is limited (to roughly 3000 characters);
- not suitable for sensitive information because the data is visible in the URL;
- useful for instances where the data will be part of a bookmark/link to a page;
- useful when debugging because the values passed to the processing script are visible.

The POST method

The POST method adds the form data inside the body of the HTTP request to the processing script so the data is not visible in the URL. This is generally more secure than the GET method.

Key points about the POST method:

- adds the form data to the HTTP request to the processing script;
- has no size limitations on the amount of data that can be submitted;
- form submissions with POST cannot be bookmarked.

Activity: Signup form - Stage 8

Amend the `method` attribute of the `<form>` element so that it uses the `get` method.

```
method="get"
```

Now that HTML for the form is complete let's add a short PHP script to process it on the server site.
Activity: Signup form - Stage 9

Go online

Create a file called `process_signup.php` in the same location as the `signup_form.php`. Enter the following PHP code. You will have some experience of server-side development from Higher Computing Science.

```php
<?php
/* script to process details from the submitted form */

// read in the values from the form
$username = $_GET['username'];
$realname = $_GET['realname'];
$email = $_GET['email'];
$message = $_GET['message'];
$terms_and_conditions = $_GET['terms_and_conditions'];
?>
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title><?php echo $message; ?></title>
<link rel="stylesheet" media="screen" href="styles.css">
</head>
<body>
<!-start form response -->
<ul>
<li>
 Username: <?php echo $username; ?>
</li>
<li>
 Realname: <?php echo $realname; ?>
</li>
<li>
 Email: <?php echo $email; ?>
</li>
<li>
 Message: <?php echo $message; ?>
</li>
<li>
<?php
    if($terms_and_conditions == "on") {
        echo "User has agreed to the terms and conditions";
    } else {
        echo "User has not agreed to the terms and conditions";
    }
?>
</li>
</ul>
</body>
</html>
```
At the moment this form just displays the values sent from the form. Test the script by loading the form, entering data and submitting it. The output should be as follows:

- **Username:** ragdoll619
- **Realname:** Jenna Sauders
- **Email:** jensos18@hotmail.com
- **Message:** I wanted to join the gaming club.
- **User has agreed to the terms and conditions**

This a simple script and it isn’t very secure. On some web browsers, it can be exploited to either inject code to carry out actions on a user’s web page OR to display HTML that was never intended to be displayed. You are going to create some harmless code to “attack” the server.

### 4.3.4 Form security

The processing script receives values from the form. Using a number of coding "tricks" it is possible to enter data that could be used to exploit the server.

**Code injection attacks**

A **code injection** attack is when a weakness in poorly written code is used by an attacker to inject code into a vulnerable script and change the execution of the script.

Modern web browsers are more sophisticated at preventing these kind of attacks. The browser checks the source code of the page with the code to be executed to ensure all the code to be executed is valid.

Try the following harmless example of Code Injection.

**Activity: Signup form - Code injection**

Load the form again and enter the following values.

- **Username:** ragdoll619
- **Real Name:** Jenna Sauders
- **Email:** jensos18@hotmail.com
- **Message:** example

And submit the form. The fields are not checked for HTML so this code allows an image to be injected into the web page. You can try other HTML including links/anchors in the message textarea.

**Cross-site Scripting**

Cross-site scripting (**XSS**) is a type of attack carried out on web applications. It allows hackers to inject client-side script into a web page that others can view. Cross-site scripting uses gaps in the security of web applications to allow malicious content to
be delivered from a compromised site. When the user visits the compromised page information can be harvest by the attacker.

Figure 4.3: Cross Site Scripting

1. An attacker finds an XSS hole in a Web application.
2. The attacker creates an attack URL for stealing sensitive information and disguises it so that it appears legitimate.
3. The attacker distributes the malicious XSS link via social engineering to unsuspecting users.
4. When the victim logs in, Javascript, which is embedded with the malicious XSS link, executes and transmits the victim’s login information to the attacker.

4.3.5 Validation

Validation is the key to protecting against code injection and cross-site scripting attacks. By validating the form data, we can protect against these exploits.

Client side validation techniques

Any validation carried out on the client site, using either JavaScript or another client technology, can be subverted. There are tools, such as interception proxy servers, that intercept the data between the client and the server and allow the hacker to change the requests before they can be sent effectively rendering all client side validation for the purpose of security useless.

However, for the typical user, carrying out some kind of client side validation is useful as it can catch errors before data is sent to the server.

Activity: Signup form - Validation

HTML5 carries out validation based on the type attributes of form collection elements. There is no specific markup required in order to activate form validation - it is on by default.

The first field, username, is text and is required. The web browser will validate the field so that it must not be empty and contains characters. As long as the user entered at least one character the field it will validate.
We can use the :valid and :invalid CSS selectors to format valid and invalid fields providing visual information to the user.

Download each of these files and place them in your images folder.

- valid.png
- invalid.png

Please go online to download these files.

Add the following CSS rules to your styles.css file.

```css
.signup_form input:focus:invalid, .signup_form textarea:focus:invalid {
/* when a field is considered invalid by the browser */
  background: #fff url(images/invalid.png) no-repeat 98% center;
  box-shadow: 0 0 5px #d45252;
  border-color: #b03535
}

.signup_form input:required:valid, .signup_form textarea:required:valid {
/* when a field is considered valid by the browser */
  background: #fff url(images/valid.png) no-repeat 98% center;
  box-shadow: 0 0 5px #5cd053;
  border-color: #28921f;
}
```

Load the form and fill out the values to see how the use of the :valid and :invalid selectors affects the form.

Further validation can be performed using the HTML5 pattern attribute. The username must be a combination of only letters and numbers; no special characters or spaces are allowed.

**Activity: Signup form - Validation 2**

Add the pattern attribute and the regular expression `^[A-Za-z0-9]+$` to the input element for username. A regular expression is a method of declaring rules to match string contents.

```html
<input id="username" type="text" name="username" value="" size="40"
      maxlength="40" placeholder="jensmith72" pattern="^[A-Za-z0-9]+$" required/>
```

Test the code and ensure that the username is only valid when it consists of letters and/or numbers.
Activity: Signup form - CSS formatting

To complete the formatting of the form add the following CSS rules to your styles.css file.

```
.form_hint {
    background: #d45252;
    border-radius: 3px 3px 3px 3px;
    color: white;
    margin-left: 8px;
    padding: 1px 6px;
    z-index: 999; /* hints stay above all other elements */
    position: absolute; /* allows proper formatting if hint is two lines*/
    display: none;
}

.form_hint::before {
    content: "\2500"; /* left point triangle in escaped unicode */
    color:#d45252;
    position: absolute;
    top:1px;
    left:-6px;
}

.signup_form input:focus + .form_hint {display: inline;}

/* change form hint color when valid */
.signup_form input:required:valid + .form_hint {background: #28921f;}

/* change form hint arrow color when valid */
.signup_form input:required:valid + .form_hint::before {color:#28921f;}
```

This CSS formats the hints so they are only shown when editing a field.

-----------------------------------------------

HTML5 can be used to carry out validation on the client side however every input must be validated on the server side and made suitable for use.

Server Side validation techniques

Variables passed to a form using the GET or POST methods must never be used without first being sanitized and validated (if required). Data normally goes through two processes on the server side before it can be reliably used.

- Sanitizing data = Remove any illegal characters from the data.
- Validating data = Determine if the data is in proper form.

Sanitization of form data

Sanitization is the process of removing any HTML entities from values passed to the
processing script. In PHP this is relatively easy to do using filters. You can read more about PHP filters at http://www.w3schools.com/php/php_ref_filter.asp

**Activity: Signup form - Sanitization**

Change the code in the `process_signup.php` form so that it correctly sanitizes the fields. Replace lines 4 to 9 with the following code.

```
//read in the values from the form
$username = filter_var($_GET['username'], FILTER_SANITIZE_STRING);
$realname = filter_var($_GET['realname'], FILTER_SANITIZE_STRING);
$email = filter_var($_GET['email'], FILTER_SANITIZE_STRING);
$message = filter_var($_GET['message'], FILTER_SANITIZE_STRING);
$terms_and_conditions = filter_var($_GET['terms_and_conditions'], FILTER_SANITIZE_STRING);

//end of reading in values and sanitizing them
```

This code removes all HTML, JavaScript and other code from the text.

**Validation**

The username and email address fields need to be validated. This can be done using a PHP function. If either of these fields is invalid, then we will force the user to return to the original form.

**Activity: Signup form - Validation 3**

Add the code to validate the email address. At the top of the `process_signup.php` file, on a new line after the `<?php` enter the following PHP function.

```
//function to validate username and email address
function validate_form($username, $email)
{
    if(!ctype_alnum($username) || !filter_var($email, FILTER_VALIDATE_EMAIL))
    {
        //one of these is invalid so return true;
        return false;
    }
    return true;
}
```

This function uses the PHP type `ctype_alnum` to validate the username because this requires that $username be alphanumeric only and the PHP filter, `FILTER_VALIDATE_EMAIL`, is used for the email address. Should either of these be invalid then a value of false is returned from the function. If they are valid then true is returned.
Add the following code under \end of reading in values and sanitizing them

```php
//check if $username or $email are invalid,
//if they are redirect back to the form.
if (!validate_form($username, $email)) {
    //there is an error so go back to where we came from
    Header('Location: signup_form2.php');
}
```

Now test the form again. Change the values in the URL to introduce errors for the script `process_signup.php` to manage. Remember to test to ensure your form values are sanitized to prevent code injection.

**Activity: Signup form - POST**

The POST method can also be used to pass values. Use the POST method and the PHP function `$_POST` to create an alternative version of the form and the processing script which makes use of the POST method.

---

**4.4 Learning points**

**Summary**

You should now know how:

- to describe the client-server model as it applies to web services and a number of server types used to host a web site;
- to create an HTML form to submit data for processing using both the GET and POST methods and can describe the purpose of each method;
- client-side and server-side methods can be used to validate data and implement data security to protect against attack.
4.5 End of topic test

End of topic test

Q6: A typical web server model consists of:
   a) a web server and a web browser making requests.
   b) a database server responding to web server requests.
   c) storage and a PHP interpreter.
   d) a web browser and MySQL database.

Q7: A web form has two parts:
   a) form validation and sanitization.
   b) objects and operations.
   c) an HTML front end and a server side processor.
   d) a form element and input elements.

Q8: An action attribute of a form contains the:
   a) address of a validation JavaScript file.
   b) address of the processor script.
   c) CSS rules for the form.
   d) input elements for the form.

Q9: The HTML5 attributes that can be used to set a mandatory field and a specific format of characters are:
   a) require and pattern.
   b) class and id.
   c) placeholder and label.
   d) presence and validation.

Q10: XSS stands for:
   a) Extensible Software Standards
   b) Cross System Software
   c) Cross Site Scripting
   d) Exclusive System Settings

Q11: PHP uses _______ to sanitise data.
   a) Regex rules
   b) HTML entities function
   c) filters
   d) escaped strings
Topic 5

Database systems and SQL

Contents

5.1 Revision ................................................................. 99
5.2 Database Server model .................................................. 100
  5.2.1 *AMP Stack ...................................................... 101
  5.2.2 Other technologies (Oracle, SQL Server, NoSQL) ............. 102
5.3 Database management tools ............................................. 103
  5.3.1 phpMyAdmin ..................................................... 103
  5.3.2 MySQL Workbench ............................................... 103
5.4 Structured Query Language (SQL) ................................... 103
  5.4.1 Statements and Syntax ......................................... 105
  5.4.2 DDL - Data Definition Language (create) ..................... 106
  5.4.3 DML - Data Manipulation Language ........................... 110
  5.4.4 Aggregate functions (count, sum, max, min, avg) ............ 117
  5.4.5 Equi-Joins between tables .................................... 122
5.5 Learning points ........................................................ 123
5.6 End of topic test ....................................................... 124

Prerequisite knowledge
From your studies at Higher you should already:

- understand the concept of relational databases including the use of primary keys/compound keys;
- understand relationships (one-to-one, one-to-many, many-to-many) and complex database operations (including queries, forms, reports and calculations);
- know the concept of database-driven websites and interactive web pages;
- know about cloud and server options to provide web hosting;
- have encountered server side validation of online form data.

Learning objectives
By the end of this topic you will be able to:
• create and execute SQL statements to create database tables (including the use of fields, keys and field types);

• use, create and execute SQL statements to insert, update and delete data from tables;

• create and execute SQL statements making use of FROM, WHERE, ORDER BY and GROUP BY clauses, aggregate functions (COUNT, SUM, MIN, MAX, AVG) and equi-joins between tables.

• describe the organisation and operation of a typical database server model;

• exemplify the use of tools to manage data within a database server.
5.1 Revision

Quiz: Revision

Q1: Data dictionaries should include details of the:
   a) definitions of the data held within each entity.
   b) name, type, size, validation and index/key status of each attribute of each entity.
   c) relationships between each entity and the operations that apply to those entities.
   d) processes which are applied to the entities within the database system.

Q2: Queries are database operations that extract data from base tables to produce answer tables. Which of the following can an answer table NOT be?
   a) A base table for a query.
   b) A table containing no rows.
   c) A table that can be used to update the underlying data.
   d) A form containing the data from an underlying table.

Q3: What is a query in a relational database system?
   a) A query is a type of data dictionary.
   b) A query is an answer table containing values which match set criteria.
   c) A query is a request for data in a prescribed format meeting set criteria.
   d) A query is a base table.

Q4: Which of the following would be a sensible size for a text field which stores a person's first name?
   a) 6
   b) 35
   c) 75
   d) 300

Q5: What is a compound key?
   a) A compound key is a foreign key which is referenced in more than one table.
   b) A compound key is two or more columns in a table which uniquely identify each row in that table.
   c) A compound key is a non-numeric primary key.
   d) A compound key is a foreign key that links to two or more tables.
5.2 Database Server model

Learning objective
By the end of this section you will be able to:

• describe the organisation and operation of a typical database server model.

A database server is much the same as any other server model. There is a store of data, which is accessed via the database server when a request is made from a client. The software on the server computer - in this case, the database server software:

• authenticates (checks that the requesting client is allowed to access the database);
• executes a query to retrieve data from the database in a specific format;
• and displays that data using the database manager CLI or presents the data to a server-side script for further processing.

Where a query contains an error the database management system software will present an error to the client or to the requesting script.
5.2.1 *AMP Stack

*AMP is a term used to design a combination of operating system, Apache Web Server, MySQL database server and PHP. The popularity of *AMP across the World Wide Web is driven by the low-cost of deployment; the components of *AMP (other than the operating system) are Open-Source applications, free software that can be used without purchasing a license.

Typical implementations of this are WAMP (Windows based), LAMP (Linux based) and MAMP (Apple Mac based). As of 2015, approximately 54% of the World Wide Web operated on Apache based servers (with Nginx, Microsoft-IIS and Google Servers making up for the rest).

High traffic (sites with lots of visitors) such as apple.com, paypal.com, adobe.com, twitter.com and many others make use of *AMP (and particularly LAMP) for the operation of their web sites.

MySQL is the key database component of *AMP and dominates online databases used.
to provide services. There are over 10 million active installations of MySQL and it is the database technology that drives popular web tools such as Wordpress, Tumblr and Twitter.

**Activity: *AMP example***

The following diagram is an example of a three tiered application server system dealing with a request from a client. Please go online to see the step-by-step animation.

---

### 5.2.2 Other technologies (Oracle, SQL Server, NoSQL)

There are many other database server technologies that are used. **Oracle Database** and **Microsoft SQL Server** are two popular closed-source, large scale database technologies which are extremely robust and well supported by both companies.

NoSQL is term used to describe a range of emerging database technologies which don't follow a traditional relational database approach. NoSQL databases evolved from the need to query sometimes millions of records in a fraction of a second, often with elements of the database running on separate servers. Facebook developed a NoSQL database management system called Cassandra, to cope with the need to process large amounts of data, across multiple servers, in multiple locations. Cassandra is now an open source project under the care of the Apache Foundation.
5.3 Database management tools

Learning objective
By the end of this section you will be able to:

• exemplify the use of tools to manage data within a database server.

The following section requires access to the *AMP technology stack. Recommended technologies for this are:

• Development Server from EasyPHP (http://www.easyphp.org/) on Windows.
• MAMP for Apple OS X (https://www.mamp.info/en/).
• XAMPP for Linux (https://www.apachefriends.org/index.html).

Each of these *AMP implementations includes phpMyAdmin.

5.3.1 phpMyAdmin
PHPMyAdmin is an open source tool written in PHP for the administration of MySQL databases via a web browser. PHPMyAdmin provides a web-based interface to MySQL to allow databases, tables and queries to be created and managed.

5.3.2 MySQL Workbench
MySQL Workbench is tool which runs on Windows, Apple OS X and Linux. It is used to manage MySQL databases, start/stop MySQL Server, set permissions on databases and to construct and execute SQL commands.

Both PHPMyAdmin or MySQL Workbench can be used to complete the following activities using MySQL.

5.4 Structured Query Language (SQL)

Learning objective
By the end of this section you will be able to:

• create and execute SQL statements to create database tables (including the use of fields, keys and field types);
• use, create and execute SQL statements to insert, update and delete data from tables;
• create and execute SQL statements making use of FROM, WHERE, ORDER BY and GROUP BY clauses, aggregate functions (COUNT, SUM, MIN, MAX, AVG) and equi-joins between tables.

Structured Query Language is a text based language for administering and manipulating
databases. There are many forms/version of SQL but for these exercises we shall be using the MySQL version.

Visual database programs, such as Microsoft Access and FileMaker Pro, place a graphic front-end on the underlying SQL commands to manage the database. Both of these packages allow a user to enter text-based SQL statements should he/she wish.

SQL has a defined standard; the ANSI (American National Standards Institute) standard requires that all versions of SQL support core commands in the same way.

**Activity: SQL database**

You are going to use SQL to create a small database to hold details about players in an eSports league. The database consists of the following tables and fields (asterisks (*) are shown to indicate foreign keys, fields which are underlined indicate primary keys):

player (username, realname, password, email, message, terms_and_conditions)

gamescore (score_id, username*, game*, score)

game (game, platform).

Some sample data from these tables is shown below:

<table>
<thead>
<tr>
<th>username</th>
<th>realname</th>
<th>password</th>
<th>email</th>
<th>message</th>
<th>terms_and_conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>shocker</td>
<td>Paul White</td>
<td>pink10red</td>
<td><a href="mailto:paul@gamers.org">paul@gamers.org</a></td>
<td>I want to play competitively</td>
<td>on</td>
</tr>
<tr>
<td>peach</td>
<td>Sally MacDonald</td>
<td>trustme1</td>
<td><a href="mailto:sally@scott.com">sally@scott.com</a></td>
<td>I’m part of an eSports team</td>
<td>on</td>
</tr>
<tr>
<td>destroyer</td>
<td>Chloe Davidson</td>
<td>shadow99</td>
<td><a href="mailto:chloe@coders.org">chloe@coders.org</a></td>
<td>I love games</td>
<td>on</td>
</tr>
</tbody>
</table>
gamescore

<table>
<thead>
<tr>
<th>score_id</th>
<th>username</th>
<th>game</th>
<th>score</th>
</tr>
</thead>
<tbody>
<tr>
<td>625</td>
<td>shocker</td>
<td>Massive RPG</td>
<td>726122</td>
</tr>
<tr>
<td>626</td>
<td>peach</td>
<td>SuperJoe</td>
<td>102928</td>
</tr>
<tr>
<td>627</td>
<td>peach</td>
<td>Massive RPG</td>
<td>625100</td>
</tr>
<tr>
<td>628</td>
<td>shocker</td>
<td>Terra 1999</td>
<td>821200</td>
</tr>
<tr>
<td>629</td>
<td>destroyer</td>
<td>Terra 1999</td>
<td>120001</td>
</tr>
<tr>
<td>630</td>
<td>peach</td>
<td>SuperJoe</td>
<td>283102</td>
</tr>
<tr>
<td>631</td>
<td>destroyer</td>
<td>SuperJoe</td>
<td>299000</td>
</tr>
</tbody>
</table>

game

<table>
<thead>
<tr>
<th>game</th>
<th>platform</th>
</tr>
</thead>
<tbody>
<tr>
<td>Massive RPG</td>
<td>X-station</td>
</tr>
<tr>
<td>SuperJoe</td>
<td>S-box</td>
</tr>
<tr>
<td>Terra 1999</td>
<td>PC</td>
</tr>
</tbody>
</table>

Using the information above, construct an entity-relationship diagram of the eSports database.

....................

5.4.1 Statements and Syntax

A single line of an SQL command is called a statement. These statements will consist of one or more clauses. Each clause contains a specific SQL keyword and some data that it acts upon. The following is an example of the SELECT command which locates and displays data according to the details entered.

```
SELECT username, realname, email, message, terms_and_conditions
FROM player
WHERE terms_and_conditions = "on"
ORDER BY username DESC;
```

In this example, each clause has been placed on a separate line and the SQL keywords have been capitalised. Notice that the semi-colon (;) is used to terminate the statement. There is no requirement to place a statement across multiple lines or to capitalise commands but for clarity in the following activities this is the approach that will be used.
Table Aliases

A table alias can be used as well. This is a shortened version of the table name which can be used to refer to columns. This is particularly useful if the SELECT statement contains references to two columns with the same name in different tables. For example:

```sql
SELECT developer.developerseq, game.developerseq
FROM developer, game
WHERE developer.developerseq = game.developerseq;
```

This is a long SQL statement but it can be shortened by using table aliases. To create an alias an alternate name for the table is entered after the table name in the FROM clause. So the above SQL statement could be shortened to:

```sql
SELECT d.developerseq, g.developerseq
FROM developer d, game g
WHERE d.developerseq = g.developerseq;
```

The MySQL keyword AS can be used in the clause for clarity, but it is not a requirement i.e.

```sql
SELECT d.developerseq, g.developerseq
FROM developer AS d, game AS g
WHERE d.developerseq = g.developerseq;
```

5.4.2 DDL - Data Definition Language (create)

To create a database we use the CREATE DATABASE command with the name of the table to be created. As always this is terminated with a semi-colon e.g.

```
CREATE DATABASE esports;
```

To check that a database has been created following this command we can use the SHOW command e.g.

```
SHOW DATABASES;
```

This should produce a list of all the databases within the current server.

Activity: Create a database

Using your MySQL management tools (PHPMyAdmin, MySQL Workbench or another tool) enter the above commands to create an esports database and confirm that it exists.

```
CREATE DATABASE esports;
SHOW DATABASES;
```

Having created the database, you can now create the tables and fields within it. You have to choose which database to use. Using MySQL this is done with the USE command.

```
USE esports;
```

Any queries executed will now only affect this database because it is the one selected.
To create a table we use the CREATE TABLE command. This creates the table and the fields as part of the table. The data types for the fields are also set at this point. The format of the CREATE TABLE command is:

```sql
CREATE TABLE table_name
(
    column_name1 data_type(size),
    column_name2 data_type(size),
    column_name3 data_type(size),
    ...
);
```

To create the tables within the esports database a data dictionary is needed so that the data types and primary/foreign keys are known.

**Activity: Create a data dictionary**

Using the sample data and entity relationship diagram from the earlier activity, create a data dictionary for the esports database.

```
..........................................
```

### 5.4.2.1 Data types in MySQL

There are many field types available when creating/maintaining a MySQL database. You can read about all the data types at [https://dev.mysql.com/doc/refman/5.0/en/data-types.html](https://dev.mysql.com/doc/refman/5.0/en/data-types.html) however an appropriate selection of these is described below.

<table>
<thead>
<tr>
<th>Type</th>
<th>Use for</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMALLINT</td>
<td>A small integer</td>
<td>The signed range is -32768 to 32767. The unsigned range is 0 to 65535</td>
</tr>
<tr>
<td>INT or INTEGER</td>
<td>A normal-size integer</td>
<td>The signed range is -2147483648 to 2147483647. The unsigned range is 0 to 4294967295</td>
</tr>
<tr>
<td>FLOAT</td>
<td>A small (single-precision) floating-point number. Cannot be unsigned</td>
<td>Ranges are -3.402823466E+38 to -1.175494351E-38, 0 and 1.175494351E-38 to 3.402823466E+38. If the number of Decimals is not set or &lt;= 24 it is a single-precision floating point number</td>
</tr>
<tr>
<td>REAL</td>
<td>A normal-size (double-precision) floating-point number. Cannot be unsigned</td>
<td>Ranges are -1.7976931348623157E+308 to -2.2250738585072014E-308, 0 and 2.2250738585072014E-308 to 1.7976931348623157E+308. If the number of Decimals is not set or 25 &lt;= Decimals &lt;= 53 stands for a double-precision floating point number</td>
</tr>
<tr>
<td>Type</td>
<td>Use for</td>
<td>Size</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>DATE</td>
<td>A date</td>
<td>The supported range is '1000-01-01' to '9999-12-31'. MySQL displays DATE values in 'YYYY-MM-DD' format.</td>
</tr>
<tr>
<td>DATETIME</td>
<td>A date and time combination</td>
<td>The supported range is '1000-01-01 00:00:00' to '9999-12-31 23:59:59'. MySQL displays DATETIME values in 'YYYY-MM-DD HH:MM:SS' format.</td>
</tr>
<tr>
<td>TIMESTAMP</td>
<td>A timestamp</td>
<td>The range is '1970-01-01 00:00:00' to sometime in the year 2037. MySQL displays TIMESTAMP values in YYYYMMDDHHMMSS, YYMMDDHHMMSS, YYYYMMDD or YYMMDD format, depending on whether M is 14 (or missing), 12, 8 or 6, but allows you to assign values to TIMESTAMP columns using either strings or numbers. A TIMESTAMP column is useful for recording the date and time of an INSERT or UPDATE operation because it is automatically set to the date and time of the most recent operation if you don’t give it a value yourself.</td>
</tr>
<tr>
<td>TIME</td>
<td>A time</td>
<td>The range is '-838:59:59' to '838:59:59'. MySQL displays TIME values in 'HH:MM:SS' format, but allows you to assign values to TIME columns using either strings or numbers.</td>
</tr>
<tr>
<td>CHAR</td>
<td>A fixed-length string that is always right-padded with spaces to the specified length when stored</td>
<td>The range of Length is 1 to 255 characters. Trailing spaces are removed when the value is retrieved. CHAR values are sorted and compared in case-insensitive fashion according to the default character set unless the BINARY keyword is given.</td>
</tr>
<tr>
<td>VARCHAR</td>
<td>A variable-length string. Note: Trailing spaces are removed when the value is stored (this differs from the ANSI SQL specification)</td>
<td>The range of Length is 1 to 255 characters. VARCHAR values are sorted and compared in case-insensitive fashion unless the BINARY keyword is given.</td>
</tr>
<tr>
<td>TEXT</td>
<td>A BLOB or TEXT column with a maximum length of 65535 (2^16 - 1) characters</td>
<td></td>
</tr>
</tbody>
</table>
Activity: Create a table

Create the esports tables using the following SQL commands.

```
USE esports;
```

This line selects the esports database for use.

```
CREATE TABLE player (
    username VARCHAR(35) NOT NULL PRIMARY KEY,
    realname VARCHAR(45) NOT NULL,
    pword VARCHAR(45) NOT NULL,
    email VARCHAR(60) NOT NULL,
    message TEXT,
    terms_and_conditions SMALLINT
) ENGINE=InnoDB;
```

The table player is created using the fields specified. The VARCHAR data type is used because this is very efficient when storing text; it only stores the number of characters used rather than storing the whole length of the field according to the declared size. The `message` text field could be quite long so the TEXT data type is used which will store a large amount of text.

MySQL doesn't really have a Boolean data type - it does have a BIT type that can be set to one bit (and used for 0 - off, 1 - on) however this is sometimes difficult to debug. Because of this we'll use the SMALLINT data type and use an integer value of 0 for false and an integer value of 1 for true. NOT NULL means that there must be value set for field when a record is created, we don't need the message field to be entered so NOT NULL isn't applied to it.

The primary key is declared using the PRIMARY KEY property. Password is a key word in MySQL so, to avoid any possible confusion, we'll call the password field pword.

MySQL supports a number of database engines; these are methods for storing the data. The InnoDB engine provides support for foreign keys so we need to tell MySQL to create the table using this engine.

```
CREATE TABLE game (
    game VARCHAR(30) NOT NULL PRIMARY KEY,
    platform VARCHAR(30) NOT NULL
) ENGINE=InnoDB;
```

The game table is created next because the primary key of game is going to be used in the gamescore table and we can't reference it until it exists!
Finally, the gamescore table is created. The score_id is an integer that will be automatically incremented each time a new row is added and it is the primary key. The username field is created using the same format as the user field in the player table, the game field uses the format from the game table.

The foreign keys are declared and the linked primary key from the appropriate other table is referred to. ON UPDATE CASCADE means that if the value of the primary key in referenced table were to change, then all related values in this table would also change (which is really useful otherwise the records would have to be updated manually!). ON DELETE CASCADE means that if a primary key row was deleted, then all related foreign key records in this table would also be deleted i.e. if we deleted a game from the game table then all the records relating to that game would also be deleted from the gamescore table.

Now that the tables have been created, it is time to enter some data.

### 5.4.3 DML - Data Manipulation Language

Data manipulation language refers to a set of commands used for selecting, inserting, deleting and updating data. Data manipulation doesn’t change the organisation of the data, the structure of the database or the permissions/privileges of those using the database. DML does all new data to be added, existing data to be removed or updated and for rows to be selected.

#### 5.4.3.1 Insert

The INSERT INTO statement is used to insert new rows into a table. The command can be used in two ways. If you use the format:

```
INSERT INTO table
VALUES (value1, value2, value3, ...);
```

Then the order of the values to be inserted has to match the default column order for
the table. In our example of game this would be:

```
INSERT INTO game
VALUES ('destroyer', 'Chloe Davidson', 'shadow99', 'chloe@coders.org',
        'I love games', '1');
```

The alternative method specifies the order of the fields and the data values then follow this order:

```
INSERT INTO table (column3, column1, column2, ...)
VALUES (value3, value1, value2, ...);
```

In the case of the game table this could be presented as:

```
INSERT INTO player (realname, pword, email, terms_and_conditions, username)
VALUES ('Charlie Love', 'xmen299', 'charlie@digitalk.org', '1', 'godsend');
```

The order of the data in this INSERT INTO statement matches the order of the fields.

**Activity: Add data to tables**

Use the following SQL statements to add three players to the players table.

```
INSERT INTO player (username, realname, pword, email, message, 
                     terms_and_conditions) 
VALUES
        ('shocker', 'Paul White', 'pink10red', 'paul@gamers.org',
         'I want to play competitively', '1'),
        ('peach', 'Sally MacDonald', 'trustme1', 'sally@scott.com',
         'I'm part of an eSports team', '1'),
        ('destroyer', 'Chloe Davidson', 'shadow99', 'chloe@coders.org',
         'I love games', '1');
```

Use the following SQL commands to add data to the remaining two tables.

```
INSERT INTO game (game, platform)
VALUES
        ('Massive RPG', 'X-station'),
        ('SuperJoe', 'S-box'),
        ('Terra 1999', 'PC');
```

```
INSERT INTO gamescore (score_id, username, game, score) VALUES
        ('627', 'peach', 'Massive RPG', '625100'),
        ('625', 'shocker', 'Massive RPG', '726122'),
        ('626', 'peach', 'SuperJoe', '102928'),
        ('628', 'shocker', 'Terra 1999', '821200'),
        ('629', 'destroyer', 'Terra 1999', '120001'),
        ('630', 'peach', 'SuperJoe', '283102'),
        ('631', 'destroyer', 'SuperJoe', '299000');
```
5.4.3.2 Update

The update statement is used to update a value or values in a row or rows. It takes the following format.

```
UPDATE table_name
SET column1 = value1, column2 = value2, ...
WHERE search_column = search_value;
```

We can use the `UPDATE` command to create a MySQL statement to change the realname and email for a specific username e.g.

```
UPDATE player
SET realname = 'Chloe Falconer', email = 'cfalconer@coders.org'
WHERE username = 'destroyer';
```

You can also use the `UPDATE` command to update change multiple rows in the same operation. E.g. to change all the passwords for users with `terms_and_conditions = 0`

```
UPDATE player
SET pword = 'reset'
WHERE terms_and_conditions = 0;
```

More on the WHERE clause

Multiple search criteria can be used in a where clause, each joined using a logical operation. Other operators can be used as well such as `>`, `>=`, `<`, `<=` etc. You can read more about operators at [https://dev.mysql.com/doc/refman/5.7/en/non-typed-operators.html](https://dev.mysql.com/doc/refman/5.7/en/non-typed-operators.html) and a selection those frequently used is shown below with some examples.
### Name | Description | Example
--- | --- | ---
AND, && | Logical AND | UPDATE users SET account_archived = 'Yes' WHERE user_login = 'Yes' AND subscription='Cancelled';
BETWEEN ... AND ... | Check whether a value is within a range of values | UPDATE user_meta SET active_status='Yes' WHERE score BETWEEN '50' AND '100';
NOT BETWEEN ... AND ... | Check whether a value is not within a range of values |  

| Operator | Description | Example |
--- | --- | ---
| $<=$ | Greater than or equal operator | UPDATE results SET grade = 'A' WHERE mark $<=$'75'; |
| $>$ | Greater than operator |  |
| $<=$ | Less than or equal operator |  |
| $<$ | Less than operator |  |
| IS NOT NULL | NOT NULL value test | UPDATE user_meta SET valid='True' WHERE dob IS NOT NULL; |
| IS NULL | NULL value test | UPDATE user_meta SET valid='False' WHERE dob IS NULL; |
| LIKE | Simple pattern matching | UPDATE user_meta SET homecity='Glasgow' WHERE postcode LIKE 'G%'; # % is wildcard, it means 'anything' |
| NOT LIKE | Negation of simple pattern matching |  |
| !=, <> | Not equal operator | UPDATE user_meta SET candidate= '1' WHERE symptoms != 'fever'; |
| ||, OR | Logical OR | UPDATE user_meta SET password_reset = 'True' WHERE password_length < 10 OR password IS NULL; |

### Activity: Update queries
Create and execute update queries to complete the following operations:

1. Change the Platform for the game "SuperJoe" to "S-Box-360".
2. Change all the scores for the user "peach" for the game "SuperJoe" to 132900.
3. Change the password (pword) for the user "destroyer" to "fluffy1928".
5.4.3.3 Delete

The DELETE statement is used to remove rows from a table. You need to be careful when using DELETE. If you use it without a WHERE clause, like this:

```
DELETE FROM table_name;
```

Then you will delete EVERYTHING in the table!!!!

Typically, the format of a DELETE statement is:

```
DELETE FROM table_name
WHERE search_column = search_value;
```

So, for example, to delete the player with the username "destroyer" from the esports database you would use the command (DO NOT DO THIS!):

```
DELETE FROM player
WHERE username = 'destroyer';
```

**Activity: Delete**

Write and execute SQL statements to carry out the following:

1. Delete the row with the score_id of 630 from the gamescore table.
2. Delete the row from gamescore where username is destroyer and the score is greater than 200,000.

.................................

5.4.3.4 Select

*Note: The following tables in this section are used as the base tables for the SQL examples which follow.*

The SELECT statement is used to select data from the database. It is used to pull data from the database to be displayed or to be processed in some way outwith the database.

The SELECT statement takes the format:

```
SELECT column_name1, column_name2, column_name3, ...
FROM table_name
```

A wildcard (*) can be used to display all fields in a table:

```
SELECT * FROM table_name
```

The SELECT statement can also be used with the WHERE clause to select specific data from a table or tables.
SELECT * FROM table_name
WHERE search_column = search_value;

In MySQL, SELECT is the most commonly used Data Manipulation Language (DML) command. In specifying a SELECT query, the user specifies a description of the desired result set (answer table) so that the database server can produce the answer table required. Commonly available keywords related to SELECT include FROM, WHERE, GROUP BY and ORDER BY.

The full syntax of the SELECT is:

```
SELECT [DISTINCT]
    select_expr, ...
    [FROM table_references
    [WHERE where_condition]
    [GROUP BY {col_name}
    [ASC | DESC], ... ]
    [ORDER BY {col_name | expr | position}
    [ASC | DESC], ...]
```

**Clauses**

A clause in SQL is a part of the SQL statement relating to a command. For example: the following SQL statement consists of three clauses:

```
SELECT *
FROM platform
ORDER BY platformseq DESC;
```

**FROM Clauses**

The FROM clause specifies which tables from the database are to be used by the SELECT statement. The most common use of the FROM clause is:

```
SELECT * FROM developer;
```

This will select all columns and rows from the developer table.

**GROUP BY Clause**

GROUP BY groups similar column values together. It is most often used with aggregate functions (such as SUM, AVG, COUNT, MAX and MIN - which we will look at later).
If we wanted to only show one row from the game for each developer then we could use the GROUP BY command to group the rows by the value in the DeveloperSeq column e.g.

```sql
SELECT DeveloperSeq FROM Game GROUP BY DeveloperSeq;
```

<table>
<thead>
<tr>
<th>DeveloperSeq</th>
<th>DeveloperSeq</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>87</td>
<td>87</td>
</tr>
<tr>
<td>54</td>
<td>54</td>
</tr>
</tbody>
</table>

This will show one record for each of the matching values. Where there is more than one matching row (i.e. where a developer has more than one game in the table) then only one row will be shown. We will use the GROUP BY command more when looking at how aggregate functions operator on the data.

Note that the query statement only shows the DeveloperSeq column. If another column were included then the rows would be likely to include different values. This would mean that the grouping would only apply where the values in BOTH fields were identical.

**ORDER BY Clause**

The ORDER BY keyword is used to sort the rows in the result set produced by a query statement. Each row is sorted according the values in the column or columns stated in the ORDER BY clause. Each column can be sorted by ascending (the default) or by descending order by using the ASC or DESC options. For example:

```
orders:

<table>
<thead>
<tr>
<th>Company</th>
<th>OrderNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Shop</td>
<td>3412</td>
</tr>
<tr>
<td>Alpha Store</td>
<td>5678</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>6798</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>2312</td>
</tr>
</tbody>
</table>
```

The above table orders can be sorted into alphabetical order using:

```sql
SELECT Company, OrderNumber FROM Orders
ORDER BY Company;
```
Which gives:

<table>
<thead>
<tr>
<th>Company</th>
<th>OrderNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha Store</td>
<td>5678</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>6798</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>2312</td>
</tr>
<tr>
<td>Software Shop</td>
<td>3412</td>
</tr>
</tbody>
</table>

To display the Companies in alphabetical order AND the OrderNumbers in numerical order:

```sql
SELECT Company, OrderNumber FROM Orders
ORDER BY Company, OrderNumber;
```

Which gives:

<table>
<thead>
<tr>
<th>Company</th>
<th>OrderNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha Store</td>
<td>5678</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>2312</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>6798</td>
</tr>
<tr>
<td>Software Shop</td>
<td>3412</td>
</tr>
</tbody>
</table>

To display the Companies in reverse alphabetical order AND the OrderNumbers in numerical order:

```sql
SELECT Company, OrderNumber FROM Orders
ORDER BY Company DESC, OrderNumber ASC;
```

Which gives:

<table>
<thead>
<tr>
<th>Company</th>
<th>OrderNumber</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Shop</td>
<td>3412</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>2312</td>
</tr>
<tr>
<td>GuildWars Den</td>
<td>6798</td>
</tr>
<tr>
<td>Alpha Store</td>
<td>5678</td>
</tr>
</tbody>
</table>

### 5.4.4 Aggregate functions (count, sum, max, min, avg)

Aggregate functions are methods of pulling together values from a column or columns. These functions are used to calculate values from the data in columns and are often used with the GROUP BY.
COUNT

COUNT is used to count the number of rows in the result set for the given SQL statement. For example, using the following two tables:

<table>
<thead>
<tr>
<th>Developer Seq</th>
<th>Name</th>
<th>Website</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bungie</td>
<td><a href="http://bungie.com">http://bungie.com</a></td>
</tr>
<tr>
<td>2</td>
<td>id Software</td>
<td><a href="http://www.idsoftware.com">http://www.idsoftware.com</a></td>
</tr>
<tr>
<td>3</td>
<td>Blizzard</td>
<td><a href="http://www.blizzard.com">http://www.blizzard.com</a></td>
</tr>
<tr>
<td>4</td>
<td>Sierra Entertainment</td>
<td><a href="http://www.sierra.com">http://www.sierra.com</a></td>
</tr>
<tr>
<td>5</td>
<td>Arena.net</td>
<td><a href="http://www.arena.net">http://www.arena.net</a></td>
</tr>
<tr>
<td>6</td>
<td>Valve Software</td>
<td><a href="http://www.valvesoftware.com">http://www.valvesoftware.com</a></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Genre Seq</th>
<th>Publisher Seq</th>
<th>Developer Seq</th>
<th>Status</th>
<th>Web Site</th>
<th>Description</th>
<th>Review</th>
<th>Game Seq</th>
</tr>
</thead>
<tbody>
<tr>
<td>Doom 3</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>R</td>
<td><a href="http://www.doom3.com">http://www.doom3.com</a></td>
<td>Science has unlocked the gates to the unknown...</td>
<td>Doom 3 is tight and straightforward action with b...</td>
<td>1</td>
</tr>
<tr>
<td>Halo 3</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>B</td>
<td><a href="http://www.bungie.net/Games/Halo3/">http://www.bungie.net/Games/Halo3/</a></td>
<td>Well known for their secrecy Bungie have rev...</td>
<td>With the second day of E3 2006 concluded, the...</td>
<td>2</td>
</tr>
<tr>
<td>World of Warcraft</td>
<td>3</td>
<td>6</td>
<td>3</td>
<td>R</td>
<td><a href="http://www.worldofwarcraft.com">http://www.worldofwarcraft.com</a></td>
<td>World of Warcraft is a massive multi-player...</td>
<td>The game is huge! I cannot quite get over th...</td>
<td>3</td>
</tr>
<tr>
<td>Half-Life 2</td>
<td>1</td>
<td>6</td>
<td>6</td>
<td>R</td>
<td><a href="http://www.half-life2.com">http://www.half-life2.com</a></td>
<td>Pistol-packing Black Mesa research scientist...</td>
<td>This game is an almighty achievement. I have...</td>
<td>4</td>
</tr>
</tbody>
</table>
SELECT COUNT(GameSeq) FROM game;

will count the total number of rows in the game table. e.g.

<table>
<thead>
<tr>
<th>COUNT(GameSeq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

If we wanted the statement to count the number of games available from each publisher we would use a GROUP BY clause, for example:

```sql
SELECT DeveloperSeq, COUNT(GameSeq)
FROM game
GROUP BY DeveloperSeq;
```

<table>
<thead>
<tr>
<th>DeveloperSeq</th>
<th>COUNT(GameSeq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

It is possible to construct more complex queries which use the COUNT function. For example:

```sql
SELECT g.DeveloperSeq, d.Name, COUNT(GameSeq)
FROM game g, developer d
WHERE g.DeveloperSeq = d.DeveloperSeq
GROUP BY g.DeveloperSeq;
```

<table>
<thead>
<tr>
<th>DeveloperSeq</th>
<th>Name</th>
<th>COUNT(GameSeq)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bungie</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>id Software</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>Blizzard</td>
<td>1</td>
</tr>
<tr>
<td>6</td>
<td>Valve Software</td>
<td>1</td>
</tr>
</tbody>
</table>
SUM

The SUM function adds together the values in the specified column. For example:

Table: orderline

<table>
<thead>
<tr>
<th>OrderLineID</th>
<th>ItemID</th>
<th>Quantity</th>
<th>OrderID</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>2</td>
<td>3</td>
<td>28776</td>
</tr>
<tr>
<td>102</td>
<td>3</td>
<td>4</td>
<td>28776</td>
</tr>
<tr>
<td>103</td>
<td>4</td>
<td>6</td>
<td>28776</td>
</tr>
<tr>
<td>104</td>
<td>3</td>
<td>4</td>
<td>28778</td>
</tr>
<tr>
<td>105</td>
<td>5</td>
<td>2</td>
<td>28778</td>
</tr>
<tr>
<td>106</td>
<td>2</td>
<td>1</td>
<td>28780</td>
</tr>
<tr>
<td>107</td>
<td>2</td>
<td>1</td>
<td>28782</td>
</tr>
</tbody>
</table>

The SQL statement:

```
SELECT SUM(Quantity) FROM orderline;
```

will calculate the total number of all products ordered for all orders.

<table>
<thead>
<tr>
<th>SUM(Quantity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>21</td>
</tr>
</tbody>
</table>

Using the GROUP BY clause, this can be changed to show the total quantity ordered for each orderID.

```
SELECT OrderID, SUM(Quantity) FROM orderline GROUP BY OrderID;
```

This produces the result set:

<table>
<thead>
<tr>
<th>OrderID</th>
<th>SUM(Quantity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28776</td>
<td>13</td>
</tr>
<tr>
<td>28778</td>
<td>6</td>
</tr>
<tr>
<td>28780</td>
<td>1</td>
</tr>
<tr>
<td>28782</td>
<td>1</td>
</tr>
</tbody>
</table>

MAX

MAX finds the maximum value present in a column depending the conditions on the SQL statement. For example, using the same data as above, to find the biggest single quantity of goods ordered the statement:

```
SELECT MAX(Quantity) FROM orderline;
```

would give the result set of:
This can be changed to show the maximum order quantities for each orderID by adding a GROUP BY clause.

\[
\text{SELECT OrderID, MAX(Quantity) FROM orderline GROUP BY OrderID;}
\]

\[
\begin{array}{|c|c|}
\hline
\text{OrderID} & \text{MAX(Quantity)} \\
\hline
28776 & 6 \\
28778 & 4 \\
28780 & 1 \\
28782 & 1 \\
\hline
\end{array}
\]

MIN

MIN finds the minimum value present in a column depending the conditions on the SQL statement. For example, using the same data as above, to find the smallest single quantity of goods ordered the statement:

\[
\text{SELECT MIN(Quantity) FROM orderline;}
\]

would give the result set of:

\[
\begin{array}{|c|}
\hline
\text{MIN(Quantity)} \\
\hline
1 \\
\hline
\end{array}
\]

This can be changed to show the minimum order quantities for each orderID by adding a GROUP BY clause.

\[
\text{SELECT OrderID, MIN(Quantity) FROM orderline GROUP BY OrderID;}
\]

\[
\begin{array}{|c|c|}
\hline
\text{OrderID} & \text{MIN(Quantity)} \\
\hline
28776 & 3 \\
28778 & 2 \\
28780 & 1 \\
28782 & 1 \\
\hline
\end{array}
\]

AVG

AVG calculates an average value from those in a column depending the conditions on the SQL statement. For example, using the same data as above, to find the average quantity of goods ordered the statement:

\[
\text{SELECT AVG(Quantity) FROM orderline;}
\]
would give the result set of:

<table>
<thead>
<tr>
<th>AVG(Quantity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.0000</td>
</tr>
</tbody>
</table>

This can be changed to show the average order quantities for each orderID by adding a GROUP BY clause.

```
SELECT OrderID, AVG(Quantity) FROM orderline GROUP BY OrderID;
```

<table>
<thead>
<tr>
<th>OrderID</th>
<th>AVG(Quantity)</th>
</tr>
</thead>
<tbody>
<tr>
<td>28776</td>
<td>4.3333</td>
</tr>
<tr>
<td>28778</td>
<td>3.0000</td>
</tr>
<tr>
<td>28780</td>
<td>2.0000</td>
</tr>
<tr>
<td>28782</td>
<td>1.0000</td>
</tr>
</tbody>
</table>

**Activity: Select statements**

Write and execute SQL Select statements to do the following:

1. To select the highest score from the gamescore table.
2. To select the lowest score from gamescore table.
3. To calculate the average score for all the rows in the gamescore table.
4. To calculate the average score for the game called Massive RPG.
5. To total the scores for the player called peach.
6. To count how many rows have Terra 1999 in them.

------------------

5.4.5 **Equi-Joins between tables**

An EQUI-JOIN is created using the WHERE clause and, normally, the primary and foreign keys of the tables to be used in the query.

Each relationship in the **Entity Relationship Diagram (ERD)** above is created by the primary key/foreign key pairing. For example, the relationship between game and gamescore is created by the primary key `game.game` and the foreign key `gamescore.game`. A query to select all the related data from both tables would be:

```
SELECT *
FROM game, gamescore
WHERE game.game = gamescore.game;
```
It is possible to combine EQUI-JOINS to link to multiple tables. For example, to bring the player details into the above SQL statement:

```
SELECT *
FROM game, gamescore, player
WHERE game.game = gamescore.game AND gamescore.username = player.username;
```

The EQUI-JOIN is contained in the WHERE clause and indicated which columns are equal for valid relationships.

So, to find the *platform* for the game with the highest score the query would be (using table aliases as well):

```
SELECT g.platform, MAX(gs.score)
FROM game AS g, gamescore AS gs
WHERE g.game = gs.game
```

**Activity: Select statements 2**

Write and execute SQL Select statements to do the following:

1. Show the username and password of the player with the lowest score.
2. Show all the scores for the game Terra 1999 with the game and the player's username and realname.
3. Show the game platform, game, score and real names of players in order of score ascending.

..........................................

5.5 Learning points

**Summary**

You should now know how to create and execute SQL statements:

- to create database tables (including the use of fields, keys and field types);
- to insert, update and delete data from tables;
- making use of FROM, WHERE, ORDER BY and GROUP BY clauses, aggregate functions (COUNT, SUM, MIN, MAX, AVG) and equi-JOINS between tables. You should also be able to describe the database server model.
5.6 End of topic test

End of topic test

This following questions makes use of the following tables:
branch (branchno, street, city, postcode)
staff (staffno, fname, lname, position, sex, dob, annualsalary, branchno*)
propertyForRent (propertyno, street, city, postcode, type, rooms, rent, owner, staffno*, branchno*)
client (clientno, fname, lname, telno, preftype, maxrent)
viewing (clientno*, propertyno*, viewdate, comments)

Q6: Which of the following produces a list of all branch offices in London or Glasgow?

a) SELECT * FROM branch WHERE city = 'London' OR city='Glasgow';

b) SELECT * WHERE branch.city = 'London' OR branch.city = 'Glasgow';

c) SELECT * FROM branch WHERE city = 'London' OR 'Glasgow';

d) SELECT * FROM staff, branch WHERE staff.branchno = branch.branchno;

Q7: Which of the following produces a list of the staff number and last name of staff who work in the branch at "13 Main Road"?

a) SELECT staffno, lname
FROM staff, branch
WHERE staff.branchno = branch.branchno
AND branch.street = '13 Main Road';

b) SELECT *
FROM staff, branch
WHERE staff.branchno = branch.branchno
AND branch.street = '13 Main Road';

c) SELECT staffno, lname
FROM staff, branch
WHERE staff.branchno = branch.branchno

b) SELECT staffno, lname
FROM staff, branch
WHERE staff.branchno = branch.branchno

..............................
Q8: Which of the following produces an answer set which shows the details for all properties being sold by the Edinburgh branch?

a) SELECT * FROM propertyForRent
   WHERE city = 'Edinburgh';

b) SELECT * FROM branch
   WHERE city = 'Edinburgh';

c) SELECT * FROM branch, propertyForRent
   WHERE branchno.city = 'Edinburgh'
   AND propertyForRent.branchno = branch.branchno

d) SELECT * FROM branch, propertyForRent
   WHERE branchno.city = 'Edinburgh'

Q9: Which of the following produces an answer set which shows the fname, lname and annualsalary of the staff member in Glasgow with the highest annualsalary?

a) SELECT fname, lname, MAX(annualsalary) FROM staff
   WHERE city = 'Glasgow'

b) SELECT fname, MAX(annualsalary) FROM staff
   WHERE staff.branchno = branch.branchno AND city = 'Glasgow'

c) SELECT staff.*, MAX(annualsalary) FROM staff, branch
   WHERE staff.branchno = branch.branchno AND city = 'Glasgow'

d) SELECT fname, lname, MAX(annualsalary) FROM staff, branch
   WHERE staff.branchno = branch.branchno AND city = 'Glasgow'

Q10: Which of the following produces an answer set which shows all the clients with a maxrent of less than 800?

a) SELECT * FROM client WHERE maxrent IS NOT 800

b) SELECT * FROM client WHERE maxrent < 800

c) SELECT maxrent FROM client WHERE maxrent < 800

d) SELECT * FROM client WHERE >maxrent 800

© HERIOT-WATT UNIVERSITY
Topic 6

Web and database integration

Contents

6.1 Revision ........................................... 129
6.2 Server side scripting requirements ....................... 130
  6.2.1 Getting started with PHP .......................... 130
  6.2.2 Syntax and PHP .................................. 131
  6.2.3 Variables and Operations ........................ 133
  6.2.4 Arrays ......................................... 134
  6.2.5 Associative Array ................................. 135
  6.2.6 Control structures ............................... 135
6.3 Connecting to database ................................ 141
6.4 Executing query ..................................... 144
6.5 Retrieving results .................................... 145
6.6 Editing/Amending data ................................. 148
  6.6.1 Insertion ...................................... 153
  6.6.2 Deletion ...................................... 156
6.7 Learning points ...................................... 160
6.8 End of topic test .................................... 161

Prerequisite knowledge

From your studies at Higher you should already know:

- the concept of a database driven web site and dynamic web pages;
- that server side scripting languages can be used for a variety of tasks including the production of web pages.

Learning objectives

By the end of this topic you should be able to use a server side scripting language (PHP) to:

- demonstrate the basic commands and syntax;
- connect to a database to process data;
- read/write data from/to a database;
• use insert, amend and deletion queries;
• process a form using server-side scripting.
6.1 Revision

Quiz: Revision

Q1: Server side script is used to:
   a) process data within the browser.
   b) execute commands using HTML.
   c) process data at the server side.
   d) securely pass values within an HTML file.

Q2: Server side validation of data is more secure because:
   a) nearly all client-side validation can be circumvented.
   b) the data is always sent to the server using an encrypted data connection.
   c) form data can only be processed by the server.
   d) cookies can be used to authenticate users.

Q3: Client-side validation of form data is useful because:
   a) it allows values to be sent to the browser.
   b) it prevents users navigating away from a form while it is partially complete.
   c) it validates data before it leaves the browser reducing the number of resubmissions for errors.
   d) allows bookmarks to be created which contain data values.

Q4: A database driven web site:
   a) presents a static web page to all users.
   b) loads some of the web site data from a sequential text file.
   c) is built using data statements and JavaScript.
   d) pulls site data from a database to create site pages.

Q5: To process a server site script your web server will require a:
   a) database server.
   b) HTML parser.
   c) server-side language module.
   d) cache.
6.2 Server side scripting requirements

**Learning objective**

By the end of this section you will be able to:

- use the basic syntax and common commands of the PHP language;
- understand and use variables and basic PHP operators/commands;
- use arrays to store data;
- use typical control structures within PHP scripts.

This section requires that you have access to a web server that makes use of the *AMP stack of technologies. You can find more information on various packages available in the introduction to the previous topic: *Database systems and SQL*.

6.2.1 Getting started with PHP

**PHP** is a server-side scripting language. This concept is not obvious, especially if you're just used to designing pages with **HTML** and **JavaScript**. A server-side scripting language is similar to JavaScript in many ways, as they both allow you to embed little programs (scripts) into the HTML of a Web page. In executing, such scripts allow you to control what will actually appear in the browser window more flexibly than is possible using HTML.

The key difference between JavaScript and PHP is that, while the Web browser interprets JavaScript once the Web page containing the script has been downloaded, server-side scripting languages like PHP are interpreted by the Web server before the page is even sent to the browser. Once interpreted, the PHP code is replaced in the Web page by the results of the script, so all the browser sees is an HTML file. The script is processed entirely by the server hence why it is called a server-side scripting language.

**Activity 1**

Create the following code in your HTML editor and save with the name `today.php` in the root folder of your web server installation then run the script by entering the URL e.g. http://localhost/today.php.

```html
<html>
<head>
<title>Today’s date</title>
</head>
<body>
<p>Today’s date (according to this web server) is</p>
<?php
    echo (date("l, f ds y."));

?>
</body>
```
Most of this is plain HTML. The line between \texttt{\textlangle ?php and ?\rangle}, however, is written in PHP. \texttt{\textlangle ?php} means "begin PHP code", and \texttt{?\rangle} means "end PHP code". The Web server is asked to interpret everything between these two tags and convert it to regular HTML code before sending the Web page to a browser that requests it.

The browser is presented with something like this:

```
<html>
<head>
  <title>Today's date</title>
</head>

<body>
  <p>Today's date (according to this web server) is
     Tuesday 11th of August 2015 02:10:29 PM</p>
</body>
</html>
```

All signs of the PHP code have disappeared. In their place, the output of the script has appeared and looks just like standard HTML. This example demonstrates several advantages of server-side scripting:

- No browser compatibility issues. PHP scripts are interpreted by the Web server and nothing else, so you don't have to worry about whether the language you're using will be supported by your visitors' browsers.

- Access to server-side resources. In the above example, we place the date according to the Web server into the Web page. If we had inserted the date using JavaScript, we would only be able to display the date according to the computer on which the Web browser was running. Now while this isn't an especially impressive example of exploiting server-side resources, we could have just as easily inserted some other information that would only be available to a script running on the Web server, for example, information stored in a MySQL database running on the Web server computer.

- Reduced load on the client. JavaScript can significantly slow down the display of a Web page on slower computers, as the browser must run the script before it can display the Web page. With server-side scripting, this load is passed to the web server.

### 6.2.2 Syntax and PHP

PHP syntax will be very familiar to anyone with an understanding of C, C++, Java, JavaScript, Perl, or any other C-derived language. A PHP script consists of a series of commands, or "statements", each of which is an instruction that the Web server must follow before proceeding to the next. PHP statements, like those in the above-mentioned languages, are always terminated by a semicolon (;).
The following is a typical PHP statement:

```php
echo ("This is a <strong>test</strong>!");
```

This statement invokes a built-in function called `echo` and passes it a string of text: `This is a <strong>test</strong>!`. Built-in functions can be thought of "things that PHP knows how to do without us having to spell out the details". PHP has a lot of built-in functions that let us do everything from sending e-mail to working with information stored in various types of databases. The `echo` function, however, simply takes the text that it is passed and places it into the HTML code of the page at the current location.

**Activity 2**

Create the following script:

```html
<html>
<head>
  <title>Simple PHP Example</title>
</head>

<body>
  <p>
    <?php echo( "This is a <strong>test</strong>!" ); ?>
  </p>
</body>
</html>
```

Copy and paste this code into a file called `test.php` and place it in the root folder of your web server. View the file by entering the URL `http://localhost/test.php` and then view the HTML source.

The browser viewing the page sees the following:

```html
<html>
<head>
  <title>Simple PHP Example</title>
</head>

<body>
  This is a <strong>test</strong>!
</body>
</html>
```

Notice the string of text contained HTML tags (`<strong>` and `</strong>`), which is perfectly acceptable.
You may wonder why we needed to surround the string of text with both parentheses (brackets) and quotes. Quotes are used to mark the beginning and end of strings of text in PHP, so their presence is required.

The parentheses serve a dual purpose. First, they indicate that echo is a function that is to be called. Second, they mark the beginning and end of the list of "parameters" that you wish to provide to tell the function what to do. In the case of the echo function, you only need to give the string of text to appear on the page, but we'll be looking at functions that take more than one parameter (for which we'll list the parameters separated by colons), as well as functions that take no parameters at all (for which we will still need the parentheses, but won't type anything between them).

### 6.2.3 Variables and Operations

Variables in PHP are identical to variables in most other programming languages. As you may already know, a variable is a name given to a location in memory into which a value may be placed.

The following statement creates a variable called $testvariable (all variable names in PHP begin with a dollar sign) and assigns it a value of 3:

```php
$testvariable = 3;
```

PHP is a "loosely typed" language, which means that a single variable may contain any type of data (be it a number, a string of text, or some other kind of value), and may change types over its lifetime. So the following statement, if written after the statement above, assigns a new value to our existing $testvariable. In the process, the variable changes from containing a number to containing a string of text:

```php
$testvariable = "Three";
```

The equals sign used in the last two statements is called the "assignment operator", as it is used to assign values to variables. Other operators may be used to perform various mathematical operations on values:

```php
$testvariable = 1 + 1; // Assigns a value of 2.
$testvariable = 1 - 1; // Assigns a value of 0.
$testvariable = 2 * 2; // Assigns a value of 4.
$testvariable = 2 / 2; // Assigns a value of 1.
```

The lines above each end with a comment. Comments are a way to describe what your code is doing by inserting explanatory text into your code and telling the PHP interpreter to ignore it. Comments begin with `//` and end at the end of the same line.

The C language standard comments which use `/* */` at the start and end of the comment also work in PHP. Comments are used by programmers to note what the program code does and are inserted into the program as it is written and amended. All the examples, from this point one, will include comments.

The +, -, *, and / operators provide the standard arithmetic operations of add, subtract, multiply and divide.
One other commonly used operator is used for string concatenation. The `.` operator is used to join strings of text together. For example:

```php
/* Assigns a value of "Hi there!". */
$testvariable = "Hi " . "there!";
```

Variables may be used anywhere an actual value can be. Look at the following example PHP code:

```php
<?php
  $var1 = "SCHOLAR";  // Assigns a value of "SCHOLAR" to $var1
  $var2 = 5;          // Assigns a value of 5 to $var2
  $var3 = $var2 + 1;  // Assigns a value of 6 to $var3
  $var2 = $var1;      // Assigns a value of "SCHOLAR" to $var2
                   // the "<br />" ensures a new line for each output
  echo($var1) . "<br />";  // Outputs "SCHOLAR"
  echo($var2) . "<br />";  // Outputs "SCHOLAR"
  echo($var3) . "<br />";  // Outputs 6
  echo($var1 . " rocks!" . "<br />" );  // Outputs "SCHOLAR rocks!"
  echo("$var1 rocks!") . "<br />";  // Outputs "SCHOLAR rocks!"
  echo("$var1 rocks!" . "<br />");  // Outputs '"$var1 rocks!'
?>
```

Look at the last two lines. Note that you can include the name of a variable inside a text string and have the value inserted in its place if you surround the string with double quotes, however, a string surrounded with single quotes will not convert variable names to their values.

### 6.2.4 Arrays

An array is a special kind of variable that contains multiple values. If you think of a variable as a box that contains a value, then an array can be thought of as a box with compartments, where each compartment is able to store an individual value.

The simplest way to create an array in PHP is with the built-in array function:

```php
$myarray = array('best', 62, 'kelly');
```

This code creates an array called `$myarray` that contains three values: 'best', '62', and 'kelly'. Just like an ordinary variable, each space in an array can contain any type of value. In this case, the first and third spaces contain strings, while the second contains a number.

**Figure 6.1: Array Structure from PHP**

<table>
<thead>
<tr>
<th>best</th>
<th>62</th>
<th>kelly</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>$myarray</td>
</tr>
</tbody>
</table>

To get at a value stored in an array, you need to know its index. Typically, arrays use numbers, starting with zero, as indices to point to the values they contain.
That is, the first value (or element) of an array has index 0, the second has index 1, the third has index 2, and so on. In general, therefore, the index of the nth element of an array is n-1. Once you know the index of the value you're interested in, you can get that value by placing the index in square brackets following the array variable name:

```php
    echo($myarray[0]); // Outputs "best"
    echo($myarray[1]); // Outputs "62"
    echo($myarray[2]); // Outputs "kelly"
```

You can also use the index in square brackets to create new elements, or assign new values to existing array elements:

```php
    $myarray[1] = 'which'; // Assign a new value
    $myarray[3] = 'cat'; // Create a new element
```

You can add elements to the end of an array by using the assignment operator as usual, except with empty square brackets following the variable name:

```php
    $myarray[] = 'new element';
    echo($myarray[4]); // Outputs "new element"
```

### 6.2.5 Associative Array

Array indices don't always have to be numbers but numbers are the most common choice of index. You can also use strings as indices to create what is called an associative array. This type of array is called associative because it associates values with meaningful indices. In this example, we associate a date with each of three names:

```php
    $birthdays['Charlie'] = '1969-01-05';
    $birthdays['Tom'] = '1959-05-16';
    $birthdays['David'] = '1962-09-09';
```

Now if we want to know Charlie's birthday, we just look it up using the name as the index:

```php
    echo('My birthday is: '. $birthdays['Charlie']);
```

This type of array is especially important when it comes to user interaction in PHP.

### 6.2.6 Control structures

All the examples of PHP code that we've seen so far have been either simple, one statement scripts that output a string of text to the Web page, or have been series of statements that were to be executed one after the other in order. If you've ever written programs in any other languages (be they JavaScript, C, or BASIC) you already know that practical programs are rarely so simple.

PHP, just like any other programming language, provides facilities that allow us to affect the flow of control in a script. That is, the language contains special statements that permit you to deviate from the one-after-another execution order that has dominated our examples so far. Such statements are called control structures. Don't get it? Don't worry! A few examples will illustrate perfectly.
6.2.6.1 IF statement

The most basic, and most often-used, control structure is the if statement. The syntax of the IF statement is:

```php
if ( condition ) {
    // Statement(s) to be executed if
    // condition is true.
} else {
    // (Optional) Statement(s) to be
    // executed if condition is false.
}
```

This control structure lets us tell PHP to execute one set of statements or another, depending on whether some condition is true or false.

The `else` clause (that part of the if statement that says what to do if the condition is false) is optional. Let's say you wanted to display the message above only if the appropriate name was entered, but otherwise, you didn't want to display any message.

Here's how the code would look:

```php
<?php

$name = $_GET['name'];
if ( $name == 'Charlie' ) {
    echo( 'Hello, how can I help you, Charlie ');
}

?>
```

The `==` (two equals signs) used in the condition above is the PHP equal-to operator that's used to compare two values to see whether they're equal.

Conditions can be more complex than a single comparison for equality.

```php
<?php
$firstname = $_GET['firstname'];
$lastname = $_GET['lastname'];
if ( $firstname == 'Charlie' and $lastname == 'Burton' ) {
    echo( 'Hi, How can I help you, Charlie ');
} else {
    echo( "Welcome to my Website, $firstname $lastname!" );
}

?>
```

In this example the condition will be true only if $firstname has a value of Charlie and $lastname has a value of Burton. The word and in the above condition makes the whole condition true only if both of the comparisons evaluate to true.
Another such operator is or, which makes the whole condition true if one or both of two simple conditions are true. If you're more familiar with the JavaScript or C forms of these operators (&& and || for and and or respectively), they also work in PHP.

6.2.6.2 WHILE statement

The while loop allows us to use a condition to determine how many times a set of statements are repeated. It looks like this:

```php
while ( condition ) {
    // statement(s) to execute over
    // and over as long as condition
    // remains true
}
```

This works very similarly to an if-else statement without an else clause. The difference arises when the condition is true and the statement(s) are executed. Instead of continuing the execution with the statement that follows the closing brace (}), the condition is checked again. If the condition is still true, then the statement(s) are executed a second time, and a third, and will continue to be executed as long as the condition remains true. The first time the condition evaluates false (whether it's the first time it's checked or the one-hundred-and-first), execution jumps immediately to the next statement that follows the while loop (after the closing brace).

This type of loop is especially useful when reading a set of results from a table. The loop can be written to continue to results until there are no results left for example.

Activity 3

Now create a new html file and enter the following code in the body of the document.

```html
<?php
$count = 1;
while ($count <= 10) {
    echo("$count ");
    $count++;
}
?>
</body>
</html>
```

Save the new HTML document as `count1.php` and then use your web browser to view the results.
The first line of the PHP code creates a variable called $count and sets its value to 1.

The second line is the start of a while loop, the condition for which is that the value of $count is less than or equal to (<=) 10.

The third line, which is within the while loop, displays the value of $count followed by a space. The fourth line adds one to the value of $count ($count++ is a shortcut for $count = $count + 1).

The closing brace (}) sets the point at which the while loop ends.

The first time the condition is checked, the value of $count is 1, so the condition is definitely true. The value of $count (which is 1) is output, and $count is given a new value of 2. The condition is still true the second time it is checked, so the value, 2, is output and a new value, 3, is assigned.

And the process continues, outputting the values 3, 4, 5, 6, 7, 8, 9, and 10. Finally, $count is given a value of 11, and the condition is false, which ends the loop. So the code will produce the string of characters: "1 2 3 4 5 6 7 8 9 10".

The while condition in this example used the less than or equal operator. The same basic operators are available in PHP as are available in MySQL.

\[
\begin{array}{|c|c|}
\hline
\text{Operator} & \text{Description} \\
\hline
= & \text{Equal} \\
!= & \text{Not equal} \\
> & \text{Greater than} \\
< & \text{Less than} \\
>= & \text{Greater than or equal} \\
<= & \text{Less than or equal} \\
\hline
\end{array}
\]

6.2.6.3 FOR statement

The For loop is used when a set number of repetitions are required. Often this is when a value is being counted. The basic syntax of the For loop is:

```php
for ( initialize; condition; update ) {
// statement(s) to execute over
// and over as long as condition
// remains true after each update
}
```
Activity 4

The following code performs the same operation as the while loop above using a for loop. Load `count1.php` and replace the code `<?php?>` with the following:

```php
<?php
for ($count = 1; $count <= 10; $count++) {
    echo( "$count " );
}
?>
```

Save the edited file as `count2.php` and then use your web browser to view the results.

6.2.6.4 Multipurpose pages

PHP can be used to allow one page to do serve more than one purpose. If you wanted each page on your web site to display the visitor's name then you have already written a suitable PHP script to do this, however, there are some problems which need to be overcome:

- the name is needed on every page of the site, not just one;
- you have no control over which page visitors to your site view first.

The first problem is easy to solve. Once we have the user's name in a variable on one page, we can pass it with any request to another page by adding the name to the query string of all links:

```html
<a href="newpage.php?name=<?php echo(urlencode($name)); ?>">A link</a>
```

The PHP code is embedded right in the middle of the HTML tag.

Similarly, a shortcut exists for when you simply want to echo a PHP value in the middle of your HTML code. The shortcut looks like this:

```html
<a href="newpage.php?name=<?php= urlencode($name) ?>">A link</a>
```

The tags `<?...?>` perform the same function as the much longer code `<?php echo(...); ?>`.

The `urlencode` function takes any special characters in the string (for example, spaces) and converts them into the special codes they need to be in order to appear in the query string. For example, if the `$name` variable had a value of "Charlie Burton", then as spaces are not allowed in the query string, the output of `urlencode` (and thus the string output by echo) would be "Charlie+Burton". This would then be automatically converted back by PHP when it created the `$name` variable in `newpage.php`.

So, the first problem of passing the variable to each page in the site is relatively easy but the name still has to be captured in the first place.
In the welcome message example earlier, there was an HTML page with a form in it that prompted the user for his or her name. The problem with this is that we couldn't force the user to enter our Web site by that page every time he or she visited our site.

The solution is to have every page of the site check to see if a name has been specified, and prompt the user for a name if necessary. This means that every page of the site will either display its content, or a prompt the user to enter a name, depending on the state of the $name variable. This will require the use of an if-else condition.

**Activity 5**

Start a new HTML document and enter the following code.

```html
<html>
<head>
    <title>Login / Echo User Demo</title>
</head>
<body>

<?php

if ( !isset($_GET['name']) ) {

    <!-- No name has been provided, so we prompt the user for one. -->
    <form action="<?php echo $_SERVER['PHP_SELF']; ?>" method="get">
        Please enter your name: <input type="text" name="name" />
        <input type="submit" value="Submit" />
    </form>

} else {
    $name=$_GET['name'];

    <p>Your name: <?php echo $name; ?></p>
    <p>This paragraph contains a <a href="nextpage.php?name=<?php echo urlencode($name); ?>">link</a> that passes the name variable on to the next document.</p>

} //endif

</body>
</html>
```

Save this file as welcome-demo.php and then view it using your web browser.

PHP code can be dropped in and out of with HTML so the open (<?php or <?) and close (?>) PHP tags can be used so that HTML code can be inserted into the condition true and condition false parts of the statement.

The next new element of PHP is that this code uses a new function called **isset** in the condition. This function returns a value of true if the variable it is given has been assigned a value (i.e. if a name has been entered), and false if the variable does not exist (i.e. if a name has not yet been entered).
The exclamation mark (the NOT operator) that appears before the name of the function reverses the returned value from true to false or vice-versa. This means that the form is displayed when the $name variable is not set.

The third new element is the use of the variable $SERVER['PHP_SELF'] to specify the action attribute of the form tag. This variable is one of several that PHP always gives a value to automatically. In particular, $SERVER['PHP_SELF'] will always be set to the URL of the current page. This provides an easy way to generate a form which will target the current page but this time will submit the $name variable with a value set.

If all the pages on the site are structured in this way, visitors will be prompted for their name by the first page they attempt to view, whichever page this happens to be. Once a name is entered and the submit button is clicked, the visitor will be presented with the exact link from that point onward, ensuring that the visitor is prompted only the once to enter his/her name.

### 6.3 Connecting to database

#### Learning objective

By the end of this section you will be able to:

- use the mysqli class in PHP to connect to a database.

As you know from previous topics, you have to connect to the database server and select the database to be used before you can execute MySQL statements. PHP has a number of commands which are used for this purpose.

The following PHP function call establishes the connection and selects the database required (or returns an error):

```php
$mysqli = new mysqli($server, $username, $password, $database);
```

For example:

```php
$mysqli = new mysqli("localhost", "root", "root","esports");
```

The items required to connect are:

- server, the IP address or name of the server on which the database server software is running;
- username, the username required to connect to the database server;
- password, the password required to connect to the database server;
- database, the name of the database, stored on the database server, that this connection will connect to.

The mysqli function shown above, for example, returns an object that includes details
of the connection that has been established. We need to store this object so that we can make use of it later.

Since the MySQL server is a completely separate piece of software, we must consider the possibility that the server is unavailable, or inaccessible due to a network failure, or because the username/password combination you provided is not accepted by the server. In such cases, the mysqli function returns an object containing details of the error.

```php
if ($mysqli->connect_errno) {
    // if the connect_errno value is set then show the error.
    echo "Failed to connect to MySQL: " .
    $mysqli->connect_errno . "");
    $mysqli->connect_error;
    die;
} else {
    echo "Connected to database"; // is okay, we connected
}
```

There are two points to note about this code. $mysqli is an object created from the constructor method of the class mysqli and connect_errno is an instance variable set in this object by the class constructor method when a connection fails. The -> is used in PHP to indicate we are referring to a value or function within the object.

The last point to note is the die function, which is your first example of a function that does not use parameters. All this function does is cause PHP to stop reading the source file at this point. This is a good response to a failed database connection, because in most cases the page will be unable to display any useful information without the database connection.

**Activity 6**

You are going to create a server side include. This is a piece of code, a module or small library of code, that you can import into your main program. In this case the include will establish the database connection for us.

Start a new document in your editor. Delete any lines of HTML which might appear pre-written in your editor (such as `<HTML> </HTML>` etc.)

Enter the following text - only this text should appear in the file.

```php
<?php
// connect to the database using mysqli API
$mysqli = new mysqli($host, $username, $password, $database);
if ($mysqli->connect_errno) {
    echo "Failed to connect to MySQL: " .
    $mysqli->connect_errno . "");
    $mysqli->connect_error;
    die;
} else {
    echo "Connected to database"; // is okay, we connected
}
?>
```
Save this file in the root of your web server as db_connection.php.

Start a new HTML document and enter the following code:

```php
<?php
// set up the connection variables for the db_connection.php include
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php"); // require the include code

?>
<html>
<head>
<title>Database Connect</title>
</head>
<body>

<?php
// just to check, let's get the name of the current database
/* return name of current default database */
if ($result = $mysqli->query("SELECT DATABASE()") ) {
    // run the query method to get the database result object
    $row = $result->fetch_row();
    // read this from the query results object as an array of data
    $default_database = $row[0];
    // set the value to the first element [0] in the array
    $result->close();
    // free the $result set (clear it)
}
?>

You are connected to the MySQL server and the
</body>
</html>
```

Save this file as database1.php. View the PHP file using your browser.

You can insert the content of a file into a PHP file before the server executes it, with the include() or require() function. The two functions are identical in every way, except how they handle errors. The include() function generates a warning (but the script will continue execution) while the require() function generates a fatal error (and the script execution will stop after the error).

This script sets the connection values for the database (the host/server, username, password and required database) as variables which are then used in the include file. For future database connections you now can reuse the db_connection.php include file and set the database connection values as required.
The `include()` and `require()` functions are used to create functions, headers, footers, or elements that can be reused on multiple pages. This can save the developer a considerable amount of time. This means that you can create a standard header or menu file that you want all your web pages to include. When the header needs to be updated, you only have to update the include file, or when you add a new page to your site, you can change the menu file (instead of updating the links on all web pages).

### 6.4 Executing query

**Learning objective**

By the end of this section you will be able to:

- carry out MySQL queries to retrieve data using PHP Scripts;
- write PHP code to trap query errors.

All SQL queries are executed using the `mysqli->query()` method of the `mysqli` class. This executes the query and returns the result set as an object for SELECT queries. For other queries it returns a true or false result depending on the success of the query.

**Example**

Return the number of rows that match a query:

```php
/* Select queries return a resultset */
if ($result = $mysqli->query("SELECT game FROM game");) {
    printf("Select returned %d rows.\n", $result->num_rows);

    /* free result set */
    $result->close();
}
```

The above code will display the number of rows from the game table.

**Activity 7**

Create a table and receive some feedback:

```php
/* Create table doesn't return a resultset */
if (!$mysqli->query("CREATE TABLE tmplogs (id INT PRIMARY KEY, username varchar(30), time timestamp ) ") === TRUE)
{
    printf("Table tmplogs successfully created.\n");
```
This code will create a table called \textit{tmplogs}. If this is successful it will display the message "Table tmplogs successfully created."

MySQL also keeps track of the number of rows affected by \texttt{INSERT}, \texttt{DELETE} and \texttt{UPDATE} queries and the number of rows in the results of a \texttt{SELECT} query. This number can be accessed using the \texttt{mysqli->affected_rows} method.

So a script to run a query and trap any errors would be:

```php
<?php

if ($mysqli->query($query_string)) {
    echo ('Query successful with ' . $mysqli->affected_rows . ' rows changed/accessed.);
} else {
    echo ('Error performing query: ' . $mysqli->error());
}

?>
```

6.5 Retrieving results

**Learning objective**

By the end of this section you will be able to:

- write MySQL queries which are executed by PHP to retrieve and process query results;
- write data to a form for processing.

The result set from a database query is provided as an object, each row from the result set is also fetched as an object. The instance variables within the object can then be individually accessed.

So if the \textit{game} table consists of the following data:

\begin{table}[h]
\centering
\begin{tabular}{|l|l|}
\hline
	extbf{game} & \textbf{platform} \\
\hline
Massive RPG & X-station \\
SuperJoe & S-box-360 \\
Terra 1999 & PC \\
\hline
\end{tabular}
\caption{game table}
\end{table}

© Heriot-Watt University
The query "SELECT * FROM game" will return this as the result set from the code:

```php
$query_string = "SELECT * FROM game";

if ($result = $mysqli->query($query_string)) {    //run the query

The query is executed within the if condition, so if the query fails we can recover from it without crashing. If the query is successful, the contents of the game table will be held in the object called $result.

To read the rows and display the data from each row, we need to read a row at a time and display its values.

The code:

```php
while ($row = $result->fetch_object()) {
```

retrieves each row from the $result (an object that holds the result set) and repeats the process until there are no rows left in the result set object.

In order to read the instance variables from each row object (these correspond to the column names) we use the code:

```php
printf ("%s (%s)", $row->game, $row->platform);
```

The code $row->game returns the game column value for the current row and the $row->platform code returns the platform column value.

So by printing these values to the screen, the resulting webpage output should be:

Massive RPG (X-station)
SuperJoe (S-Box-360)
Terra 1999 (PC)

Create the full script using the code below.
Activity 8

Create a copy of the file database1.php called database2.php. Amend the code as follows:

```php
<?php
    $host="localhost";
    $username="root";
    $pword="root";
    $database="esports";

    require("db_connection.php");

?>
</html>
<head>
<title>Database Query Test</title>
</head>
<body>
    <?php
        // set up a query to use
        $query_string = "SELECT * FROM game";

        if ($result = $mysqli->query($query_string)) {
            // run the query
            // fetch each row as an object
            while ($row = $result->fetch_object()) {
                // display the game and platform fields from the row object
                printf ("%s (%s)", $row->game, $row->platform);

                echo "<br />"; // start a new line
            }
        }
        $result->close(); // free the $result set (clear it)
    } else {
        echo ("Error performing query: " . $mysqli->error() );
    }

    $mysqli->close(); // close the database connection
?>
</body>
</html>
```

..........................................

© HERIOT-WATT UNIVERSITY
6.6 Editing/Amending data

Learning objective
By the end of this section you will be able to:

- use PHP scripts and HTML forms to add/update/amend data stored in a MySQL database.

To edit or change data in the data we need to first read the database and display the values to be edited in a suitable format.

A simple way to do this is to display each from the table and then have an edit button which will display a form to update the specific record.

Activity 9
Create a copy of the file database2.php called database3.php. Amend the code as follows:

```php
<?php
    $host="localhost";
    $username="root";
    $password="root";
    $database="esports";
    require("db_connection.php");

    ?
    <html>
    <head>
    <title>Database Display</title>
    <style>
    table, td, th {border: 1px solid black;}
    </style>
    </head>
    <body>
    <?php
    // set up a query to use
    $query_string = "SELECT * FROM game";
    ?>
    <table>
    <thead>
    <th class="outline">Game</th><th>Platform</th><th></th>
    </thead>
    <tbody>
    if ($result = $mysqli->query($query_string)) {
        // run the query
        while ($row = $result->fetch_object()) {
```
This code reads every row from the game table and displays each row with an **Edit** hyperlink.

<table>
<thead>
<tr>
<th>game</th>
<th>platform</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Massive RPG</td>
<td>X-station</td>
<td>Edit</td>
</tr>
<tr>
<td>SuperJoe</td>
<td>S-box-360</td>
<td>Edit</td>
</tr>
<tr>
<td>Terra 1999</td>
<td>PC</td>
<td>Edit</td>
</tr>
</tbody>
</table>

Notice that the PHP function `urlencode` is used to encode the game values so that they are ready for passing using the **GET** method.
Clicking on the hyperlink will activate a script to display the row values in a form for editing. The script required for this is:

```php
<?php
    $host="localhost";
    $username="root";
    $password="root";
    $database="esports";
    require("db_connection.php");
    $game = filter_var($_GET[\'game\']);
?
<html>
<head>
    <title>Database Update</title>
</head>
<body>
    <form action="update-row-game.php" method="get" name="updateform">
        <!-- set up a query to use
        $query_string = "SELECT * FROM game WHERE game.game = \\" . $game . \\"";
        if ($result = $mysqli->query($query_string)) {
            //run the query

        while ($row = $result->fetch_object()) {
            /*
            <!-- keep track of the original primary key value to update the row --
            <input type="hidden" name="gameid" value="<?php echo urlencode($row->game); ?>">

            <!-- values which can be edited --
            Game: <input type="text" value="<?php echo $row->game; ?>"
                    name="game" size="40"><br />
            Platform: <input type="text" value="<?php echo $row->platform; ?>"
                    name="platform" size="40"><br />

            <!-- button to submit form --
            <button class="submit" type="submit">Update Row</button>
            </form>
            
        }
    ?></form>
</body>
$result->close(); //free the $result set (clear it)

} else {
    echo "Error on Game Table Query";
}
$mysqli->close();
?>
</tbody>
</table>
</body>
```
This code (save it as edit-data.php) produces a very simple web form containing the database values. Notice that a hidden input is used to hold the original value of the primary key. This is because if the primary key value is changed we need a record of the previous key value in order to find the row and update it. The form displays the data as follows:

**Game:** Massive RPG  
**Platform:** X-station

These values can be edited in the form. e.g.

**Game:** Super Massive RPG  
**Platform:** X-station 2

Update Row
To process the result of this change we need to have a script to read the form data and update the database:

```php
<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php");
$gameid = filter_var($_GET[‘gameid’]); //read the previous primary key
$game = urldecode($gameid); //decode the previous key value
$game = filter_var($_GET[‘game’]); //read the value to be written for game
$platform = filter_var($_GET[‘platform’]); //read the value to be written for platform

<html>
<head>
<title>Database Update</title>
</head>
<body>

//set up a query to use
$query_string = "UPDATE game SET game = ‘".$game . "’, platform = ‘".
 . $platform . “’ WHERE game.game = ‘".$gameid . “’ “;

if ($result = $mysqli->query($query_string)) { //run the query
    echo "Completed: Updated Row in Table"; //show successful output
    echo "<a href=database3.php>Back</a>"; //navigation back to records
    $result->close(); //free the $result set (clear it)
} else {
    echo "Error: Unable to Update Row in Table";
    //show error because it’s failed
}

$mysqli->close();

?>
</tbody>
</table>
</body>
```

Save this script as `update-row-game.php`.
Notice that the script reads all three values from the form using the GET method. The hidden primary key value, which was encoded in the previous script, is not decoded using the function `urldecode` and `filter_var` is used again to sanitize the values passed to the script.

An UPDATE query is used to set the new values for each of the columns and the old primary key value is used in the WHERE clause to locate the row so that it can be updated.

### 6.6.1 Insertion

Inserting values can be done in a similar manner to updating values. To do this we will add a link to the record display script to insert a new record, present a blank version of the update form and have a new script that will insert this new record.

**Activity 10**

Open `database3.php` and add the following line of code between the `<table>` and `</body>` tags (around line 45).

```html
</table>

<a href="add-game-row.php">Insert a new row</a>
</body>
```

Now create a new script called `add-game-row.php` which will include this form code.

```html
<html>
<head>
<title>Database Insert</title>
</head>
<body>
<form action="insert-row-game.php" method="get" name="insertform">
Game: <input type="text" value="" name="game" size="40"><br />
Platform: <input type="text" value="" name="platform" size="40"> <br />

<!-- button to submit form -->
<button class="submit" type="submit">Insert Row</button>
</form>
</body>
</html>
```
This is how the script will display the form:

```
Game: 
Platform: 
[Insert Row]
```
To process the data from this form, create the following script called `insert-row-game.php` which will be called from the `add-game-row.php` script.

```php
<?php
    $host="localhost";
    $username="root";
    $password="root";
    $database="esports";

    require("db_connection.php");
    $game = filter_var($_GET['game']);
    // read the value to be written for game
    $platform = filter_var($_GET['platform']);
    // read the value to be written for platform
?

    <html>
    <head>
    <title>Database Insert</title>
    </head>
    <body>
    <?php
    // set up a query to use
    $query_string = "INSERT INTO game (game, platform)
        VALUES (" . $game . "," . $platform . ");"

    if ($result = $mysqli->query($query_string)) {  // run the query
        echo "Completed: Inserted Row in Table";  // show successful output
        echo "<a href=database3.php >Back</a>";  // navigation back to records
    } else {
        echo "Error: Unable to Insert Row in Table<br />";
        // show error because it's failed
        print_r($mysqli->error_list);
        // display a list of the MySQL errors that occurred
        echo "<br /><a href=database3.php >Back</a>";
        // navigation back to records
    }
    $result->close();  // free the $result set (clear it)
    $mysqli->close();
?
    </tbody>
    </table>
    </body>
```
This script has some additions. The error checking is improved because now the script will report that an error has occurred (line 30) and also report the type of error or errors that have occurred (line 31). There is then an option to go back to the list of database rows.

.................................

6.6.2 Deletion

Deleting a row is can be done with from the table display script. To delete a row all that is required is the primary key value. If we add a DELETE link next to the EDIT link, this can be used to delete that specific row.

Activity 11

Update the script database3.php as follows.

Around line 24 edit the code to add an additional open and close th tag.

```
<th class="outline">Game</th><th>Platform</th>
```

Around line 36

(after

```
echo"<td><a href=edit-data.php?game=" . urlencode($row->game) . ">Edit</a></td>"
```

) add the line:

```
echo "<td><a href=delete-data.php?game=" . urlencode($row->game) . ">Delete</a></td></tr>"
```

MAKE SURE to remove the </tr> from the end of the previous line of code, otherwise your table will display incorrectly.

The completed script will be:

```
<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php");

?>
<html>
<head>
<title>Database Query Player</title>
```
<?php
// set up a query to use
$query_string = "SELECT * FROM player";
//
?>

<table>
<thead>
<th class="outline">Username</th><th>Realname</th><th>Password</th>
<th>Email</th><th>Message</th><th>Terms and Conditions</th>
</thead>
<tbody>
<?php
if ($result = $mysqli->query($query_string)) { // run the query
while ($row = $result->fetch_object()) {
    echo "<tr>
    echo "<td>" . $row->username . "</td>
    echo "<td>" . $row->realname . "</td>
    echo "<td>" . $row->pword . "</td>
    echo "<td>" . $row->email . "</td>
    echo "<td>" . mb_convert_encoding($row->message, 'utf-8', 'iso-8859-1') . "</td>
    echo "<td>" . $row->terms_and_conditions . "</td>
        echo "<td><a href=edit-data-player.php?username= . urlencode($row->username) . ">Edit</a></td>
    echo "<td><a href=delete-data-player.php?username= . urlencode($row->username) . ">Delete</a></td>
</tr>
}
$result->close(); // free the $result set (clear it)
}

mysqli->close();
?>
</tbody>
</table>
<a href="add-player-row.php">Insert a new row</a>
</body>
</html>
The table now displays as:

<table>
<thead>
<tr>
<th>game</th>
<th>platform</th>
<th>Edit</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Massive RPG</td>
<td>X-station</td>
<td>[Edit]</td>
<td>Delete</td>
</tr>
<tr>
<td>SuperJoe</td>
<td>S-box-360</td>
<td>[Edit]</td>
<td>Delete</td>
</tr>
<tr>
<td>Terra 1999</td>
<td>PC</td>
<td>[Edit]</td>
<td>Delete</td>
</tr>
</tbody>
</table>

**Insert a new row**

This code will call the delete script with the encode primary key value for the row to be deleted.
Create a new script called `delete-data.php`.

```php
<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php");
$game = filter_var($_GET['game']);

?>
<html>
<head>
<title>Database Delete</title>
</head>
<body>
  <?php
  // set up a query to use
  $query_string = "DELETE FROM game WHERE game.game = ?" . $game . ";
  if ($result = $mysqli->query($query_string)) {
    // run the query
    echo "Success, the row for game=" . $game . " has been deleted.<br />
    echo "<br /">a href=database3.php >Back</a>";
    // navigation back to records
  } else {
    echo "Error: Unable to Delete Row in Table<br />
    echo "<br /">a href=database3.php >Back</a>";
    // display a list of the MySQL errors that occurred
    echo "<br /">a href=database3.php >Back</a>";
    // navigation back to records
  }
  $result->close(); // free the $result set (clear it)
  $mysqli->close();
?>
</html>
```

When run, the following confirmation will appear:

**Success, the row for game=Test123 has been deleted.**

[Back](#)
Exercise

Using what you have learned from the previous exercises, complete scripts for the player table in the Esports database which will allow records to be amended, inserted and deleted.

6.7 Learning points

Summary

You should now be able to:

• connect to a database to process data;
• read/write data from/to a database;
• use insert, amend and deletion queries;
• process a form using server-side scripting.
6.8 End of topic test

End of topic test

Q6: Which four parameters are required to connect a database server using the mysqli API?

a) Client, server, username, password.
b) Username, password, database, tablename.
c) Username, server, password, database.
d) Client, tablename, username, server.

Q7: The PHP echo command:

a) displays output to a text file.
b) displays output.
c) duplicates a variable.
d) writes a variable as HTML.

Q8: The GET method is more secure than the POST method for transmitting data from the client to the server.

a) True
b) False

Q9: To execute a query, stored in a variable $query, the following PHP code is required:

a) $mysqli->query($query);
b) $result->fetch_object($query);
c) $mysqli->prepare($query);
d) $result->mysqli->error($query);

Q10: Which function should be used to sanitize POST and GET values submitted by a form?

a) urlencode
b) htmlspecialchars
c) filter_var
d) sanitize

© HERIOT-WATT UNIVERSITY
Topic 7

Implications: Legal, ethical, environmental, economic and social

Contents

7.1 Revision ................................................................. 165
7.2 Intellectual property rights .............................................. 166
  7.2.1 Protecting IPR via patents, registered designs and trademarks ........................................... 166
  7.2.2 Enforcement .................................................... 167
7.3 Storage of user data .................................................... 168
7.4 Increasing use and power of intelligent systems .................. 169
7.5 Energy (data centres, low carbon equipment) ...................... 169
7.6 Online marketing (web, e-mail, text) ............................... 171
  7.6.1 Cookie tracking .................................................. 172
7.7 Analytics ................................................................. 173
7.8 Cyber security risks and precautions ................................ 175
  7.8.1 Cyber crime ...................................................... 176
  7.8.2 Cyber prevention ............................................... 177
7.9 Tracking, privacy, online safety ..................................... 177
  7.9.1 “Right to be forgotten” ........................................... 178
7.10 Social media ............................................................. 178
7.11 Implications of “Big data” ............................................ 179
7.12 Learning points ......................................................... 180
7.13 End of topic test ....................................................... 181

Prerequisite knowledge

From your studies at Higher you should already:

• know about a range of contemporary software development languages, software development environments, intelligent and online systems;

• be able to describe and give the implications of:
  ◦ the Computer Misuse Act;
TOPIC 7. IMPLICATIONS: LEGAL, ETHICAL, ENVIRONMENTAL, ECONOMIC AND SOCIAL

- the Data Protection Act;
- the Copyright, Designs and Patents Act;

- be able to describe environmental implications including: lifetime carbon footprint (manufacture, use, disposal) and environmental benefits;

- know of the economic and social impact of computing science including:
  - competitive advantage, global marketplace, business costs, maintainability, scalability;
  - censorship and freedom of speech, privacy and encryption, global citizenship, online communities.

Learning objectives

By the end of this topic you should be able to describe and consider the implications of:

- Intellectual property rights relating to the development of technologies;
- social control via data and media;
- issues relating to the storage of large amounts of user data;
- Intelligent systems;
- energy consumption;
- online marketing including (web, email and text/SMS/IM);
- Data analytics;
- tracking, privacy and online safety;
- social media/networking;
- Big data.
7.1 Revision

Quiz: Revision

Q1: The Regulation of Investigatory Powers Act requires that:

a) employers make use of digital communication methods.
b) staff hand all work related encryption keys to their employers.
c) employers pay for systems to assist with electronic interception of communications.
d) staff allocate copyright for all developments to their employer.

Q2: It is possible that replacing an item of technology is more environmentally friendly than continuing to use an existing item. This could be because:

a) it reduces the need for technology items to go to landfill.
b) new technology items are more energy efficient.
c) new technology items make use of additional toxic minerals.
d) it increases the use of packaging and materials.

Q3: Life-time carbon footprint refers to the environmental impact of the:

a) manufacture of an item of technology.
b) manufacture and distribution of an item of technology.
c) manufacture and recycling of an item of technology.
d) entire life of the item including manufacture, distribution, deployment and recycling/disposal.

Q4: Encryption safeguards web traffic from being read. This uses SSL encryption which is an example of:

a) Symmetric encryption.
b) Caesar cipher.
c) Asymmetric encryption.
d) Key exchange encryption.

Q5: Keylogging is an attack that is used to:

a) gather information about encryption keys.
b) access database technology embedded in secure systems.
c) record security credentials entered via the keyboard.
d) break secure passwords using distributed processing.
7.2 Intellectual property rights

Learning objective
By the end of this section you will be able to:

• describe the purpose and implication of copyright, trademarks, designs and patents.

Existing intellectual property rights help individuals stop others stealing or copying:

• the names of products or brands;
• inventions;
• the design or look of products;
• things that are written, made or produced.

Copyright, patents, designs and trade marks are all types of intellectual property protection. In the U.K. copyright exists automatically on any and all original works and a design right applies to the shape and configuration of a design (how it fits together), however patents, registered designs and trademarks all have to be registered to enable an individual or organisation to exert rights over the work. Intellectual property only applied to something unique that is created. An idea on its own is not intellectual property. For example, an idea for a great mobile phone app doesn’t count, but an actual app that you have written would.

7.2.1 Protecting IPR via patents, registered designs and trademarks

Intellectual property that is an original work is automatically covered by copyright law in the U.K. however there are additional intellectual property rights that must be applied for. An individual or organisation will own intellectual property if they meet at least one of the following points:

• They have:
  ◦ created the item (and it meets the requires for a patent or a design);
  ◦ purchased the property rights from the creator or previous owner;
  ◦ brand that could be a trademark e.g. a well-known product name.

Typically, unless your contract of work states an alternative, any work you create while directly employed by someone will belong to the person/organisation employing you.

Applying for trademark, registered design and/or patents takes a certain amount of time for the required processes and searchers for similar properties to be completed.
### Type of Protection

<table>
<thead>
<tr>
<th>Type of Protection</th>
<th>Examples of Intellectual Property</th>
<th>Time to Allow for Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trademarks</td>
<td>Product names, logos, jingles.</td>
<td>4 months</td>
</tr>
<tr>
<td>Registered designs</td>
<td>Appearance of a product including shape, packaging, patterns, colours and decoration.</td>
<td>1 month</td>
</tr>
<tr>
<td>Patents</td>
<td>Inventions and products e.g. machines and machines parts, tools, medicines.</td>
<td>Approximately 5 years</td>
</tr>
</tbody>
</table>

It is typical for more than one type of protection to be used to protect intellectual property, for example, a rights holder could:

- register a name and logo as a trademark;
- protect a piece of software by registered its unique user interface as a registered design;
- patent a completely new process for creating the software using an innovative technique;
- use copyright to protect images of the software.

Once protected via one or more of the above protections, your intellectual property is yours to exploit. If others infringe on your rights you have the power to take action against those others.

#### 7.2.2 Enforcement

It is the responsibility of the intellectual property owner to take action if IP is used without permission. Intellectual property rights can be infringed if someone in the U.K.:

- uses, sells or imports a product or process similar to that of a rights holder;
- makes, offers or sells your registered design for commercial gain;
- uses a trademark similar or identical to one you have registered.

There are three possible ways of resolving and intellectual property infringement.

- Reach an agreement with the infringing party to stop using your IP or reach agreement on its use (perhaps via an IP licence agreement).
- Use mediation or another type of dispute resolution to reach a solution.
- Take legal action if agreement isn’t possible via other means.

Enforcement via the courts is typically expensive and often infringing parties will often settle out of court due to the possible costs involved.

**Activity: Patent Infringement**

Research two or three examples of technology patent infringement. These cases often attract large settlement damages or licence fees.
7.3 Storage of user data

Learning objective
By the end of this section you will be able to:

• explain some of the issues relating to the storage of large amounts of user data and how the Data Protection Act applies to this.

Increasingly, large amounts of data are stored by individuals and third parties. This data covers a wide variety of uses including personal data (data relating to an individual such as photos, location information, personal documents etc.) and business data (data relating to the operation of a business or organisation).

We create a huge volume of data every year. All our activities online generate data: visiting web sites, making purchases, placing phone calls, using social media etc. all generate data that is stored.

This data can be used to profile our activities and behaviours, to predict what we may do, and to suggest products or services that we may find attractive (based on previous behaviour). The same data can also be used to steal our identity with a wide variety of potentially serious implications for our credit worthiness and financial stability.

There are U.K. and European laws that prevent unauthorised access and use of personal data. The Data Protection Act in the U.K. offers significant protection against the unauthorised use of personal information.

UK businesses have to process data in accordance with the eight principals set out in the data protection act. These are that data must:

• be processed fairly and lawfully;
• be processed for specified lawful purposes;
• be adequate, relevant and not excessive;
• be accurate and up to date;
• not be kept longer than is necessary;
• be processed in accordance with the rights of individuals;
• be kept secure;
• not be transferred outside the European Economic Area with adequate protection.

These principals mean that, for example, credit card data held by businesses must be encrypted and that personal data can't be transfered outside of the EEA without contractual or legal protections being in place. More information on the requirements of the data protection act can be found on the ICO web site (https://ico.org.uk)
7.4 Increasing use and power of intelligent systems

Learning objective

By the end of this section you will be able to:

- demonstrate some understanding of increasing use and power of intelligent systems.

From self-driving vehicles to rescue robots to domestic appliances to digital assistants, intelligent systems are becoming more commonplace in our society. These systems gather data from their environment and increasingly, without human interaction, automatically act on that information, enabling benefits such as faster responses in crisis situations, more efficient resource management and safer performance of difficult tasks.

As processor power, data storage and connectivity have all increased it has become possible to process increasing amounts of data in reduced amounts of time. This has enabled intelligent systems to be developed which demonstrate the properties of human intelligence and understanding.

There are a wide variety of intelligent systems and intelligent agents available.

Activity: Mobile technology

Research current mobile technology, which offer personal assistant capabilities, such as Google Now, Microsoft Cortana and Apple Siri.

Compile a short written report on your findings.

7.5 Energy (data centres, low carbon equipment)

Learning objective

By the end of this section you will be able to:

- outline the energy usage and energy saving issues which affect data storage and data centres;
- demonstrate an awareness of low carbon equipment.

With our increasing reliance on digital systems for business and personal data, entertainment and social media, there has been a significant increase in the use of energy to power the servers and infrastructure to provide the online services that we use every day. These services are often clustered together in data centres.

Data centres

These data centres generate large amounts of heat, so require specialised cooling systems, use large amounts of energy to power the computer systems and cooling
systems and generate significant volumes of carbon emissions. Companies such as Amazon, Google, Microsoft, Facebook and Apple all operate large data centres across the world and many provide data services to other businesses to generate income. For example: Netflix, an online video streaming service, is hosted entirely on Amazon’s Web Services platform based in Amazon data centres.

These data centres are adopting new technologies to reduce energy usage and to reduce carbon emissions as a result of their operation. The European Union has established a Code of Conduct for Energy Efficiency in Data Centres (http://iet.jrc.ec.europa.eu/energyefficiency/ict-codes-conduct/data-centres-energy-efficiency) which details a number of commitments for data centre operators and service suppliers.

**Video: Google**

Watch the video Inside a Google Data Centre:

https://www.youtube.com/watch?v=XZmGGAbHqa0

**Video: Facebook**

Watch the video on the Facebook Open Compute project which is a leader in energy efficiency and low carbon equipment.

https://www.facebook.com/Engineering/videos/10150555918930484/

**Low Carbon Equipment**

Innovations in the provision of computing equipment has been a key part in reducing the carbon impact of technology.

European regulations, known as RoHS, REACH, and WEEE, apply to computing products: especially their manufacture and disposal. These regulations ensure that hazardous materials or chemicals that can be included in devices are disposed of properly, and that electronics are effectively reused, recycled and recovered. Since 2009 there has also been a European directive requiring the eco-design of energy using products.

The most prominent feature for the average consumer is the EU energy labelling that appears on most consumer white goods now, ranking their energy consumption from A to G. Following advancements in energy efficiency, throughout the industry, new categories of A+, A++ and A+++ had to be introduced in 2010.
The final driver towards green IT is the way that purchasing decisions have been influenced. Companies and consumers purchasing computing devices now consider the carbon footprint of the technology and its environmental impact. This is particularly pronounced with public procurement, with European and government policies requiring public sector organisations to purchase more energy efficient products and services. There is a similar trend with companies concerned about their carbon impact, and also with the cost of their energy use following steep energy price rises in recent years.

7.6 Online marketing (web, e-mail, text)

**Learning objective**

By the end of this section you will be able to:

- explain how advertisers use the web to target individual users with advertisements.

Online marketing is an umbrella term for the use of targeted, measurable and interactive marketing of products and services using digital technologies to generate sales. It typically uses web based and mobile based technologies but also has applications across gaming consoles and other media.

Typically, users interact with advertisements or may sign-up for free trials of services or membership of web sites. This provides marketers with contact information for direct
marketing via e-mail/instant message or SMS.

7.6.1 Cookie tracking

Cookie tracking, cookie profiling or web profiling is the use of persistent or permanent cookies to track a user’s overall activity online. This tracking does not just happen when you are on a particular site, but it occurs the whole time you are browsing. This kind of profiling activity is often done by marketers who buy advertising rights on thousands of popular websites in order to collect and collate cookie information and create a single "profile" of a user.

Internet advertising, as it is called, targets potential customers based on the way they browse the Internet. This is the very reason why most websites flash banner ads on their pages. This matter may not be a big deal for some, but others take their privacy seriously and are uneasy about being “followed around” and profiled.

Figure 7.2: Cookie tracking

6. FINGERPRINTING PROVIDER
The fingerprinting provider checks the information against its database and returns to the ad broker the unique number or code it uses to identify you.

5. PRINT INFO
The ad broker in turn relays this information to its fingerprinting provider.

2. GET AD
As your browser loads that page, it also requests an advertisement from an online ad broker.

4. USER IDENTIFICATION
The fingerprint script reports this information back to the ad broker’s server.

3. FINGERPRINT SCRIPT
The ad broker responds by sending the advertisement to your browser along with a script that quietly gathers sufficient information about your computer and browser to identify your uniquely.

7. GET TRACKED
The ad broker can then use this identification to keep track of your online activities anytime you visit a site that displays one of its advertisements.
Cookie profiling is the only way for marketers to target potential customers and obtain a possible product purchase from them. By knowing a user's browsing habits, including sites visited, age, marital status, and political and religious affiliations, they can show him or her advertisements that are appealing, advertisements that he or she will care to patronize. This is a certain way for marketers to increase their profit by widening their customer base.

For a cookie guide please see the following video: https://www.youtube.com/watch?v=coWuhy3CjVE

7.7 Analytics

Learning objective

By the end of this section you will be able to:

• explain how analytics can use large sets of data to display insights on user or other behaviour.

Analytics are a tool used by web services companies, individuals and businesses to understand the behaviour of users. Analytics are used to discover patterns of behaviour, such as to identify frequently visited pages/content. Analytics are based on large sets of data gathered from operations, for example, data on the pages visited on a web site, the number of visits, the time a user spent on each page etc. This data can then be displayed in a visual manner, using analytics, to identify useful business insights.
TOPIC 7. IMPLICATIONS: LEGAL, ETHICAL, ENVIRONMENTAL, ECONOMIC AND SOCIAL

Activity: Web Analytics

Research the use of Web Analytics to generate information about site visitors, location and type. A good place to start is:

http://www.google.com/intl/en_uk/analytics/
7.8 Cyber security risks and precautions

Learning objective
By the end of this section you will be able to:

- demonstrate an awareness of cyber security issues.

Cyber security is a very important part of our modern lives. More and more of our daily activity depends on the data services that we use. If these services are attacked as a result of cyber crime that may have a significant impact on us.

Cyber risks can be grouped into three main areas.

Cyber crime
Conducted by individuals working alone, or in organised groups, intent on extracting money, data or causing disruption, cyber crime can take many forms, including the acquisition of credit/debit card data and intellectual property, and impairing the operations of a website or service.

Cyber war
A nation state conducting sabotage and espionage against another nation in order to cause disruption or to extract data.

Cyber terror
An organisation, working independently of a nation state, conducting terrorist activities through the medium of cyberspace.

Organisations that have to consider measures against cyber war or cyber terror include governments, those within the critical national infrastructure, and very high-profile institutions. It is unlikely that most organisations and individuals will face the threat of cyber war or cyber terror.
7.8.1 Cyber crime

Cyber crime is a seeing huge growth as we depend more and more on digital systems. Typical cyber crime threats are from different kinds of malware some of which are shown in the table below:

<table>
<thead>
<tr>
<th>Type</th>
<th>Aim</th>
<th>Technique</th>
</tr>
</thead>
<tbody>
<tr>
<td>Viruses</td>
<td>Gain access to, steal, modify and/or corrupt information and files from a targeted computer system.</td>
<td>A small piece of software program that can replicate itself and spread from one computer to another by attaching itself to another computer file.</td>
</tr>
<tr>
<td>Worms</td>
<td>By exploiting weaknesses in operating systems, worms seek to damage networks and often deliver payloads which allow remote control of the infected computer.</td>
<td>Worms are self-replicating and do not require a program to attach themselves to. Worms continually look for vulnerabilities and report back to the worm author when weaknesses are discovered.</td>
</tr>
<tr>
<td>Spyware/Adware</td>
<td>To take control of your computer and/or to collect personal information without your knowledge.</td>
<td>By opening attachments, clicking links or downloading infected software, spyware/adware is installed on your computer.</td>
</tr>
<tr>
<td>Trojans</td>
<td>To create a 'backdoor' on your computer by which information can be stolen and damage caused.</td>
<td>A software program appears to perform one function (for example, virus removal) but actually acts as something else</td>
</tr>
</tbody>
</table>

In addition to software attacks of the types given above, cyber criminals use a range of other methods to infect computers with malware or to gather/steal data.

<table>
<thead>
<tr>
<th>Attack</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Phishing</td>
<td>An attempt to acquire users' information by masquerading as a legitimate entity. Examples include spoof emails and websites. See 'social engineering' below.</td>
</tr>
<tr>
<td>Pharming</td>
<td>An attack to redirect a website's traffic to a different, fake website, where the individuals' information is then compromised. See 'social engineering' below.</td>
</tr>
<tr>
<td>Drive-by</td>
<td>Opportunistic attacks against specific weaknesses within a system.</td>
</tr>
<tr>
<td>MITM</td>
<td>'Man in the middle attack' where a middleman impersonates each endpoint and is thus able to manipulate both victims.</td>
</tr>
<tr>
<td>Social engineering</td>
<td>Exploiting the weakness of the individual by making them click malicious links, or by physically gaining access to a computer through deception. Phishing and phishing are examples of social engineering.</td>
</tr>
</tbody>
</table>
7.8.2 Cyber prevention

The UK Government has estimated the cost of cybercrime to the UK to be in excess of £27bn per annum, and the main loser - at a total estimated cost of £21bn - is UK business. In January 2015 GCHQ stated that 8 in every ten of the biggest British companies have suffered a serious cyber attack, costing the UK economy tens of millions of pounds annually.

As individuals and organisations, we can prevent cybercrime using a number of tools/approaches.

- Use regularly updated anti-virus/anti-malware software to prevent, detect and remove infections.
- Use secure passwords for the sites and services we make use of.
- Use encryption on our devices to prevent access from malware or if the device is compromised.
- Use biometrics (such as fingerprint or face recognition) to prevent access to our technology devices.
- Always ensure that SSL (Secure Socket Layer) is enabled when carrying out transactions online.
- Use a firewall to protect your home network from unauthorised incoming and outgoing data traffic.
- Educating users / employees on the dangers and how to recognise and protect against attack.

In recent years, cyber security has moved more towards cyber resilience for small, medium and larger organisations. Cyber resilience combines the approaches used to deal with and prevent attacks with the techniques used to ensure that an organisation can continue to function effectively even if it has suffered from an attack.

Activity: Cyber security

Visit https://www.cyberstreetwise.com/ to research how you can ensure your own cyber security.

7.9 Tracking, privacy, online safety

Learning objective

By the end of this section you will be able to:

- highlight issues relating to tracking privacy and online safety including the "Right to be forgotten" ruling.
The ability of advertisers to track our online activity is a potential threat to our privacy and online safety. Advertisers, internet service providers and social media companies use our online behaviours to profile us as individual users and then target marketing materials: adverts, offers etc. to match our preferences.

This has become an increasing concern for Internet users. Recently web browsers and mobile services have become available to block third-party cookies which track users and to disable advertisements which include tracking code.

7.9.1 "Right to be forgotten"

The Right to be forgotten affects all of us. Given the nature of the Internet and World Wide Web, information from our past which would not normally be relevant, can still be located many years after it was published. This can have an adverse affect on individuals. The "Right to be forgotten" ruling forces Internet search providers to remove search results which point to articles/information which relate to personal data about an individual which should no longer appear.

Activity: Right to be forgotten

Read through the "Right to be forgotten" information sheet:


7.10 Social media

By the end of this section you will be able to:

• outline key legal and ethical issues in relation to the use of social media.

Social media

Social media refers to a range of online communications tools focused on communities of users and their interactions. Common platforms such as Facebook, Twitter, Youtube, Google+, Instagram, Tumblr and many more encourage users to share images, videos and text based content and response to the content shared by others.

These businesses connect many millions (and in Facebook's case billions) of people together and make their profits from advertising and selling data about their users.

However, the profits from these business can also be used for positive social change. For example Facebook is a pioneer, via internet.org (http://internet.org) of bringing Internet access to communities in the developing world.

Legal issues

There are a number of issues that can apply to social media. A small selection are:
• Libel - if you make a defamatory allegation about someone you can be sued for libel. It is the same as publishing a false and damaging report in a newspaper.

• Malicious Communications - if you make "grossly offensive" remarks on social media you can be charged under the Communications Act 1988.

• Breach of copyright - posting images that you don't own the copyright to can result in action.

**Cyberbullying and trolling**

The ease of communication that comes with social networking sites, email and mobile phones means that it is can also be easier to be unpleasant to other people. Cyberbullying involves abuse of another person using threats, insults and hurtful remarks and messages over the internet. There have been numerous reports of people who have been driven to suicide by persistent cyberbullying. Internet trolls post messages and comments that try to evoke an emotional response from other people.

### 7.11 Implications of “Big data”

**Learning objective**

By the end of this section you will be able to:

• explain how developments in the use of Big data can provide significant insights or discoveries relating to human or other behaviour.

The idea behind the phrase ‘**Big data**’ is that many things we do in our lives leave a digital trace (or data), which we (and others) can use and analyse. The advances in capturing and analysing Big data allow us to decode human DNA in minutes, find cures for cancer, accurately predict human behaviour, pinpoint marketing efforts, prevent diseases and much more. However, the same technology can be used to profile individuals, identify patterns in behaviour and predict how individuals or groups are likely to behave.

Big data refers to our ability to collect and analyse the vast amounts of data we are now generating in the world. The ability to harness this large amounts of data is completely transforming our ability to understand the world and everything within it.

Some examples of how Big data is being used are:

• Supermarkets are combining their loyalty card data with social media information to detect and leverage changing buying patterns. For example, it is easy for retailers to predict that a woman is pregnant simply based on the changing buying patterns. This allows them to target pregnant women with promotions for baby related goods.

• Google's self-driving car is analysing a gigantic amount of data from sensor and cameras in real time to stay on the road safely.

• A hospital unit that looks after premature and sick babies is generating a live steam
of every heartbeat. It then analyses the data to identify patterns. Based on the analysis the system can now detect infections 24hrs before the baby would show any visible symptoms, which allows early intervention and treatment.

- The Security services are combining data from social media, CCTV cameras, phone calls and texts to track down criminals and predict the next terrorist attack.

**Activity: Big data**

Research two additional uses of Big data and record your findings. Ensure that you consider the legal, ethical, environmental, economic and social implications that may apply.

..........................................

### 7.12 Learning points

**Summary**

You should know be able to describe and consider the implications of:

- Intellectual property rights relating to the development of technologies;
- Social control via data and media;
- Issues relating to the storage of large amounts of user data;
- Intelligent systems;
- Energy consumption;
- Online marketing including (web, email and text/SMS/IM);
- Data analytics;
- Tracking, privacy and online safety;
- Social media/networking;
- Big data.
7.13 End of topic test

End of topic test

Q6: Which two rights do you automatically acquire for intellectual property?

a) Patent and trademark.
b) Patent and design right.
c) Design right and trademark.
d) Design right and copyright.

Q7: Patents can be used to defend a claim about a product. How are patent claims possibly progressed?

a) Through mediation and patent sale.
b) Through mediation, licencing or the courts.
c) Through development of alternative patents.
d) Through the courts and court of appeal.

Q8: Self-driving vehicles are an example of:

a) intelligent systems.
b) remote control robotics.
c) voice control systems.
d) transit control systems.

Q9: Data centres are reducing their energy usage through the:

a) use of cooling systems.
b) development of easy to upgrade servers.
c) use of low energy components.
d) use of robotics.

Q10: Web profiling is the process of:

a) tracking a user’s behaviour across a number of web sites to gather business data.
b) tracking a user login at a specific web site.
c) issuing a Do Not Track message from a browser.
d) sharing security details with other users.
Topic 8

End of Unit 2 test
End of Unit 2 test

Q1: Agile approaches to software development can respond quickly to (choose all that apply):
   a) changes to the functions required within the software.
   b) demands created by a change in the targeted user group.
   c) changes to the overall development budget.
   d) changes to related systems which integrate with the software being developed.

Q2: Within Scrum, a release is produced when:
   a) all of the product backlog items have been completed.
   b) a design sprint is completed.
   c) the product owner agrees the product backlog.
   d) a user guide is created.

Q3: A business case presents:
   a) the user requirements for a development project.
   b) the social and economic feasibility of project.
   c) a summary of the benefits of a project to the business.
   d) a technical outline of a project.

Q4: The role of the ScrumMaster in a Scrum project is to:
   a) ensure that the Scrum team make best use of Scrum methods during the project.
   b) coach the Scrum team on user interface design.
   c) train the product owner to ensure he/she is aware of how Scrum operates.
   d) host the daily 'town hall' meetings.

Q5: In Scrum, each design sprint consists of three phases:
   a) Analyse, Design, Implement
   b) Version, Control, Evaluate
   c) Design, Build, Test
   d) Design, Implement, Evaluate
Q6: The waterfall model is not appropriate for many medium to large scale development projects because:

a) all the requirements for the project need to be known at the start of the development or significant costs will be incurred for revisiting earlier phases.
b) all the resources to deliver the project must be available for the duration of the project which may not be possible.
c) it is not possible to allocate development work across larger teams with this method.
d) systems analysts are unable to work collectively on medium/large scale projects.

Q7: The project management triangle consists of four constraints. These are:

a) scope, budget, requirements and testing.
b) scope, time, user interface and iteration.
c) scope, cost, function and time.
d) scope, time, cost and quality.

Q8: A project manager defines the tasks to be completed in the work breakdown structure and estimates the________ required to complete each task.

a) time (or cost)
b) resources (or cost)
c) effort (or cost)
d) quality

Q9: The project baseline is used:

a) as the definitive version of the plan against which progress is measured.
b) to define the start date of the project.
c) to plan resource allocation.
d) as the means of planning the work breakdown and team operations.

Q10: A critical path is calculated by adding up the total:

a) person hours/days for the whole project.
b) person hours/days of float for the whole project.
c) person hours/days for tasks which have no float which run beginning to end throughout the project timeline.
d) person/hours for all parallel tasks which run beginning to end throughout the project timeline.
Q11: When processing an HTML form the POST method is slightly more secure than
the GET method because it:

a) encrypts the data sent to the server.
b) sends the data via an HTTP tunnel rather than within the URL.
c) sends the data via HTTP headers rather than within the URL.
d) routes data through a number of servers for security.

Q12: The action attribute of an HTML form element specifies:

a) the script which will process the data.
b) the name of the form.
c) the CSS class of the form.
d) an JavaScript action for the form.

Q13: An HTML form is being created that will require a user to enter a value in a text
field. Select the attribute of the INPUT element that will force a user to enter a value
before the form can be submitted:

a) validated.
b) regex.
c) required.
d) necessary.

Q14: Code injection is an attack which:

a) acts as a 'man-in-the-middle' collecting data between client and server.
b) uses a faulty SSL certificate to allow attacks on the server.
c) exploits poorly secured form variables to execute code on the server.
d) uses CSS selectors to execute JavaScript on the client.

Q15: A pattern defines a rule for the client side validation of data in a form field. Patterns
as specified using:

a) regular expressions.
b) Boolean algebra.
c) JavaScript.
d) fuzzy logic.
Q16: Which one of the following techniques is used to represent the motivations, goals and frustrations of users so that the development team can understand their needs?

a) Wireframe.
b) Use case.
c) Persona.
d) User scenario.

Q17: A paper prototype is:

a) sketch of the user interface which users can comment on.
b) version of the application drawn on paper with which users can interact.
c) pseudocode design.
d) template of buttons, menus, and text boxes that users can discuss.

Q18: A design team consists of multi-disciplinary skills and perspectives because the:

a) team should only be focused on coding rather than user experience design.
b) team is directed and managed by the product owner to complete the application.
c) needs of the project require only user experience experts to design the application.
d) team needs to draw on the views of number of areas: graphic designers, programmers, users experience experts, project managers and others to be successful.

Q19: Which SQL statement is used to extract data from a database?

a) SELECT
b) GET
c) EXTRACT
d) OPEN

Q20: Which SQL statement is used to insert new data in a database?

a) ADD RECORD
b) ADD NEW
c) INSERT INTO
d) INSERT NEW

Q21: With SQL, how do you select all the columns from a table named "developer"?

a) SELECT developer
b) SELECT [all] FROM developer
c) SELECT * .developer
d) SELECT FROM developer
Q22: With SQL, how do you select all the records from a table named "developer" where the value of the column "firstname" starts with an "a"?

a) SELECT * FROM developer WHERE firstname LIKE '%a'
b) SELECT * FROM developer WHERE firstname = 'a'
c) SELECT * FROM developer WHERE firstname LIKE 'a%'
d) SELECT * FROM developer WHERE firstname = '%a%'

Q23: With SQL, how can you return all the records from a table named "developer" sorted descending by "firstname"?

a) SELECT * FROM developer ORDER BY firstname DESC
b) SELECT * FROM developer ORDER firstname DESC
c) SELECT * FROM developer SORT BY firstname DESC
d) SELECT * FROM developer SORT 'firstname' DESC

Q24: With SQL, how can you delete the records where the "firstname" is "Peter" in the developer table?

a) DELETE firstname = 'Peter' FROM developer
b) DELETE ROW firstname = 'Peter' FROM developer
c) DELETE FROM developer WHERE firstname = 'Peter'
d) DELETE 'Peter' FROM developer

Q25: Which of the following is the correct PHP syntax to connect to a database?

a) $mysqli = connect ("server", "username", "password") for "database";
b) $mysqli = new mysqli("server", "username", "password","database");
c) $mysqli = ("server", "username", "password","database");
d) $mysqli = new connection("server", "username", "password","database");

Q26: Which of the following PHP code is required to execute a query stored in a variable $query?

a) $result = $mysqli->query($query);
b) $result->fetch_object($query);
c) $result = $mysqli->prepare($query);
d) $result->mysqli->error($query);

Q27: Relating to the use of technology, cyberbullying and trolling are examples of what?

a) Social issues.
b) Legal issues.
c) Economic issues.
d) Professional issues.
Q28: Which of the following is not related to the use of ‘Big data’?

a) Processing supermarket sales across multiple stores and online.
b) Analysing stock market data.
c) Storing a word processing file.
d) Booking a flight online.

Q29: Viruses, phishing and spyware/adware are examples of?

a) Cyber crime.
b) Trolling.
c) Hacking.
d) Unauthorised access.

Q30: The "Right to be forgotten" requires that Internet search engines:

a) remove all search results that match a specific user's name.
b) remove search results that point to information that is no longer relevant from our past.
c) prevent the use of a specific name in searches.
d) remove all data on a specific person from their database.
Glossary

**AMP**
a combination operating system, Apache web sever, MySQL and PHP commonly used to provide web services.

**Acceptance criteria**
the agreed sign-off criteria for an item in the product backlog. When the item meets this criteria then it is delivered and marked as complete in the product backlog.

**Activity**
an item from the project plan that is carried out using an amount of effort.

**Agile**
an approach to software development that is flexible and focuses on working software over documentation and complete sets of requirements.

**Agile model**
Agile software development is a group of software development methods in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development, early delivery, continuous improvement, and encourages rapid and flexible response to change.

**Analytics**
is information resulting from the systematic analysis of data or statistics.

**Base line plan**
is the agreed final plan before moving into the development phase. The progress of the project is measured against the base line plan.

**Benchmark usability tests**
are different from conventional usability tests. Participants carry out testing from their home or place of work. Rather than providing feedback to a team member, the user works with an automated system that asks his/her to perform tasks, measures his/her performance and asks for subjective comments.

**Big data**
is extremely large data sets that may be analysed computationally to reveal patterns, trends, and associations, especially relating to human behaviour and interaction.

**Budget**
the amount of money available to complete a project.

**Business case**
is a justification for a proposed project on the basis of its expected commercial benefit.

**Business need**
a process or work flow which is required by a business/organisation.
CLI
Command Line Interface.

Client
is the organisation responsible for requesting the software development project and may for the delivery of the project.

Client-server Model
The client-server model of computing is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients.

Code injection
the exploitation of a computer bug that is caused by processing invalid data. Injection is used by an attacker to introduce (or "inject") code into a vulnerable computer program and change the course of execution.

Constraints
the variables that limit the activities of the project: scope, time, cost and quality.

Context of use
is the actual conditions under which a given software product/application is used, or will be used in a normal day to day working situation.

Cookies
an HTTP Cookie is a small piece of data sent from a website and stored in a user's web browser while the user is browsing that website.

Cookie tracking/cookie profiling
is a process where a program or data is used by a third party to identify a user or computer. The most commonly used tool for this is a tracking cookie - a small piece of data that identifies a certain user or a certain computer, with the help of a web browser configured to store cookies.

Cost
the financial value of a resource used to deliver part of a project (may also apply to the whole project).

Critical path
the sequence of activities determining the minimum time needed to complete a project.

CSS
Cascading Style Sheets, a style sheet language used to describe the look and formatting of a document written in a markup language. It is currently on version 3 of the CSS standard (CSS3).

Cyber crime
are criminal activities carried out by means of computers or the Internet.
**Cyber resilience**

is defined as the ability of systems and organisations to withstand cyber events, measured by the combination of mean time to failure and mean time to recovery.

**Cyber security**

is the body of technologies, processes and practices designed to protect networks, computers, programs and data from attack, damage or unauthorized access.

**Database Server**

a computer running a server based version of a database management system which can respond to requests from clients.

**Data centre**

is a large group of networked computer servers typically used by organizations for the remote storage, processing, or distribution of large amounts of data.

**Data dictionary**

is a set of information describing the contents, format, and structure of a database and the relationship between its elements.

**Data modelling**

is a process used to understand the data involved in business processes.

**Data Protection Act**

the Data Protection Act 1998 (DPA 1998) is an act of the United Kingdom (UK) Parliament defining the ways in which information about living people may be legally used and handled. The main intent is to protect individuals against misuse or abuse of information about them.

**Deliverable**

an artifact (a thing) which is handed over to the client. This includes documentation, compiled code and evidence of testing.

**Design right**

the shape and the configuration of objects are automatically protected by design right in the UK.

**Effort**

physical or mental activity required to achieve part of the project objectives.

**Entity relationship diagram**

(ERD) is a data modelling technique that graphically illustrates a database’s entities and the relationships between those entities.

**Environment**

is the situation in which an application or software product is used/will be used.

**ERD**

Entity Relationship Diagram - is a data modelling technique that graphically illustrates a database’s entities and the relationships between those entities.
Field studies
research into the success of an application or to verify the needs of a target groups which is carried out “in the field” - in real life settings with real users.

Float
the amount of time that an activity can be delayed without causing the delay to subsequent activities or the project overall.

Focus group
a group of people assembled to participate in a discussion about a product before it is developed/launched.

Functional requirements
define specific behaviour or functions within a system or product.

Gantt chart
a chart in which a series of horizontal lines shows the amount of work done or production completed in certain periods of time in relation to the amount planned for those periods.

GET method
a means of passing data to a script via the URL.

High-fidelity prototypes
from a user testing point of view, a high-fidelity prototype is close enough to a final product to be able to examine usability questions in detail and make strong conclusions about how behaviour will relate to use of the final product.

HTML
Hypertext Markup Language, a standardized system for tagging text most commonly used to define the structure and content of a document.

Human computer interaction (HCI)
relates to the design and use of computer technology, focusing particularly on the interfaces between people (users) and computers.

IDE
Integrated Development Environment, a software application that provides facilities to computer programmers for software development. An IDE normally consists of a source code editor, build automation tools and a debugger.

Integration testing
is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

Intellectual property protection
the protections that are available for intellectual property: patents, design rights and trademarks.
Intellectual property rights
are the rights given to persons over the creations from their minds.

Intelligent Agent
an Intelligent Agent (IA) is an autonomous entity which observes through sensors
and acts upon an environment using actuators (i.e. it is an agent) and directs its
activity towards achieving goals.

Intelligent systems
a system with artificial intelligence.

Interviews
a face to face meeting of between the developers and one or more target users.

ISO 9241-210 Human-centred design for interactive systems
ergonomics of human-system interaction, provides guidance on human-system
interaction throughout the life cycle of interactive systems. An international
standard detailing the development of interactive systems.

Iterative
a repeating process.

Iterative prototyping
is a design methodology based on a cyclic process of prototyping, testing,
analysing, and refining a product or process.

JavaScript
an object-oriented computer programming language commonly used to create
interactive effects within web browsers.

Licence agreement
is a legal contract between two parties, known as the licensor and the licensee. In
a typical licensing agreement, the licensor grants the licensee the right to produce
and sell goods, apply a brand name or trademark, or use patented technology
owned by the licensor.

Low-fidelity prototypes
an early prototype that is sketchy and incomplete, that has some characteristics of
the target product but is otherwise simple, usually in order to quickly produce the
prototype and test broad concepts.

Mediation
is an intervention in a dispute in order to resolve it

Milestones
a significant stage or event in the development of a project.

MySQL
a popular open source database.
Objectives
what the project is intended to achieve.

Operational requirements
are the basis for system requirements.

Paper prototyping
in human-computer interaction, paper prototyping is a widely used method in the user-centered design process, a process that helps developers to create software that meets the user's expectations and needs—in this case, especially for designing and testing user interfaces.

Patent
a government authority or licence conferring a right or title for a set period, especially the sole right to exclude others from making, using, or selling an invention.

Performance and satisfaction criteria
the criteria used by the client/end-user which detail what the applications/software product must do (and how well it should do it) in order to be accepted.

PHP
a server-side scripting language designed for web development but also used as a general-purpose programming language.

PhpMyAdmin
a tool written in PHP intended to handle the administration of MySQL over the WWW.

Primary Key
is a single column or combination of columns that uniquely identifies each row in a table.

Product backlog
is the single most important artifact in the Scrum approach to agile development. The product backlog is, in essence, an incredibly detailed analysis document, which outlines every requirement for a system, project, or product.

Product owner
is the member of the team responsible for defining and prioritizing the Product Backlog so as to streamline the execution of program priorities, while maintaining conceptual and technical integrity of the features or components the team is responsible for.

Programmer
a person who writes computer programs.

Project initialisation documentation
bundles the information which was gathered at the start of the project.
Project management

the application of processes, methods, knowledge, skills and experience to achieve the project objectives.

Project manager

the person in overall charge of the planning and execution of a particular project.

Project plan

a formal, approved document used to guide both project execution and project control.

Project proposal

is a detailed description of a series of activities aimed at solving a certain problem.

Project sponsor

a senior management role that typically involves approving or supporting the allocation of resources for a venture, defining its goals and assessing the venture’s eventual success.

Project timeline

is a tool to plan and organise the tasks within a project.

Prototyping session report

a formal report from a session with users working with prototypes of any time.

Quality

the standard of something as measured against other things of a similar kind.

Registered design

a design refers to the features of a shape, configuration, pattern or ornament applied to an article by any industrial process. If you register a design, you will be protecting the external appearance of the article. Registered Designs are used primarily to protect designs for industrial use.

Regular expression

a sequence of symbols and characters which define a string or pattern to be searched for within a longer piece of text.

Release

a package of software which is released to the customer.

Requirements specification

is a description of a software system to be developed, typically it will include full details of the product to be development.

Right to be forgotten

the European Union passed its so-called "Right to be forgotten" law, which compels search engines like Google to remove search listings about people if they get the appropriate court orders or provide information to service prod.
Scope
the detail of what a project is to achieve. Changes to the scope of the project have a direct impact on the time and cost of the project.

Script
Program code which can be executed.

Scrum
is an iterative and incremental agile software development framework for managing product development.

ScrumMaster
is the facilitator for a product development team that uses scrum - a development methodology that allows a team to self-organize and make changes quickly. The scrum master manages the process for how information is exchanged.

Scrum team
is a collection of individuals working together to deliver the requested and committed product increments.

Server
a computer or computer program which manages access to a centralised resource or service in a network.

Social media
websites and applications that enable users to create and share content or to participate in social networking.

Software development
is the computer programming, documenting, testing, and bug fixing involved in creating and maintaining applications.

Sprint
a period of development when a fixed set of product items (the sprint backlog) is developed.

Sprint backlog
the list of features to be developed as part of a sprint in Scrum.

Sprint planning meeting
a meeting prior to the start of the sprint where the team chooses which items from the product backlog to develop as part of the sprint.

Surveys
a series of questions used to evaluation or reflect on an application.

Systems analyst
a person who analyses a complex process or operation in order to improve its efficiency, especially by applying a computer system.
Task analysis
is the process of learning about ordinary users by observing them in action to understand in detail how they perform their tasks and achieve their intended goals.

Tasks
what the user has to achieve with the applications/software product.

Technical documentation
detailed documentation about the technical requirements for the software, details of installation and system requirements.

Test methods
are approved procedures for ensure that an application meets it’s requirements.

Timescale
a period of time to complete an activity.

Trademark
a symbol, word, or words legally registered or established by use as representing a company or product.

Unit testing
is a software development process in which the smallest testable parts of an application, called units, are individually and independently scrutinized for proper operation.

URL
Uniform Resource Locator, a reference (an address) to a resource on the Internet. A URL has two main components: Protocol identifier: For the URL http://scholar.hw.ac.uk/, the protocol identifier is http. Resource name: For the URL http://scholar.hw.ac.uk/, the resource name is scholar.hw.ac.uk/ - the resource can include the server, port, path and filename of the resource plus any internal anchor and/or data.

Use cases
is an approach used in system analysis to identify, clarify, and organize system requirements. Use cases are made up of sets of possible sequences of interactions between systems and users in a particular environment and related to a particular goal.

User
the end user of an application.

User documentation
a technical communication document intended to give assistance to people using a particular system.

User experience
the overall experience of a person using a product such as a website or computer application, especially in terms of how easy or pleasing it is to use.
User experience (UX)
the overall experience of a person using an application, especially in terms of how easy or pleasing it is to use.

User persona
is a representation of the goals and behavior of a hypothesized group of users. In most cases, personas are created from data collected from interviews with users.

User profile
is a simple representation of a user. It is an initial step in the creation of a user persona.

User scenarios
describe the stories and context behind why a specific user or user group comes to your site or use your application. They note the goals and questions to be achieved and sometimes define the possibilities of how the user(s) can achieve them on the site.

User stories
are short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system.

Validation testing
is an independent procedure that is used for checking that a product, service, or system meets requirements and specifications and that it fulfills its intended purpose.

Waterfall model
is a sequential design process, used in software development processes, in which progress is seen as flowing steadily downwards (like a waterfall) through the phases of Conception, Initiation, Analysis, Design, Construction, Testing, Production/Implementation and Maintenance.

Web profiling
businesses can profile Web users based on the sites they travel to, how long they stay on each page, and what they do on these sites. This information can then be used to target the users with advertising.

Web server
a computer system that processes requests via HTTP, the basic network protocol used to distribute information on the World Wide Web.

WEEE

Weekly backlog review
a regular process of reviewing the product backlog and re-prioritising if required.
Wireframes

an image or set of images which displays the functional elements of a website or application, typically used for planning interactions and interface elements.

Work breakdown structure

is a hierarchical and incremental decomposition of the project into phases, deliverables and work packages. It is hierarchical in layout and can be drawn as a diagram.

XSS

Cross-site scripting, a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side script into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy.
Answers to questions and activities

1 Project design process

Quiz: Revision (page 3)

Q1:  b) Usability testing
Q2:  d) a questionnaire
Q3:  c) it is able to deal with all errors experienced during execution of the program.
Q4:  b) Wire-framing
Q5:
   a) sensory/vibration feedback when option selected
   b) audio description
   c) high contrast display
Q6:  c) meet the functional needs of the program and be accessible for the target user.
Q7:  a) that the interface keeps interactions with the user to a minimum to meet the functional requirements of the program.

Activity: Roles in the Agile model (page 8)

Q8:

<table>
<thead>
<tr>
<th>Role</th>
<th>Responsibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product owner</td>
<td>Selects items from the product backlog for the sprint.</td>
</tr>
<tr>
<td></td>
<td>Represents the views of the users to the developers.</td>
</tr>
<tr>
<td></td>
<td>Presents a vision for the project.</td>
</tr>
<tr>
<td>ScrumMaster</td>
<td>Encourages the use of agile methods.</td>
</tr>
<tr>
<td></td>
<td>Ensures the developers do not over commit to work during a sprint.</td>
</tr>
<tr>
<td>Scrum team member</td>
<td>Updates colleagues with his/her progress on a particular development.</td>
</tr>
<tr>
<td></td>
<td>Is part of a team of between 5 and 9 people.</td>
</tr>
</tbody>
</table>
Activity: Types of feasibility (page 14)

**Q9:**

<table>
<thead>
<tr>
<th>A project that cannot be delivered in the time available because of the complexity of the task</th>
<th>has an issue with schedule feasibility.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A project which has insufficient finance to hire the necessary staff to complete the project when required</td>
<td>has an economic feasibility issue.</td>
</tr>
<tr>
<td>A project that processes banking transactions to indicate spending habits in contravention of data protection laws</td>
<td>has an issue with legal feasibility.</td>
</tr>
<tr>
<td>A project which cannot proceed because the software to link two data sources into the project is not available</td>
<td>has an issue with technical feasibility.</td>
</tr>
</tbody>
</table>

End of topic test (page 28)

**Q10:** c) it is unsuitable for very large complex projects.

**Q11:** b) a release

**Q12:** d) Design, Build, Test.

**Q13:** a) Feasibility study

**Q14:** c) Economic, Technical

**Q15:** d) coach and encourage the scrum team to make use of the agile.

**Q16:** a) product owner to present the top items in the product backlog.
2 Project management

Quiz: Revision (page 33)

Q1: c) perfective maintenance
Q2: d) Tutorial
Q3: b) Programmer
Q4: c) Reliability

Activity: Gantt chart (page 44)

Q5: | Activity          | Effort | Predecessor | Days |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1 System investigation</td>
<td>4 days</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 Design and consultation</td>
<td>7 days</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3 Develop application code</td>
<td>4 days</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>4 Develop user experience</td>
<td>6 days</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>5 Create data sources</td>
<td>7 days</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>6 Test application</td>
<td>8 days</td>
<td>3, 4, 5</td>
<td></td>
</tr>
<tr>
<td>7 Submit application to app store</td>
<td>3 days</td>
<td>6</td>
<td></td>
</tr>
</tbody>
</table>

End of topic test (page 47)

Q6: a) project objectives.
Q7: c) cost, time, scope and quality.
Q8: d) By reducing the quality of the final product or removing items from the scope.
Q9: b) effort (or cost)
Q10: c) sequentially or concurrently.
Q11: c) 30 person days
Q12: c) 2 person days
Q13: a) baseline
3 Information systems design techniques

Quiz: Revision (page 53)

Q1: d) wireframe
Q2: c) usability and accessibility
Q3: a) viewable and appropriate on any device type.
Q4: d) search engines
Q5: b) shortcut keys and text-based commands.

Activity: Development of prototypes (page 68)

Q6:

End of topic test (page 72)

Q7: b) wireframe
Q8: c) a user story
Q9: a) version of the application drawn on paper with which users can interact.
Q10: c) interviews and focus groups.
Q11: c) team needs to draw on the views of number of areas: graphic designers, programmers, users experience experts, project managers and others to be successful.
4 Web development

Quiz: Revision (page 75)

Q1: c) a class.
Q2: c) A DIV grouping element.
Q3: b) `<meta name="keywords" content="snowboarding, Scotland, mountains">`.
Q4: a) Optimization for web search.
Q5: c) a page which is constructed by a server script or application.

End of topic test (page 95)

Q6: a) a web server and a web browser making requests.
Q7: c) an HTML front end and a server side processor.
Q8: b) address of the processor script.
Q9: a) require and pattern.
Q10: c) Cross Site Scripting
Q11: b) HTML entities function
5 Database systems and SQL

Quiz: Revision (page 99)

Q1: b) name, type, size, validation and index/key status of each attribute of each entity.

Q2: d) A form containing the data from an underlying table.

Q3: c) A query is a request for data in a prescribed format meeting set criteria.

Q4: b) 35

Q5: b) A compound key is two or more columns in a table which uniquely identify each row in that table.

Activity: SQL database (page 104)

Expected answer

Activity: Create a data dictionary (page 107)

Expected answer

<table>
<thead>
<tr>
<th>Table</th>
<th>Fieldname</th>
<th>Data type</th>
<th>Size</th>
<th>Key</th>
<th>Required</th>
</tr>
</thead>
<tbody>
<tr>
<td>player</td>
<td>username</td>
<td>Text</td>
<td>35</td>
<td>PK</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>realname</td>
<td>Text</td>
<td>45</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>password</td>
<td>Text</td>
<td>45</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>email</td>
<td>Text</td>
<td>60</td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>message</td>
<td>Text</td>
<td>&gt;255</td>
<td></td>
<td>No</td>
</tr>
<tr>
<td></td>
<td>terms_and</td>
<td>Boolean</td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>_conditions</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>gamescore</td>
<td>score_id</td>
<td>Int</td>
<td></td>
<td>PK</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>username</td>
<td>Text</td>
<td>35</td>
<td>FK</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>game</td>
<td>Text</td>
<td>30</td>
<td>FK</td>
<td>Yes</td>
</tr>
<tr>
<td>game</td>
<td>score</td>
<td>Int</td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>game</td>
<td>Text</td>
<td>30</td>
<td>PK</td>
<td>Yes</td>
</tr>
<tr>
<td></td>
<td>platform</td>
<td>Text</td>
<td>30</td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>

Activity: Update queries (page 113)

Expected answer
1. 
   
   ```sql
   SET platform = 'S-Box-360'
   WHERE game = 'SuperJoe';
   ```

2. 
   
   ```sql
   UPDATE gamescore
   SET score = 132900
   WHERE username = 'peach' AND game = 'SuperJoe';
   ```

3. 
   
   ```sql
   UPDATE player
   SET password='fluffy1928'
   WHERE username='destroyer'
   ```

**Activity: Delete** (page 114)

**Expected answer**

1. 
   
   ```sql
   DELETE FROM gamescore
   WHERE scoreid = 630;
   ```

2. 
   
   ```sql
   DELETE FROM gamescore
   WHERE username = 'destroyer' AND score > 200000;
   ```

**Activity: Select statements** (page 122)

**Expected answer**

1. SELECT MAX(score) FROM gamescore;
2. SELECT MIN(score) FROM gamescore;
3. SELECT AVG(score) FROM gamescore;
4. SELECT AVG(score) FROM gamescore WHERE game = 'Massive RPG';
5. SELECT SUM(score) FROM gamescore WHERE username = 'peach';
6. SELECT COUNT(*) FROM gamescore WHERE game = 'Terra 1999'

**Activity: Select statements 2** (page 123)

**Expected answer**
1.

```sql
SELECT p.username, p.pword, MIN(gs.score)
FROM player AS p, gamescore AS gs
WHERE p.username = gs.username#
```

2.

```sql
SELECT gs.score, gs.game, p.username, p.realname
FROM player AS p, gamescore AS gs
WHERE p.username = gs.username
```

3.

```sql
SELECT g.platform, gs.game, gs.score, p.realname
FROM player AS p, gamescore AS gs, game AS g
WHERE p.username = gs.username AND gs.game = g.game
ORDER BY gs.score ASC
```

End of topic test (page 124)

Q6: a) SELECT * FROM branch WHERE city = 'London' OR city= 'Glasgow';

Q7: a) 

```sql
SELECT staffno, lname
FROM staff, branch
WHERE staff.branchno = branch.branchno
AND branch.street = '13 Main Road';
```

Q8: a) 

```sql
SELECT * FROM propertyForRent
WHERE city = 'Edinburgh';
```

Q9: d) 

```sql
SELECT fname, lname, MAX(annualsalary) FROM staff, branch
WHERE staff.branchno = branch.branchno AND city = 'Glasgow'
```

Q10: b) SELECT * FROM client WHERE maxrent < 800
6 Web and database integration

Quiz: Revision (page 129)

Q1:  c) process data at the server side.
Q2:  a) nearly all client-side validation can be circumvented.
Q3:  c) it validates data before it leaves the browser reducing the number of resubmissions for errors.
Q4:  d) pulls site data from a database to create site pages.
Q5:  c) server-side language module.

Exercise (page 160)

Expected answer

To display player:

```php
<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";
require("db_connection.php");
?
<html>
<head>
<title>Database Query Player</title>
<style>
table, td, th {border: 1px solid black;}
</style>
</head>
<body>
<?php
//set up a query to use
$query_string = "SELECT * FROM player";
//
?
</table>
</head>
</body>
```

© Heriot-Watt University
if ($result = $mysqli->query($query_string)) { //run the query

while ($row = $result->fetch_object()) {
    echo "<tr>";
    echo "<td"> $row->username "</td>";
    echo "<td"> $row->realname "</td>";
    echo "<td"> $row->pword "</td>";
    echo "<td"> $row->email "</td>";
    echo "<td"> mb_convert_encoding($row->message, "utf-8", "iso-8859-1") "</td>";
    echo "<td"> $row->terms_and_conditions "</td>";
    echo "<td"><a href=edit-data-player.php?username= . urlencode($row->username) ."Edit</a></td>";
    echo "<td><a href=delete-data-player.php?username= . urlencode($row->username) ."Delete</a></td></tr>

$result->close(); //free the $result set (clear it)
}

$mysqli->close();
?>
</tbody>
</table>
<a href="add-player-row.php">Insert a new row</a>
</body>
</html>

To edit the player values:

<?php
$host="localhost";
$username="root";
$pword="root";
$database="esports";

require("db_connection.php");
$username = filter_var($_GET['username']);
?
<html>
<head>
<title>Database Update</title>
<link rel="stylesheet" media="screen" href="styles.css">
</head>
<body>
<?php
//set up a query to use
$query_string = "SELECT * FROM player WHERE username = ",
$username . \\
";
if ($result = $mysqli->query($query_string)) {  //run the query

while ($row = $result->fetch_object()) {
  ?
  <form class="signup_form" action="update-row-player.php"
  method="post" name="signup_form">

<!-- keep track of the original primary key value to update the row -->
<input type="hidden" name="usernameid" value=""?>

<!-- collection of form associated elements goes here -->
<ul>
  <h2>Edit Data</h2>
  <span class="required_notification">* Denotes Required Field</span>
</ul>
<li>
  <label for="username">Username:</label>
  <input id="username" type="text" name="username"
  value=""?> size="40" maxlength="40"
  placeholder="jensmith72" pattern="^[A-Za-z0-9]+$" required />
  <span class="form_hint">A combination of letters and numbers only.</span>
</li>
<li>
  <label for="realname">Real Name:</label>
  <input id="realname" type="text" name="realname"
  value=""?> size="40" maxlength="40"
  placeholder="Jenny Smith" required />
  <span class="form_hint">Please enter your firstname and lastname.</span>
</li>
<li>
  <label for="email">Password:</label>
  <input id="password" type="password" name="password"
  value=""?> size="40" maxlength="60"
  placeholder="***********" required />
  <span class="form_hint">Please enter a password</span>
</li>
<li>
  <label for="email">Email:</label>
  <input id="email" type="email" name="email"
  value=""?> size="40" maxlength="60"
  placeholder="jenny.smith@example.com" required />
  <span class="form_hint">Use the format "name@domain.com"</span>
</li>

<!-- end of user ID sign up fields -->
<li>
  <label for="message">Message:</label>
  <input id="message" name="message" type="text" value=""?> size="40" maxlength="60"
  placeholder="Your message here..." required />
  <span class="form_hint">Enter your message here.</span>
</li>

To update the row:

```php
<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php");
$usernameid = filter_var($_POST["usernameid"]);
$username = filter_var($_POST["username"]);
$realname = filter_var($_POST["realname"]);
?>
```
```php
$pword = filter_var($_POST['pword']);
$email = filter_var($_POST['email']);
$message = filter_var($_POST['message']);
$terms_and_conditions = filter_var($_POST['terms_and_conditions']);

// set up a query to use
$query_string = "UPDATE player SET
    username = " . $username . " ,
    realname = " . $realname . " ,
    pword = " . $pword . " ,
    email = " . $email . " ,
    message = " . $message . " ,
    terms_and_conditions = " . $terms_and_conditions . "
    WHERE player.username = " . $usernameid . ""
    ;

if ($result = $mysqli->query($query_string)) {
    // run the query
    echo "Completed: Updated Row in Table"; // show successful output
    echo "<a href=database4.php >Back</a>"; // navigation back to records
} else {
    echo "Error: Unable to Update Row in Table<br /></br>"; // show error because it’s failed
    print_r($mysqli->error_list);
    // display a list of the MySQL errors that occurred
    echo "<a href=database4.php >Back</a>"; // navigation back to records
}

$result->close(); // free the $result set (clear it)
$mysqli->close();

?>
</tbody>
</table>
</body>
</html>

To insert player form:
<!DOCTYPE html>
<html>
<head>
<meta charset='utf-8'>
<title>Sign-up form</title>
<link rel='stylesheet' media='screen' href='styles.css'>
</head>
<body>
<!--start form for signup -->
<form class='signup_form' action='insert-row-player.php' method='get' name='signup_form'>

<!-- collection of form associated elements goes here -->
<ul>
<li>
<h2>Sign Up</h2>
<span class='required_notification'>* Denotes Required Field</span>
</li>
<li>
<label for='username'>Username:</label>
<input id='username' type='text' name='username' value=''
 size='40' maxlength='40' placeholder='jensmith72'
 pattern='^[A-Za-z0-9]+$' required />
<span class='form_hint'>A combination of letters and numbers only.</span>
</li>
<li>
<label for='realname'>Real Name:</label>
<input id='realname' type='text' name='realname' value=''
 size='40' maxlength='40' placeholder='Jenny Smith' required />
<span class='form_hint'>Please enter your firstname and lastname.</span>
</li>
<li>
<label for='email'>Password:</label>
<input id='password' type='password' name='pword' value=''
 size='40' maxlength='60' placeholder='***********' required />
<span class='form_hint'>Please enter a password</span>
</li>
<li>
<label for='email'>Email:</label>
<input id='email' type='email' name='email' value=''
 size='40' maxlength='60' placeholder='jenny.smith@example.com' required />
<span class='form_hint'>Use the format "name@domain.com"</span>
</li>

<!-- end of user ID sign up fields -->
<li>
<label for='message'>Message:</label>
<textarea id='message' name='message' cols='40' rows='6'></textarea>
<span class='form_hint'>A brief message, why you want to sign up.</span>
</li>
</ul>
</form>
</body>
</html>
To insert player row:

```php
<?php
    $host="localhost";
    $username="root";
    $password="root";
    $database="esports";

    require("db_connection.php");
    $player = filter_var($_GET['player']);
    $realname = filter_var($_GET['realname']);
    $password = filter_var($_GET['password']);
    $email = filter_var($_GET['email']);
    $message = filter_var($_GET['message']);
    $terms_and_conditions = filter_var($_GET['terms_and_conditions']);
    //store the correct value for terms and conditions

    if ($terms_and_conditions == "on") {
        $terms_and_conditions = 1;
    } else {
        $terms_and_conditions = 0;
    }

?>
```

```html
<html>
<head>
    <title>Database Insert</title>
</head>
<body>
    <?php

    //set up a query to use
```
$query_string = "INSERT INTO player VALUES (".$username ", ",".$password ", ",".$email . ",".$message . ",".$terms_and_conditions . ");

if ($result = $mysqli->query($query_string)) { //run the query
    echo "Completed: Inserted Row in Table"; //show successful output
    echo "<a href=database4.php />Back</a>"; //navigation back to records
} else {
    echo "Error: Unable to Insert Row in Table<br />";
    //show error because it's failed
    print_r($mysqli->error_list);
    //display a list of the MySQL errors that occurred
    echo "<br /><a href=database4.php />Back</a>";
    //navigation back to records
}
$result->close(); //free the $result set (clear it)
$mysqli->close();

To delete player row:

<?php
$host="localhost";
$username="root";
$password="root";
$database="esports";

require("db_connection.php");
$game = filter_var($_GET['username']);

?>
<html>
<head>
<title>Database Delete Player</title>
</head>
<body>
<?php
//set up a query to use

© HERIOT-WATT UNIVERSITY
$query_string = "DELETE FROM player WHERE username = ";
$username . ";"
if ($result = $mysqli->query($query_string)) { //run the query
    echo "Success, the row for player" . $username . ", has been deleted.<br />
    echo "<a href=database4.php >Back</a>";
    //navigation back to records
} else {
    echo "Error: Unable to Delete Row in Table<br />
    //show error because it’s failed
    print_r($mysqli->error_list);
    //display a list of the MySQL errors that occurred
    echo "<a href=database4.php >Back</a>";
    //navigation back to records
}
$result->close(); //free the $result set (clear it)
$mysqli->close();

</body>
</html>

End of topic test (page 161)

Q6:  c) Username, server, password, database.
Q7:  b) displays output.
Q8:  b) False
Q9:  a) $mysqli->query($query);  
Q10: c) filter_var
7 Implications: Legal, ethical, environmental, economic and social

Quiz: Revision (page 165)

Q1: b) staff hand all work related encryption keys to their employers.
Q2: b) new technology items are more energy efficient.
Q3: d) entire life of the item including manufacture, distribution, deployment and recycling/disposal.
Q4: c) Asymmetric encryption.
Q5: c) record security credentials entered via the keyboard.

End of topic test (page 181)

Q6: d) Design right and copyright.
Q7: b) Through mediation, licencing or the courts.
Q8: a) intelligent systems.
Q9: c) use of low energy components.
Q10: a) tracking a user's behaviour across a number of web sites to gather business data.
8 End of Unit 2 test

End of Unit 2 test (page 184)

Q1:
   a) changes to the functions required within the software.
   b) demands created by a change in the targeted user group.
   c) changes to the overall development budget.
   d) changes to related systems which integrate with the software being developed.

Q2:   b) a design sprint is completed.

Q3:   c) a summary of the benefits of a project to the business.

Q4:   a) ensure that the Scrum team make best use of Scrum methods during the project.

Q5:   c) Design, Build, Test

Q6:   a) all the requirements for the project need to be known at the start of the development or significant costs will be incurred for revisiting earlier phases.

Q7:   d) scope, time, cost and quality.

Q8:   c) effort (or cost)

Q9:   a) as the definitive version of the plan against which progress is measured.

Q10:  c) person hours/days for tasks which have no float which run beginning to end throughout the project timeline.

Q11:  c) sends the data via HTTP headers rather than within the URL.

Q12:  a) the script which will process the data.

Q13:  c) required.

Q14:  c) exploits poorly secured form variables to execute code on the server.

Q15:  a) regular expressions.

Q16:  c) Persona.

Q17:  b) version of the application drawn on paper with which users can interact.

Q18:  d) team needs to draw on the views of number of areas: graphic designers, programmers, users experience experts, project managers and others to be successful.

Q19:  a) SELECT

Q20:  c) INSERT INTO

Q21:  d) SELECT * FROM developer

Q22:  c) SELECT * FROM developer WHERE firstname LIKE 'a%'

© HERIOT-WATT UNIVERSITY
**Q23:** a) SELECT * FROM developer ORDER BY firstname DESC

**Q24:** c) DELETE FROM developer WHERE firstname = 'Peter'

**Q25:** b) $mysqli = new mysqli("server", "username", "password","database");

**Q26:** a) $result = $mysqli->query($query);

**Q27:** a) Social issues.

**Q28:** c) Storing a word processing file.

**Q29:** a) Cyber crime.

**Q30:** b) remove search results that point to information that is no longer relevant from our past.