MODULE DESCRIPTIONS

ADVANCED DATA SCIENCE AND MACHINE LEARNING

This advanced module aims to equip students with the knowledge and skills to design, implement, and evaluate machine learning models and predictive analytics solutions. Students will learn both supervised and unsupervised learning techniques, model selection strategies, and model evaluation frameworks. Practical deployment considerations, including scalability, reproducibility, and interpretability of models, will be emphasised. The module prepares students to handle complex data science projects independently, focusing on the ethical, reliable, and effective use of data-driven models.

ADVANCED PROGRAMMING AND ALGORITHM ANALYSIS

In 1976, Niklaus Wirth (considered to be one of the "fathers of programming") published a textbook covering the fundamental topics of computer programming, entitled "Algorithms + Data Structures = Programs". This book highlights the inherent relation between algorithms and data structures and their importance as one of the essential subdisciplines of theoretical and practical approaches to computing. This module follows the content in this seminal work to show the importance of tailoring the algorithm chosen for a particular task to match the kind of input data expected.
Since computing is mainly about the storage and retrieval of information, the first part of this module introduces the concept of abstract data types (ADTs) to create a variety of complex data structures, including stacks, queues, linked lists, maps, hash tables and various forms of ordered trees. Associated operations required to insert, delete and manipulate objects in each of these ADTs are also covered. Thereafter, the module focuses on a variety of searching and sorting algorithms, including worst-case time and space analysis of these algorithms.
This module would be useful to any computer programmer wishing to write more efficient code as it introduces more complex data structures to replace the simple ones used in the prerequisite introductory programming modules, namely, Introduction to Programming and Object-Oriented Programming and shows how to analyse algorithms in order to choose the most efficient one to solve a given problem.

AUTOMATA AND FORMAL LANGUAGES

The theory of computation (which encompasses automata and formal languages) essentially governs the development of computational models (involving both mathematics and logic) that will execute efficiently and more importantly, to completion. Since all computational devices that implement logic apply the theory of computation, studying this is important to define the limitations of both the computational devices and the applications executed.
This module aims to introduce automata theory, computability theory and complexity theory. Automata theory is the study of abstract computing devices (or machines) that automatically follow a series of predetermined steps and the types of computational problems that can be solved by each device. In this module, finite state automata, push-down automata and Turing machines are investigated to find the limitations of each. Computability theory explores what types of problems can and cannot be solved by each of the abstract computing devices, while complexity theory involves the analysis of algorithms to estimate resource usage in terms of time and space.
This module provides the answers as to why certain classes of problems cannot be computed in a finite amount of time so that programmers do not waste time on impossible efforts. A simple algorithm that appears doable for a small input set, might be not computable for larger or different input sets. Understanding which problems can be solved algorithmically is one of the main objectives of this module, which would thus appeal to programmers and computing professionals wishing to understand the theoretical limitations of their craft.

CLOUD COMPUTING

The availability of high-capacity networks, low-cost computers and storage devices as well as the widespread adoption of hardware virtualisation, service-oriented architecture and autonomic and utility computing, has led to growth in cloud computing.
The aim of this module is to equip students with a sound knowledge of the underlying principles of cloud computing, its basic functioning and architecture. They will acquire the insight and skills needed to evaluate different cloud platforms and services, develop an understanding of the process (including how to manage the process), and gain experience in developing cloud applications. In addition, students will gain hands-on experience in using a Cloud Platform such as Microsoft Azure and using API technology to create a hybrid cloud.

COMPUTATIONAL THINKING AND INTRODUCTION TO PROGRAMMING

Efficient and effective problem solving is a skill that is of benefit to any individual in his/her everyday life. In the field of computing, problem solving as well as familiarity with computer programming concepts, development of computer programs, and the syntax and semantics of programming languages are fundamental to successful software development.
This module first introduces computational thinking as a systematic approach to problem solving with the solution expressed in such a way that it can be carried out by a computer. Thereafter, all aspects of programming, from designing an algorithm and coding this algorithm to develop a program, to debugging and executing the program, are covered. The main focus of the module is on the coding aspect, where students will be exposed to the Python programming language, from the basic programming constructs (sequencing, decisions and looping) and data structures (such as simple types and lists) to more advanced constructs (such as the use of functions and recursion) and more complex data structures (such as dictionaries).
No prior programming experience is required to complete this module. Students will be taken from first principles to being able to develop a non-trivial Python program. The module will be useful to students from diverse fields by offering a systematic way of approaching problem-solving and providing a solid introduction to programming.
Having completed this module, students will have a solid foundation of problem-solving and programming experience, thereby facilitating the comprehension of the more advanced aspects of programming, including object-oriented design and the use of abstract data structures, as taught in later modules.

COMPUTER ARCHITECTURE

To fully understand complex programming language constructs as well as algorithms for systems program, and to be able to create high performance programs, knowledge of computer architecture is fundamental. All computer science graduates should have knowledge of the low-level workings of a computer, and not merely see it as a black box. The theoretical approach to computer architecture presented in this module is complemented by a practical study of the MIPS processor and its related assembly language.
This module aims to introduce the basic computer architectural components using a simple microprocessor without Interlocked pipeline stages (commonly known as the MIPS processor). The main topics addressed are: computer arithmetic as well as logic for basic computer instructions, a simple von Neumannn architecture including its control, memory storage and I/O interface, and extensions to various aspects of this architecture in terms of pipelined control and data paths, as well as shared memory multiprocessors. Associated practical work involves using the MIPS assembly language running on a MIPS emulator to reinforce the theoretical knowledge of register usage, memory handling and other low-level constructs.
This module provides a foundation for students interested in developing systems software (including operating systems, interpreters, compilers, etc.) or user applications with enhanced performance, as well as those wishing to specialize in the computer architecture and engineering field.

COMPUTER NETWORKS

The growth of network capable devices (smart devices), constant additions to the Internet of Things (IoT), always online services, and the need for global communications has integrated networks and network technologies into the global everyday life. Because of this, a general level of understanding of networks is a cornerstone in any computer scientist's repertoire of knowledge.
This module serves as an introduction to networking, the design and implementation of networks, and the current technologies and protocols that are typically implemented within networks. Furthermore, this course provides additional knowledge on how to configure and include network communications into one's own programs and systems, and gives one knowledge on the use of commonly used network monitoring and debugging tools.
Those working with or merely having an interest in networks and digital communication in general would find this introductory module useful.

DATA AND DECISION-MAKING

In this module students will learn the importance of data in the decision-making process. They will be exposed to the fundamentals of decision making and how to assess risk and uncertainty while supporting this with well formulated and appropriate data. Students will further learn how to apply this knowledge as part of a larger project team for highly complex and more specialised decision-making environments where they act as facilitators rather than the actual analysts or decision makers.
To achieve the module outcomes, students will learn how to separate empirical quantifiable data (known factors) from uncertain and unverifiable assumptions (unknowns) often required in business decision making. With this separation established, students will be guided in how to apply the known and unknown correctly in order to manage risks associated with the degree of uncertainty. Learning how to assess whether the balance between the known and unknown data components will be achieved through the use of sensitivity analysis and the illustration of the relative impact that the various data points can have on the final decision and ultimately how the final decision will impact the venture over time.

DISCRETE MATHEMATICS

Although few computer scientists primarily work on discrete structures alone, there are many areas of computer science that require the ability to work with concepts associated with discrete structures. Thus, discrete mathematics is considered foundational material in this degree.
This module aims to develop discrete mathematical skills appropriate for students studying Computer Science as well as other scientific disciplines. The main skills addressed are: mathematical reasoning, which is necessary to read, understand and construct mathematical arguments; combinatorial analysis, which is the ability to count or enumerate objects and which is required in designing algorithms; working with discrete structures, which are the mathematical structures used to represent discrete objects, and their relationships; algorithmic thinking, which relates to the use of algorithms to solve certain classes of problems, and in so doing using mathematical principles to ensure the correctness of the algorithm and the efficient use of computational and memory resources; and finally modeling, whereby models are constructed using discrete mathematics to represent solutions in the real-world.
Topics for this module have been chosen to emphasise the Computer Science connection and include: sets, relations and functions, number theory, basics of counting, basic logic, proof techniques, discrete probability, and trees and graphs.

ELECTRONICS FOR COMPUTING

Automation in data processing is the cornerstone of the information age and modern processing technology builds the foundation that these systems run on. In order for one to understand the limitations of modern hardware, and how to best leverage its capabilities, one needs to understand the components that constitute this hardware.
This module serves as an introduction to electronics and aims to make one familiar with definitions of common terminology and theory used within electronics. Furthermore, this module provides a gateway to circuit analysis and design, as well as use of digital circuits to perform computation. Additionally, introduces high-level concepts about computation that serve as a basis into analysis, design and implementation of application specific hardware solutions.
This module would be of use to anyone with an interest in the electronic circuitry used in modern computers.

FOUNDATIONS OF DATA SCIENCE

The purpose of this module is to introduce students to the foundational concepts, processes, and tools used in data science. Students will develop the ability to collect, clean, explore, analyse, and visualise data to generate meaningful insights. The module emphasises critical thinking, problem-solving, and the practical application of statistical and computational techniques to real-world datasets. It bridges introductory statistics and more advanced data science and machine learning concepts, preparing students for professional data-driven roles or further study in the field.

FUNDAMENTALS OF DATA ANALYTICS

The amount of data generated and consumed on a daily basis is increasing at an alarming rate due to the ubiquitous use of mobile technology like smartphones and tablets; however, in its raw form, this data is useless. It is thus, not unexpected that data analytics is currently one of the fastest growing fields world-wide. At its simplest, data analytics is the systematic computational analysis of data. However, there are two distinct processes included in this simplistic definition. The first is data analysis, which focuses on analyzing the past to determine what happened and why (using statistical and diagnostic analysis techniques). The second process is data analytics, which uses data to predict what is likely to happen in the future (using predictive analysis techniques), and also to apply the insights gained from these various analysis techniques (including text analysis or data mining) to solve a current problem. This is also referred to as prescriptive analysis.
Computer programmers are ideally suited to working in data analytics, as many of the tools required are common to both fields and the processes to be followed are not dissimilar to those used in software analysis and design. This module introduces the fundamentals of data analytics, starting with an overview of the typical analytics lifecycle, and then focusing on each of the stages in this lifecycle. Methods and techniques are covered theoretically for each of the various types of data analysis, namely statistical, diagnostic, and predictive, and then these are practically implemented using the Python programming language and various associated libraries. Prescriptive analysis is dealt with by considering real life datasets and showing how analytics can be used to provide visualizations of actual and predicted results.
Given the widespread need for data analytics across all disciplines, this module would be both insightful and of practical use to anyone with basic programming skills.

FUNDAMENTALS OF INFORMATION TECHNOLOGY

The increased importance and global reach of computing technology in today's society was the basis for the emergence of the information technology discipline. Information Technology is the study of systemic approaches to select, develop, apply, integrate, and administer secure computing technologies to enable users to accomplish their personal, organizational and societal goals.
This module provides an overview of the discipline of IT, and how it relates to other computing disciplines. The goal is to help students understand the diverse contexts in which IT is used and the challenges inherent in the diffusion of innovative technology.

INFORMATION SECURITY AND CYBERSECURITY

With the growing ease of access to both hardware and software required to perform malicious actions on the Internet, there is a growing requirement globally to secure both developing and existing systems. Failure to do so can result in significant financial, operational, and reputational damage. In the worst case, this can cause an entity to cease operating entirely.
Students who take this module are required to have completed the "Computer Networks" module as this course builds on the knowledge acquired from this module. This module serves as an introduction to the design and implementation considerations that are part of the everyday security professional's on the job thought process. Additionally, this module provides knowledge of many security technologies and techniques that have been, and currently are, implemented in global technologies. Given the proliferation of cyber and network attacks, this module would be useful to anyone serious about safe-guarding information as well as computing infrastructure.

INFORMATION SYSTEMS PROJECT MANAGEMENT

Information Systems development projects are initiated for a variety of reasons, e.g. to address an existing problem, leverage a potential opportunity, support a business merger, or achieve regulatory compliance. The project manager plays a critical role in achieving project success, by effectively planning, leading and monitoring a project from inception to delivery.
In this module, students will learn the key responsibilities of a project manager and the scope of activities that are typically performed by project managers during the different stages of an information systems development project. In doing so, they will familiarise themselves with the methods and tools used to plan project activities and to monitor progress and budgets. They will also learn the importance of maintaining accurate project documentation. Role-playing scenarios will be used to practice relevant communication skills including techniques for conflict resolution. During the module, students will submit several individual and group assignments relating to different aspects of the project manager's job.

INTEGRATED RESEARCH PROJECT

Solving problems using IT is what computer scientists are trained to do. For some problems, IT solutions might already exist whereas for others, novel algorithms, application systems or hardware prototypes might be called for, which in turn implies some form of research.
This module requires students to identify a "problem" within a computer science research field (as identified in the Research Principles module) and to carry out an appropriate research experiment to solve it. The type of research experiment used is limited only in the sense that it must show practical use of some computer science concepts or theory studied in the degree curriculum and that the solution has some societal impact. To encourage team work and ensure a broader literature search, the initial literature review field may be completed by small groups of students working in a similar research field.
The full research project lifecycle (as outlined in the topics for this module) will need to be carried out, culminating in the project report as the final deliverable. In addition to this final deliverable, a scientific short paper needs to be written for dissemination of the project results to a wider audience, typically by way of a conference submission (although this submission is not required for successful completion of the module).

INTRODUCTION TO ARTIFICIAL INTELLIGENCE

With the vast increases in computational power, Artificial Intelligence has gained considerable attention resulting in huge development of AI applications in the past two decades. Most students will have heard of machine learning, autonomous cars and drones, and image recognition, amongst other applications of AI, all of which have resulted from the recent developments in AI. Given the prominence of such AI applications in our society and the fact that these are set to grow in the future, an understanding of the basic concepts and techniques in AI is crucial to all Computer Scientists.
This module aims to introduce a broad variety of subfields of AI to provide a solid foundation for later specialisation (at a post-graduate level) in one or more of the AI application topics. Specifically, this module introduces concepts and relevant algorithms in basic and advanced search strategies, knowledge representation and reasoning, intelligent systems, basic classification and clustering techniques, natural language processing and finally, image processing. The module follows a balanced approach by combining the theoretical aspects with practical implementations of the algorithms to increase understanding.
Topics for this module have been chosen to highlight foundational topics in the expansive AI field, and in so doing to provide the necessary understanding for future work in this field. This module would be invaluable to anyone seeking to deploy an AI-based application in the future, not only for the programmers of such an application, but also for the domain experts that are needed to provide the underlying knowledge to invoke the AI.

INTRODUCTION TO DATABASES

Information and raw data are the life blood of the various computing disciplines, either for their use in creating a solution to a given problem, or for storing after having been generated by information systems and other computational applications. As such algorithms for the efficient and effective collection, transformation, storage, access and updating of data, as well as for the representation, organization and presentation of the resulting information, constitute vital knowledge for computing professionals.
Relational databases, which rely on the use of structured tables to store data, have traditionally been used for data storage. However, with the ever-increasing volume of data needed to be processed and stored by information systems, non-traditional, unstructured alternatives have recently become more popular. The main aim of this module is to introduce students to the theoretical concepts of databases as well as provide practical experience in designing and extracting information from both relational and semi-structured databases.
This module introduces fundamental database concepts related to the design and creation of relational databases, as well as the extraction of information in the form of query building using SQL (structured query language). Also covered in the module are considerations of transactional processing and security of databases as well as an introduction to non-relational database models using XML to apply the theoretical concepts in semi-structured databases.
Having completed this module, students will be able to manipulate a variety of databases, as well as design, implement and manage a database system.

INTRODUCTION TO PARALLEL COMPUTING

With the proliferation of multi-core and parallel hardware, knowledge of parallel computing is no longer reserved for those who specialize in high performance computing. The performance of executing application programs on entry-level laptops and personal computers can easily be enhanced by taking advantage of the multiple processing cores available on a single device. To do this effectively, however, the user must understand the basic concepts of parallel computing and also the errors to avoid.
This module aims to introduce parallel algorithms and execution on both shared memory and distributed systems. The core foundational topics covered are: fundamentals of parallel and distributed computing, communication and synchronization primitives for shared memory systems, parallel algorithms, deadlock and performance issues. Practical exercises for this module can all be executed on multi-core CPUs.
The module is aimed at programmers who wish to take full advantage of their parallel hardware. The module is designed specifically to show that accessing increased performance via parallel computation does not have to be difficult or overly complex. Due to the extensive coverage of the theoretical concepts of parallel computing in general, this module would also be of interest to those wishing to gain a basic understanding of parallel and distributed programming.

OBJECT-ORIENTED ANALYSIS

This module will develop students' understanding of the object-oriented approach to information systems development, as well as their ability to apply a variety of object-oriented techniques that are commonly used during the systems analysis phase of a project.
The underlying concept of object-oriented analysis is that software systems are modelled as collections of cooperating objects. This approach supports the reuse of common objects, reduces system complexity, and facilitates future maintenance of the developed system. The Unified Modelling Language (UML) will be used to create appropriate visual models (diagrams) that support and enhance communication between analysts, developers and other stakeholders. Students will submit a number of individual assignments during the module to demonstrate their ability to apply object- oriented principles and UML modelling techniques. At the end of the module, students will submit a group project integrating the principles and methods that have been covered.

OBJECT-ORIENTED DESIGN

This module builds on the Object-Oriented Analysis module. In the Object-Oriented Design module, students will develop the knowledge and skills needed to create a detailed and unambiguous design specification for the system components, processes and interactions that together will deliver the functionality required within the final software product. Students will construct a variety of conceptual, logical and physical models as well as supporting text-based artefacts, that together define the architecture of the system, the associations between entities within the system, and the technology-specific composition of the system. Students will also be expected to understand and apply basic principles of good object-oriented design. Students will submit a number of individual practical assignments during the module in order to develop competence in the use of relevant UML modelling techniques. At the end of the module, students will submit a group project integrating the principles and methods that have been covered.

OBJECT-ORIENTED PROGRAMMING

One of the main competences required for employment in any of the computing disciplines is good programming skills. The object-oriented paradigm is extensively used in industry for developing large information systems. This module aims to impart both the theory and practice of this paradigm and further develop the overall programming skills of the students by introducing a second programming language, Java.
Students must have completed the "Computational Thinking and Introduction to Programming" module as prerequisite before attempting this module, which builds on the first principles of programming already covered and focuses on the design and implementation of larger more complex programs though a widely used industry language. In addition to providing the corresponding Java syntax for the constructs previously covered in the Python programming language, this module introduces object-oriented programming concepts, such as encapsulation and information hiding, data objects and inheritance, as well as additional imperative programming concepts and constructs, such as garbage collection and reference types. Practical experience using an object-oriented language is included by way of computer-based programming assignments.

OPERATING SYSTEMS

Often referred to as the brain of the computer, the operating system provides an interface between the programmer and the hardware. Included in the operating system software are various modules to perform the basic tasks required by an intermediary, such as file management, process scheduling, memory management, input/output and peripheral device control. In essence, the operating system abstracts the low-level functionality of the underlying hardware away from the user. However, in order to develop application programs that interact with various other devices or files, or those that require efficient use of memory or some other computer resource, knowledge of the operating system functionality and aspects of its implementation is essential.
This module aims to introduce the fundamentals of modern operating systems which allow multi-tasking and multi-users on a single computer. The main topics addressed are: creation and scheduling of processes, use of light-weight concurrent threads, synchronization primitives used to avoid deadlock and ensure race-free conditions for multiple processes and threads, creation and use of virtual memory, persistent filesystems, and security aspects related to each topic. Associated practical work is included to reinforce the theoretical knowledge of process scheduling policies, virtual memory paging policies, disk filesystem, memory handling and other synchronization issues.
This module provides a foundation for students interested in developing systems software (including operating systems, interpreters, compilers, etc.) or user applications with enhanced performance requirements, as well as those wishing to specialise in network and/or database administration.

PROGRAMMING LANGUAGE IMPLEMENTATION

Developing a compiler for a programming language makes extensive use of knowledge gained in a number of typical undergraduate computer science modules, including advanced data structures and programming, computer architecture, theory of computation, software engineering, and programming paradigms, amongst others. This module thus serves to consolidate many of the "stand-alone" modules previously studied. Although few computer science graduates will likely work in compiler development after graduating, a number of the techniques covered in this module would be most useful in a variety of occupational fields. Easily constructed parsers, for example, can be used to extract specific information from sources with a variety of formats (textual data, numerical data, log data).
This module covers the entire front-end and back-end compiler development cycle: from defining programming language grammars, through the syntax and semantic analysis phases, to the generation of intermediate bytecode that can be interpreted by a virtual machine. Topics for this module include programming language specification, character handling and syntax analysis, semantic constraint analysis, and intermediate code generation and interpretation. Native machine code generation is not covered in this module, mainly because it is a specialized process that is highly dependent on the hardware architecture on which the compiler will run and thus cannot be generalised.
By understanding how languages are compiled, programmers will be able to use the languages better, and therefore, this course is recommended not only for those wishing to specialise in translation of programming languages, but also for coders who take programming and the creation of efficient code seriously. Moreover, since much of the translation process can be generalised to the processing of other data (as alluded to above), this module would be of interest to general scientists with a programming background who need to parse files of structured data efficiently.

PROGRAMMING PARADIGMS

Programming paradigms relate to different classifications of programming languages based on certain features or characteristics. Some of these characteristics define how a particular language is executed, whilst others relate to the style of syntax and semantics used by the language. Knowledge of the main programming paradigms allows a programmer to choose the correct one for a particular application and once the paradigm has been determined, an appropriate programming language can be adopted. Two of the most widely used paradigms, imperative programming and object-oriented programming, have already been covered in this degree offering, however, without mention of the specific features that determine to which paradigm each of the languages belongs.
This module covers the features and characteristics of three of the most commonly used programming paradigms, namely, procedural programming, object-oriented programming, and functional programming, as well as introducing specialised aspects of logic and scripting programming languages. Languages from the various paradigms are compared with regard to their execution model, intrinsic data types and operations, control structures and suitability for use in solving various problems. This module would be useful to software developers and application programmers in selecting the best programming paradigm in terms of efficiency and ease of programming amongst others, as well as systems developers that require knowledge of various aspects of programming language theory.

SOFTWARE ENGINEERING

Software engineering is the use of an organized and regulated approach for the design, development, testing, documentation and maintenance of software by applying principles from engineering, project management, computer science, programming, cost management and other areas.
The Software Engineering module specifically aims to lay a solid foundation in the discipline of developing and maintaining software systems that behave reliably and efficiently, are affordable to develop and maintain, and satisfy all the requirements that customers have defined for them. The student will acquire broad knowledge about the Software Engineering process and the range of methods, tools, and techniques utilised, illustrated by means of a variety of case studies.

STATISTICS

In this module, students will gain a good theoretical and practical understanding of statistical techniques and experimental design as applied to commercial problems. The topics covered in this module address the full breadth of the components of statistics, consisting of descriptive statistics, inferential statistics and statistical modelling.

TECHNOLOGY AND SOCIETY

The cyber world is a world with countless interactions in different forms. Online interaction has become part of our daily routine and it presents a number of challenges to both developers, owners and users. Given the different types of personal and corporate data that can be generated, it is important to establish awareness of possible ethical contraventions in which we can unintentionally participate. The development of new technologies also potentially poses threats to the livelihoods of many. Ethical behaviour by all stakeholders is crucial in sustaining the fair development and implementation of technologies. It is also important to ensure fair online interaction and the sharing of information. This module aims to establish an initial awareness of ethical issues in students as they start their studies in technology. It will assist with understanding the rights of people in the world of technology.

VIRTUAL SYSTEMS AND SERVICES

New software, from operating systems to applications, constantly demands more resources - more data, more processing power, more memory. Virtualization makes a single physical machine act like multiple machines, thus saving the cost of more servers and workstations.
The Virtual Systems and Services module introduces this important technology which is in widespread use. The aim is to equip students with a sound knowledge of the underlying principles and concepts of virtualization, and to provide them with the basic skills needed to be able to implement virtualization for desktops, servers, and network platforms. Examples of virtualized applications will be discussed, and students will be required to explore and evaluate further examples.

WEB AND MOBILE DEVELOPMENT

In the digital age, standalone applications have become largely obsolete, and the minimum interface requirements for user applications is a web interface. Some applications also require a mobile device interface through either an installed app on the device or the web interface. Thus, web and mobile programming skills are becoming necessary for application programmers.
Website development is one of the fastest developing sub-areas of computer programming, whilst mobile app development is another lucrative area for specialization. The main aim of this module is to introduce students to the theoretical concepts of design and programming for the web and mobile devices, whilst considering best practices in human-computer interaction and focusing on usability.
This module introduces the fundamental concepts related to the design and implementation of both static and dynamic webpages and basic mobile apps. In terms of web development it focuses on both front-end and back-end design and implementation, whilst for the mobile development, emphasis is placed on usability and working with limited resources. Foundational concepts in human-computer interfaces are covered to ensure good design practices are followed in both the web and mobile development. This module would be a useful addition to anyone with basic programming and database experience, who wishes to develop webpages and/or mobile apps, thus gaining a skill that will always be in demand.