MODULE SPECIFICATION

Part 1: Information			
Module Title	Fundamentals of Software Development		
Module Code	UFCFQM-30-1	Level	Level 4
For implementation from	2020-21		
UWE Credit Rating	30	ECTS Credit Rating	15
Faculty	Faculty of Environment & Technology	Field	Computer Science and Creative Technologies
Department	FET Dept of Computer Sci & Creative Tech		
Module type:	Project		
Pre-requisites	None		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description
Overview: Students will learn the basic concepts of software design, data structures, programming, problem solving, programming logic, and fundamental software design techniques. This will include a review of traditional and contemporary software development methods including agile development. They will develop a holistic view of software engineering practice including gathering requirements, designing a solution, implementing a solution in a programming language, testing the completed application and deploying the solution to end users. **Educational Aims:** The purpose of this topic is to introduce the students to the fundamental concepts of systems development through programming, computational thinking and data structures. They will analyse models of application development so that they can understand the key processes related to building functioning applications and appreciate the complexity of application development. **Outline Syllabus:** Development lifecycles and their differences (e.g. Waterfall model, Agile) Application of programming principles to business requirements (e.g. design, code, test, correct, deploy and document) including the approach required Role of legacy systems in program development Programming paradigms and the reasoning behind selecting it for business requirements (e.g.

Event Driven, Object Orientation, Procedural)

Analyse business' software requirements to create a sound solution

Develop a program using an industry recognised language and development lifecycle model (e.g. Design, implement, test and debug)

Create modifications to a created to program to meet program requirements

Testing facilities and tracking to be able to debug created program code to understand and rectify problems within the code (e.g. white box, black box, unit testing)

Usage of industry standards to create robust and efficient code and understanding the necessity of this practice for both solo and group projects

Creation of end user training (e.g. User guide, Tooltips, help facilities)

Teaching and Learning Methods: Introductory lectures are supported by seminars, case studies, and practical workshops. In addition, this module will be supported by interactive forums and learning tools.

Independent learning includes hours engaged with essential reading, assignment and completion. Study time will be organised each week with a series of both essential and further readings.

This module will be based on ensuring that student's practical skills are developed in programming. Every session will incorporate designated practical work to complete in order to ensure that students understand and implement principles of good practice.

Part 3: Assessment

This module is assessed by a combination of techniques: a report (3,000 words) and a practical build.

Component A1 – Report

Students will be required to write a 3,000-word report that will identify the core concepts that a software developer will need to follow in the development of software applications. Students will need to discuss the stages of a different development lifecycles and their stages.

Students will be required to understand the application of programming principles throughout the stages of the lifecycle model and how to approach each stage. Within the development stage, they should also be able to identify the reasoning behind selecting a paradigm for the development task.

Students will need to be able to demonstrate their understanding of why legacy systems are used within a business environment to test new or existing projects.

Component A2 – Practical Build

Students will be given a business specification from which they will produce a solution. They will need to design their systems and apply their knowledge of the development lifecycle models to create a sound system.

The task will include practical software design, development, implementing, testing and debugging. The testing of the program will need to be robust and thorough, using techniques such as white and black box testing. After the testing, students will be required to modify their existing code base and apply new fixes or modules from their requirements and testing. The program will need to be fully documented and conform to industry standards.

Students will be required to create end user training that should be able to be delivered with a successful system.

Opportunities for formative assessment exist for the assessment strategy used. Verbal feedback and written feedback is given to all students providing a personal platform for improvement.

First Sit Components	Final Assessment	Element weighting	Description
Project - Component A		60 %	Fully Documented (e.g. technical, user guide, algorithms) and Implemented System
Report - Component A	✓	40 %	Report (3,000 words)
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Project - Component A		60 %	Fully Documented (e.g. technical, user guide, algorithms) and Implemented System
Report - Component A	✓	40 %	Report (3,000 words)

Part 4: Teaching and Learning Methods

Learning Outcomes

On successful completion of this module students will achieve the following learning outcomes:

Module Learning Outcomes	Reference
Understand basic programming concepts.	MO1
Understand programming principles including design, code, test, correct, deploy and document from supplied specifications, using agreed standards and tools.	MO2
Understand the stages of a software development lifecycle.	MO3
Understand the similarities and differences between agile and waterfall software development methodologies.	MO4
Be aware of the role and position of legacy systems in organisations and how new development environments interface and integrate with them.	MO5
Understand how teams work effectively to produce software.	MO6
Understand software design approaches and patterns and can interpret and implement a given design.	MO7
Analyse business and technical requirements and select appropriate solutions.	MO8
Design, implement, test, and debug software to meet a requirement's specification.	MO9
Select the relevant paradigm for a given set of business requirements.	MO10
Develop moderately complex software solutions and software modifications to specified requirements.	MO11
Debug own code and understand structure of programmes in order to identify and resolve issues.	MO12
Identify and apply best practices and standards.	MO13

Contact Hours

Independent Study Hours:	
Independent study/self-guided study	228
Total Independent Study Hours:	228
Scheduled Learning and Teaching Hours:	
Face-to-face learning	72

	Total Scheduled Learning and Teaching Hours:	72
	Hours to be allocated	300
	Allocated Hours	300
Reading List	*The reading list for this module can be accessed via the following link:* https://rl.talis.com/3/uwe/lists/363CA84E-881E-B1F4-1B7D-9861A48A8EA9.html	

Part 5: Contributes Towards
This module contributes towards the following programmes of study: