STUDENT AND ACADEMIC SERVICES

MODULE SPECIFICATION

Part 1: Information

Module Title	Collaborative Software Development Project		
Module Code	UFCFCR-30-3	Level	Level 6
For implementation from	2019-20		
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:	Standard		
Pre-requisites	None		
Excluded Combinations	None		
Co- requisites	None		
Module Entry requirements	None		

Part 2: Description

Overview: The primary role of a software engineer is to be able to design, build and test high-quality software solutions following best practices and industry standards. They will typically be working as part of a larger collaborative team, in which they will have responsibility for significant elements of the overall project. The developer will need to be able to interpret requirements, specification documentation and designs in order to develop and test software that meets its requirements, even when these requirements may change. Using a collaborative environment allows developers to bring together independent modules together to create a complete solution

Educational Aims: See Learning Outcomes

Outline Syllabus: In completion of this module learners should be able to:

Undertake the roles and responsibilities that are required from a software engineer at every stage of the development lifecycle

Collaborate as a team to apply systems analysis and design to a project specification creating artefacts
(e.g. use case)

Interpret and implement a design that's compliant security requirements (e.g. functional and nonfunctional)

Collaboratively create a program based on user requirements, embracing an industry based methodology

Create software using industry standard build processes, and tools for configuration management,
version control and software build (e.g. GitHub, Bitbucket, Tortoise SVN), release and deployment

Create effective and secure software solutions using contemporary software development languages, producing high quality code with sound syntax applying secure and robust development techniques to increase code resilience

Participate in code reviews, debugging and refactoring processes to improve code quality and efficiency

Thoroughly test a solution to ensure resilience of code and that it meets the functional and nonfunctional
requirements (e.g. black box, white box, unit testing)

Teaching and Learning Methods: Introductory lectures are supported by seminars, case studies, visits and practical workshops. In addition this module will be supported by interactive forums and learning tools. Students must have access to a suitable publicly accessible hosting platform and database server to be able to complete this module.

300 hours study time of which 108 hours will represent scheduled learning. Scheduled learning includes lectures, seminars, tutorials, demonstration, practical classes and workshops; external visits; supervised time in studio/workshop.

Part 3: Assessment

This module is assessed by a combination of techniques: an examination and a practical portfolio.

Group Presentation (includes the following):

Critical Review of Project Lifecycle
Demonstration of the completed project
Evaluation of team performance

Practical Portfolio (includes the following):

Design Documentation
Allocation of Roles and Modules
Completed Project Code Base
Collaboration Logs for all code branches (Including code commits and comments)
Meeting Minutes / Code Reviews and Audits
Testing Documentation

Opportunities for formative assessment exist for the assessment strategy used. Verbal feedback is given and all students will engage with personalised tutorials setting SMART targets as part of the programme design.

First Sit Components	**Final Assessment**	**Element weighting**	**Description**
Portfolio - Component B		70 %	Portfolio - Design, Build and Test a Software Project built using a Collaborative Approach
Presentation - Component A	✓	30 %	Group Presentation (30 Minutes)
Resit Components	**Final Assessment**	**Element weighting**	**Description**
Portfolio - Component B		70 %	Portfolio - Design, Build and Test a Software Project built using a Collaborative Approach
Presentation - Component A	✓	30 %	Individual Presentation (20 Minutes)

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**
Identify, justify and review all stages of the software development lifecycle for a completed project.	MO1
Analyse, interpret and implement a design compliant with functional, nonfunctional and security requirements	MO2
Create effective and secure software solutions using contemporary software development languages to deliver the full range of functional and non-functional requirements using relevant development methodologies.	MO3
Perform code reviews, debugging and refactoring to improve code quality and efficiency	MO4
Test code to ensure that the functional and non-functional requirements have been met, using functional and unit testing.	MO5
Deliver software solutions using industry standard build processes, and tools for configuration management, version control and software build, release and deployment into enterprise environments	MO6
Evaluate the effectiveness of teamwork and roles in completing a collaborative software development project	MO7

Contact Hours

Independent Study Hours:	
Independent study/self-guided study	192
Total Independent Study Hours:	192
Scheduled Learning and Teaching Hours:	
Face-to-face learning	108

	Total Scheduled Learning and Teaching Hours:	108
	Hours to be allocated	300
	Allocated Hours	300
Reading List	*The reading list for this module can be accessed via the following link:* https://uwe.rl.talis.com/index.html	

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