Module Specification

Programming Data Intensive Applications

Version: 2026-27, v1.0, Approved

Contents

Module Specification **1**

Part 1: Information **2**

Part 2: Description **2**

Part 3: Teaching and learning methods **4**

Part 4: Assessment **4**

Part 5: Contributes towards **5**

Part 1: Information

Module title: Programming Data Intensive Applications

Module code: UFCEQC-15-M

Level: Level 7

For implementation from: 2026-27

UWE credit rating: 15

ECTS credit rating: 7.5

College: College of Arts, Technology and Environment

School: CATE School of Computing and Creative Technologies

Partner institutions: None

Field: Computer Science and Creative Technologies

Module type: Module

Pre-requisites: None

Excluded combinations: None

Co-requisites: None

Continuing professional development: Yes

Professional, statutory or regulatory body requirements: None

Part 2: Description

Overview: This module explores the principles and practices of building scalable data-intensive applications, covering distributed architectures, storage and processing frameworks, optimisation, and deployment.

Features: Not applicable

Educational aims: This module develops students' ability to design and build robust, maintainable data applications beyond scripting level. It focuses on software

engineering best practices, the use of frameworks and architectural patterns, and integration with diverse data sources. Students will also gain experience in creating APIs, dashboards, and visualisation tools, preparing them to deliver high-quality data-intensive applications in research and industry contexts.

Outline syllabus: Topics may vary, but indicative subjects will include:

Software Engineering for Data Applications:
Principles of software engineering best practice (modularity, reusability, maintainability).
Version control workflows and collaborative development (e.g. Git, GitHub/GitLab).
Test-driven development and automated testing for data applications.
Code quality assurance, documentation, and maintainability considerations.

Frameworks, Patterns, and Data Integration:
Architectural patterns for data applications (e.g. Model-View-Controller MVC, layered architectures).
Object-relational and object-document mapping (ORM/ODM).
Use of Python frameworks (e.g. Flask, FastAPI, Django) for building data applications.
Data integration techniques for Structured Query Language (SQL)/NoSQL databases and external services.

Data Access, Presentation, and Interaction:
Designing and implementing Application Programming Interfaces (APIs) for data access and integration.
Building interactive dashboards and visualisations (e.g. Plotly Dash, Streamlit).
Usability considerations in presenting data to end-users.
Case studies of maintainable, user-oriented data applications.

N.b. Specific languages and tools are indicative and may be replaced by similar alternatives

Part 3: Teaching and learning methods

Teaching and learning methods: Teaching on this module will combine lectures with practical sessions to provide both theoretical understanding and hands-on experience. Lectures will introduce core principles, frameworks, and case studies of data-intensive systems, while practical sessions will allow students to apply these concepts through programming tasks, experiments with distributed processing frameworks, and deployment exercises.

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

MO1 Apply software engineering best practices to the development of robust, maintainable data-intensive applications.

MO2 Design and implement data applications using appropriate frameworks, architectural patterns, and data integration techniques.

MO3 Integrate and present data through APIs, dashboards, or visualisation tools, with consideration for usability and maintainability.

Hours to be allocated: 150

Contact hours:

Independent study/self-guided study = 120 hours

Face-to-face learning = 30 hours

Reading list: The reading list for this module can be accessed at readinglists.uwe.ac.uk via the following link https://uwe.rl.talis.com/modules/ufceqc-15-m.html

Part 4: Assessment

Assessment strategy: Group Project: Students will work in teams to design, implement, and evaluate a data-intensive application. The project will assess their ability to apply programming and data management skills, address scalability and optimisation challenges, and demonstrate deployment considerations. It will also

evaluate teamwork, project management, and effective communication of results. Formative feedback will be given throughout.

To ensure fairness, individual adjustments will be applied to account for differences in contribution within each team, based on evidence such as project logs, peer evaluation, and individual reflections.

Resit will match the main sit

Assessment tasks:

Group work (First Sit)
Description: Group Work (design, implement, and evaluate a data-intensive application)
Weighting: 100 %
Final assessment: Yes
Group work: Yes
Learning outcomes tested: MO1, MO2, MO3

Group work (Resit)
Description: Group Work (design, implement, and evaluate a data-intensive application)
Weighting: 100 %
Final assessment: Yes
Group work: Yes
Learning outcomes tested: MO1, MO2, MO3

Part 5: Contributes towards

This module contributes towards the following programmes of study:

Data Science [Frenchay] MSc 2026-27

Data Science [Frenchay] MSc 2026-27