# SETAP – Software Engineering Theory and Practice

Academic Year 2023/2024

## Coursework Specification (Referral/Deferral)

Module co-ordinator: Dr Claudia Iacob

Email: iacob@port.ac.uk

# 1. Overview

You are required to design and develop a prototype of a project management app, documenting the development stages as follows:

Chapter 1: Problem specification (25% of the final mark)
Chapter 2: Design (25% of the final mark)
Chapter 3: Implementation (25% of the final mark)
Chapter 4: Testing (25% of the final mark)

This is an individual assignment.

## Chapter 1: Problem Specification

The Problem Specification chapter must include:

- Detailed description of the process used to elicit requirements
- Overview of user requirements gathered for the system
- Detailed translation of the user requirements into system requirements, both functional and non-functional.

## Chapter 2: Design

The Design chapter must include:

- Detailed model and description of the system's architecture
- Detailed specification of at least 3 representative use cases. These should not include Log In, Log Out, and Register.

## Chapter 3: Implementation

The Implementation chapter must include:

- A link to a 3-5 minute demo of the system implemented.
- A link to the code repository

## Chapter 4: Testing

The Testing chapter must include:

- A complete test plan for the system OR
- A link to the automated tests that cover the functionality implemented.

# Marking Scheme

| Chapter | Marks | Expectations |
|---|---|---|
| **Problem Specification** | 10 | Detailed description of the process used to elicit requirements |
| | 5 | Overview of user requirements gathered for the system |
| | 10 | Detailed translation of the user requirements into system requirements, both functional and non-functional |
| **Design** | 15 | Detailed model and description of the system's architecture |
| | 10 | Detailed specification of at least 5 representative use cases |
| **Implementation** | 20 | A 3-5 minute demo of the system implemented |
| | 5 | Quality of the code submitted |
| **Testing** | 25 | Complete test plan for each system requirement specified for the project or automated tests |

**Problem specification**

- **Process**
  - **0-3:** No description of the requirements gathering process or brief discussion of the design of the process without any evidence of the data collection and/or analysis
  - **4-6:** Use of inadequate requirements gathering method;
    Brief/incomplete description of the design of the method and the motivation for using it;
    Brief description of the data collected with no/little evidence of the data; Brief/no discussion of the patterns identified in the data and the process used to identify these patterns;
  - **7-10:** Use of a single requirements gathering method; Adequate choice of method;
    Clear and complete description of the design of the method and the motivation for using it;
    Brief description of the data collected and evidence of the data; In-depth discussion of the patterns identified in the data and the process used to identify these patterns;

- **User requirements**

  - **0-1:** Brief list of user requirements;
    Requirements presented as notes;
    No indication of the source of each user requirement;
  - **2-3:** Complete list of user requirements;
    Requirements presented as clear statements;
    No indication of the source of each user requirement;
  - **4-5:** Complete list of user requirements;
    Requirements presented as clear statements;
    The source of each user requirement is referenced back to the data collected via the requirements gathering process.

- **System requirements**
  - **0-3:** No system requirements or incorrect identification of the user functional and non-functional requirements.
  - **4-6:** Correct identification of the user functional and non-functional requirements, but no/limited translation of these requirements into detailed system requirements.
  - **7-10:** Correct identification of the user functional and non-functional requirements;
    Clear and complete translation of each user requirement into its corresponding system requirements;

**Design**

- **Architecture**
  - **0-5:** Simplified model, limited or no connection to the system's requirements.
  - **6-10:** Complex model, connection to the system's requirements clear; Detailed explanation of the model.
  - **11-15:** Complex model, connection to the system's requirements clear; Detailed explanation of the model; Architectural pattern identified and choice discussed.

- **Use case modeling**
  - **0-3:** Identify 5 use cases by name.
  - **4-6:** Draw a correct use case diagram including at least 5 use cases. Brief specification of at least 5 use cases.
  - **7-10:** Draw a correct use case diagram including at least 5 use cases. Detailed specification of at least 5 use cases.

**Implementation**

- **Demo**
  - **0-8:** Limited complexity, limited number of features demoed, interface design can be improved.
  - **9-10:** Working prototype, medium complexity, good number and range of features demoed, well designed interface.

    - **11-15:** Working prototype, elements of high complexity, good number and range of features demoed, interface design can be improved.
    - **16-20:** Finished product, high complexity, complete and diverse features demoed, well designed interface.
- **Quality of code**
    - **0-2:** Basic use of version control.
    - **3-5:** Good use of version control (using issue tracker, branches, GitHub wiki), well written code comments.

**Testing**

- **Test plan**
    - **0-10:** Random test cases identified for the system.
    - **11-20:** Test cases identified for all units of code, but unclear if all possible types of inputs considered.
    - **21-25: T**est cases identified for all units of code, good evidence of partition testing.

**OR**

- **Automated tests**
    - **0-10:** Limited test case coverage - random functions are associated with unit tests.
    - **11-20:** Good test case coverage - main functions are associated with unit tests.
    - **21-25:** Complete test case coverage - all functions are associated with unit tests.