**Summative Assignment**

|  |  |
| --- | --- |
| **Module code and title** | COMP2281 Software Engineering (Team work) |
| **Academic year** | 2024-25 |
| **Coursework title** | Technical Report |
| **Coursework credits** | 4 credits |
| **% of module’s final mark** | 20% |
| **Lecturer** | Effie Law |
| **Submission date\*** | Friday, March 14, 2025 14:00 |
| **Estimated hours of work** | 8 hours |
| **Submission method** | Ultra |

|  |  |  |
| --- | --- | --- |
| **Additional coursework files** | *None* | |
| **Required submission items**  **and formats** | *Technical Report document in PDF format.*  *You must include the following information the filename: GroupNN(where NN = your group number, zero-padded e.g. Group 1 ->Group01, Group 21 -> Group21, etc.)* |  |

\* This is the deadline for all submissions except where an approved extension is in place.

Late submissions received within 5 working days of the deadline will be capped at 40%.

Late submissions received later than 5 days after the deadline will receive a mark of 0.

It is your responsibility to check that your submission has uploaded successfully and obtain a submission receipt.

Your work must be done by yourself (or your group, if there is an assigned groupwork component) and comply with the university rules about plagiarism and collusion. Students suspected of plagiarism, either of published or unpublished sources, including the work of other students, or of collusion will be dealt with according to University guidelines (<https://www.dur.ac.uk/learningandteaching.handbook/6/2/4/>).

# Software Engineering (COMP2281) 2024/25

# Technical Report

## Overview

The Technical Report is to be submitted in pdf format via Ultra by ***2pm on 14th March 2025***. We appreciate it can be difficult to submit all the code of your final working solution in a way that makes it possible for staff to run and mark. If your final product is accessible online you should provide an additional text file containing the URL with your submission or outline very clearly at the very beginning of the Technical Report where the system can be found and provide the link, if applicable. Any aspects of your system not accessible online should be zipped and submitted through Ultra too. To successfully mark your Technical Report, staff will need accounts with full access to all parts of your system, which can then be later deleted as required.

## Form and Content

There is no required format or style for the Technical Report—you are free to choose your own. However, it is strongly recommended that you follow the **report structure** outlined below.

The Technical Report should be a maximum of 20 A4 pages, using 11-point font, single-spaced, with 2 cm margins on all sides. This includes tables and figures. The length should be appropriate to the complexity of the project and its features.

The quality of the report will be evaluated based on the **clarity, thoroughness**, and **rigour** in addressing the following two key aspects:

* **How the system has been developed**
* **How the system should be used by the client and other stakeholders, including end-users.**

For the first aspect, the technical descriptions should be detailed and self-contained, enabling other developers, not involved in the project, to continue development independently.

For the second aspect, the user instructions should cover all system functionalities and features, which were co-created with the client/stakeholders during the early project phase (as documented in "Requirements Specification"), tested (as documented in "Test Plan"), and refined throughout development.

The final product should include the submitted code, which must align with the Technical Report. Additionally, the report should address important issues such as system maintenance, future development plans, and the assessment of the system's potential ethical and societal impacts.

## Report Structure and Mark Scheme

Below is the proposed structure for the Technical Report. It is strongly recommended that you use this structure and the numbered headings provided to organize your document. You may adapt the structure as needed to address the specific characteristics of your project.

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Cover**

You must include a single cover page indicating:

* The title of the document (i.e. “Technical Report for <Project Name>”)
* The authors of the document (i.e. full names and CIS usernames for each group member)
* The group number of the software engineering team
* The date the document was prepared.

**Section 1: Introduction (subtotal: 20%)**

* 1. Present a clear summary of the project to contextualise its main motivation and goals (5%)
  2. Provide usable access to the developed system, along with clear instructions on how to set it up and run it (5%)
  3. Present the status of each behavioural requirement in a table. Provide the code (e.g., BR1.2 is “behavioural requirement for feature 1, scenario 2”) and a succinct description (no need to restate the entire user story). Indicate whether the requirement remains unchanged or has been modified (if modified, provide the updated description), and explain the extent to which it is met or not met, with justification if applicable. (10%)

**Section 2: Technical Development** (**subtotal: 40%)**

* 1. Clearly describe the source materials that form the basis for the conceptualisation and development of the system. (5%)
  2. Provide a clear and appropriately detailed technical description of how each system functionality was developed. Include, where relevant, some or all of the following aspects: (30%):
* A high-level overview of the system architecture.
* The design principles and patterns used.
* The technologies and platforms used (e.g., programming languages, databases, frameworks).
* Relevant diagrams (e.g., UML diagrams, class diagrams, sequence diagrams) to illustrate system design.
* The software development process followed by the team.
* The role of the behaviour-driven development (BDD) approach in the implementation phase.
  1. Clearly describe how the system's usability and user experience aspects were addressed, providing an appropriate level of detail. (5%).

**Section 3: Use Instructions (subtotal: 20%)**

* 1. Installation: Describe system requirements, including the minimum and recommended hardware requirements (e.g., CPU, RAM, and storage) and operating systems. Provide step-by-step instructions for installing and configuring the software. (5%)
  2. Deployment: Explain how to deploy the system on a local machine and how to set up the database. If relevant, include instructions for setting up virtual machines, containers or cloud functions. (5%)
  3. Launching: Provide instructions on creating a user account or logging into the system. Explain different user roles (if applicable). Guide users through any first-time setup steps, such as creating admin accounts or configuring user settings. (5%)
  4. Troubleshooting: List common error messages and solutions for resolving them. Explain where users can find logs or diagnostic information to troubleshoot issues. (5%)

**Section 4: Maintenance and Implications (subtotal: 20%)**

* 1. Provide useful and usable information how the system can be maintained (5%).
  2. Provide useful and usable information how the system’s possible future development can be implemented (5%).
  3. Describe potential ethical and societal impacts of the system in its current and possible future status (10%).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*