COMP2005
Software Development Tools and Practices

20 CREDIT MODULE

ASSESSMENT: 100% Coursework **W1: 30% Set Exercises**
W2: 70% Report

MODULE LEADER: Dr Mark Dixon

MODULE AIMS

- To explore software development best practices such as usability, unit testing, TDD, version control and quality management.
- To develop an awareness of current industry tools for automation of builds and release of software systems.
- To create appropriate test plans and test cases making use of modern testing tools and processes.
- To examine a variety of techniques to evaluate a given UI.

ASSESSED LEARNING OUTCOMES (ALO):

1. Analyse the issues that are important in selecting a set of tools for the development and release of a software system.
2. Describe and distinguish the different types of and levels of testing.
3. Create and document a set of tests for a medium sized software project.
4. Demonstrate the capability to use software tools in support of the deployment of a software product.

Overview

This document describes the assessment of *COMP2005 Software Development Tools and Practices*. The module is assessed via **100% coursework**, across two elements: *30% Set Exercises* and *70% Report*.

The submission and expected feedback dates are presented in Table (1). All assessments are to be submitted electronically via the respective DLE module pages before the stated deadlines.

	Submission Deadline	Feedback
Set Exercises (30%)	16:00-19:00 Tue 18th March 2025	after submission
Report (70%)	15:00 Thu 1st May 2025	within 20 working days

Table (1): Assessment Deadlines

All assessments will be introduced in class to provide further clarity over what is expected and how you can access support and formative feedback prior to submission. Whilst the assessment information is provided at the start of the module, you are not expected to start this immediately (as you will often not have sufficient understanding of the topic). The module leader will provide guidance in this respect.

As with any software development project, the details of this brief may change slightly over time. You must check for changes and announcements (both email and in teaching sessions) regularly.

This document was last updated: Tuesday, 04 February 2025

Assessment 1: Set Exercises

Task:

This coursework contributes **30%** of the overall module mark for COMP2005 and is an **individual assignment**. It will be administered via the DLE quiz facility (where you will need to enter short free-text responses to the exercises presented) and as such will be run entirely online. Students will be given a 2 hour period in which to submit. They will be allowed up to 3 attempts within that period and the final mark will be an average of all attempts taken. The mark should be available just after submission.

Note: This is separate from the lab exercises that are handed out each week (which are not assessed).

Assessment 2: Report Automated Software Testing

This assignment contributes **70%** of the overall module mark for COMP2005 and is an **individual assignment**. The **final submission** must be submitted to the DLE by the specified submission dates.

Software Testing Task

For the purposes of this assignment, you have been provided with a Hospital Maternity Unit Web-Service API that:

- can be found at:
  https://web.socem.plymouth.ac.uk/COMP2005/api/
- serves information about a UK Hospital Maternity Unit, including information with dates regarding patient admissions
- includes its own specification (showing endpoints and return information)

Your task is to create software that fulfils the business requirements specified below, and to ensure that this software functions as expected by **implementing** and **documenting** a set of tests. You must demonstrate a range of testing approaches (such as Unit testing, using mock objects, Integration testing, System tests, and Usability testing). An important part of the assignment is how to plan and design your tests. You should consider edge cases and corner cases to ensure that the system is properly tested, as well as considering aspects such as code coverage. The software development tools and techniques discussed during the module should be incorporated.

This task has 2 parts:

- Part A: build a web-service API (using Java Spring Boot or Spark) that provides endpoints for the following functionality (to fulfil the business requirements specified by the customer):
  - F1 - A list of patients who have never been admitted
  - F2 - A list of patients who were re-admitted within 7 days of being discharged
  - F3 - Identify which month has the most admissions

- F4 - A list of patients who have had more than one member of staff

Note: This part has no GUI (as it is a web-service). It must connect to the existing API, parse the returned data and provide its own endpoints to fulfil the requirements specified above. Your code will be run by executing the different tests that you will provide.

- Part B: build a Desktop application (using Java Swing UI), that:
  - connects to 1 of these endpoints (you are free to choose which)
  - allows a user to interact with the data via a GUI
  - apply some form of useability test to evaluate the GUI
  - document software modifications made as a result of the usability testing

Deliverables

There are three deliverables for this assignment:

D1: A single report file **(in PDF format)** describing the test strategies (the **tables in the report** should describe the different kinds of tests (i.e., unit tests, integration tests, functional tests, usability testing), inputs, and any (pre) conditions) for the task. The report must be a document of no more than 2,000 words. You should use screen shots and sketches to illustrate key aspects. The report must include:

- A link to your code on GitHub
  (no other source will be marked, you must ensure it can be accessed).
- A link to your video on YouTube
  (no other source will be marked, you must ensure it can be accessed).

The report should have a complete description of the testing strategy. It should include:

- A detailed discussion and analysis.
- A description of the test cases you have designed.
- A description of the structure and role of any mock objects you have used.
- An explanation of how to run the **unit and integration tests**, and **continuous integration** (in code environment).
- Your functional test plan.
- An evaluation (critical analysis) of your test strategy, explaining why you have tested the software in the way that you have.

D2: The code you have written (both the software and the tests). This must use the GitHub Classroom repository created for you (which will be described in class) and a link included in your report (D1). This should contain all of the code you have written yourself and should (including copies of the folders and files needed to run your application).
Use your real names for your GitHub account, not nick names.
This should contain all of the code you have written yourself and should (including copies of the folders and files needed to run your application).
Failure to submit a working link within your report is likely to lead to a substantially reduced (failing) mark.

D3: A video of ***5 minute*s** (only the first 5 minutes will be marked) in which you demonstrate your software's functionality working, and the different tests. A link must be included in your report (D1) and the video must be accessible. The video **must be uploaded as an unlisted YouTube video**. The video will be used to assess the implementation of the software, and failure to submit a working link within your report is likely to lead to a substantially reduced (failing) mark.

Assessment and Feedback

The marking criteria in the following table will be used (subject to moderation – usually no more than 10% mark reallocation may be undertaken after submission for moderation purposes):

Analysis and Discussion	10
Unit Tests	20
Integration Tests	20
System Tests	10
Metrics (such as *Code Coverage*)	10
Usability	10
Use of Tools, Practices, and Systems (such as *Version Control, CI / CD*)	10
Evaluation	10
Overall (Task 1 & Task 2)	**100%**

Table (5): Assessment 2 Marking criteria

Threshold Criteria (These are indicative only)

Fail

- The submission has no or inadequate evidence of appropriate tests (such as unit and integration)
- Evidence shows little or no understanding of software development practices.
- Very limited evidence of test cases are provided (suggesting little or none of the system has been tested)
- Little or no evidence of a functional test plan has been provided.
- There is no or inadequate evidence of usability testing
- Little or no evidence that the project makes sensible use of software development tools and practices
- The quality of the report is extremely low

40-49%

- You have submitted some evidence of unit and/or integration tests, containing some bugs.
- Evidence of some test cases is provided but they are poorly designed. No edge cases or corner cases are considered.
- Evidence suggests limited understanding of software development practices.
- Evidence shows an attempt at a functional test plan, but it is limited in scope and not correctly presented.
- Evidence shows limited software development tools and techniques have been used, but mostly ineffectively.
- Limited evidence of usability (key aspects of what was undertaken are not clear).
- A basic report is submitted but it is not well organised and needs more detail.

50-59%

- Evidence suggests unit and integration tests are well implemented and mostly function without error.
- Evidence of basic test case design is reasonable but few or edge cases or corner cases are considered.
- Evidence suggests average understanding of software development practices.
- Evidence suggests functional test plan contains the correct information but is limited in the scope of its tests.
- Evidence of useability testing is present.
- Evidence of a range of software development tools and techniques have been used but to limited effect.
- Submitted report describes the work in some detail and follows a reasonable structure.

60-69%

- Unit and integration tests are fully implemented and execute properly.
- Well-designed test cases have been provided. Some edge and/or corner cases have been tested.
- Good understanding of software development practices.
- Strong evidence of usability testing is present.
- You have provided a generally complete functional test plan that is well presented.
- You have used a range of software development tools and techniques, mostly to good effect.
- Your report is mostly complete, lacking detail in only a few areas. The organization and presentation are good.

70%+

- Unit and integration tests are provided. All execute without error and the structure of the test suites is outstanding.
- The system is exhaustively tested – most or all conceivable cases are tested.
- High-level understanding of software development practices.
- An exhaustive functional test plan has been provided and its presentation is outstanding.
- Extensive evidence of usability testing is present (including detailed description of the method used, the number of participants, tasks used, questions used, results and modifications to the software suggested and implemented).
- Your use of software development tools and techniques is to a professional standard.
- You have produced an excellent report that documents the software well, is well written and correctly organized.

General Guidance

Extenuating Circumstances

There may be a time during this module where you experience a serious situation which has a significant impact on your ability to complete the assessments. The definition of these can be found in the University Policy on Extenuating Circumstances here: https://liveplymouthac.sharepoint.com/sites/x70/SitePages/Extenuating-circumstances.aspx

Plagiarism

This is not a collaborative piece of work. It is an individual work assignment and must be all your own work. Any plagiarism will be dealt with using the appropriate process; consult your student handbook if you are unaware of this as it could jeopardize your studies. This link leads you to a

comprehensive 6-minute talk by Jason Truscott on how to avoid plagiarism. https://plymouth.cloud.panopto.eu/Panopto/Pages/Viewer.aspx?id=6b1ef663-cf89-4de6-81af-acb6012c74b6

All of your work must be of your own words. You must use references for your sources, however you acquire them. Where you wish to use quotations, these must be a very minor part of your overall work. To copy another person's work is viewed as plagiarism and is not allowed. Any issues of plagiarism and any form of academic dishonesty are treated very seriously. All your work must be your own and other sources must be identified as being theirs, not yours. The copying of another persons' work could result in a penalty being invoked.

Further information on plagiarism policy can be found here:

- Plagiarism: https://www.plymouth.ac.uk/student-life/your-studies/essential-information/regulations/plagiarism
- Examination Offences: https://www.plymouth.ac.uk/student-life/your-studies/essential-information/exams/exam-rules-and-regulations/examination-offences
- Turnitin (http://www.turnitinuk.com/) is an Internet-based 'originality checking tool' which allows documents to be compared with content on the Internet, in journals and in an archive of previously submitted works. It can help to detect unintentional or deliberate plagiarism. It is a formative tool that makes it easy for students to review their citations and referencing as an aid to learning good academic practice. Turnitin produces an 'originality report' to help guide you. To learn more about Turnitin go to: https://guides.turnitin.com/01_Manuals_and_Guides/Student/Student_User_Manual

Turnitin, available here: https://help.turnitin.com/new-links.htm, is an Internet-based 'originality checking tool' which allows documents to be compared with content on the Internet, in journals and in an archive of previously submitted works. It can help to detect unintentional or deliberate plagiarism. It is a formative tool that makes it easy for students to review their citations and referencing as an aid to learning good academic practice. Turnitin produces an 'originality report' to help guide you.

Referencing

The University of Plymouth Library has produced an online support referencing guide which is available here: http://plymouth.libguides.com/referencing. Another recommended referencing resource is "cite them right" which is available here: Cite Them Right - Home (citethemrightonline.com). This an online resource which provides you with specific guidance about how to reference lots of different types of materials.