Module	Engineering1 (Eng1) - COM00019
Assessment Title	Assessment 2, Cohort 2
Team	Dragonite (Team 21)
Members	Omar Omar, Rhianna Edwards, Okan Deniz, Craig Smith, Omar Galvao Da Silva, Joel Wallis
Deliverable	Continuous Integration

Continuous Integration

Methods and approaches

General

- Continuous integration emphasises smaller more frequent commits and builds of a project to ensure the team is working on as similar code as possible throughout development.
- On the whole it helps to remove the risks of large commits being incompatible with other programmers code and relieves the anxiety this may cause a programmer about to commit a large section of code to the project. [2]

Badges

- Being able to create different versions and branches of the project has many benefits.
- Having different branches can allow for modular design to be done in isolation of other modules being worked on. These individual branches make sure a programmer does not change anything important for other programmers. Important changes that need to be made can be committed to the main branch which can be merged with these branches ensuring that they are available for each programmer.
- Version control is useful when considering errors and the iteration process. If a new version causes issues then looking at the changes between the versions can make it easier to find what is causing issues in the project. It also acts as an archive of progression for the project which might be of interest to stakeholders.
- It encourages modular design for the project which helps increase the code's quality.

Code Linting

- Linting is a powerful tool for creating and maintaining projects.
- For creation it highlights bugs in the project which would later prevent a build of the project. Having these be spotted in smaller chunks of code from the commits allows for easier, quicker and less stressful fixes to the project due to less errors needing to be looked into. This also reduces the strain on testing as only major issues should then need to be fixed at that stage rather than the smaller issues linting can help catch earlier on.
- Maintaining the code is aided through code standards and inconsistencies being detected. Future teams will now have to contend with messy code and will spend less time needing to decipher what our team has created.

Code Coverage

- Coverage gives a breakdown on how much of the project's code is used or utilised during its execution. This is displayed in reports broken down into function, statement, branches, conditions and/or line coverage.
- This increases the efficiency of the project by indicating how much of the is being used allowing for unneeded code to be pruned away; this reduces file size and execution time.
- It also highlights if certain code is being skipped due to conditions not being accessed. This allows for the conditional clause to be investigated by the team and the issue being fixed without seeing which part of the project is affected by this not being executed properly.
- For this project we are aiming for 80% code coverage. This ensures that the majority of the program will work without issues but also prevents time being wasted on reaching 100% code coverage which can be used on. [1]

Tools Used

- Travis
  - The team decided to use Travis as the continuous integration tool. Travis is a web based tool which builds the code on a virtual machine to test if the current commit will allow the project to build. It has Github integration which was easy to set up. Without needing to install anything or go through setting up plugins the time needed to get the tool working is lower than other tools, such as Jenkins, to set up and understand. The virtual machines also allow us to run the project on different operating systems; we can use this to test the project on different platforms that our client may wish to use the project on. [3]
  - 

- IntelliJ
  - Our IDE comes with coverage testing. It has its own coverage calculator and the option to use 3rd party calculators such as JaCoCo. As we were already using IntelliJ for writing the code we decided it would be easier to use a program we already are already familiar with than use a separate calculator.
  - The code coverage is able to run on individual tests which can help with modular testing of the project.
  - Evidence of code coverage is found in the Software Testing Report

Bibliography

[1] https://www.atlassian.com/continuous-delivery/software-testing/code-coverage
[2] https://bitbar.com/blog/top-continuous-integration-tools-for-devops/
[3] https://docs.travis-ci.com/