Date: 13/01/25

UniTycoon
TESTING

Group 6

TEAM6 Game Studios

Work Undertaken by:

Thomas Koukouris
Adam Khan
Oliver Herron
Sam Jordan
Nathan Hopper
Fergus Irvine

Testing Method and Approaches:

The ideal testing method for the code would be split between manual and automatic, the manual being actively working through the code and spotting various errors, launching the exe and spotting errors and working through them, etc. and automatic would be largely based in UnitTests. While we had been working on UnitTests, with some of them working with parts of the files of the code, due to constant changes to the code and failure for packages to properly instantiate between each other they were quickly redundant. While we do have some UnitTests, and retain them in their folder, they don't work in the code for reasons we don't quite understand given 2 ran fine for a while, so it was largely error handling the main code such that it (the main code) would work. We assume there's something wrong with the dependencies.

In that regard our approach to error handling and Unit testing was mostly manual, the tester went through the code to manually adjust it accordingly.

In manual testing the approach was through someone that is aware of both testing and software engineering can parse through the code, adjusting it where possible as per the edge/test cases, the tester based on the documentation provided regarding the code and the code itself edited parts of the code such that it wouldn't be likely to cause an issue in the future. Error handling was done through methods, to make them more secure and streamlined in catching out errors via observing the edge cases and refactoring them into try, excepts where possible. Basic if statements, try-excepts and switch cases were used where possible to make the code meet the aforementioned requirements (functionality, readability and bug free code completion), toward the end based on user requirements the tester adjusted certain things within the timeframe to make them as suitable to the Players as possible. This degree of testing was also applied to the website.

The UnitTests that were made but not able to properly be implemented are in the website alongside a brief description of testing the product.

A Report on the tests:

UniSim:

- ErrorHandling: the main method of testing was done via error handling through the code, as previously mentioned due to the failure of properly linking implementation to unitTests, largely due to issues with the dependencies we were unable to implement the tests properly within the timeframe. The method of going through error handling was based on multiple ideas – Clarity and Readability, Error catching and quality of life, these make up the overall performance of the code. The tester ensure that the code was working, ways to increase clarity, such as minimising the amount of statements, or efficiently refactoring a loop, changing the method outright and reducing the amount of clutter (i.e pieces of code not necessarily useful, or that can be refactored into its own method). Error catching was also a priority, ensuring that any potential bugs or errors that may arise would be caught via try, excepts, or if.. statements that would point out the error, or in some capacity affect the code so that it would prevent any game breaking bugs. For instance there was a missing texture when we ran the game in FinalScreen which the try except caught and we used that to identify the issue and readded the texture to the exit button. Overall this ensured that the code was as good as we could get it to be. Another important aspect of error handling were the visual and audio feedback the player gets while using the game, this was fairly easily done via manual handling as we played through the game multiple times with various people and got feedback that we acted on, from actual bugs to enhancing the feedback from the game. Some of these were the firstscreen.java for instance, for an intuitive UI we made it short and simple with the space bar being the key for navigation, the method show() for instance of how the testing was, was nested inside a try except to catch any errors and point them out without breaking the game or code, and housed the textures and writing to illustrate the area of the game a player was in. The general heuristics are what we kept up and had in mind throughout the error handling process alongside bug fixes.

- UnitTests: While unit tests couldn't be integrated, there were some unitTests that were made and put into a folder during the time we were trying to figure out what exactly was wrong with it. Mostly the tester wanted to test one specific functionality in representation of the file, however other tests were instantiated in conjunction. UnitTests were largely focused on the events, however some were created for files outside the events, such as to make sure music was working or elements of the menu were up to snuff.

Website:

The method for testing the website was simply seeing if it was functional, based on user critiques we adjusted certain elements of the site to finalise and create a unique and stand out product. There were some issues along the way, but that's where the testing came into play, alongside generally making it more robust and user friendly, the issues we had we solved as to abide by the general heuristics of a navigational website. For example, there was an issue exclusive to mobile devices where the about section of the webpage got stuck, so we manipulated the css for it to function.

Conclusion:

Despite the shortcomings of the unittest failing to be imported, some were made regardless to show evidence of the knowledge of the coding ideas behind the unit Tests, the manual testing makes up for the functionality they would've offered in some capacity. The code was adjusted accordingly, and methods were followed to ensure that the code was readable and functional so as to launch the game with minimal amounts of problems. Additionally, the visual and narrative feedback of the game was also a priority while errorhandling and bug fixing, that is the central focus, so we did everything we could to maximise this.