Requirements

Group 7

Charlie Meader, Sylvia Bulch, Emilija Dudko, Sam Laljee,Eisvinas Daujotas,Max Sweetman

Introduction

Functional requirements are written down to allow project members to stay focused and be on the same goal for specific tasks required in the system by the customers. These functional requirements are useful to look at before development to spot any potential new requirements linked to already existing ones and potentially spot errors.

We have chosen to include maintainability requirements in our requirement list as these are essential in the development of software that is easy to understand, as well as being easy to find anything that needs to be changed, make said changes, and then also ensure that these changes have not resulted in additional bugs being introduced to our software. These ensure that when developing software, project members develop it in a way that allows it to be easily read by other project members, as well as debugged, should the need arise. We are also including portability requirements as these detail which system our software is expected to run on, as well as whether the software can be moved between different systems.

Security requirements are used in order to know how much data about the user will be collected and how it will be processed. This could include any account information for the user and how it is presented publicly, or how it is stored for example password processing and secure/encrypted storage. Security requirements also consider any vulnerabilities that the software could have.

Performance and reliability requirements describe how fast the software should be running as well as how crucial it is for the software not to run into any issues or crashes. In the context of a video game this can also include details on how quickly the system should respond to user input.

The requirements are listed in the following way, with a letter representing the type of requirement, followed by a number representing the individual requirement ID, for example F.1.1 would represent “The avatar should be able to perform a restricted number of activities per day depending on the time or energy”. [1]

Sorted requirements

Functionality requirements(F)

In the product brief some basic functional requirements outlined which we have to implement are:

1. The game should last 7 days.
    - 1.1. The avatar should be able to perform a restricted number of activities per day depending on the time or energy.
    - 1.2. The day will end when you sleep.
    - 1.3. The game will end when the 7th day is over.
    - 1.4. The game doesn’t progress by real-world time only by actions performed.
    - 1.5. Each day contains 16 hours (and the rest is used for sleep)
2. The game must have a map for the player to move an avatar around in which includes some buildings on or off campus.
3. The avatar should be able to interact with specific objects that are next to them which are relevant to the game.
4. There should be a certain amount of buildings for the player to interact with and to perform activities that affect the game
    - 4.1. At least one building should allow the player to study.
    - 4.2. At least one building should allow the player to sleep.
    - 4.3. At least three buildings should allow the player to do recreational activities
    - 4.4. At least one building that allows the player to eat

5. The game should give the player a score upon finishing the game that is calculated by how well they performed in their exam.
6. Their exam performance is calculated by how much the player has studied before the 7 days run out and is affected by several factors.
    - 6.1. Whether the player has studied in different buildings
    - 6.2. Studying more than the minimum daily requirement

    6.3. Eating enough food daily
    6.4. How well the player balances recreational activities with studying

7. The student can perform activity with each activity relating to the following
    7.1. Each activity consumes time and energy
    7.2. Each day the student has an energy bar.
    7.3. Performing an activity will consume a percentage of energy that the student has for the day, depending on activity.
    7.4. Performing an activity will also move time forward by hours depending on the task.[2]
8. Allow the character to be controlled with the WASD keys or the arrow keys
9. Allow the character's score to be calculated and displayed at the end of the game
    9.1. Every time the character studies, their energy decreases.
    9.2. Every time the character eats, their energy decreases.
    9.3. Every time the character interacts with the lake, their energy decreases.

Performance/Reliability requirements(P)

1. The game must be able to run on a medium specification computer with a smooth number for frames per second.

2. In the product there shouldn't be any game breaking bugs/issues which would potentially crash the pc or make the game unplayable.

Security requirements(S)

1. The game will not save any user data, e.g. the username and password, as it will not have a login page

Maintainability requirements(M)

1. Use regular comments throughout the code as well as readable variable names and descriptive code comments in order to ensure that it is easy for a team member to pick up where another finished.

2. Make it easy to check that changes made have not introduced bugs, as well as ensuring that previous versions are saved, meaning that the program can be rolled back to a different version if necessary.[3]

Portability requirements

1. The game should be displayed properly for all screen ratios and resolutions using adaptability in the program.

Target audience

The target ages for the audience for the game should be around 16 to 20 years old.

References

[1] K. Jędrzejko, “Software requirements engineering – the driving force behind successful IT projects ,” Software Mind, https://softwaremind.com/blog/software-requirements-engineering-the-driving-force-behind-successful-and-efficient-it-projects/ (accessed Feb. 22, 2024).

[2] A guide to functional requirements (with examples), https://www.nuclino.com/articles/functional-requirements (accessed Feb. 22, 2024).

[3] S. Crouch, “Developing maintainable software,” Software Sustainability Institute, https://www.software.ac.uk/guide/developing-maintainable-software#:~:text=The%20maintainability%20of%20software%20depends,have%20not%20introduced%20any%20bugs (accessed Feb. 22, 2024).