Requirements

Cohort 3 - Group 4
AJAJARA

Adam Fuller
James Goude
Abbie Spears
Rosie Sherwood
Artem Frolov
Alexander Gibson

Our requirements engineering and elicitation process conform to the standards outlined in the IEEE requirements engineering documentation [1]. Following extensive research of proposed standards for software design and requirements elicitation, we collectively agreed that the standards presented in this document were the most descriptive in nature and lent themselves to an intuitive and accessible categorization of our user's needs.

The main incentive behind this decision was the usage of the modal verbs 'May', 'Shall' and 'Should' as descriptors for the priority of our requirements [1, p.19], which unequivocally outlines the importance of each requirement in terms that are comprehensible to both the design team and the client. This idea is supported by some of the most advanced organisations in the technology industry, namely NASA, who employ this standard in their own requirements matrix [2], emphasising the robust nature of this documentation quality.

The requirements were outlined to the team in a meeting with the client, who requested a multitude of features to be included in the final iteration of our product. These specifications have been refined by the team as a collective, ensuring that the resulting components *'solved a problem, achieved a goal or addressed a stakeholder concern'* and most importantly were *'qualified by measurable conditions'* [1, p.18]. The latter of these two factors will become increasingly important during the testing phases of our software engineering where results will need to comply with agreed constraints to measure their success. The initial requirements proposed are often subject to change, with 42% of similar projects facing issues with the articulation of user needs [3, p.14], however in order to mitigate this possibility, we will be holding further meetings with the client in order to validate current requirements and ensure they satisfy their requests.

Within our requirements table, the elicitations at a user level are broken down into functional (things the system must do) and non-functional requirements (qualities and properties the system must have). It has been noted that the Agile software methodology can conflict with the documentation and delivery of non-functional requirements as it *'prioritises the incremental delivery of functional features'* [4, p.1]. As a team, we feel that this issue is addressed appropriately in the IEEE documentation [1, p.24] and have recognised these requirements prior to the design process in order to maximise functional output without sacrificing the utilities desired by the user. Any Assessment 2 requirements that were not included in the original Req1 document have been added into the table and are highlighted in blue for clarity. Any yellow highlighted text indicates where existing text has been modified for Assessment 2.

The research considered has developed our understanding of the requirements elicitation process and provided us with measures to accurately identify user needs and describe their measurement in the testing stage of our design. Our findings will provide us with a rigid foundation from which to draft and build our product for the client and we are confident that issues relating to user requirements can be addressed promptly through effective communication between the team and the stakeholder, aided by weekly sprints under the agile methodology employed in our design.

User requirements

ID	Description	Priority
UR_OS_COMPATIBILITY	The game shall run on Windows, Mac, and Linux operating systems.	Shall
UR_HARDWARE_COMPATABILITY	The game shall run on minimal specification hardware.	Shall
UR_DISPLAY_SCALABILITY	The game should run correctly on different DPI displays.	Should
UR_COLOURBLIND_ACCESIBILITY	The game shall be accessible to colour blind users.	Shall
UR_PAUSE	The user shall be able to pause the game.	Shall
UR_TIMER	The game shall run for 5 real-world minutes, representing 3 in-game years.	Shall
UR_TIME_TRACKER	The game should use weeks and semesters so the user can track the in-game time.	Should
UR_AUDIO	There should be background music and sound effects.	Should
UR_COUNTER	The game shall have a counter with the amount of buildings that have been placed.	Shall
UR_LOCATIONS	There should be at least one building type to sleep, one to learn and one to eat. There should be at least two different types of buildings/areas for recreational activities.	Shall
UR_LOCATIONS_MOVE	The user should be able to move buildings once they've been placed.	Shall
UR_LOCATIONS_REMOVE	The user should be able to remove buildings once they've been placed.	Shall
UR_LOCATIONS_SIZES	Different locations shall be different shapes and sizes so the user can easily differentiate them.	Should

UR_MAP	The user shall be shown a map including different geographical features to build their university around.	Shall
UR_LOCATIONS_UPGRADE	Buildings may be able to be upgraded to allow additional features or capacity.	May
UR_MENU	Game should start with a menu screen.	Should
UR_TOOLTIPS	The game may have tutorial-style tooltips.	May
UR_SATISFACTION	Student Satisfaction Score will determine if the player wins or loses and if they place on the leaderboard. Will be determined by the player's in-game choices. Will be visible to the player throughout playthrough.	Shall
UR_WIN	Game will determine if the player has won or lost based on student satisfaction. Will be informed if they won or lost at the end of their playthrough.	Shall
UR_EVENTS	Will have at least 3 events that occur during the playthrough.	Shall
UR_LEADERBOARD	Will display the names and scores of the top 5 scorers.	Shall
UR_ACHIEVEMENTS	Will include an achievement system which can alter the player’s final score.	Shall
UR_CONTROLS	The user may be able to customise their controls	May

Functional requirements

ID	Description	User Requirements
FR_STOP_TIME	The user should be able to stop the time at any point during the game. The user should not be able to place buildings while the game is paused.	UR_PAUSE
FR_AUDIO_MUTE	The game should have a button that	UR_AUDIO

	allows the user to mute the sounds.	
FR_LOCATIONS_PLACEABILITY	The system shall clearly display on the grid whether a building can be placed.	UR_LOCATIONS
FR_COUNT_DISPLAY	The system shall update a counter of building types to the user each time one is placed.	UR_COUNTER
FR_LOCATION_PLACEABILITY	The system shall show to the user whether they can place a building in a location on a grid or not.	UR_LOCATION_PLACEABILITY
FR_LOCATIONS_SIZE	Different locations should take up different amounts of squares on the grid.	UR_LOCATIONS_SIZE
FR_TIMER_DISPLAY	The timer shall display the in-game time in months and semesters in accordance with the user's wishes.	UR_TIMER
FR_SOUNDTRACK	The game should play a soundtrack when the title screen is loaded.	UR_AUDIO
FR_CONTROLS_CUSTOM	There will be options in settings for the user to easily customise their experience.	UR_CONTROLS
FR_EVENT	The effect of events within the game will be impactful but also fair. Users should be aware the event has occurred.	UR_EVENTS
FR_USERNAME	User should be able to choose a username at the start of the game.	UR_LEADERBOARD
FR_DATABASE	The username and score of users whose score puts them on the leaderboard will be saved after each game.	UR_LEADERBOARD
FR_RESET	In-game currency and satisfaction score should be reset after each play-through,	UR_LEADERBOARD
FR_SATISFACTION_VARIABLES	Student satisfaction will take into account the variety, proximity and location of different buildings placed by	UR_LOCATIONS UR_WIN UR_SATISFACTION

	the user. Also affected by the user's reaction to the events.	UR_EVENTS
FR_CURRENCY	Will implement an in game currency to add challenge to the game. Placing more buildings will require more money - certain buildings will earn the player more money.	UR_LOCATIONS UR_WIN
FR_LOCATIONS_MOVE	The system shall allow the Player to left click and drag an existing building to move it.	UR_LOCATIONS_MOVE
FR_LOCATIONS_REMOVE	The system shall allow the Player to right click on an existing building to remove it.	UR_LOCATIONS_REMOVE

Non-Functional requirements

ID	Description	User Requirements	Fit Criteria
NFR_JVM	The system shall support the JVM used by all 3 operating systems to provide compatibility between OS.	UR_OS_COMPATIBILITY	Complete a successful run-through of the game on all 3 systems.
NFR_DISPLAY_SIZES	The game should be playable on both laptop and large whiteboard-sized screens.	UR_DISPLAY_SCALABILITY	Test game on varying display sizes to ensure textures scale appropriately.
NFR_HARDWARE_COMPATABILITY	The game should run on minimal specification hardware, as expected to be used by the average user.	UR_HARDWARE_COMPATIBILITY	Test game on personal laptops which reflect average user setup.

NFR_PLAYABILITY	The game should be operable by users who have picked the game up for the first time.	UR_TOOLTIPS	Have testers who have never played the game before play and collect their feedback on which game mechanics they found intuitive or unintuitive and provide guidance on its usage.
NFR_COLOURBLIND_ACCESSIBILITY	The game shall be playable by users who are visually impaired or colourblind.	UR_COLOURBLIND_ACCESSIBILITY	Interview a colourblind QA tester to check if the game is accessible for their needs.

References

[1] ISO/IEC/IEEE, *Systems and software engineering — Life cycle processes — Requirements engineering*, 2nd ed. 2018.

[2] NASA, *NASA Software Engineering Requirements: Appendix C - Requirements mapping matrix*, 2022.

[3] Cristina Palomares, Xavier France, Carme Quer, Panagiota Chatzipetrou, Lidia López, Tony Gorschek, *The state-of-practice in requirements elicitation: an extended interview study at 12 companies*, Springer-Verlag London, 2021.

[4] Aleksander Jarzębowicz, Paweł Weichbroth, *A Qualitative Study on Non-Functional Requirements in Agile Software Development*, Gdańsk University of Technology, 2021.