Requirements

Cohort 3 Group 6 - Carbon Goose

Members: Bailey Horsley, Owen Jones, Rory Ingram, Ken Jacob, Abishek Kingslin Raj, Louis Polwarth, Adam Johnson

As a team we discussed what requirements and details needed to be addressed, particularly with the implementation team about the game specifications. We reviewed similar games to familiarise ourselves with the different art styles used (e.g. Pixel art) as the assessment brief did not specify this. Everything discussed was presented as questions in our interview document, which was later referenced whilst conducting the client interview. The questions were used to clarify certain ambiguous aspects of the project brief and specific system requirements as well as to collect feedback on our client's needs and wants.

We presented our findings from our research to our client to negotiate and draw out what elements we wanted to be incorporated into our game. With the help of proposing varied ideas and game mood boards, we tackled the issue of clients not being clear on what they wanted from the game and allowed our client to make more informed decisions about the game. Prior to the commencement of the interview, both parties agreed on audio recording the session. As a team, we took down notes and answers for respective questions on our interview document, allowing us to listen back and ensure that the requirements would be exactly meeting what was specified.

Our single statement of need was derived after our interview with our client. This was because we needed to gather insights into the context of the game and base it on our client's objectives. The SSOS was later proposed to our client and was finalised.

"The game shall enable young people, mainly those in sixth form, to have a short, simple and fun experience while building their own dream campus."

After eliciting the requirements, we formatted them into systematic and logical tables, allowing the architecture aspect of the project to efficiently build upon this using the clear identification codes. This was inspired by Hull et al.'s Table 3.3 [1], where requirements elicited from the client viewpoint were translated into clear tables including respective requirements' description, label/identifier (ID) and priority type.

The requirements were split into 3 distinctive tables, categorised as Functional requirements (FRs), Non-functional requirements (NFRs), and User requirements (URs). By doing so, it allowed us to clearly identify what was required for the system capabilities (using FRs and NFRs) and what the client's objectives were for the game (with URs). We agreed upon a priority system ("shall"/"should"/"may") to ensure we implemented the requirements that were most essential to meeting the client's needs.

It also makes the project traceable as it becomes easy to monitor the extent requirements were met. By having the fit criteria for the NFR, it provides tangible measurements for the success of the project, ensuring we are meeting the stakeholder's needs beyond the extent of just adding functionality. It also allows those working on the project in future to be able to see the reasons for implementing certain functions and details the way they were and ultimately prevent them from straying too far away from the client's original requirements.

[1] **Hull, E., Jackson, Ken, & Dick, Jeremy. (2011).** ***Requirements engineering [electronic resource] / Elizabeth Hull, Ken Jackson and Jeremy Dick.*** **(3rd ed.). Springer.**

User Requirements

ID	Description	Priority
UR_FIVE_MINUTES	The user can play the game in five minutes	Shall
UR_PAUSE	The user must be able to pause the game completely whenever they want to	Shall
UR_INFLUENCE_SATISFACTION	The actions of the user must be able to influence the student satisfaction metric	Shall
UR_SETTINGS	The user should be able to alter their experience playing the game via game settings	Should
UR_BUILDING	The user will be able to create a campus through the creation and interaction with buildings and roads	Shall
UR_TEACH_DURING_GAME	The user should be able to learn how to play the game without the need for a tutorial	Should
UR_DEVICE_ACCESSIBILITY	The user will be able to access the game on different types of devices	Shall
UR_CLEAR_DISPLAY	The user will be able to understand what the visuals they see on the map mean	Shall
UR_QUICK_START	The user should be able to get into the to playing of the game without having to first wait too long	Should
UR_BASIC_UI	The user can interact with the game via the keyboard or mouse	Shall
UR_FLEXIBLE_UI	The user can interact with the game using either the keyboard and mouse depending on which is more convenient for the user	Should
UR_SHORTCUTS	The user has access to shortcuts to make it faster to carry out some of the actions in the game faster	May
UR_EVENT_INTERACTION	The user will able to interact with a variety of planned and unplanned events	Shall
UR_LEADERBOARD	The user will be able to view a leaderboard with the names and scores of the top 5 players	Shall
UR_ACHIEVEMENTS	The user will unlock achievements for specific milestones (e.g., maintaining 80% satisfaction)	Shall

System Requirements

Functional Requirements

ID	Description	User Requirements
FR_OPERATING_SYSTEMS	The game will work on Linux, Mac, and Windows operating systems	UR_DEVICE_ACCESSIBILITY
FR_TEXTUAL_DESCRIPTIONS	The game provides textual descriptions during gameplay to help the user understand how to play	UR_TEACH_DURING_GAME
FR_CALCULATE_SATISFACTION	Distance of buildings and events will influence student satisfaction	UR_INFLUENCE_SATISFACTION
FR_COMPLETE_PAUSE	Pause will halt the timer, stop new events and user game actions from occurring, other than unpausing	UR_PAUSE
FR_EVENTS	The game will generate events randomly as well as having fixed events that every player will encounter	UR_EVENT_INTERACTION
FR_TIMER	The game will count down the 5 minutes and display the time remaining on the screen	UR_FIVE_MINTUES
FR_MAP	The game will have a map where buildings be located	UR_PLACE_BUILDING
FR_MUTE	There is an option to directly mute the game's sound	UR_SETTINGS
FR_ALTER_VOLUME	The user can adjust the volume of the game to go higher or lower than it starts off being	UR_SETTINGS
FR_PLACE_BUILDING	The user will be able to place down buildings on the map	UR_BUILDING
FR_MOVE_BUILDING	The user can freely move the location of buildings on the map	UR_BUILDING
FR_HOVER_INFO	Upon hovering over buildings with a cursor, information about the building and the related student satisfaction stats will be provided	UR_TEACH_DURING_GAME
FR_START_GAME	The game will have a button to press to begin	UR_QUICK_START
FR_MOUSE_INPUT	The game should contain several assets which can be interacted with using the mouse, such a clicking on a building	UR_BASIC_UI
FR_KEY_INPUT	The game should be able to handle keypresses appropriately throughout the entire duration of the 5 minutes playing it	UR_SHORTCUTS
FR_ROAD	The game must allow the user to build roads and paths between buildings which students can travel across	UR_BUILDING
FR_LEADERBOARD	The game will maintain a leaderboard that displays the names and scores of the top 5 players.	UR_LEADERBOARD
FR_ACHIEVEMENTS	The game will unlock achievements for specific actions and modify the final score	UR_ACHIEVEMENTS

Non-Functional Requirements

ID	Description	User Requirements	Fit Criteria
NFR_CLEAR_GRAPHICS	The game's graphics will be clear and have contrast between buildings	UR_CLEAR_DISPLAY	Users will have negligible difficulty distinguishing the appearance of buildings
NFR_SCALABLE	The game will be able to scale to fit various screen resolutions and aspect ratios	UR_DEVICE_ACCESSIBLE	The game works on both laptops and TV-sized monitors
NFR_SIMPLE	The game must be simple enough to be able to be picked up easily	UR_TEACH_DURING_GAME	A new player must be comfortable with playing the game within 60 seconds
NFR_ENJOYABLE	The game must provide an enjoyable experience to the user in a short period of time	UR_FIVE_MINUTES	The majority of players find their experience was enjoyable by the end of the 5 minutes
NFR_FORGIVING	The game will allow the player to alter previous decisions and mistakes, such as misplaced buildings	UR_BUILDING	There are no complaints about from users about the game unfairly punishing them for previous decisions
NFR_FAST_LOAD	Assets should start loading before the user has decided indicated they want to begin	UR_QUICK_START	There should be no significant wait time for the game to start
NFR_CONCURRENCY	The game should be able to handle both mouse and keypresses that may be occuring at the same time and possibly conflicting	UR_FLEXIBLE_UI	The game should handle simultaneous mouse and keyboard input without crashing as though they were done separately
NFR_PLAYABLE	The combination of all of the functionality must lead to a game which can be played from start to finish	UR_FIVE_MINUTES	The gameplay must not contain any breaks or bugs in it that prevent the user from playing it completely
NFR_VARIETY	There must be a variety of types of buildings that can be placed	UR_BUILDING	The game must contain one building type for food, one for accommodation, one for education, and **two** for recreational activities
NFR_INTUITIVE	The leaderboard and achievements must be visually clear and easy to interpret.	UR_LEADERBOARD, UR_ACHIEVEMENTS	Users will understand their achievements and leaderboard position without needing additional explanation.