Requirements

Group 9

Pluto Pioneers

James Lawson, James Rooke, Dema Williams, Nicholas Barker,
Sam Slade, Joshku Gauntlett, Seiya Ikeda

Requirements Introduction and Explanation

For our Project, we elicited requirements based on succinct techniques specified from textbooks such as “Writing effective use cases” by Alister Cockburn and “Software engineering” by Ian Sommerville. “Requirements engineering involves three key activities. These are discovering requirements by interacting with stakeholders, converting these requirements into a standard form and checking that the requirements actually define the system that the customer wants” [1]. We took information from our user’s product brief and interviews we had with our client and created tables to show the requirements of the User and the System itself. We also came up with a single statement of need for our project.

We got User and System requirement definitions from the textbooks. From “Software Engineering”, User and System requirements are defined as follows: “User requirements are statements, in a natural language plus diagrams, of what services the system is expected to provide to system users and the constraints under which it must operate. The user requirements may vary from broad statements of the system features required to detailed, precise descriptions of the system’s functionality” [1] “System requirements are more detailed descriptions of the software system’s functions, services and operational constraints. The system requirements document should define exactly what is to be implemented. It may be part of the contract between the system buyer and the software developers” [1]. Often, one user requirement will facilitate many software requirements. We tried to follow these definitions accurately, whilst making our requirements.

It was important to write our user requirements using natural language sentences, this meaning that each requirement was written so that it didn't require any technical knowledge to understand and that each sentence should express only one requirement. We presented our requirements using tables which store information about the requirements, such as giving them an ID and a description. We separated the system requirements into functional and non functional requirements. Every system requirement was formed from a user requirement. We presented the requirements in this format as we believed it would be the most simple and effective method to document them, writing them in such a way that they all could easily be identified and understood by anyone who might need to. It also acts as a checklist for our team that is making the project to look back upon and thus makes sure we are meeting all the criteria that our customer gave us.

We created use cases to show the system’s behaviour. In a use case, an ‘actor’ initiates an interaction with the system to accomplish some goal. This shows the system's effectiveness in good or bad situations. Use cases are, in every sense, requirements “when properly written they accurately detail what the system must do”. However, “They are not all the requirements”. They constitute only a small amount of all the requirements you need to collect, so it is important to not solely rely on them [2]. The use cases are available to view on the website.

Single Statement of Need

The system will be a game targeted at 16-20 year olds in which they have a week to prepare for exams and must balance their time between studying, eating, doing recreational activities, and sleeping.

User Requirements

ID	Description	Priority
UR_map_movement	The game shall enable the player to move the character around the map and interact with entities on said map.	Shall
UR_time_limit	The game shall last for 7 days and each day the player will perform a number of activities before running out of time.	Shall
UR_activity_places	The game shall include: places to study, place to sleep, recreational activities, places to eat.	Shall
UR_score_system	The game shall provide a score at the end based on how effectively the player has studied.	Shall
UR_energy_system	The game shall include an energy mechanic where the player has limited energy to spend on tasks.	Shall
UR_energy_consumption	The game shall consume energy and time for each activity.	Shall
UR_load_time	The game should have minimal load times.	Should
UR_UX	The game should offer a pleasing user experience.	Should

System Requirements

Functional

ID	Description	User requirement
FR_character_control	The game shall allow the player to control a character, with keyboard controls	UR_map_movement
FR_map_movement	The game shall allow the character to move around a map with the keyboard controls	UR_map_movement

ID	Description	User requirement
FR_entity_interaction	The game shall allow the character to interact with entities on the map with the mouse	UR_map_movement
FR_entity_activities	The entities will allow the character to perform certain activities	UR_map_movement
FR_Time_Calculation	The game should have designated time to preform and then be used to calculate remaining time	UR_time_limit
FR_time_costs	Different activities shall have different time costs	UR_time_limit
FR_7_day_game	The game shall have 7 in-game days	UR_time_limit
FR_day_times	The game day shall be 16 hours long, plus 8 hours dedicated to sleep	UR_time_limit
FR_study_places	The game shall include at least one and no more than 2 places to study	UR_activity_places
FR_sleep_places	The game shall include one place to sleep	UR_activity_places
FR_recreational_places	The game shall include at least 3 and no more than 6 places for recreational activities	UR_activity_places
FR_eating_places	The game shall include at least one and no more than 3 places to eat	UR_activity_places
FR_Score_Calculation	The game should calculate a score based on amount studied and time spent relaxing	UR_score_system
FR_different_location_score	The game’s score will be better if the player has studied in multiple location over the days	UR_score_system
FR_study_time_improvement	The game’s score will be better if the player has studied more than the minimum amount	UR_score_system
FR_study_time_regression	The game’s score will be worse is the player has over studied	UR_score_system
FR_Eating_Score_improvement	The games score will be better if the player eats at reasonable times	UR_score_system
FR_recreational_score	The games score will be better if the player performs recreational activities	UR_score_system
FR_energy_allocation	Each day will have a certain amount of energy that the player will be able to use and won’t be able to use more energy than allocated	UR_energy_system
FR_Energy_Calculation	The game should have each action lose energy when performed	UR_energy_consumption

ID	Description	User requirement
FR_energy_costs	Different activities will have different energy costs	UR_energy_consumption

Non-functional

ID	Description	User Requirement	Fit criteria
NFR_loading_time	The system shall have minimal load times	UR_load_time	Excluding initial start up, nothing should take longer than 0.5 seconds to load
NFR_action_response	Any action the user takes should be indicated visually to the user	UR_UX	Any result from an action should take place within 0.5 seconds
NFR_user_interaction	The User interface should be clean and easily understandable	UR_UX	THE UI should be kept as simple as possible and not overload the player with information

Reference list

[1] I. Somerville, *Software Engineering, Global Edition*. Pearson Education, 2015.

[2] A. Cockburn, *Agile Software Development*. Harlow, England: Addison Wesley, 2002.