Implementation 2

Cohort 4 Group 6
Javengers

Braithwaite, Max
Faruque, Amber
Fu, Zhuoran
Kocaman, Melike
McDermott, John
Rissen, James
Scott, Charlotte

Blue highlight means changed for part 2, no highlight is unchanged.

D2a - Introduction

Elicitation

Requirements were created and organised according to the requirements of the customer during the customer meeting. We organised a meeting with our customer and planned a series of questions (Planning & Notes) to ask so we could clarify what the user wanted from our game and gauge an idea of what was most important to them.

Presentation

All the requirements were given a priority according to the importance discussed with the customer whose priority took precedence over negotiation between the assigned team members. We decided to display the requirements using a table for each group of requirements to aid readability and make it accessible to the whole group.

We split requirements into 4 groups, *user*, *functional*, *non-functional* and *constraint requirements*.

Each is given an ID and description. Giving the requirements a unique ID allows us to reference them efficiently throughout the project so we can ensure that as many of the requirements as possible are met. User requirements are displayed with their respective priority:

- *Shall* - The game must include this,
- *Should* - The requirement is encouraged, or
- *May* - The requirement is desirable.

Functional requirements are displayed with their respective preceding user requirement. Non-functional requirements are displayed with their preceding user requirement and also a criteria for success according to their description. constraint requirements, due to their nature are displayed with their ID and description alone, they are implied as mandatory requirements. We chose to display the tasks in this tabular manner so we could easily see all the requirements and for ease of editing in concurrent customer meetings.

Research

All members of our group watched the video on requirements engineering on the VLE and the members of the requirements team read the chapter about requirements from the Software Engineering textbook by Ian Sommerville [1] to further understand the types of requirements and how to present them most effectively. We took the layout of the table from the video lecture about requirements engineering that we all watched on the VLE as we all thought that that would be the most efficient way to display all of the information needed surrounding the requirements. We also referred to the IEEE Guide to Software Requirements Specification [2] for further clarification about what is to be included.

For the following part of the project, our research was mainly gained from playtesting that was conducted by volunteers. The volunteers played the game and produced a set of data as listed on the evaluation document which can be found on our website, this was directly used to aid in our addition of new requirements. As a group, we would brainstorm solutions to the problems raised by playtesters, and then smaller subsections of the group would refine these ideas resulting in a more developed set of requirements, more tailored to the intended user group (a category which our playtesters fell into).

D2b - Statement of Requirements

Single Statement of Need (SSON)

The game should be a single player maze-style game game to simulate escaping university with hidden and visible events found along the way that trigger events.

Requirements

User Requirements		
ID	Description	Priority
UR_LOADING	Load the game on Windows, Mac and Linux easily	Shall
UR_CHARACTER_CUSTOMISATION	Players can customise some parts of their character	May
UR_MAP	Player moves through various area on a hidden map	Shall
UR_EVENTS	Events are triggered when the player hits some visible and hidden checkpoints	Shall
UR_EVENTS_LOCATION	Hidden and visible events to be in random positions each time the game is played	Should
UR_ENERGY	Energy or health levels for the player that change due to events	May
UR_EVENTS_LINKED	Visible and hidden events to be linked and dependent on each other	May
UR_MUSIC	Background music that can be controlled (eg muted)	Shall
UR_SOUND_EFFECTS	Sound effects triggered by special events	Should
UR_INSTRUCTION	One page instruction screen	Shall
UR_TIMER	Timer counting down while playing	Shall
UR_TIME_LIMIT	Game should be played to completion in under 5 mins	Shall
UR_DISPLAY	Suitable to be displayed on a large monitor while playing	Shall

UR_COLOURS	Game accessible for people who are colourblind	Shall
UR_CONTROLS	Player can use the mouse or keyboard to move the character	Shall
UR_CAMERA	The camera remains stationary and the character will move around the screen	Should
UR_MENU	There is a menu system for the user to operate and navigate the game using.	Shall
UR_UI	There should be a pleasing UI for the user to interact with, it should be easy for User to navigate the system	Shall

Functional Requirements		
ID	Description	User Requirement
FR_TITLE_SCREEN	The system will display the title screen when the game loads.	UR_LOADING
FR_CHARACTER_CUSTOMISATION	The character chosen has elements that can be changed and customised.	UR_CHARACTER_CUSTOMISATION
FR_CUSTOMISATION_MENU	The game needs a selection menu for picking the character customisations.	UR_CHARACTER_CUSTOMISATION
FR_MAP	The game has a series of hidden areas that the character can move through and between. The player must be able to distinguish between these tiles.	UR_MAP
FR_LIMITS	The character has a clear limit of where they can go in the game.	UR_MAP
FR_GOOSE_PIECES	The player can locate pieces of the stolen goose statue in order to complete the game.	UR_EVENTS
FR_POSITIVE_EVENTS	When the user moves to a certain location then it will trigger a positive effect that benefits the character (x3)	UR_EVENTS
FR_DRINK_POSITIVE	The player can drink something from a vending machine that gives them a temporary buff.	UR_EVENTS
FR_SCORE_BONUS	The player can collect items around the map that directly increase their score.	UR_EVENTS
FR_MAP_ACCESS	The player can gain access to a one time map that allows them to see where they need to go.	UR_EVENTS
FR_NEGATIVE_EVENTS	When the user moves to a designated area on the map, it will cause a negative event to happen that hindrances the gameplay (x5)	UR_EVENTS

FR_LIGHTS_NEGATIVE	The player may lose their vision for a short period of time due to the lights failing.	UR_EVENTS
FR_SECURITY_NEGATIVE	The player may be apprehended by campus security who are trying to frame you for the stolen goose.	UR_EVENTS
FR_COFFEE_NEGATIVE	The player slips in coffee, slowing them down.	UR_EVENTS
FR_FROG_NEGATIVE	The player can interact with a frog that will slow down the player.	UR_EVENTS
FR_DRINK_NEGATIVE	The player can drink too much and have a sugar crash.	UR_EVENTS
FR_HIDDEN_EVENTS	When the user collides with an inthe map it will trigger for a surprise event to happen (x3)	UR_EVENTS
FR_WIND_HIDDEN	The user can activate a “windy” effect for the main path - speeding them up one way and slowing them down another.	UR_EVENTS
FR_VANITY_HIDDEN	The player can gain vanity items during the game that do not affect gameplay.	UR_EVENTS
FR_PROFFESSOR_HIDDEN	The player can interact with a professor who provides no assistance.	UR_EVENTS
FR_EVENTS_LOCATION	Each time the game is played, the events are randomised in some way through a set of pre-chosen locations	UR_EVENTS_LOCATION
FR_ENERGY	There is an energy system for the player that is used to allow the player to sprint.	UR_ENERGY
FR_EVENTS_LINKED	Some events may require a different event to be triggered before they can be triggered.	UR_EVENTS_LINKED
FR_MUSIC	If the game is being played then background music needs to be playing.	UR_MUSIC
FR_MUSIC_VOLUME	The volume of the background music can be changed and muted by the user.	UR_MUSIC
FR_SOUND_EFFECTS	Some events may trigger a sound effect that will be heard over the top of the background music.	UR_SOUND_EFFECTS
FR_INSTRUCTION	From the instruction screen, the user can read the game controls	UR_INSTRUCTION
FR_BUTTON	The user should be able to click on buttons to get from one screen to another. This includes a	UR_UI

	“return” button or a way to return to the previous screen easily.	
FR_MAIN_MENU	there should be a menu from which the player can navigate to all other areas of the game including a new instance of the game.	UR_MENU
FR_PAUSE	A pause button can be clicked while playing that will temporarily stop the timer and pause the game.	UR_MENU
FR_LEADERBOARD	There should be a leaderboard menu that is accessible through the menu system that displays players achievements/highscores	UR_MENU
FR_TIMER	Timer will be counting down on the screen while the game is being played	UR_TIMER
FR_TIMER_END	Once the timer reaches 0 then the game is over	UR_TIMER
FR_CONTROLS	The player can use the mouse or keys appropriately to move the character on screen	UR_CONTROLS
FR_CAMERA	As the player moves around the screen, the camera will be stationary until the player moves to the next room	UR_CAMERA

Non-Functional Requirements			
ID	Description	User Requirement	Fit Criteria
NFR_LOADING	Game loads quickly on the different operating systems	UR_LOADING	The game loads in less than 5 seconds.
NFR_TIME_LIMIT	Game can be played to completion in under 5 mins	UR_TIME_LIMIT	The player should either escape or the timer run out (or be caught by the dean) in under 5 mins.
NFR_SHAPES	Shapes to be used as well as colours to distinguish objects in the game	UR_COLOURS	Someone who is colourblind should have the same understanding for the game as shapes are used alongside colours.
NFR_INSTRUCTION_SCREEN	The instruction screen must be simple and easy to understand	UR_INSTRUCTION	The aim and controls of the game can be understood in max of 20 words and all on one screen

NFR_INSTRUCTION_LOAD	Once the game loads, the user is taken to an instruction screen	UR_INSTRUCTION	Taken to the instruction screen in less than 1 second

Constraint Requirements	
ID	Description
CR_JAVA	Implemented in Java 17
CR_DEADLINE	3 events implemented by 10.11.2025 in first version of the game
CR_OPERATING_SYSTEMS	Run on computers with Windows, Mac and Linux

References

[1] I. Sommerville, *Software Engineering,* ed. 10 Essex, United Kingdom: Pearson Education, 2015

[2] IEE Computer Society, “IEEE Guide to Software Requirements Specifications” February. 1984. [Online]. Available:https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=278253