Requirements

Group 2 - The Debug Thugs

Bader Albeadeeni
Dan Hemsley
Jennifer Bryant
Oliver Elliott
Mathilde Couturier-Dale
Rosie-Mae Connolly
William Mutch

a) Introduction

Requirements for the escape the maze game were gathered through an interview with the customer and analysed by the team using the IEEE 29148-2018 as a guide. Requirements engineering is about collecting, discussing, writing down, and checking what a system needs so that everyone involved agrees on the project's goals. The customer explained the game should be a short family friendly, single player university-themed maze game that's easy to play and is suitable for a wide range of players.

To gather and refine the requirements, we asked the customer questions about the game's features and its limitations. From this, we identified specific expectations: a five-minute time limit, both visible and hidden obstacles, a scoring system, pause and unpause options, also accessibility features like color-independent cues. Visibility was left open for the team's interpretation, allowing flexibility in the design. The inclusion of background music was optional. The customer decided that some features, such as leaderboards, a health system, or multiple difficulty levels, were unnecessary. These points were summarized in the analysis of the interview document, which confirmed the main requirements for maze generation, timing, scoring, and asset licensing. Further discussing the record in the method selection & planning helped us finalize these details and plan the work in an agile way, so we could make improvements during later sprints.

The specification follows the structure that was given in our requirements engineering module (Kolovos 2025). It starts with a single statement of need (SSON) to summarise the main goal of the system, followed by user requirements describing what the player wants to achieve, and system requirements explaining how the software supports that. System requirements are split into functional (what the game does) and non-functional (how well it does it) categories. Each requirement has its own ID and a verification method, inspection, analysis, demonstration, or testing to make sure everything can be traced through the design development process.

By following IEEE 29148-2018 and the ENG1 project guidance, we aim to make the game requirements clear, measurable, and maintainable across the full project journey

Single Statement of Need:
To create a video game of a player escaping from a maze, set in a university.

The following Functional Requirements are planned for Assessment 2:
FR_EVENT_NEG (full 5+); FR_EVENT_POS (full 3+); FR_EVENT_HID (full 3+); FR_EVENT_CONNECT; FR_SCORE_TIME; FR_SCORE_EVENT; FR_GAMBIT_BONUS; FR_GAMBIT_SHORTCUT.

Statement of Requirements

User requirements

ID	Description	Priority	Rationale
UR_MAZE	The system shall have a playable, university-themed maze with clear boundaries and a designated exit.	High	Defines the core theme and objective of the game.
UR_TIME	The system shall enforce a real-time, 5-minute time limit to escape the maze in.	High	Creates a sense of urgency for the player and a condition for losing the game.
UR_EVENTS	The system shall include at least 5 visible positive, 3 visible negative, and 3 hidden events.	High	Provides challenges and rewards in the game.
UR_AUDIENCE	The system should be playable and enjoyable for a “casual audience”.	Medium	Defines the target user for the game.
UR_ACCESSIBILITY	The system should be usable by players with common disabilities (e.g. hearing or colour vision deficiency).	Medium	Specific stakeholder requirement. Ensures an inclusive experience for the player.
UR_PAUSE	The system shall start in a paused state and allow the player to pause the game at any point.	High	Provides usability and game state control.
UR_EVENT_COUNTER	The system shall display a counter of the interactions the player had with each type of event.	High	Provides the player with key feedback.
UR_PROJECT	The system shall have a design justifiable to the stakeholder, and make use of 3rd party libraries.	High	Ensures the project maintains stakeholder approval and technical efficiency.
UR_SCORE	The system may generate a final score based on the player’s escape time and the events they interacted with.	Low	Gives the player a motivation to replay the game to achieve a higher score. **Not required for Assessment 1.**

Functional Requirements

ID	Description	User Requirement(s)	Verification Method
FR_MAP_THEME	The system shall display a university-themed map.	UR_MAZE	Inspection
FR_MAP_PROTAGONIST	The system shall provide a playable, moveable character.	UR_MAZE	Test
FR_MAP_LIMITS	The system shall stop the player moving through limit/wall objects.	UR_MAZE	Test
FR_MAP_EXIT	The system shall go to a “Win” state when the player reaches the designated exit location.	UR_MAZE	Test
FR_TIMER_RUN	The system shall implement a timer with a 5-minute countdown.	UR_TIME	Test

FR_TIMER_LOSE	The system shall go to a “Lose” state when the timer reaches 00:00.	UR_TIME	Test
FR_TIMER_UI	The system shall display the timer on the UI.	UR_TIME	Inspection
FR_START_MENU	The system shall load a Start Menu (“start paused” state) before gameplay begins.	UR_PAUSE, UR_ACCESSIBILITY, UR_AUDIENCE	Test
FR_PAUSE_TOGGLE	The system shall let the player pause and un-pause the game at any time.	UR_PAUSE	Test
FR_PAUSE_HALT	The system shall halt the timer and gameplay when the game is paused.	UR_PAUSE	Test
FR_PAUSE_MENU	The system shall display a Pause Menu with the options ‘Resume’, ‘Settings’, and ‘Quit’.	UR_PAUSE, UR_AUDIENCE	Inspection, Test
FR_SETTINGS_MENU	The system shall provide a Settings Menu which can be accessed from the Start and Pause Menus.	UR_ACCESSIBILITY, UR_AUDIENCE	Inspection, Test
FR_SETTINGS_SOUND	The system shall enable or disable game audio based on an option in the Settings Menu.	UR_AUDIENCE, UR_ACCESSIBILITY	Inspection, Test
FR_SETTINGS_SUBTITLES	The system shall enable or disable text captions based on an option in the Settings Menu.	UR_ACCESSIBILITY	Inspection, Test
FR_EVENT_NEG	The system shall implement at least 5 visible hindering events.	UR_EVENTS	Inspection, Test
FR_EVENT_POS	The system shall implement at least 3 visible beneficial events.	UR_EVENTS	Inspection, Test
FR_EVENT_HID	The system shall implement at least 3 hidden events, invisible until triggered.	UR_EVENTS	Inspection, Test
FR_EVENT_CONNECT	The system shall support connected events, where one event resolves another hindering event (e.g. key for locked door).	UR_EVENTS	Test
FR_EVENT_COUNT_POS	The system shall display and increment a counter for visible positive event interactions.	UR_EVENT_COUNTER	Inspection, Test
FR_EVENT_COUNT_NEG	The system shall display and increment a counter for visible negative event interactions.	UR_EVENT_COUNTER	Inspection, Test
FR_EVENT_COUNT_HID	The system shall display and increment a counter for hidden event interactions.	UR_EVENT_COUNTER	Inspection, Test
FR_SCORE_TIME	The system shall calculate a base score, where a faster escape time means a higher score.	UR_SCORE	Test, Analysis
FR_SCORE_EVENT	The system shall increase or reduce the base score based on event interactions.	UR_SCORE, UR_EVENTS	Test, Analysis
FR_GAMBIT_BONUS	The system shall implement optional events that offer the player a score bonus if they accept a challenge.	UR_SCORE, UR_EVENTS	Test

FR_GAMBIT_SHORTCUT	The system shall implement optional events that offer the player a shortcut if they accept a score penalty.	UR_SCORE, UR_EVENTS	Test
FR_STATE_WIN_UI	The system shall display a win screen showing final score/time when the “Won” state is entered.	UR_MAZE, UR_SCORE	Inspection, Test
FR_STATE_LOSE_UI	The system shall display a lose screen when the “Lose” state is entered.	UR_TIME	Inspection, Test

Non-Functional Requirements

ID	Description	User Requirement(s)	Verification Method
NFR_JAVA_VERSION	The system shall be developed using Java version 17.	(Project Constraint)	Inspection
NFR_FILE_FORMAT	The system shall be distributed in a single executable JAR file including all dependencies.	(Project Constraint)	Test
NFR_CODE_STANDARDS	The system’s source code shall follow standard Java 17 conventions for formatting, naming and comments.	UR_PROJECT	Inspection
NFR_CODE_MODULARITY	The system’s software architecture shall be modular.	UR_PROJECT	Inspection
NFR_LICENSES	All libraries, tools and assets used in the system shall have the correct license, and be documented.	UR_PROJECT	Inspection
NFR_OS_COMPATABILITY	The system shall be able to run on Windows, macOS, and common Linux distributions that support Java 17.	UR_AUDIENCE	Test
NFR_UI_LEGIBILITY	Important system UI elements shall be presented clearly and legibly.	UR_TIMER, UR_EVENT_COUNTER, UR_ACCESSIBILITY, UR_AUDIENCE.	Inspection
NFR_LOAD_TIME	The system shall load and display the Start Menu within 5 seconds of launching.	UR_AUDIENCE	Test
NFR_PAUSE_RESPONSE	The system shall pause/resume gameplay within 0.5 seconds of the respective commands.	UR_PAUSE, UR_AUDIENCE	Test
NFR_TIMER_ACCURACY	The system’s timer shall be accurate to within +/- 1 second of real-time over 5 minutes.	UR_TIME	Test

References