REQUIREMENTS

GROUP 5 - BITCRUSHED BOB

Maryam Mathews
Joseph Hinde
Jacob Mace
Will Aston
Zathia Jacquesson-Ahmad
Bulganchimeg Munkhjargal
Evan Weston

Introduction

The requirements for this project were developed through a structured requirements engineering process, consisting of requirements elicitation and specification, and verification and validation. The process was scaled appropriately for the scope and nature of this project, with a focus on clearly identifying what the game should do, how it should perform, and any constraints that would shape it's development.

Requirements Elicitation and Negotiation:

The primary method of requirements elicitation was a semi-structured interview with the customer. This interview allowed the team to gather information about the key requirements for the game that needed to be clarified before development could proceed, for example, information about system requirements and core gameplay mechanics.

From this session, the customer provided a clear Single Statement of Need (SSON): "A short, single-player game in a university setting, for young adults to have an immersive, escaping-uni experience". This initial interview served as the main source of requirements. Beyond that, the elicitation process continued informally, with only occasional follow-up questions asked for clarification. These follow-ups typically occurred during milestone reviews, or when the development team needed confirmation on design decisions.

Requirements Presentation:

Following elicitation, the team documented the requirements into a structured format. This process began by outlining user requirements, which described what the player should experience and be able to do in the game. These were then translated into more detailed system requirements, which described what the game system must implement. The system requirements were categorised into functional requirements and non-functional requirements. Due to the game's relatively limited scope, the specification process was largely completed early in the project, with only minor revisions being made later.

Verification and Validation:

Once the requirements were specified, the team carried out verification and validation to ensure they were complete, consistent, and aligned with the original vision. As part of the verification process, we compared the specified requirements against the ISO/IEC 25010 software quality model to ensure they align with recognised quality characteristics for software systems. This comparison focused on assessing whether the requirements addressed the nine quality characteristics outlined in the product quality model. For validation, the team conducted internal reviews of the brief and interview notes to confirm that the documented requirements accurately represented the customer's intended user experience.

User Requirements

ID	Description	Priority
UR_AUDIENCE	The game will be aimed at young adults.	Shall
UR_UX	The UI design needs to be easily understandable.	Shall
UR_TIME	The game should last approximately 5 minutes.	Shall
UR_RESULTS	The game should show a score at the end of each run.	Shall
UR_TUTORIAL	There will be an optional tutorial at the start of the game.	Should
UR_CHARACTER_SELECTION	The game can have an option to customise the playable character.	May
UR_UNIVERSITY	The game should have a university setting, and provide an escape-from-uni experience.	Shall
UR_THEME	All art is consistent with a main theme and art style.	Shall
UR_DIFFICULTY	The user should be able to change a difficulty setting to have a new game experience.	Should
UR_MUSIC	Music plays a theme while the character plays.	May
UR_ACCESSIBILITY	The game should be as widely accessible as feasible for a small project, in difficulty or for disability.	Shall
UR_HARDWARE	Should be playable on any regular PC and look to fit an average monitor.	Shall
UR_DATA	The game must not save any user data.	Shall
UR_RELIABILITY	The game should not crash or be unavailable.	Shall
UR_COUNTER	The game must have a counter that displays how many of each event have been interacted with.	Shall
UR_EVENTS	The game must have at least one benefitting event, one hindering event, and one hidden event.	Shall
UR_PAUSE	The game should allow the player to pause and resume the game at any time.	Shall
UR_END	The game will end when the player reaches the exit or the 5-minute timer expires, and displays a result screen showing score.	Shall

System Requirements - Functional Requirements

ID	Description	User Requirements
FR_DESKTOP_PLATFORM	The system shall run smoothly on standard desktop systems without requiring high-end hardware.	UR_HARDWARE
FR_MAP_STRUCTURE	The system shall provide a maze-like map representing a university environment with clear boundaries and defined escape paths.	UR_UNIVERSITY
FR_PLAYER_MOVEMENT	The system shall allow the player to move within the maze using keyboard or mouse controls.	UR_UX
FR_PAUSE_RESUME	The system shall suspend the game timer when paused, and resume it when unpaused.	UR_PAUSE
FR_VISIBLE_HINDRANCES	The system should include at least five visible hindrance events that obstruct the player’s progress.	UR_DIFFICULTY
FR_VISIBLE_BENEFITS	The system should include at least three visible beneficial events that enhance the player’s performance.	UR_DIFFICULTY
FR_HIDDEN_EVENTS	The system shall include at least three hidden events that are invisible until triggered, producing interactive or humorous effects.	UR_DIFFICULTY
FR_EVENT_INTERACTION	The system shall allow the player to interact with events, resolving obstacles or collecting items as needed.	UR_EVENTS
FR_SCORING_SYSTEM	The system shall calculate and display a final score based on time taken and events triggered when the player escapes or time expires.	UR_RESULTS
FR_TIMER_LIMIT	The system shall run each game session for a maximum of 5 minutes.	UR_TIME
FR_EXIT_CONDITION	The system shall end the game when the player reaches the exit or when the timer expires, displaying a results screen.	UR_END
FR_TUTORIAL_MENU	The system shall include a tutorial or instruction section explaining controls and objectives.	UR_TUTORIAL
FR_MAIN_MENU	The system should include a main menu with options such as Start Game, Tutorial, and Settings	UR_TUTORIAL, UR_UX
FR_USER_INTERFACE	The system shall display key information during gameplay, including timer, score, and status messages.	UR_RESULTS
FR_NO_DATA_STORAGE	The system should not store any user data, scores, or session history.	UR_DATA

System Requirements - Non-Functional Requirements

ID	Description	User Requirements	Fit Criteria
NFR_PORTABILITY	The game shall have cross-platform desktop support and not require any high-performance hardware	UR_HARDWARE	Game passes a standard playtest on Windows, macOS, and Linux with no manual configuration changes
NFR_ACCESSIBILITY	The game shall have a universally understandable UI.	UR_UX	All key events must use at least two feedback channels (e.g. flashing colours and text).
NFR_USABILITY	The game will provide a tutorial such that someone inexperienced with games will be able to play.	UR_TUTORIAL UR_ACCESSIBILITY	Playtest shows 90% of first-time players can complete the tutorial without assistance.
NFR_SECURITY	The game shall not store any user data or personal information	UR_DATA	No files related to user-identity, scores, or progress are saved to the disk after the session ends.
NFR_MAINTAINABILITY	Codebase will be modular and well documented in order to allow for future extensions and development	N/A	Game logic and systems are separated into independent modules, with each major module including docstring explaining purpose and usage.
NFR_RELIABILITY	The game will not crash upon being given reasonable input, as well as in edge-cases	UR_RELIABILITY	Playtest with 10 sessions shows no crashes or freezes under normal or extreme inputs.
NFR_CONTENT_CONSTRAINTS	The game will be family-friendly, and not feature any events or themes that may be considered harmful or inappropriate.	UR_AUDIENCE	Internal review upon every addition confirms narrative and visual cohesion.
NFR_THEME_CONSISTENCY	A consistent narrative and thematic experience shall be maintained across all elements of the game (visuals, sounds, events, etc.)	UR_THEME	All game assets and events must support the aesthetic direction, with no conflicting elements. Internal review upon every addition to confirm narrative cohesion.
NFR_LEGALITY	The game shall only include third-party assets with appropriate licenses.	N/A	All third-party assets must be documented and reviewed before implementation to ensure that they have the appropriate licences.
NFR_PERFORMANCE	The game shall maintain at least 30 FPS under normal gameplay conditions	UR_HARDWARE	Performance test confirm stable framerate on standard desktops
NFR_SCALABILITY	The system should allow for future expansion (e.g. adding new maps or difficulty)	N/A	Modular code structure allows addition of at least 5 new event types without affecting existing functionality
NFR_ERROR_HANDLING	The game shall handle unexpected input gracefully without crashing	UR_RELIABILITY	Test edge cases (such as invalid user input) and verify the game does not terminate.