Requirements for the game were elicited from both the product brief and the initial customer meeting. To prepare for the customer meeting, we first brainstormed different interview questions, and then we refined them, organising questions into three categories: User Requirements, Functionality Requirements and Graphical requirements, and marking question priorities. As related questions were grouped together, this made it easier to identify similar questions which could be merged into one, and allowed us to produce a more structured interview script, so that the process of interviewing the customer was more efficient.

From the meeting transcript, we summarised the customer's points and used this to produce tables for user and system requirements. All requirements have been given a unique, meaningful ID. User requirements all have priorities, to ensure that the customer's most important requirements can be assigned sufficient time and resources. System requirements have been produced based on these user requirements, giving more specific technical requirements. System requirements have been further divided into functional requirements and then non-functional requirements with specific and measurable fit criteria, as well as constraint requirements.

Many of the user requirements were taken directly from the product brief, such as UR_Negative, UR_Positive and UR_Hidden corresponding to the different event types. As much as the customer meeting was about identifying any additional requirements, it was also about clarifying which potential requirements were not needed, for example there is no expectation to save/load previous attempts or have a leaderboard to record past scores.

In the customer meeting, the client mentioned wanting some way to teach the player how to play the game, reflected by the UR_Tutorial requirement. They also mentioned that the maze should remain unchanged between runs, covered by UR_Consistent_Maze. Although the client didn't have specific performance targets to give, the game is required to run smoothly on an average computer, and the player should be able to play the game with a keyboard and mouse (see UR_Performance, UR_Standard_Computer and UR_Desktop_Inputs). A settings menu was less of a priority for the client, and only required if we implement features such as audio. Likewise, no specific accessibility features were required, although the client did mention that it is good to keep accessibility in mind e.g., ensuring text is large enough to easily read. The requirements around settings and menus (UR_Accessibility, UR_Audio, UR_Main_Menu and UR_Settings) were all given lower priorities of 'Should' or 'May' to reflect this.

Although the brief mentioned the target audience of the game was a casual audience, as with many of the other requirements, asking the right questions in the customer meeting gave us a bit more insight, revealing that the game was to be aimed at students (or those around a similar age), and that the game must be family friendly. The customer also reiterated the constraints of the project, including the fact that any 3rd party assets must be licensed, which have been used to produce a list of constraint requirements.

Many of the features, such as what the different negative events actually are, or how the Dean works, the customer has left up to us to decide, and so we have tried to ensure that the requirements leave options open. Key design choices like how the player is taught the game don't need to be made at this stage of the project.

User Requirements

ID	Description	Priority
UR_Player	The game should have a player character which can move around the maze.	Shall
UR_Negative	The game should have five negative events that hinder the player from progressing.	Shall
UR_Positive	The game should have three positive events (that benefit the player).	Shall
UR_Hidden	The game should have three hidden events.	Shall
UR_Time_Tracker	The game should track how long the game lasts. This should be shown to the player.	Shall
UR_Lose_Time	The player should lose if they haven't escaped after 5 minutes.	Shall
UR_Performance	The game should appear to run smoothly on an average computer.	Shall
UR_Pause	The game will start paused and allow you to pause at any time.	Shall
UR_Tutorial	The game requires a system to teach the player how to play and the rules of the game This could be through a demo/tutorial or text.	Shall
UR_University	The game should take place in a university-like maze. The map should evoke university-like qualities/places/scenery.	Shall
UR_Map_Limits	There should be clear limits to where the player can go.	Shall
UR_Dean	The game should involve a Dean, who is chasing the player. This can be abstract - e.g., the player might not actually see the Dean.	Shall
UR_Dean_Object	The game could involve a Dean that is visible to the player and chases the player. If it collides with the player it will end the game.	May
UR_Fast	Escaping as fast as possible will give you a better base score.	Shall
UR_Audience	The game is aimed at a casual audience of students. The game should have a constant difficulty level that is enjoyable for anyone to play as their first game.	Shall
UR_Licensed_Assets	The game should only use third party assets that are fully licensed for the production of the game.	Shall
UR_Standard_Computer	The user should be able to play the game on a device with average hardware.	Shall
UR_Desktop_Inputs	The user should be able to interact and play the game using a keyboard and mouse/trackpad.	Shall
UR_Accessibility	The game can have input remapping and easy to read text however this is not a primary concern.	May

UR_Audio	Implement music and/or sound effects. The volume of the game should be adjustable to the users preference	Should
UR_Score	The game must have a score counter that is shown to the player at least once at the end of their game.	Shall
UR_Consistent_Maze	The maze layout itself has to remain the same for each run.	Shall
UR_Main_Menu	The game should have a main menu with the ability to navigate to the settings. This could be combined with the pause menu.	Should
UR_Settings	The game should have a settings menu, which allows the player to update the audio and accessibility settings (if they exist).	Should

Functional Requirements

ID	Description	User_Requirements
FR_Main_Menu	The main menu will be displayed as soon as the game is launched. It will show a background image with several buttons on it that are linked to tutorials, settings page, and starting the game. Players will be transferred to the next sequence by clicking buttons.	UR_Main_Menu
FR_Tutorial	There will be a tutorial icon appearing on the Main menu screen and on the settings page. Players will be granted access by clicking the icon.	UR_Tutorial
FR_Audio	The volume of background music and general sound will be displayed on the settings page with bars. Each of these elements will be adjustable by clicking arrows near it.	UR_Audio
FR_Map_Limits	There are visible boundaries inside the university map that show the area that players are allowed to go. Players will not be able to move past the boundaries. If the coordinate of the player's current location neighbors with that of the map's boundaries, the movement of the player will be restricted to a certain direction.	UR_Map_Limits
FR_Performance_Display	On the up right side of the screen, fps(frame per second) will be displayed real-time with small text. Players can turn it off by modifying it in the settings page.	UR_Performance
FR_Score_Counter	The system will calculate the final score depending on the number of achieved hidden events and the passed time for escaping the uni and display it on the screen after the game ends.	UR_Score
FR_Event_Counter	The user is displayed a counter of the number of each remaining event that decrements on each event triggered	UR_Event_Counter
FR_Pause	The user is able to stop the in game timer by pressing a key on a keyboard and unpause the game by pressing the same button again	UR_Pause
FR_Fast_	The users score decrements every second passed	UR_Fast

Score		
FR_Player_Displayed	There is a “Player” displayed on the screen	UR_Player
FR_Player_Movement	There is an algorithm that converts the players key presses into movement of the player on screen	UR_Player
FR_Timer	There is a timer displaying the remaining time the player has	UR_Time_Tracker
FR_Losing_Time	A timer decrements in real time starting from five minutes once it reaches zero the game ends and the player is shown a loss screen	UR_Lose_Time
FR_Pause_Menu	When the game is paused a Series of UI buttons are displayed for the user to select	UR_Pause_Menu
FR_Settings	A menu the user can adjust their settings in	UR_Settings
FR_Chasing_Dean	An object that chases the player through the maze and ends the game if it collides with the player.	UR_Dean_Object

Non Functional Requirements

ID	Description	User_Requirement_ID	Fit Criteria (specific)
NFR_Time_constraint	The game should track the time of how long the game lasts	UR_Time_Tracker	The entire gameplay should be <5mins
NFR_Assets	The assets that the game use such as interactable objects, background image and music	UR_Licensesd_Assets	The game should only use third party assets that are fully licensed for the production of the game/ open sourced
NFR_Usabilty	The Audience at which the game is aimed at	UR_Audience	The target audience is very casual and the game difficulty should make it possible for someone to play it as their first game. 90% of players should be able to escape the maze after 30 minutes of playing.
NFR_sys_reqs	The minimum required specifications of the device to play the game.	UR_Standard_Computer	The user should be able to play it on average hardware.
NFR_accessebility	The accessibility features in the game	UR_Accessibility	The game can have input remapping and easy to read text however this is not a primary concern.