3^3

Requirements

3^3 Studios | Team 27

James Hooper, Daniel O'Brien, Connor Davies, Joe Krystek-Walton, Mitchell Simpson, Yiyang Ye

Requirements

Introduction

When determining the requirements for this software engineering project, we identified 3 key areas to obtain the information required to express the requirements for the project. These were the product brief, the stakeholders, and our own team decisions. Our first step was to analyse and quantify the requirements from the product brief - creating an initial list of all the requirements from the document and discussing the implications of these with the team. Many of these have been classified as functional requirements of the system.

Once the requirements from the product brief were categorised and discussed, we then moved onto the second stage, which involved interviewing a stakeholder to obtain additional requirements, constraints, and expectations of the system. We arranged a meeting with the client and asked them predetermined questions, discussed between members of our development team beforehand and were specifically written for achieve a few key objectives:

- Elaborate on requirements from the product brief to achieve a greater understanding of the specifics regarding them and any potential constraints.
- Obtain new functional requirements for the project. (additional critical functionality)
- Gather some expectations of the user for the project.
- Understand if and where there was flexibility for deciding our own requirements.

These questions covered a lot of topics regarding the project including the style of the game, a target audience, mechanics, and other aspects that previously had not been covered in the product brief. While the interview was being conducted with the customer of the product, members of our team noted down key information / transcripts of the answers provided, to ensure we had the correct, unbiased information provided by the stakeholder. This data was again aggregated, and discussions were made with our team around the answers provided.

Once we had gathered all the required data from the product brief alongside additional prompts from the stakeholders, another discussion was performed in order to determine the final initial requirements of the project, mostly prompted from the answers provided by the customer. These requirements were determined from specifics of the system that the stakeholder expressed to us were our decision to make, and were mostly non-functional requirements, such as how certain required features of the game would be implemented.

Taking all of this data into consideration, we have decided to format the provided requirements into 4 tables. These will consist of constraints provided that will have an impact on the development of the system, the user's requirements and expectations from the system, the functional system requirements, and the non-functional system requirements. Additionally, this statement of requirements will include criteria for quantifying the success of the requirement.

Constraints

ID	DESCRIPTION
CON_PLATFORM	Design for a desktop experience, considering aspect ratio
CON_PROG_LANG	Must be developed with the Java programming language
CON_REAL_TIME	Game must run in real time (not turn based)
CON_TOP_VIEW	Game must be from a top down view

User Requirements

ID	DESCRIPTION
UR_NAME	Game must be named “Auber”
UR_UX	The user interface should be easy and intuitive to navigate
UR_MAP	The game displays a map to the user which can be navigated and interacted with
UR_PLAYER	The player can navigate the map, heal, and arrest hostiles
UR_HOSTILES	Hostiles act like other characters on the ship, but attempt to sabotage key systems and have special abilities
UR_LOGIC	Game is played in real time, with no turns. Player wins if all hostiles are arrested and loses if more than 15 key systems are destroyed.

Functional System Requirements

ID	DESCRIPTION	USER REQUIREMENTS
FR_MENU	The system should provide a menu to allow the player to start and configure the game	UR_UX
FR_MAP	The system should have a fixed playable map area consisting of at least distinct 4 types of rooms	UR_MAP
FR_MAP	The map will be made of tiles on which the player and the AI can traverse, from tile to tile	UR_MAP
FR_TELEPORTER	The map should contain teleport pads in areas which the player will use to teleport around the map	UR_MAP
FR_KEY_SYSTEMS	The map should contain at least 15 key systems which can be sabotaged by infiltrators and defended by the player. These cannot be repaired.	UR_MAP

FR_ALIENS	There must be at least 2 types of alien characters on the map	UR_MAP
FR_HOSTILES	There are eight hostiles on the map	UR_HOSTILES
FR_SABOTAGE	The hostiles can sabotage key systems on the map	UR_HOSTILES
FR_SPECIAL_ABILITIES	The hostiles must have at least 3 special abilities	UR_HOSTILES
FR_PRISON	Holds hostiles once they have been arrested. Arrested hostiles cannot escape.	UR_MAP
FR_ATTACK_NOTIF	The player is notified of sabotage attempts	UR_UX
FR_HEAL	Player can heal in the infirmary	UR_UX
FR_ARREST	Player can arrest hostiles	UR_UX
FR_REAL_TIME	Game must run in real time (not turn based)	UR_LOGIC
FR_WIN_CONDITION	Game is won when all eight infiltrators have been arrested	UR_LOGIC
FR_LOSS_CONDITION	Game is lost when more than 15 key systems have been destroyed	UR_LOGIC
FR_TOP_VIEW	Game must be from a top down view	UR_UX
FR_MAP_SIZE	Map must have 10 rooms	UR_MAP
FR_SYSTEM_HEALTH	Key systems should have set, non-regenerating health	UR_LOGIC
FR_SYSTEM_DESTROY	Systems should be destroyed by attackers in 10 seconds	UR_LOGIC
FR_PLAYER_SPEED	Player's movement speed should be faster than infiltrators	UR_PLAYER
FR_HOSTILES_RUN	Hostiles should run from the player's teleportation gun	UR_HOSTILES
FR_ARREST	Player can't arrest hostiles if they aren't attacking anything	UR_PLAYER
FR_RESPAWN	Player should respawn at cloning bay if killed	UR_PLAYER
FR_HOSTILES_ATTACK	Hostiles' attacks should do 10% of the player's health	UR_HOSTILES
FR_HOSTILES_ABILITIES	Hostiles should have following abilities: Blinding player, player slowdown, temporary invisibility, hallucinations	UR_HOSTILES
FR_ROOM_SIZES	Rooms should be different shapes and sizes	UR_MAP
FR_HOSTILES_SPAWN	All hostiles spawn on the map from the beginning of the game	UR_HOSTILES
FR_SYSTEMS_ATTACKED	A maximum of three systems can be attacked at once	UR_LOGIC
FR_ALIENS_COUNT	There should be 24 non hostile aliens on the map	UR_MAP
FR_HOSTILES_SPECIAL	There should be four hostiles with special abilities	UR_HOSTILES
FR_PLAYER_TILES	Player should have free movement between tiles; should not snap to tiles	UR_PLAYER
FR_TELEPADS	There should be three telepads on the station	UR_MAP
FR_BRIG	The brig should appear as a room on the map	UR_MAP

FR_MINIMAP	The game should display a minimap that shows what room the player is in	UR_MAP
FR_PLAYER_VIEW	The game should be zoomed in on the player	UR_MAP
FR_TELEPAD_DESTINATION	The player can teleport from any telepad to any other telepad	UR_PLAYER
FR_PLAYER_SPAWN	Player's spawn location on the map is fixed	UR_PLAYER
FR_ALIEN_SPAWN	Spawn locations for aliens should be random	UR_MAP
FR_HOSTILES_SPAWN	Spawn locations for aliens should be random	UR_HOSTILES

Non-Functional System Requirements

ID	DESCRIPTION	CRITERIA FOR SUCCESS	USER REQUIREMENTS
NFR_ENJOYABLE	Game should be enjoyable to play by the ENG1 cohort	Majority of subjects tested report enjoying the game	UR_UX
NFR_DURATION	Game should last around 5-10 minutes	Majority of games finish within this timeframe	UR_LOGIC
NFR_DIFFICULTY	Game should be enjoyable and not overly difficult	Majority of subjects win the game	UR_LOGIC
NFR_SCALABLE	Window size should scale with the game based on the dimensions	Game dimensions transform in respect to the window size	UR_UX