Engineering 1

Assessment 1

Cohort 3

Team 23

Seif Hussein

Joshua Mckean

Harry Draper

Sebastian Armstrong

Thomas Maalderink

Tikhon Likhachev

Requirements Elicitation:

Requirements needed a process to be gathered. Our process started with first reading the product brief that we had, identifying the main points and areas, and highlighting any areas that we had questions about or needed clarification on. This was done with all the team members attending and discussing the product brief. Then our team met with the client to ask about the questions that we had from the brief, the client told us the features and functionalities that he expects from the product. He stated the single statement of need for the product which contains the overall goal of the product and what capabilities the system should provide and an expectation of the outcome of the product. We then held a meeting as a team and turned the clients needs and expectations, as well as details from the product brief, into two types of requirements; User Requirements and System Requirements, which are divided into functional requirements and non-functional requirements.

Requirements are represented in the form of 3 tables. The first table is User Requirements. It contains a column which represents the unique ID of the user requirement, the ID contains the name of the user requirement to make it meaningful. This helps when the requirement is referred to in a different context. The second column is a description of the requirement, and the third column is the priority of the requirements which has 3 levels - 'should', 'shall' and 'may'. The second table is the functional requirement of the product. The first column is a meaningful ID which uniquely identifies the functional requirement, the second column is the description of the requirement which explains what the requirement states, and the third column is the ID of the user requirement which this functional requirement satisfies and builds on. The third table is the non-functional requirements. It contains four columns, the first column being the meaningful ID which uniquely identifies the non-functional requirement, the second column is the description of the requirement, the third column is the ID of the user requirement which this requirement satisfies, and the fourth column is the fit criteria of this requirement that quantifies and measures the requirement intention, which is called the fit criterion.
All the IDs of all requirements are prefixed with UR/FR/NFR to denote the type of requirement. The requirements take the shape of tables because there is a big chance that some of the requirements change over the time or new requirements may be introduced, so the table form is easy to update and modify and we should always be open and ready to any change of requirements.

SSON: The system should support players to control various cooks around various cooking stations around the kitchen and to be able to interact and use ingredients to prepare dishes that are requested by customers.

User Requirements:

ID	Description	Priority
UR_GAME_MODES	The system should support scenario game mode	Should
UR_SUPPORT_MULTIPLE_COOKS	The system should support multiple cooks	Should
UR_CONTROL_COOKS	The system should allow players to control any cook	Should
UR_SUPPORT_MULTIPLE_COOKING_STATIONS	The system should support multiple cooking stations.	Should
UR_FLIP_PATTY	The system should allow the user to flip patties	Should
UR_CUT_INGREDIENTS	The system should allow the user to cut/chop certain ingredients	Should
UR_INTERACT	The system should allow players to interact using the cook with cooking stations and ingredient stations	Should
UR_HAVE_RECIPES	The system should have recipes for salads and burgers	Should
UR_HAVE_PANTRY	The system should have a places that contains ingredients	Should
UR_GOOD_EXP	The system shall offer a good user experience for the player	Shall
UR_CUSTOMERS_ORDER	The system should allow for 5 customers to come and order	Should
UR_COLLISIONS	The system should make collisions to prevent the user from leaving the map	Should
UR_WINNING_SCREEN	The system should display a win screen	Should

Functional Requirements:

ID	Description	User Requirements
FR_SELECT_COOK	The system should make a player choose which cook they want to control	UR_SUPPORT_MULTIPLE_COOKS
FR_MOVE_COOK	The system should provide a way for the player to control the cooks' movement around the kitchen	UR_CONTROL_COOKS
FR_CHOOSE_COOKING_STATION	The system should allow the player to choose which cooking station they want to operate on	UR_SUPPORT_MULTIPLE_COOKING_STATIONS
FR_SWITCH_COOK	The system should provide a way for players to switch between different cooks	UR_CONTROL_COOKS
FR_INTERACT	The system should provide operations of like chop, flip, grab an ingredient, grab a dish, and place what they holding	UR_INTERACT
FR_SHOW_RECIPE	The system should show the recipe for the dish that the customer ordered	UR_HAVE_RECIPES
FR_USE_PANTRY	The system should provide operations that can be executed in the ingredients pantry to collect ingredients	UR_HAVE_PANTRY
FR_UNLIMITED_PANTRY	The system should never allow the ingredients to run out	UR_HAVE_PANTRY
FR_COLLISIONS	The system should cause collisions with certain objects.	UR_COLLISIONS
FR_NOTIFY_CUSTOMER_ORDER	The system should notify the user the order that the customer ordered when customer arrive	UR_CUSTOMERS_ORDER
FR_LOSE_REP_POINT	The system should make the player lose a reputation point if the player fails to serve the customer within the time limit	UR_REPUTATION_POINTS
FR_WIN_SCENARIO	the system should make the player win if the player still have reputation points after the last customer left	UR_WINNING_SCREEN

Non-functional requirements:

ID	Description	User Requirements	Fit Criteria
NFR_EASY_CONTROLLING_COOKS	Controlling the cooks should be easy for players and does not need training	UR_CONTROL_COOKS	2 controlling mechanisms: 1: keyboard 2: mouse
NFR_EASY_SWITCHING_COOKS	Switching between cooks should be easy for players and does not need training	UR_SUPPORT_MULTIPLE_COOKS	Switching time should be on average 1 sec, at maximum
NFR_ACCURATE_REPUTATION_POINTS	The reputation points left should always be accurate	UR_REPUTATION_POINTS	99% accurate, 1% + or - 1
NFR_EASY_INGREDIENTS_INTERACTING	Interacting with ingredients in the pantry should be easy	UR_HAVE_PANTRY	99% accurate, 1% + or - 1
NFR_EASY_INFRONT_COOK_INTERACTING	Interacting with what's in front of the cook should be easy	UR_INTERACT_INFRONT	95% of players will be able to easily interact with what’s in front of the cook
NFR_GAME_NOT_DIFFICULT	The game should not be difficult to play for the average person	UR_GOOD_EXP	80% of people will be able to win the game in the first time. 95% of people will be able to win the game in the second time
NFR_ENJOYABLE	The game should be fun	UR_GOOD_EXP	Ask people for their opinion after finishing the game. 95% of people should find it enjoyable
NFR_NO_CRASHING	The game should not crash	UR_GOOD_EXP	At least 99% of the time, the game doesn't crash
NFR_ACCESSIBILITY	The game should be accessible for t all users	UR_GOOD_EXP	95% of people should be able to access it easily
NFR_NO_CHEATING	The user should not be able to cheat	UR_GOOD_EXP	All users should not be able to do un permitted actions
NFR_FAST_LOAD	The game should load quickly	UR_GOOD_EXP	Game should load in < 1 minute after pressing start
NFR_NO_TRAINING	The game shall be playable for users with no training	UR_GOOD_EXP	90% of people will be able to play it without training

NFR_READABLE_MESSAGES	Messages from the system should be in plain english	UR_GOOD_EXP	100% of messages should be in plain english
NFR_FAST_RESPONSE	The game should have instant response times	UR_GOOD_EXP	In < 2.5 seconds after click and on average < 1 second after click
NFR_CHILD_FRIENDLY	Game should be child friendly	UR_GOOD_EXP	No violence or +18 scenes and figures should be shown