Engineering 1 Group Assessment 2
Requirements
Cohort 3 Group 23

Team Members:
Harry Draper
Seif Hussein
Tikhon Likhachev
Thomas Maalderink
Joshua McKean
Sebastian Armstrong

Eliciting Requirements

The approach we took to eliciting requirements went as follows:

- We listed the requirements outlined in the brief and highlighted any ambiguities
- We set up a meeting with the customer to address ambiguities in requirements
- We edited the list of requirements according to the customer's responses
- We finalised the list of requirements once there were no ambiguities

From the outset of the project, the paper provided our team with a brief outlining the general purpose of the game and a list of features we had to include. After writing up these features into a table of individual user requirements, we found some apparent ambiguities with the features listed in the brief. To address these ambiguities effectively, we set up a meeting with the customer to discuss how we should approach these issues. We discussed the single statement of need and asked for feedback on our current statement, before updating it and then sending it to Tommy again via email for final review. In the event of a delay, we discussed how we would handle potentially cutting requirements from the product to save time and were told if we had to we should contact him by email. We asked which platforms and interfaces we should develop for - for example, developing for mobile or allowing players to use controllers instead of a keyboard. Tommy's answer was to develop only for desktop computers and a keyboard/mouse interface. We asked about the gameplay experience, covering customisation we should offer users (to be reserved for assessment 2), average game time (5-10 minutes), and graphical scaling on high resolution systems (not critical). Our gameplay audience was discussed and we clarified that we were not allowed viscera or gore due to the child nature of our audience. Finally, we discussed accessibility options and were told to use shapes instead of colour to help colour blind players enjoy their experience.

We then discussed these as a team and created our lists of requirements. We found no more ambiguities in the list of requirements after the customer meeting and can conclude from this that the approach we took was effective. We also settled on our final single statement of need: "Build a single-player game where the player, a prospective student or family member at a University open day, manages kitchen staff cooking and assembling recipes in the Piazza restaurant."

To aid in assembling our list of requirements, we also conducted market research into similar games to gain understanding of how such a game may look.

User Requirements

We defined requirements based on the brief and customer interviews. The requirements are presented in tabular form to allow ease for finding a requirement both by code and description, which is made easier by the vertical alignment of columns. We ensured the requirement codes were all unique and intuitively named for convenient referencing. Our market research also provided a good context for how the player generally interacts with the system and what sorts of features will be needed.

(Each user requirement may have many functional and/or non-functional requirements, or neither.)

Requirement Code	Description
UR_COOKS	Keep track of and display 3 cooks which the user can control (only one cook can be controlled at a given time)
UR_WHICH_COOK	Keep track of and display which cook is currently controlled by the user
UR_SWITCH_COOK	Swap which cook is currently controlled by the user
UR_MOVE_COOK	Move the cook which is currently controlled by the user
UR_COOK_STATION	Keep track of and display which station the cook which is currently controlled by the user is in front of
UR_COOK_INTERACT	Interact with the station in front of the cook which is currently controlled by the user
UR_MODE_SELECT	Screen allowing user to choose between scenario or endless mode
UR_MODE_SCENARIO	Customers are limited to a certain number wanting to be served in a specific time
UR_MODE_ENDLESS	Unlimited customers wanting to be served until all reputation points are lost
UR_CUSTOMER_ARRIVAL	Create new customers to be served at intervals according to the game mode
UR_CUSTOMER_ORDER	Keep track of and display all current customer orders and their recipes
UR_COOK_STACK	Keep track of what items are in the cooks stack they are carrying
UR_CUTTING_STATION	There is a station where cooks can cut ingredients
UR_BAKING_STATION	There is a station where cooks can bake ingredients
UR_SERVING_STATION	There is a station where cooks can serve the food to customers
UR_INGREDIENT_PANTRY	There is a pantry which allows players access to ingredients
UR_FRYING_STATION	There is a station where cooks can fry ingredients
UR_TAKE_INGREDIENT	Cooks can take an ingredient from the pantry, adding it to the top of their ingredient stack
UR_DROP_INGREDIENT	Cooks can drop the top ingredient from their stack onto a cooking station
UR_UNLIMITED_INGREDIENT	The pantry provides unlimited ingredients to the cooks
UR_EARNING	Cooks can earn money to invest in cooking stations
UR_UNLOCK	Cooks can unlock cooking stations
UR_POWER_UPS	Cooks can use power ups

UR_SCORE_POWER	Increases the score by 1
UR_MONEY_POWER	Increases money by 5
UR_REPUTATION_POWER	Increases reputation by 1
UR_CUSTOMER_POWER	Decreases customer spawn rate for a limited period
UR_ORDER_POWER	Fulfils the next order without the need to cook it
UR_COUNTER_WAIT	There is a counter for customers to wait at
UR_GAME_END	Once the last customer has been served, move to a screen displaying how long the user took to complete the scenario
UR_GAME_SAVE	User can save their game
UR_GAME_LOAD	User can load saved games
UR_GAME_DIFFICULTY	User can select the game difficulty when starting a new game
UR_REPUTATION	Cooks gain a reputation point for every customer served
UR_RECIPE	Keep track of the recipes for both salads and burgers
UR_HARD_MODE	There is a hard mode which increases difficulty for the user
UR_NORMAL_MODE	There is a normal Mode leaves the default difficulty for the user
UR_EASY_MODE	There is an easy mode which decreases the difficulty for the user
UR_IDLE_COOK	Cooks do nothing when they are idle
UR_ACCESSIBLE	The game must be accessible to users who are colourblind
UR_PLAYABLE	The game must be playable for a typical user

System Requirements

Each functional and non-functional requirement is linked back to one or more user requirements for reference. They use the wording “the system must/should/may" to denote the importance of the requirement.

Functional Requirements:

Requirement Code	Description	User Requirement This Fulfils
FR_COOKS	The system must keep track of 3 cooks which can be controlled by the user	UR_COOKS
FR_WHICH_COOK	The system must keep track of which cook is currently being controlled by the user	UR_WHICH_COOK
FR_DISP_COOK	The system must display the 3 cooks, with some indication of which cook is currently under the user’s control	UR_WHICH_COOK, UR_COOKS
FR_SWITCH_COOK	The system must allow the user to swap which cook is currently being controlled by the user	UR_SWITCH_COOK
FR_COOK_STATION	The system must keep track of which station the cook currently controlled by the user is in front of	UR_COOK_STATION
FR_DISP_STATION	The system must display the station the cook currently being controlled by the user is in front of	UR_COOK_STATION

FR_MODE_SELECTION	The system must allow the user to choose between different modes	UR_MODE_SELECT
FR_DIFFICULTY_SELECTION	The system must allow the user to choose between 3 difficulties for both endless and scenarios modes	UR_GAME_DIFFICULTY
FR_HARD_MODE	The system will decrease waiting time for customers to increase difficulty and chef speed is decreased	UR_HARD_MODE
FR_NORMAL_MODE	The system will leave the default waiting time for the customers and chef speed	UR_NORMAL_MODE
FR_EASY_MODE	The system will increase the waiting time for the customers to decrease difficulty and chef speed is increased	UR_EASY_MODE
FR_COOK_INTERACT	The system must allow the user to interact with the station that the cook controlled by the user is in front of	UR_COOK_INTERACT
FR_MODE_SCENARIO	The system must limit customer arrivals to a certain number and a certain time in scenario mode	UR_MODE_SCENARIO
FR_MODE_ENDLESS	The system will not limit customer arrivals and will continue until the user runs out of reputation points	UR_MODE_ENDLESS
FR_CUSTOMER_ARRIVAL	The system must introduce new customers to be served at intervals according to the game mode's configuration	UR_CUSTOMER_ARRIVAL
FR_CUSTOMER_ORDER	The system must keep track of all current customer orders.	UR_CUSTOMER_ORDER
FR_DISP_ORDER	The system must display all current customer orders and their recipes	UR_CUSTOMER_ORDER
FR_COOK_STACK	The system must allow a cook to hold multiple ingredients in a LIFO stack	UR_COOK_STACK
FR_CUTTING_STATION	The system must have a station where cooks can cut ingredients	UR_CUTTING_STATION
FR_EARNING	User can earn money for each order they complete which can be accumulated	UR_EARNING
FR_UNLOCK	User can use money earned to unlock cooking stations in order to complete different recipes	UR_UNLOCK
FR_REPUTATION	Cooks gain a reputation point for every customer served to a max of 3. If all points are lost the game is over.	FR_REPUTATION
FR_POWER_UPS	Cooks can use power ups	FR_POWER_UPS
FR_SCORE_POWER	Increases the score by 1	FR_SCORE_POWER
FR_MONEY_POWER	Increases money by 5	FR_MONEY_POWER
FR_REPUTATION_POWER	Increases reputation by 1	FR_REPUTATION_POWER
FR_CUSTOMER_POWER	Decreases customer spawn rate for a limited period	FR_CUSTOMER_POWER

FR_ORDER_POWER	Fulfils the next order without the need to cook it	UR_ORDER_POWER
FR_BAKNG_STATION	The system must have a station where cooks can bake ingredients	UR_BAKING_STATION
FR_SERVING_STATION	The system must have a specific location where finished meals are deposited.	UR_SERVING_STATION
FR_INGREDIENT_PANTRY	The system must have a pantry (station) which allows players access to ingredients	UR_INGREDIENT_PANTRY
FR_FRYING_STATION	The system must have a specific location where ingredients to be fried are cooked.	UR_FRYING_STATION
FR_TAKE_INGREDIENT	The system must allow cooks to take an ingredient from the pantry, adding it to the top of their ingredient stack	UR_TAKE_INGREDIENT
FR_DROP_INGREDIENT_ KEY	The system may have a keyboard input that triggers popping an ingredient off the stack so that it appears in the “inventory” of the station the cook is nearest to, within a certain radius.	UR_DROP_INGREDIENT
FR_UNLIMITED_INGREDIEN T	The system must not limit the amount of each ingredient provided by the pantry	UR_UNLIMITED_INGREDIENT
FR_COUNTER_WAIT	The system must have a counter for customers to wait at	UR_COUNTER_WAIT
FR_GAME_SAVE	User can press a button to save their game and leave	UR_GAME_SAVE
FR_GAME_LOAD	User can press a button to load a previously saved game	UR_GAME_LOAD
FR_GAME_END	Once the last customer has been served in scenario mode or when the user fails in endless mode, **the system must move to a screen displaying how long the user took to complete the scenario**	UR_GAME_END
FR_RECIPE	The system must keep track of the recipes for each item (salad, burger, pizzas or jacket potatoes)	UR_RECIPE

Non-Functional Requirements:

Requirement Code	Description	User Requirements	Fit Criteria
NFR_VISIBLE_COOK	Cooks should be visually distinct from background assets	UR_COOKS	A typical user* must be able to recognise the cooks 100% of the time
NFR_OVERLAP	Cooks should not visually overlap at any point	UR_WHICH_COOK UR_SWITCH_COOK	The cooks must never graphically overlap
NFR_CURRENT_COOK	The system should clearly show currently selected cook either through outline or icon	UR_WHICH_COOK	A typical user must be able to identify the selected cook in < 2 seconds
NFR_INFORM_SWITCH	Visual or sound effects should indicate when the cook has been switched	UR_SWITCH_COOK UR_WHICH_COOK	A typical user must be able to identify the newly selected cook in < 2 seconds
NFR_CLEAR_STATIONS	Stations should be clearly visible	UR_COOK_STATION	A typical user must be able to identify a station in <2 seconds.

NFR_SHOW_INTERACT	System should clearly show when a station is being interacted with, and the state of this interaction	UR_COOK_INTERACT	A typical user must be able to identify if and how a station is being interacted with in <2 seconds
NFR_CUSTOMER_ANNOUNCE	System should effectively inform the player of the arrival of new customers	UR_MODE_SCENARIO UR_CUSTOMER_ARRIVAL	A typical user must be aware a customer has arrived within 2 seconds of the customer arriving
NFR_ITEM_STACK	The system should show a stack of items the cook is currently holding	UR_COOK_STACK	A typical user must be able to tell what items they are holding within 2 seconds
NFR_END_OF_GAME	Audio and visual effects should show clearly that the game has ended	UR_GAME_END UR_MODE_SCENARIO	A typical user must be able to tell the game has ended within a second of it ending.
NFR_FRAMERATE	The game must be smooth enough to play consistently	UR_PLAYABLE	The game must have an average framerate of 30fps
NFR_STABILITY	The game must not crash while it is being played	UR_PLAYABLE	The game must not crash >99% of the time it is played.
NFR_DIFFICULTY	The game must change difficulty when the user selects different difficulties	UR_GAME_DIFFICULTY	A typical user should be able to tell they are playing a different difficulty within 2 seconds of starting the game.
NFR_LOAD	The game must load previously saved games without crashing	UR_GAME_LOAD	The game must not crash >99% of the time when a game is loaded and must load quickly

* A typical user is assumed to have no significant perceptual or cognitive impairments.