Requirements
Group 30 Triple 10
Team Members:
Kelvin Chen, Amy Cross, Amber Gange, Robin Graham, Riko Puusepp, Labib Zabaneh

Part a)

After arranging a meeting with the stakeholders, the next process was to decide on the requirements needed for the project. Using the provided product brief, which specified all the compulsory requirements needed for the first assessment, we were able to elicit the majority of requirements needed for the product. Next, we prepared a list of questions for the stakeholders, where members of the group inquired about additional specifications/details that should be incorporated into the product. Then, the questions were posed to the stakeholders, which consisted of an ENG1 lecturer and The University of York Communications Office. The questions were mainly feature-related, sound effect related, user-related, and some were non-feature-related. The feature-related questions included questions such as how long a burger should cook and etc. The sound effect-related questions were about the overall audio of the game. The user-related questions included questions such as who will be the users of the game and if they are familiar with the technology of the game. The non-functional questions were asked about the system's usability, availability, and security. There were also questions posed inquiring about The University of York Communications Office's requirements, of which they had none.

After finishing the meeting with the stakeholders, we had a clear understanding of the product's requirements and next we had to start documenting the requirements. First, we had to conduct research[1] into requirements specification and presentation. We examined published documents[2] and also extracted information from the book to help our process. For requirements specification, the general consensus was that the best notation to use for the requirements is natural language, seeing that after going through the "Software engineering" book[3] and searching through various articles, it was clear that was the optimal choice. Natural language specification is easy to understand and expressive, meaning that it is comprehensible to everyone and can be used for all the requirements we've elicited. We also decided that each requirement should have a consistent, meaningful name instead of a number id, which makes it easier to identify them later on. Lastly, for the requirements presentation, we agreed to use three tables–one for user requirements and two for system requirements. Tables are organised, simple, and easy to follow. Furthermore, by using tables we were able to easily organise the requirements in hierarchical order and order them by appropriate priority. Tables are also easy to extend, which makes it easier to add and refine requirements throughout the development lifecycle if necessary.

The next step of our project was to fully implement all of the requirements from the product brief. This was done by highlighting all the requirements, for example, the requirement 'You start with 3 reputation points' and then creating an instance of this requirement UR_REP_POINTS. We also received some more requirements from the stakeholders like adding 5 powerups that the player can use during the game so we created an instance of this, FR_POWERUPS. This allowed us to fully create the game to stakeholders' specifications to create a fully functional game that is fun to play.

Part b)

SSON: "The game shall enable players to control multiple chefs to prepare and cook food at each customer's request."

User Requirements

ID	Description	Priority
UR_CONTROL_COOKS	The game shall allow the player to control three chefs individually.	Shall
UR_GAME	Game is single-player	Shall
UR_INGREDIENTS	The player shall be able to collect ingredients via a pantry	Shall
UR_COOK_FOOD	The player shall be able to make a salad, burgers, pizzas and jacket potatoes	Shall
UR_SERVE_FOOD	The player shall be able to serve food to customers	Shall
UR_CUSTOMERS	The system shall allow customers to arrive at the counter and wait to be served	Shall
UR_FAILING_STEPS	The player shall be able to overcook or fail making food or serving customers	Shall
UR_WRONG_INGREDIENT	The player shall be able to get rid of ingredients that they have accidentally create	Shall
UR_UX	The game shall offer a pleasant user experience	Shall
UR_INSTRUCTIONS	The instructions to cook food shall be displayed to the user along with controls	Shall
UR_SCALABILITY	The game shall be able to be displayed on both big and small screens	Shall
UR_ACCESSIBILITY	The system should be accessible for all users	Should
UR_ENJOYABILITY	The game should be engaging and enjoyable	Should
UR_SETTINGS	The game should provide the option to customise settings to the player’s preference	Should
UR_MENU	The system provides a menu for the player to begin a scenario and which is returned to once complete	May

UR_REP_POINTS	Each game starts with the player having 3 Reputation Points	Should
UR_UPGRADES	Allow upgrades throughout gameplay	Should
UR_DIFFICULTY	User can choose different modes ‘Scenario’,‘Endless’ and difficulties	Should

Non-functional requirements

ID	Description	User Requirements	Fit Criteria
NFR_DOCUMENTATION	The system shall have a guide that details all its functions	UR_INSTRUCTIONS	Clear instructions on how to play the game
NFR_OPERATABILITY	The system shall be operable by customers that have no previous experience with the game	UR_INSTRUCTIONS	Easy to understand interface with clear instructions
NFR_ACCESSIBILITY	The system shall be operable by those with accessibility issues	UR_ACCESSIBILITY	Cater for accessibility needs
NFR_USABILITY	The system shall contain no technical jargon	UR_INSTRUCTIONS	Not use any complicated terminology
NFR_GRAPHICS	The graphics of the game shall be clear and easy to understand. The graphics shall also be child friendly.	UR_UX	Large easy-to-read buttons, clear stations, clear ingredients
NFR_OS	The game shall be able to be played on multiple operating systems	UR_ACCESSIBILITY	The system will support Windows/Linux
NFR_SCENARIO_TIME	The scenario should have a fast-paced rhythm	UR_ENJOYABILITY	The scenario will take around 5-6 minutes

Functional Requirements

ID	Description	User Requirements
FR_CHANGE_PLAYABLE _ CHARACTER	The system shall let the user switch control between playable characters	UR_CONTROL_CHEFS
FR_MOVE_PLAYABLE_C HARACTER	The system shall have controls that move the playable character	UR_CONTROL_CHEFS
FR_CHANGE_COOK	The system shall allow the player to begin a task with one cook, then switch to the other without affecting the continuation of said task	UR_CONTROL_CHEFS
FR_SHOW_COOK	The system shall highlight which cook is currently being controlled by the player	UR_CONTROL_CHEFS
FR_CUSTOMER_ARRIVAL	Customers arrive at different intervals starting by themselves then in groups	UR_CUSTOMERS
FR_RECIEVE_FOOD	Customers only wait a set period of time before they leave	UR_CUSTOMERS
FR_REP_POINT_LOSS	If customer leaves without food a reputation point is lost	UR_REP_POINTS
FR_MENU	The system shall display a main menu that is initially loaded when the program is started	UR_MENU
FR_START	An option shall be provided to begin the scenario	UR_MENU
FR_GRAB_ITEMS	The system shall allow the player to grab various in-game items	UR_INGREDIENTS
FR_STATIONS	There is cooking stations for cutting, baking, frying	UR_COOK_FOOD

	and serving	
FR_STEP_VALIDATION	Each preparation step shall require input from the player and the correct ingredient(s)	UR_COOK_FOOD
FR_COOK_BUSY	A cook shall carry out a preparation step for a given period of time, making this cook inoperable until the task is complete	UR_COOK_FOOD
FR_PLACE_ITEMS	The system shall let the player place items after grabbing them.	UR_SERVE_FOOD
FR_REMOVE_ITEMS	The system shall let the player completely remove items from the game	UR_WRONG_INGREDIENT
FR_OVERCOOKING	The system shall allow items to be overcooked or overbaked	UR_FAILING_STEPS
FR_DISH_VALIDATION	The system shall provide verification of a dish when served to check if correct	UR_SERVE_FOOD
FR_KITCHEN_UPGRADES	User can invest earnings to enable other cooking stations	UR_UPGRADES
FR_POWERUPS	User can obtain 5 different powerups during the game	UR_UPGRADES
FR_STAFF_UPGRADES	User can invest earnings to call kitchen staff back from leave	UR_UPGRADES
FR_FULL_SCREEN	The system should let the user play on full screen mode	UR_SCALABILTY
FR_TIMER	The system should have a timer that shows how much time has elapsed	UR_TIME_TO_COMPLETE

FR_INVENTORY	The system shall limit the player to how many ingredients can be stored at once	UR_INGREDIENTS
FR_INVENTORY_STACK	Each new ingredient will be added to the top of the stack and when interacting with a station will drop top ingredient	UR_INGREDIENTS
FR_PANTRY	The player shall choose which ingredients to take from the pantry	UR_INGREDIENTS
FR_INGREDIENTS_INFINITE	The system shall allow the player an infinite number of each ingredient	UR_INGREDIENTS
FR_ZERO_REP_POINTS	If all reputation points are lost then player loses the game	UR_REP_POINTS
FR_ENDLESS	Endless mode will keep track of the most customers you have managed to serve	UR_DIFFICULTY
FR_DISPLAY_ORDER	The system shall display current customer orders for the player to easily see	UR_ACCESSIBILITY
FR_SAVE_CHANGES	The system should remember the user's settings	UR_SETTINGS
FR_VERIFY_SETTINGS'_CHANGES	The system should verify if the user would like to save the changes	UR_SETTINGS
FR_SOUND	The system may have music/ sound effects	UR_SETTINGS
FR_MUTE_SFX	The system shall let players mute sound effects/music	UR_SETTINGS