Requirements

Group 18

Team B

Olivia Betts
Zac Bhumgara
Nursyarmila Ahmad Shukri
Cameron Duncan-Johal
Muaz Waqas
Oliver Northwood
Teddy Seddon

Introduction

Our requirements were elicited through the supplied assessment brief, as well as communication with the client (which occurred in the form of both an in-person meeting and via email).

Initially, we had a single statement of need from the customer via the supplied assessment brief: “You are to build a single-player game that requires managing the staff around a kitchen, who will be preparing various dishes requested by customers coming into the Piazza Restaurant.” From here, we communicated more directly with the client through email and in-person meetings. This allowed us to create a full list of requirements, which we split into two major sections:

- User requirements, which are tasks users should be able to do with the system, and written for non-technical people that will be involved in the process
- System requirements, which are written for technical implementations, and describe *how* the system will deliver the user requirements

We then split the system requirements further, into functional requirements (actions the system must do to provide useful functions) and non-functional ones (qualities the system has).

They are presented below in three tables (user, functional, non-functional). We chose this above a more textual presentation (eg, a use case) as it presents the information more succinctly, and the act of giving each requirement a unique ID also makes it easier once the development cycle moves on to software architecture and implementation, as we already have a basic structure in place.
Each user requirement also has a priority: “shall” for the highest priority requirements that *must* be featured; “should” in the middle, for requirements that ideally shouldn’t be missing but could be; and “may” for the lowest priority requirements.

Many rows in the NFR and FR tables are extensions of specific URs, and in these cases a link to the relevant UR row is provided. Additionally, many of the NFRs have further specific fit requirements to assess if they have been implemented and to what extent this implementation has been successful. In these cases, these requirements have been detailed.

User Requirements

ID	Description	Priority
UR_UX	Game should look appealing, and should rely on different shapes and visuals to tell items apart instead of colours and sounds	Shall
UR_FAMILY_FRIENDLY	Game must not have any swearing, violence, or graphic content	Shall
UR_LICENCED	All assets and technologies used must be appropriately licenced and attributed	Shall
UR_OFFLINE	The game shouldn't need a network connection to run	Should
UR_SIMPLE	The game design, assets, features, etc, should be kept simple where possible. Gameplay should be simple to pick up	Should
UR_SYS_REQUIREMENTS	No specific system requirements for the game, should be able to run on a standard computer	Should
UR_ARCADE_LIKENESS	The game will have the ability to "play itself" after a certain time has passed with no input. It will have no save profiles, but a global leaderboard. One person shouldn't be able to occupy the machine for hours	Shall
UR_DOCUMENTATION	The code must be fully documented. Another team should be able to pick up the code and work on it	Should

Functional Requirements

ID	Description	User Requirements
FR_GAMEMODES	Can be played in **Scenario Mode** (configurable amount of customers to serve in time limit - default 5) or **Infinite Mode** (try achieve high score by serving as many customers as possible. Time between customers and time they wait will decrease as game progresses)	
FR_CUSTOMERS	Customers arrive at intervals and need to be served within time limit. They will wait at a counter and pay once order is received	
FR_MULT_COOKS	A player should be able to switch between three different cooks.	UR_SIMPLE
FR_CONTROLS	The game will use a mouse and a keyboard to control cooks.	UR_SYS_REQUIREMENTS UR_OFFLINE
FR_COLLISIO	If a cook bumps into another cook, nothing will happen e.g. no food	UR_SIMPLE

N	spilled.	
FR_RECIPES	Each recipe has multiple steps that must be carried out before it can be served. The recipes are **salad** (chop lettuce, onions, tomato + serve), **burger** (form patty, fry it, toast buns + serve), **pizza** (make dough, puree tomatoes, slice mozzarella, make pizza, cook in over, slice + serve), and **jacket potato** (cut potato, cook, add beans, add cheese, cook again, + serve)	
FR_ITEM_INTERACTION	A cook can hold none, one, or two items, added to the top of their held stack. If they hold two, they can't do any action until they put an item down. They should be able to drop the top item or whole stack on the cooking station.	
FR_PREPARE	A cook can prepare ingredients at **stations** (cutting, baking, frying, serving). This can include chopping lettuce, adding cheese, or toast burger buns. It may have multiple steps, like flipping a burger patty to cook on both sides. Ingredients don't run out	
FR_FAIL_STEP	When an item burns etc, that step will have to be re-completed (not the entire recipe)	
FR_INVEST	At start, not all stations are available (eg, can only make salad.) Money earned can be used to buy more	
FR_REPUTATION_POINTS	"Lives". Start with 3, (and cannot gain more in Scenario Mode). Failing to serve customer will lose point. Game ends when all points lost	
FR_GAIN_REP	In Endless Mode **only**, reputation points can be regained via uncommon methods (eg serve customer especially fast)	
FR_DIFF_INCREASE	Customers initially arrive 1 at a time, as difficulty increases this can go up to 2 or 3 at once	

Non-Functional Requirements

ID	Description	User Requirements	Fit Criteria
NFR_DIFICULTY	The game will be playable by new players who have never had any experience playing before.	UR_UX UR_SIMPLE	95% of players should understand all game concepts.
NFR_SYSTEM_INTEGRATIONS	The game should be playable on all PC machines of any modern standard.	UR_SYS_REQUIREMENTS	The game will run on a PC with 4GB RAM and 10 GB internal storage drive with Java installed.

NFR_SYSTEM_MONITOR	The game resolution should adjust accordingly to any sized monitor screen.	UR_UX	The game will run on a minimum 720×480 resolution monitor.
NFR_VISIBILITY	The game should be visible from a distance to attract visitors.	UR_UX	The game should be appealing and attractive from 2 to 3 metres away.
NFR_GRAPHICS	Graphics are not restricted but should be clear and concise to all users of all	UR_UX	Individual assets should be distinguished after being put under a black & white filter.
NFR_TIME_LIMITS	The game should last a shorter amount of time so one person can't hold all access to the machine.	UR_ARCADE_LIKENESS	An individual user shouldn't be able to play in 'endless' mode for longer than 10 minutes as difficulty should reach too great a level.
NFR_BACKGROUND_ATTENTION	The game should be played in the background by an algorithm to attract customers when nobody is playing on the machine.	UR_ARCADE_LIKENESS	
NFR_LEADERBOARD	There should be no option to save a game to a user profile, but rather a global leaderboard where players can enter a name associated with their score in the endless game mode.	UR_ARCADE_LIKENESS	
NFR_VIOLENT_GRAPHIC_CONTENT	The game shall not contain any graphic violence or swearing as the game should be appropriate for children.	UR_FAMILY_FRIENDLY	
NFR_COLOUR_SOUND	There shall be no reliance on the sounds or colours in the game as it should be accessible to all. The colours should however be distinct enough for colour blind players to distinguish different assets.	UR_UX	
NFR_MAINTENANCE	The game's code shall be commented upon and documented for future maintenance or updates.	UR_DOCUMENTATION	
NFR_MANUAL	The game shall require no physical manual but should have a brief in-built instructions panel within the game.	UR_SIMPLE	