Requirements

Assessment 2

Group 5 :

Callum Watson

Jack Guan

Craig Slomski

Chase Mo

Kamrul Islam

Yaseen Khan

Requirements Description

SSON: PizzaPanic is a single player arcade-style game designed to be used on an open-day, therefore should provide an easily playable, accessible and enjoyable experience.

We have a main client, who is our customer, who wants to sell the game to other audiences and they will be the main point of communication during development of the game for requirements. A meeting with our customer, the main client, was arranged where we could discuss and negotiate desirable requirements. Prior to the meeting, our group gathered together and discussed a range of questions to ask the customer, by grouping different possible features of the game gathered through the product brief. We then agreed on a set of questions to ask the client, so we could get more specialised requirements.

During the meeting with the client, we learnt that the University of York wants our team to make a game shown on Open Days that will attract upcoming computer science undergraduate students to the University of York as this is when the majority of visitors see and make the decision about whether to apply to the university. They will play this game during taster sessions to get a feel of how practical sessions work at the University like in previous years. This in turn will help increase overall Computer Science undergraduate admissions to the university and thus increase profits for the department. We were also given a brief by our main client that details the important game features that we must include in phase one of development. transcribed the meeting to make sure we have a note of what the client wanted. The customer also pointed out certain features of the game that they would like, how the game should focus more on implementation than the design and aesthetics, and how the game should be easily picked up by players, and run on a short 5-10 minute loop.

The list of requirements were then summarised into 16 user requirements, for which we formatted them in the form of tables, where each user requirement has a unique identifiable ID, followed by description of what each user requirement is about. Each of them is then classified priority wise based on their importance to the game and whether they were completely necessary as clients tend to underestimate how long certain requirements take to be implemented or they want a certain requirement that the group implements that they can't function without. The priorities in decreasing order of priority are Shall, Should and May. The user requirements had to be split into functional and non-functional requirements based on how the user requirements are described. Functional requirements refer to different features to be integrated within the system, whereas non-functional requirements are mainly associated with the style and quality of the system.

The functional requirements were also formatted as a table, with a unique functional ID, followed by its description. Each functional requirement is linked to its respective user requirement from which it is derived from. The non-functional requirements table consists of an ID, description, respective user requirements ID and a fit criteria which specifies the qualification of the non-functional requirement.

The layout of the tables of system requirements was inspired from the content covered in the requirements lecture slides to make sure that the requirements were in an organised fashion for implementation. We could only derive a few non-functional requirements as the user requirements elicited were not specific enough, and focused more on the prominent features the system should have rather than qualities of the system.

User Requirements

ID	User Requirement Description	Priority
UR_EXPERIENCE	The game must be fun and enjoyable to attract students to study at the University of York.	Shall
UR_DURATION	Game should be on a short 5–10-minute game loop that can be easily picked up by players.	Shall
UR_DIFFICULTY	Game must have a balanced learning curve where the difficulty increases gradually for the player. The game can't be too difficult, but players need to be challenged.	Shall
UR_LEADERBOARD_2	Games should have a leader board system so players can compare their scores to others.	May
UR_TUTORIAL	Game must have a simple tutorial screen which specifies controls and objectives of the game.	Should
UR_CONTROL	Game consists of a simple keyboard control scheme.	Shall
UR_SOUND	Sound effects for cues for player actions. (Background music optional)	Shall
UR_DISCARD	Allows players to discard ingredients when they go wrong in the middle of the recipe.	Shall
UR_RECIPE	List of recipes to be displayed on screen and taken off screen when the player has completed them. The priority of the recipes needs to be specified. Each recipe should be limited to three ingredients.	Shall
UR_ACTION_TIME	Timers should be displayed above workstations indicating the time left for the current action, i.e., cooking the dish, indicating if it's overcooked	Shall
UR_LAYOUT	Game should have the same map layout and fixed workstation for each round. Customers will wait at the collection counter which will act as the boundary of the game screen.	Shall
UR_STYLE	Graphics of the game need to be simple and Arcade like.	Should
UR_ACCOMMODATIONS	The game should be able to accommodate people with disabilities i.e., colour blindness and hard of hearing.	Should
UR_LEVEL_COMPLETION	For phase 1, order completion time is tracked, and scores obtained. Impossible to fail the round or take wrong orders. Certain steps can be failed, resulting in longer completion time.	Shall
UR_COMPATIBILITY_2	Complete implementation must be able to run on multiple OS systems and screen sizes.	Shall
UR_FUNCTIONAL	The game needs to be functional and implemented well over the aesthetics of the game.	Shall
UR_CONTINUE_2	The user should be able to continue the game from their last incomplete game (no win/loss).	Shall

UR_FAIL_STATE_2	Users should be able to fail preparation steps such as burning food and will have to repeat the step to continue. Users should also be able to fail and lose the game if reputation points reach 0.	Shall
UR_CUSTOMERS_2	Customers should have randomly generated orders and arrive individually or in groups (not all at once). Number of customers served should be displayed during runtime.	Shall
UR_MODE_2	Mode entails different difficulty levels (easy, normal and hard) as well as the endless mode.	Shall
UR_SCENARIO_2	Additions to the scenario mode based on feedback from original implementation (from Assessment 1) - should be configurable.	Shall
UR_PROGRESS_2	Users should be able to progress into further difficulty levels, in turn unlocking new features as they progress.	Shall
UR_PERKS_2	Perks/power ups will be implemented to allow for a dynamic game experience, increasing enjoyment and playability of the game.	must
UR_BONUS_2	Potentially offer users 'double point' or 'VIP' customers who will earn you an increased amount of money.	Should

Functional Requirements

ID	Functional Description	User Requirements
FR_INCREASED_DIFFI CU LTY	The system will have the difficulty increased with every customer, by asking for more complicated recipes as time goes on.	UR_DIFFICULTY
FR_UPDATE_LEADER BOARD_2	The system should update the leader board every time the player completes a level with their time. The system should allow users to enter their name with timing which will be restricted to 3 characters.	UR_LEADERBOARD
FR_DISPLAY_TUTORIAL	Should display a tutorial screen with clear and simple instructions for the user so they know how to play. If a new station is introduced, then another short tutorial will be displayed. Tutorials displayed before the start of a level.	UR_TUTORIAL
FR_TUTORIAL_SKIP	The system should allow for the Tutorial screens to be skipped with key press.	UR_TUTORIAL
FR_CONTROL_MOVE ME NT	The system should allow the player chefs to move on screen with the WASD keys.	UR_CONTROL
FR_CONTROL_ACTIO N_ KEY	The action key to switch between chefs is 1,2,3 and the action key to grab, work on and discard an item is E key.	UR_CONTROL

FR_ITEM_REMOVAL	The system must allow the player to remove items from the game if they accidentally ruin an order or want to discard an item in the trash station using the action key E. This item needs to be removed from the playable area.	UR_DISCARD
FR_RECIPE_DISPLAY	The system should display the dish ordered by the customer, and when a particular dish is completed, it must be served to the customer before being removed from the screen. Incorrect dishes cannot be served to the customer.	UR_RECIPE
FR_RECIPE_PRIORITY	The recipe displayed is the top priority. The priority of the dishes being in first come first serve order from when the customer has asked for the dish.	UR_RECIPE
FR_ACTION_TIMER_BAR	For each workstation that takes time to complete an action, a countdown to be displayed above that station. If an action runs out of time, then the player has failed in that step, and needs to redo the action. The ruined dish should be discarded.	UR_ACTION_TIME
FR_FIXED_LAYOUT	The system has a consistent map layout displaying fixed workstations for each level. Each workstation should have its own designated action function.	UR_LAYOUT
FR_SCORE_TRACKING	The system should produce a score which is the tracked time, longer time taken to complete an action will result in a lesser score. Failed steps will result in more completion time which affects the score.	UR_LEVEL_COMPLETION
FR_COMPATIBILITY	The game should be able to run on different screen resolutions, old and new computers.	UR_COMPATIBILITY_2

FR_DURATION_TIMER	The system needs a timer in the corner of the game screen tracking how long the player takes to finish a level, with the maximum duration lasting no more than 10 minutes.	UR_DURATION
FR_GAME_ASSET_STYLE	The game assets should have a simple design and be easily recognizable, the animations being not too complicated.	UR_STYLE
FR_SAVE_GAME_2	Saving the game should be implemented with extensive testing - ensure that as much detail is saved and loaded into the game.	UR_CONTINUE_2
FR_REPUTATION_POINTS_2	Reputation points will aid in reaching a fail state. Start with 3 reputation points and lose 1 for failing to serve a customer within a specified time limit.	UR_FAIL_STATE_2
FR_CUSTOMER_ARRIVAL_2	Customers should arrive individually or in groups rather than all at once.	UR_CUSTOMERS_2
FR_DIFFICULTY_2	Difficulty level should be choosable and either unlocked or require unlocking.	UR_MODE_2
FR_ENDLESS_2	Endless mode should be implemented - a fail state for endless mode would include reputation points running out and losing the game, as well as existing fail states within the game itself. A running total of the customers served will be shown on the game screen.	UR_MODE_2, UR_FAIL_STATE_2, UR_CUSTOMERS_2

FR_COMPATIBILITY_2	The end product should be developed and fully tested to provide compatibility for multiple operating systems.	UR_COMPATIBILITY
FR_PROGRESS_2	New stations and recipes should be unlocked as the user progresses further into the game's difficulty levels.	UR_PROGRESS_2
FR_PERK_DROPS_2	Power ups/perks should be dropped at random intervals during the game - ensure there is a balance as to give users a slight advantage but not too much.	UR_PERKS_2

Non-Functional Requirements

ID	Description	User Requirements	Fit Criteria
NF_SIMPLE_EN GLISH	The system shall be easily interpretable by players by usage of simple English and easy symbols.	UR_EXPERIENCE	Jargon shall not be used in the game
NF_USER_EXPER IENCE	The game should be easily playable by players who haven't heard of the game before or have very limited knowledge on how to play the game.	UR_EXPERIENCE	Players should have basic knowledge on how to play the game after the tutorial. Players must also have knowledge on how to operate a computer.
NF_SOUND_EFF ECT	The sound effect should be played when the user does an action.	UR_SOUND	The sound should be activated within 70 milliseconds of the player action key.
NF_CONTROL_S CHEME	Needs a responsive Control scheme needs to be responsive	UR_CONTROL	Key presses should be registered within 70 milliseconds.
NF_RECIPE_COMPLEXITY	Each dish has a certain amount of ingredients going into it, which determines the dish complexity.	UR_RECIPE	The number of ingredients per dish should be limited to 3, which 1 being the minimum.
NF_BOUNDARY_ LAYOUT	The system should display a counter where the customers wait, which is the boundary of the game screen.	UR_LAYOUT	No playable object should be outside of the boundary
NF_ACCESSIBILITY	The system should have game options for people with accessibility requirements.	UR_ACCOMMO DATIONS	The system shall be operable by students and staff with impaired vision, hearing and mobility.
NF_DOCUMENTATION	The game's code should well be documented to allow the game to be department to modify and update the game implementation in the future	UR_FUNCTIONA L	Each main class in the code should be accompanied by well written documentation/Doc String.

NF_CODE_2	Refactor the code from the original implementation to use data gathered instead of hardcoded (for interaction).	UR_FUNCTIONAL	The tryInteract function is hard-coded with variables within the code and should be refactored to instead use data gathered from gameplay.