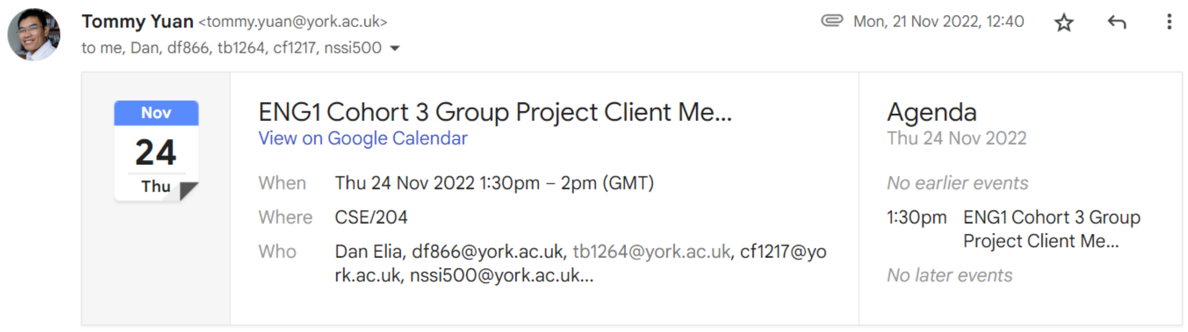
Requirements

2a

We elicited the requirements for our project using a combination of the project brief and a short, 30 minute interview with the client. It was important to us to analyse the briefing document in great detail in order to ensure we had a good understanding of what was required of us before proceeding to the client interview. We spent one of our first meetings focusing largely on understanding and decomposing the product brief. Once we were confident in our understanding, we scheduled a short client interview with a stakeholder, Tommy Yuan, in order to elicit any further requirements that were not in the brief. We took short notes in the meeting, though refrained from making any audio or video recordings due to privacy concerns. Below you can see the meeting we scheduled with him via Google Calendar.



Tommy Yuan <tommy.yuan@york.ac.uk>
to me, Dan, df866, tb1264, cf1217, nssi500
Mon, 21 Nov 2022, 12:40

Nov 24 Thu

ENG1 Cohort 3 Group Project Client Me...
View on Google Calendar

When	Thu 24 Nov 2022 1:30pm – 2pm (GMT)
Where	CSE/204
Who	Dan Elia, df866@york.ac.uk, tb1264@york.ac.uk, cf1217@york.ac.uk, nssi500@york.ac.uk...

Agenda
Thu 24 Nov 2022
No earlier events
1:30pm ENG1 Cohort 3 Group Project Client Me...
No later events

This meeting has significantly influenced the details of our requirements, for example we learned that the game would be used on open days and played by prospective students on a large screen. This means that the game must be adaptable to this environment and have an appropriate interface. We have presented the requirements as a series of user and system requirements, as this is an important distinction to pay attention to during development. The user requirements help us to understand which features we need to implement, and the system requirements allow us to make decisions about how we should implement said features. Each feature has a unique code which makes it easier to identify. This will be useful in later documentation when we may need to reference certain requirements. It is also clear from the code whether it is a user or system requirement - user requirements begin 1.X and system requirements 2.X. The layout of our requirements is based on research we carried out into example requirement documents and academic literature on the subject. We followed guidelines set out in the paper "The Formal Reference Model for Software Requirements" (Nazaruka, Osis 2019). A full citation can be found in the bibliography at the end of this document. Additionally, to better understand the difference between system and user requirements, we referred to online resources such as the software engineering stack exchange forum. Details of this can be found in the bibliography.

2b

1. User Requirements

1.1: Users must be able to move and control two chefs, with the ability to switch between them

- **1.1.1:** Chefs must be able to interact with the station in front of them
- **1.1.2:** If the chef interacts with a station with an item on it, they pick the item up.
- **1.1.3:** A chef may pick up as many objects as they want
  - **1.1.3.1:** Items the chef is holding are displayed above its head
- **1.1.4:** A chef can place an item on an empty station. It is taken from the top of the stack they are carrying.
- **1.1.5:** The chefs must be able to be moved independently

1.2: There must be 5 customers that will arrive periodically during the game

- **1.2.1:** The customers order must be displayed clearly and visually in the interface
- **1.2.2:** When the user successfully creates a dish that satisfies the customers order and places it on the counter, the customer will leave
- **1.2.3:** If the user places the wrong dish on the counter, this dish is discarded and the customer will stay there until the correct dish is placed.

1.3: There must be a pantry which a cook can interact with in order to collect ingredients

- **1.3.1:** The pantry has an infinite supply of ingredients, it cannot run out
- **1.3.2:** The user can choose which ingredient they take from the pantry

1.4: There must be a station for cutting ingredients, which is used in preparing recipes

- **1.4.1:** When a user places an ingredient on the cutting station, it takes a fixed amount of time for the cutting to finish.
- **1.4.2:** The user must be able to cut lettuce, tomatoes and onions by placing them on the cutting station
- **1.4.3:** The user can't place anything else on the cutting station. If they try, nothing will happen

1.5: There must be a station for grilling/frying, which is used in preparing recipes, specifically the burger recipe

- **1.5.1:** When the user places an ingredient on the grill, it takes a fixed amount of time for the item to finish grilling
- **1.5.2:** The user must flip an item partway through grilling, where it will then take another fixed amount of time before it is fully cooked.
- **1.5.3** Items placed on the grill can't burn, they can be left on the grill for as long as the user wants
- **1.5.4:** The user must be able to grill patties
- **1.5.5:** The user can't place anything else on the grill. If they try, nothing will happen.

1.6: There must be a station that is used for toasting the buns, which is used in the burger recipe

- **1.6.1:** When the user places buns in the toaster, it takes a fixed amount of time for them to finish toasting
- **1.6.2:** Items in the toaster can't burn
- **1.6.3:** The user can't place anything else in the toaster other than buns. If they try, nothing will happen.

1.7: There must be at least one station where users can assemble final dishes

- **1.7.1** Users must place items on this station in the correct order, as is specified in the recipe

1.7.2 Any item can be placed in this area, it is up to the user to ensure that it is the correct item

1.7.3: Items placed are added to the top of the stack at the station

1.7.4: If items have been placed in the correct order in this station, the full set of correct items will turn into a dish

1.8: There must be recipes for burgers and salads in the game, which can be ordered by customers and created by the user's chefs

1.8.1: To create a burger, the user must form a patty, fry it, toast the buns and assemble it in the correct order as specified by the customer.

1.8.2: To create a salad, the user must chop the lettuce, tomatoes and onions and assemble it in the correct order as specified by the customer.

1.9: Once all the customers are served, the game must end immediately and an ending screen displayed

1.9.1: The timer must immediately stop once all customers are served

1.9.2: The end screen must clearly and concisely display how long it took the user to complete the game

1.9.3: There must be an option to restart the game, or to exit completely.

1.10: There must be a simple yet effective homepage that is displayed upon start up

1.10.1: It must have a start button, which launches the game play

1.10.2: It must have an exit button, which exits the game completely.

2. System Requirements:

2.1: There must be full keyboard controls to move the cooks, switch between them, display their inventory and interact with stations and objects.

2.1.1: Cooks must collide with stations and counters, they may not stand on a tile where there is already an object such as a grill, this is how they interact with the station

2.1.2: Controls must be robust, an incorrect or invalid key press should be ignored by the system and shouldn't cause errors.

2.1.3: Controls must be intuitive - WASD to move and Q to switch chefs

2.2: Random orders must be generated from customers

2.2.1: There must be a counter to keep track of the fixed amount of orders in the game: 5, one per customer

2.2.1.1: There should be separate counters for each of recipes

2.2.1.2: When the user fulfils an order, the appropriate counter is decremented.

2.2.2: Orders must appear at random intervals

2.2.3: Orders must be decided randomly - either they are a burger or a salad. This decision is not predetermined.

2.3: There must be functionality for handling and using items

2.3.1: Items carried by a chef must be implemented as a stack, ie first in first out

2.3.2: When a chef picks up a new item, a new label must be created in the HUD to signify this in an inventory which can be accessed by pressing a key.

2.3.3: The label must be deleted when the user drops an item

2.4: There must be appropriate hit detection throughout the environment

2.4.1: Chefs must collide with counters, stations and the pantry so there must be appropriate hit detection

- **2.4.1.1:** There should be a slight overlap between the chef's head and the counter, to convey their height.
- **2.4.2:** Chefs shouldn't collide with each other, two chefs may pass through and stand on the same tile freely.

2.5: There must be an appropriate system for preparing items and recipes

- **2.5.1:** There must be timers to alert the user of when ingredients are prepared (grilled/chopped etc)
- **2.5.2:** There must be validation in place so that only appropriate items can be placed on the corresponding stations
- **2.5.3:** An error must be raised if the user tries to create a recipe that doesn't exist
  - **2.5.3.1:** These error messages must be concise and aesthetically fit with the rest of the game in a way that makes sense.

2.6: There must be a functional main menu screen upon the game start up

- **2.6.1:** Design and layout must be very simple
- **2.6.2:** There must be a button to start the game
- **2.6.3:** There must be a button to exit the game

2.7: The game must perform well, even on machines with limited processing power

2.8: The UI and graphics must scale well to be played on screens of various sizes

- **2.8.1:** The game should be playable on all PC machines of any modern standard
- **2.8.2:** The game resolution should adjust accordingly to any sized monitor screen
- **2.8.3:** The graphics are not restricted but should be clear and concise to all users

2.9: Design of UI should take into account those with visual impairments

- **2.9.1:** Fonts for text must be clear and easy to read
  - **2.9.1.1:** The background of text must also aid readability
  - **2.9.1.2:** The text itself should be concise and simple to understand - it should clearly explain what is happening to the user
- **2.9.2:** Where possible, information should be represented graphically, not textually.

2.10: There must be an end screen that is displayed once all 5 customers have been served

- **2.10.1:** When this screen is displayed, the timer must immediately stop, and its value must be output to the screen
- **2.10.2:** There must be a button to restart the game
- **2.10.3:** There must be a button to exit the game

2.11: The code must be easy to read and maintainable

- **2.11.1:** There should be comments throughout
- **2.11.2:** The code should follow a suitable format that aids readability and maintainability.

2.12: The code should be developed in an object oriented style

- **2.12.1:** The chef, ingredients, stations and the customer should all be coded as classes
  - **2.12.1.1:** These classes should have appropriate attributes to define different instances of them (eg the two chefs, different ingredient types)
  - **2.12.1.2:** Classes should also have correct methods in order to make the game function correctly
- **2.12.2:** Encapsulation should be used throughout the code in order to make it more robust
- **2.12.3:** Inheritance should be used to create subclasses

Bibliography

Nazaruka, E. and Osis, J. (2019) “The formal reference model for software requirements,” *Communications in Computer and Information Science*, pp. 352–372. Available at: https://doi.org/10.1007/978-3-030-22559-9_16.

5had3sofQu4rtz *et al. What is the difference between user requirements and system requirements?*, *Software Engineering Stack Exchange*. Available at: https://softwareengineering.stackexchange.com/questions/264113/what-is-the-difference-between-user-requirements-and-system-requirements