Change Report

Team 4 - Undercooked

Fin Cochrane
Sehran Ahmed
Sam Davis
Hamza Salman
Owen Thomas
Zhenyi Xu

Changes

We started with a general idea of what needed to be done - the new requirements needed to be implemented as part of the brief, and other changes made to the existing architecture of the project. We used a github project board to keep track of these as milestones.

Less notable changes were kept track of primarily using detailed commit notes on github, along with written notes on Discord during meetings. Any important changes were also mentioned in the main Discord chat to ensure everyone was aware.

Changes were reviewed by other members of the team following major implementation milestones, such as implementing a new feature.

Requirements

Extended from https://eng1-team3.github.io/project_eng1_team3/Req1.pdf

Given the requirements for the second assignment, we needed to update the requirements table to include specific new requirements, and update existing entries to correspond with the new brief.

Updates to existing entries

User Requirements

ID	Description	Priority
UR_NUMBER_OF_CUSTOMERS	The game shall support varying numbers of customers, chosen by the player	Shall
UR_COOKS	The game shall let the player control 3 or more cook characters, one at a time. Cooks should be locked out for certain of time when they are used up	Shall
UR_REPUTATION_POINTS	The game shall have 3 reputation points that act as lives for the player	Shall

Functional Requirements

ID	Description	User Requirement
UR_PREPARE_STAGE	The user needs to interact with a cooking station during preparation. If they failed, they will have to start the preparation from the beginning	UR_RECIPES

Non-Functional Requirements

No updates were needed to existing entries as all existing non-functional requirements still apply to the game with the new features.

New entries

User Requirements

ID	Description	Priority
UR_ENDLESS_MODE	The game shall include a mode to play forever until all reputation points are lost	Shall
UR_CUSTOMERS_AT_A_TIME	The game shall increase the amount of customers arriving at a time as the game progresses, in the endless mode	Shall
UR_EARNINGS	The game shall allow the user to earn money by completing orders, and be able to buy new cooks and stations with this money	Shall
UR_CUSTOMER_TIME_LIMIT	The customers will each have a time limit to be served before leaving.	Shall
UR_FAIL_PREPARATION	The user will be able to fail a preparation step, such as burning when frying.	Shall
UR_SAVEGAME	The game shall include the ability to save and load progress	Shall
UR_POWERUP	The game shall have power ups that alter gameplay	Shall
UR_DIFFICULTY	The game shall have multiple difficulty modes	Shall

Functional Requirements

ID	Description	User Requirement
FR_PIZZA	The system will allow the user to combine specific ingredients to make a pizza	UR_RECIPES
FR_JACKET_POTATO	The system will allow the user to combine specific ingredients to make a jacket potato	UR_RECIPES
FR_BAKING	The system should be able to let users use the baking station to bake ingredients as part of the cooking process	UR_COOKING_STATIONS
FR_PURCHASE	The user can purchase more stations or cooks with their money.	UR_EARNINGS
FR_EARN	The user is rewarded with money upon completing orders	UR_EARNINGS
FR_GAME_OVER	When users have 0 remaining reputation points the game will end in a game over	UR_REPUTATION_POINTS

	and update the leaderboard if in endless mode.	
FR_SAVE	The system will be able to save status of on screen items	UR_SAVEGAME
FR_LOAD	The system will be able to load a saved status	UR_SAVEGAME
FR_POWERUP	Collectable power ups will alter the gameplay	UR_POWERUP
FR_DIFFICULTY	Different difficulties should make the gameplay more challenging	UR_DIFFICULT Y

Non-Functional Requirements
No updates were needed to existing entries.

Architecture

Extended from https://eng1-team3.github.io/project_eng1_team3/Arch1.pdf

Since we changed the architecture of the project quite a bit due to refactoring, the UML diagrams had to be completely redone: Updated UMLs (too large to fit in one diagram)

Method Selection and Planning

Extended from https://eng1-team3.github.io/project_eng1_team3/Plan1.pdf

Methodology and Tools
We didn't follow an exact method to allow more flexibility within the team. The vague timetable we followed was meeting on Wednesday, arranging tasks based on importance, and organising other meetings when needed.

Discord was chosen for communication, as it provided us with voice chat for meetings and the ability to share both text and images easily.
We used a GitHub repository branched from the original project for version control and sharing code, along with hosting the website.
The main IDEs we used were IntelliJ and VScode.

Team Organisation
We began work on this part of the project by organising ourselves into two teams: implementation and documentation. These were based on individual strength, and also willingness to work on different parts. However these teams were still flexible, and members could help with either one depending on what was needed.

Meetings were held on Wednesdays in person, and we would hold meetings whenever needed over Discord. Individual tasks were assigned at the end of each meeting. Anything important discussed in these meetings, such as code screenshots or tasks, was sent to the

Discord chat to keep track of and keep anyone unable to attend up to date with everyone else.

Planning
Our initial plan involved splitting up each requirement to make it easier to keep track of. While the requirements were updated, work was begun on implementation.
Upon beginning implementation, multiple issues were found that needed to be fixed before we could work on implementing the new requirements. Refactoring the code was focused on before moving onto requirements.

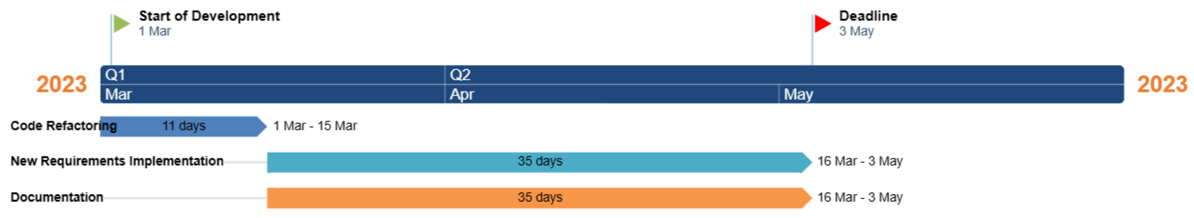
To begin with, our plan was to complete the code refactoring and as much of the documentation as currently possible before the last meeting before term break on March 15th, but development wasn't progressing as quick as expected. We then chose to push this deadline back a few weeks until the end of the month, as this would still give us more than enough time to work on the rest of the project. In the meantime, we assigned more people to work on the code refactoring to speed up the process.

Progress on refactoring slowed due to the break, however it was largely completed on April 1st. We had a short discussion over messages on what to do next, and began implementing the new requirements.

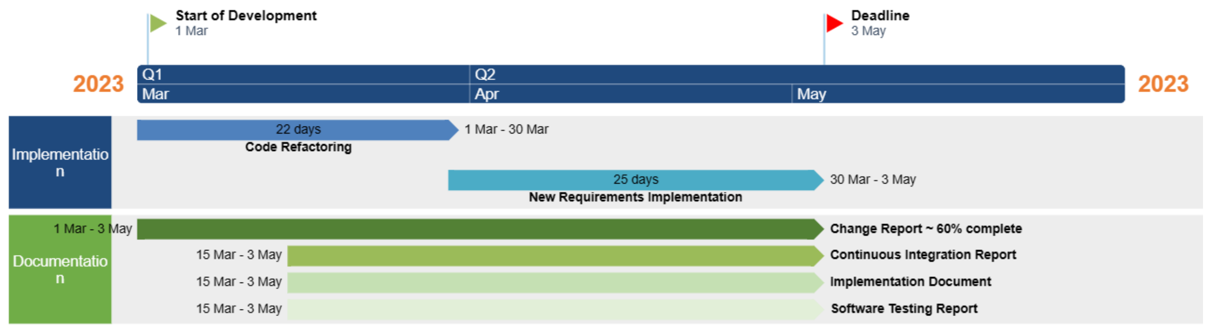
Following the break, we continued working on the implementation of new features, along with starting development on the tests. New features were able to be implemented very quickly due to the refactoring, and so further work on documentation could begin with the implementation fully completed aside from tests.

We ended up a little short of time so we had to work much harder for the final few days before the deadline to update the website and finish off work on everything else.

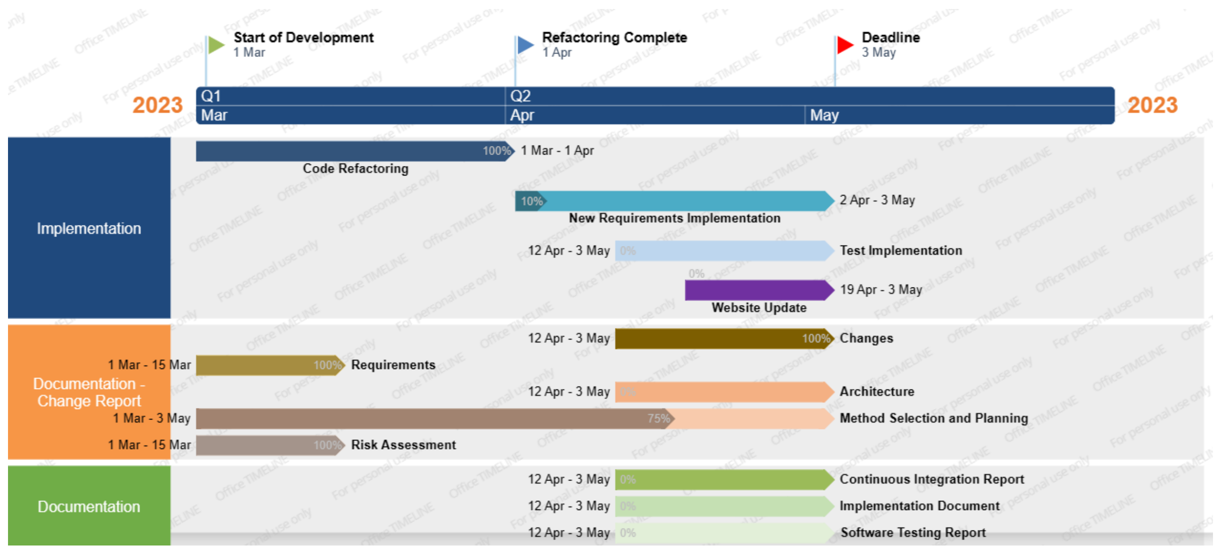
Gantt Charts
Created with [Office Timeline Online]()
Initial gantt chart 1/3/2023



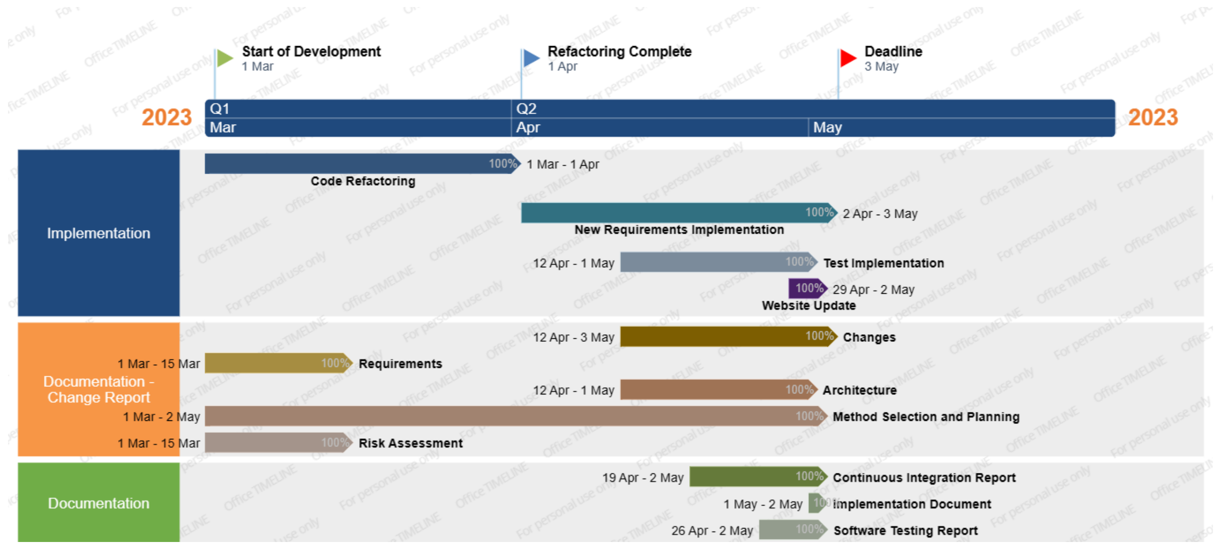
2nd gantt chart 15/03/2023



3rd gantt chart 12/04/2023



Final gantt chart 02/05/2023:



Risk Assessment

Extended from https://eng1-team3.github.io/project_eng1_team3/Risk1.pdf

We felt that there was no need to change or add to the risk management document (aside from changing those responsible for each risk to be members of our own team), as the project itself is relatively low risk, and even with our updated requirements, the existing risk register represents any possible risk. We did however make some preparations for possible risks, for example, we communicated with team 3 to adapt problems brought by different code styles.