

Change Report  
Eng1 Group 29

Adam Hewlett  
Ani Thomas  
Dan Kirkpatrick  
Dominik Hagowski  
Matthew Crompton  
Niko Chen

## Process, Tools and Conventions

### Requirements

Original:

<https://aaddmn.github.io/new.aaddmn.github.io/Deliverables/Assessment-1-Original/Req1.pdf>

For the requirements, we found that the requirements were laid out in a tabular format with an ID, Description and Priority. There were 3 different tables: User Requirements, Non-Functional Requirements and Functional Requirements. This layout kept it easy to understand. The naming conventions in the table were also very easy to understand, with a prefix signalling which requirement each row falls under and the rest signifying what the actual requirement is, such as UR\_TIME\_TO\_COMPLETE. We decided to keep this naming convention and layout the same as it allowed for easy readability, as well as keeping the majority of content the same as it was found to be accurate.

As for what was changed or added, we added in new requirements under User and Functional requirements which are listed below in the table, as well as updating some of the pre-existing requirements. This was in order for the new requirements to be in line with the assessment 2s new requirements such as the additional recipes and cooks. Any changes are highlighted in red

ID	Description	Priority/User Requirement
UR_CONTROL_COOKS	The game shall allow the player to control <b>three or more chefs</b> individually.	Shall
UR_COOK_FOOD	The player shall be able to make salads, burgers, <b>pizzas and jacket potatoes</b>	Shall
UR_SERVE_FOOD	The player shall be able to serve salads, burgers, <b>pizzas and jacket potatoes to customers</b>	Shall
<b>UR_MODE</b>	<b>The game will offer 2 game modes, scenario or endless</b>	<b>Shall</b>

UR_POWERUPS	The game will offer 5 different power ups in endless mode	Shall
UR_SAVES	The game will allow you to save the game in a specific state, and resume the game from that state	Shall
UR_MONEY	The player shall be able to earn money and buy more staff and cooking stations	Shall
UR_CUSTOMERS	The game shall have a fixed/or infinite number of customers to serve that require one dish each	Shall
UR_FAILING_STEPS	The player shall be able to overcook or fail making food or serving customers	Shall
FR_COOK_AND_BAKE	The System shall allow the player to cook and bake certain foods	UR_COOK_FOOD
FR_GAME_MODES	The system shall let the player pick between a scenario mode or a endless mode	UR_MODE
FR_POWER_UPS	The system will have 5 obtainable powerups in endless mode which last for 30 seconds	UR_PowerUps
FR_SAVE_GAME_STATE	The system will allow the game to be saved while playing	UR_SAVES
FR_RESUME_GAME_STATE	The system will allow the player to resume a game from a saved state	UR_SAVES
FR_EARN_MONEY	The system will allow the player to earn money through selling food	UR_MONEY
FR_SPEND_MONEY	The system will allow the player to spend money on more chefs or cooking stations	UR_MONEY
FR_OVERCOOKING	The system shall allow items to be overcooked or overbaked	UR_FAILING_STEPS

## Architecture

Original:

<https://aaddmn.github.io/new.aaddmn.github.io/Deliverables/Assessment-1-Original/Arch1.pdf>

In terms of architecture, we continued to use the object oriented design frame that the previous group had used from the last assessment. This, along with the use of a Responsibility Driven Design(RDD) strategy appeared to have worked very well for them so we decided to follow suit. Furthermore, after reading Section B of the architecture, we ended up agreeing with most of the points that they made, and decided to leave the majority of this section untouched, along with using the ideas and skills that they had laid out. We added points to this section on the new features that we had added and how the related to the requirements, such as the addition of more chefs being related to UR\_Control\_Cooks

However we had to revamp the majority of the UML and sequence diagrams to include the new features of the game, such as the new game mode, powerups, shops and new recipes. Also added separation lines between the object attributes and methods. We also decided to use a different UML Diagram created as well as Sequence Diagram tool as we weren't comfortable with PlantUML which is the tool that the other group used. We decided that using a tool that we knew how to use from the previous exercise would be advantageous.

## List of Changes:

Figure 1:

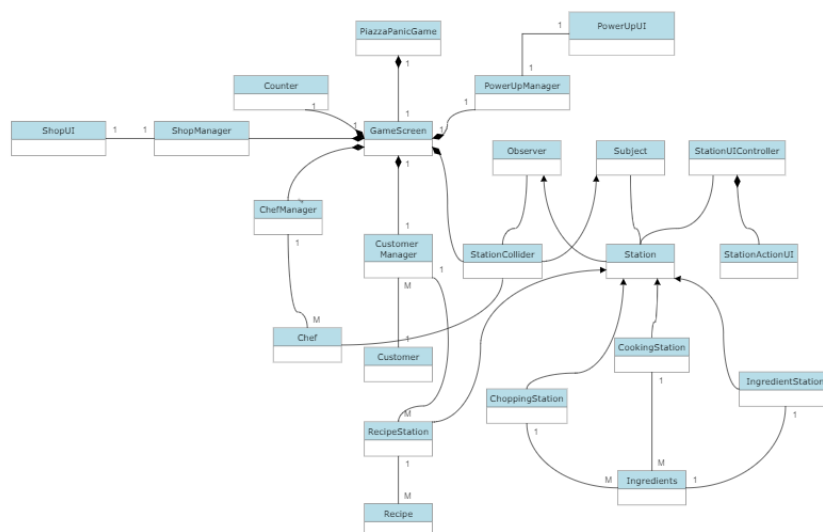


Figure 1 has been an overview of the classes involved within the system architecture, and has been adapted with clearer entity relationships as well as updated to incorporate Powerups, the Shop and a counter. We decided to follow the theme of how Team 32 designed their game, using a Shop and Powerup manager to manage the Power Ups and Shop, which will then have a 1 to 1 relationship with the UIs of both of these. Naturally, these

will be attached to the GameScreen. As well as this, a Counter has been added. This counter refers to the lives and timer which we look at in more depth later

Figure 2:

Figure 2 looks at many of the UI elements, and has been adapted to incorporate ShopManger and PowerUpManger along with the other previous components

Figure 3:

Figure 3 has been left mainly the same, however we decided to add more attributes to the Cooking and Chopping Station.

Figure 4:

Figure 4 was kept completely the same except to add the reputation to the UIOverlay, as we had decided to not touch any of these

Figure 5:

A new Figure 5 was created to display how the PowerUp and Shop Manager would link to the game screen. We decided that using a UIController for each would be the easiest so that the code does not get messy

Figure 6:

Figure 6 is a state diagram for controlling the chefs, and has been extended to show how this would work with additional chefs, obtained through the shop.

Figure 7:

Added a serves state to make the the state diagram more understandable

Figure 8:

New state diagram to show how the game would work in endless mode

Figure 9:

Kept the same - we won't be changing how this works

Figure 10:

Kept the same

Figure 11:

New sequence diagram to show how a salad is created

Figure 12:

New sequence diagram to show how a pizza is created

Figure 13:

New sequence diagram to show how a jacket potato is created

Figure 14:

Kept the same

#### Risk assessment and mitigation

Original:

<https://aaddmn.github.io/new.aaddmn.github.io/Deliverables/Assessment-1-Original/Risk1.pdf>

Updated:

Risk ID	What's been changed?	Why?
R_PRO JECT_01	A group member misunderstands their task and completes the wrong thing	There may be a misunderstanding of the last few weeks of the task. It needs us to complete together. Then we can submit on time. Because we are a team.
R_PRO JECT_02	A member leaves the project	The fact that a member left the project was an accident, but it didn't mean we didn't have to submit a full project, so we needed to get the whole thing done as soon as possible
R_PRO JECT_06	We misjudge the time it takes to complete the project, and do not finish it before the deadline	It is necessary to finish the work within the time limit
R_PRO DUCT_02	Assets that are being used are not allowed to be included in open source projects	The asset creator may not cooperate or cannot be contacted.

### Method Selection and planning

Original:

<https://aaddmn.github.io/new.aaddmn.github.io/Deliverables/Assessment-1-Original/Plan1.pdf>

For the Method Selection and Planning, we looked through the previous teams planning and decided what we needed to update. As for their section A, after reading through the section we realised that this team had used all the same tools that had and were planning to use, and after reading their reasoning of the other group, we ended up agreeing with all of their points for using GitHub, IntelliJ, Discord and Google Drive. The other team did a really good job explaining the benefits of these tools over others, such as VScode, and made points that our team hadn't thought of, such as the much larger community of developers that use it and are ready to help. For these reasons, we decided to leave section A mainly the same, while adjusting some of the points for our group

For Section B, the approach to the team organisation, which changed the majority of this section as we approached the team organisation of this project completely differently to that of team 32. As described in the updated Method Selection and Planning, we did not select a particular hierarchical structure with a project lead and such, as we decided that having everyone check that each other is doing the work well would be more convenient for us as that is the way that we did the first assessment. However, this time we decided rather than have everyone working individually, that we would work on different sections in pairs as this would make it easier to get help when stuck, as someone else would be working on the same section as you and make it easier for them to understand the problem. We decided that this was a better approach then what we did for assessment 1 where we each worked individually because it means that we are forced to be more structured and punctual with what we were working on, as our partner would be helping and need to be informed about what is going on and what had been updated/needed to be done.

Name	Section
Dan	Change Report
Niko	Change Report
Matty	Implementation
Adam	Implementation/Website
Ani	Testing
Dominik	Testing/CI

These were the pairs and sections that we decided on.

For part C, we had to change the whole section to suit our planning. Last assessment we had an issue with the timings of what we were doing so this time we decided to come up with a much more structured plan which can be seen below.

		Start Date	End Date
Change Report	<ul style="list-style-type: none"> <li>• Update Requirements</li> <li>• Update Architecture</li> <li>• Update Risk Assessment</li> <li>• Update Method and Planning</li> </ul>	01/03/2023 08/03/2023 15/03/2023 19/04/2023	07/03/2023 14/03/2023 22/03/2023 21/04/2023
Implementation	<ul style="list-style-type: none"> <li>• Add new Ingredients</li> <li>• Add more Chefs</li> <li>• Add Power Ups</li> <li>• Add Shop</li> <li>• Add Save</li> </ul>	01/03/2023 08/03/2023 08/03/2023 08/03/2023 17/04/2023	07/03/2023 08/03/2023 14/03/2023 20/03/2023 23/04/2023
Testing	<ul style="list-style-type: none"> <li>• Summarise testing methods</li> <li>• Carry out test, provide precise URLs</li> <li>• Report on tests</li> </ul>	08/03/2023 15/03/2023 20/04/2023	14/03/2023 20/04/2023 25/04/2023
CI	<ul style="list-style-type: none"> <li>• Summarise CI methods</li> <li>• Set UP CI infrastructure</li> <li>• Give brief report</li> </ul>	15/03/2023 19/03/2023 20/03/2023	18/03/2023 20/04/2023 25/04/2023
Website	<ul style="list-style-type: none"> <li>• Add all new links to work</li> </ul>	01/05/2023	01/05/2023

As part of our team-forming session, we outlined the work that needed to be done and created a rough timeline of when we hoped to start and complete all of these. The group unanimously decided that we would keep the work during the Easter break to a minimum to give us time off as well as time to revise for other modules, and with this in mind, managed to come up with the plan as seen above.

The beginning focus of our tasks was decided to be the Change report as well as get started on the Implementation. We decided to split into groups to complete these as per Part B, and at the points where Testing and CI was not scheduled to start yet, the group members assigned to these could revise for their own modules and help out when needed

We decided we would just meet weekly at the ENG1 sessions as that was all that would be needed. As a result of the work we had done for Assessment 1, we were more comfortable with the work we had to carry out this time as this is why we decided that we would not need an extra meeting.

By starting with the change report this will make sure we understand and know what information we are going to update within the implementation. Following this, if we discover



any other methods or architecture points that we want/need to add or change, it would be easy to look at what we were planning to do and compare it to what we are trying to do now.

Of course the Implementation and Testing would come next. Putting these 2 together means we can create and implement functions and tests for these functions at the same time. Naturally this means any problems that we encounter can be fixed as we go along

The CI must be done as after/ during the implementation of the game, which is why it is done later

Finally with all sections done, these new documents can be linked to the updated website