Change Report

Group 28

Piazza Panic

By OuseWorks

Ben Harris
Joshua Gill
Niamh Hanratty
Amy Raymond
Matthew Czyzewski
Matt Rohatynskyj

Part A

We used GitHub and gantt charts to keep track of progress with updating documents and code.

- Everytime we would change something in one of the 4 documents, we would make a note of it on GitHub projects to remember to justify why we made those changes in the Change2 document.
- Weekly gantt charts show what documents we updated and when

We made a checklist on Google docs using direct quotes from devCharles' feedback.

- We made a list of changes we needed to make on each document and this also meant we had a list of changes, which would remind us to mention them in the Change2 document.
- Note that this checklist was only made in sprint 5, so isn't a list of all the changes we made, but instead is an example of how we managed our progress.

We saved copies of all devCharles' deliverables. We then edited the copies using the 'comments' and 'suggest edits' mode in google docs. This meant:

- We wouldn't overwrite the old document without the suggested changes being looked over and approved by at least one other person in the team
- For smaller changes (i.e grammatical or formatting changes) individuals still had freedom to edit documents without review, speeding up the process

We continued to use our review and approve system (from Assessment 1), for pull requests submitted to our Github repository by individuals to keep track of changes to code. This was an efficient way to keep track of changes because:

- The team was already familiar with this method
- Details for each pull request were easily stored and accessed in one place
- Automated checks would indicate conflicts and issues with the edited code
- Merging could be delayed until other edits were made
- Individuals could keep track of who was working on what

Part B
Requirements Document

devCharles' Doc
Updated version

We didn't feel the need to change much of the Requirements Part A, which is the introduction.

- This is because we felt like their explanations of the tables were sufficient, and as we didn't make any changes to the actual structure of the tables, this could remain the same.
- devCharles used the same priority rating system that we used during assessment 1 and we feel like their explanation of the different types of requirements is sufficient.
- The only thing we added to Part A was a note clarifying why we took that particular approach to tabulating the requirements and we added a reflection stating how well we think that approach worked.

We also added a section for references at the end of the document as we thought this formatting was better than having links throughout the document. We made sure the requirements we added were in the EARS format that devCharles used.

The biggest changes we made to the tabulated requirements (Part B) are adding the user requirements UR_SCENARIO and UR_ENDLESS. This is so that we can distinguish between which system requirements are needed for the scenario mode and for the fully implemented game. For example, FR_TIMING and FR_TOGGLE_CUSTOMERS are solely for the scenario mode as the brief states that the player shall choose how many customers to serve in scenario mode for assessment 2. Similarly, requirements such as FR_MONEY, FR_MORE_COOKS, FR_PREP_FAIL and FR_UNLOCK_STATIONS are specifically for endless mode as they are extra features that build on the basic game. We made sure to keep some of the base requirements which appear in both gameplay modes the same; these are the functional requirements that use the user requirement, UR_GAME_PLAY.

The new requirements for Assessment 2 are:

- Implement five special power ups that chefs can obtain.
    - Although, at the stage of editing the requirements document we didn't know what our five power ups would be, we added the functional requirement, FR_POWER_UP, so that we could later make it a priority to implement.
- Implement support for different levels of difficulty in the game (e.g. easy, normal, hard).
    - FR_DIFFICULTY
- Implement facilities that allow players to save the state of the game at any point and resume a saved game later.
    - FR_SAVE_GAME

Architecture Document

devCharles' Doc
Updated version

Behavioural Diagrams

- In the feedback, it states that devCharles should have included more behavioural diagrams as well, therefore we have added a new state diagram and sequence diagram to show how the power ups and game save system would work.
- Behavioural diagrams are necessary as they show how the system will work to fulfil the necessary requirements.

Class diagram

- We didn't feel the need to change the explanations and justification behind their choices on the classes that they implemented for the main game but in the updated architecture document, we explain the classes and components that we added to devCharles' existing diagrams.
- We felt like the existing diagrams were sufficient for the scenario mode, however they didn't cover the requirements for the endless mode
    - For example, we had to add sections for power-ups, money, tips, unlocking new stations etc.
- New system components classes such as powerUpsSystem needed to be created from scratch, so these architecture diagrams would help give us an understanding of where we needed to start with the implementation.
- We also updated a lot of the component classes as we needed to account for things like more recipes being added, checking up on the food, pricing, and unlocking power-ups and stations which were all necessary for the requirements for assessment 2

Entity-Component and System-Component diagrams

- We didn't change much as we realised devCharles hadn't included the specifics like the types of food that customers could order, so therefore there was no need to add the new food types.
- The only things we added were the PowerUpComponent and the PowerUpsSystem which let us implement the powerups for requirement FR_POWER_UP.

Class-Responsibility-Collaboration (CRC) Cards

- We didn't change the actual contents of the CRC cards to match the code that was implemented, as CRC cards are meant to be a brainstorming tool to make a note of initial ideas.
- We thought it was still necessary to make some small modifications as CRC cards are supposed to include the responsibilities (purpose for the class) and the collaborators (which other classes that specific class might interact with) which we felt like devCharles had failed to achieve.

- We used devCharles' descriptions of their classes on their original document to aid us with this. There is a direct link to these cards in the document, however devCharles' original cards are still on the website to view.

Tools

- We changed what tools we used as we personally preferred to use PlantUML and used the gizmo google drive extension for the new diagrams as we felt like it was easier coding UML rather than using a UI to do it.
- We used diagrams.net to edit their previous diagrams as this was easier than remaking them from scratch.
- We changed the discussion about the alternatives, as we found using intelliJ ultimate didn't work well within our team, so we used the alternatives discussed above.

Once again, we added references to all tools used or considered in the process of making our architecture.

Method selection and planning Document

devCharles' Doc
Updated version

Part A

- We decided to switch our method to an agile delivery scrum method with Ben being our scrum master and doing weekly stand-ups as this is what we used for assessment 1 and it worked well.
    - We considered using the waterfall method that devCharles used but we didn't tend to do our tasks in isolated phases and instead did our tasks in parallel.
- We used PlantUML as we were already familiar with it and devCharles' has issues with Monday.com
    - Therefore we updated our description on these software, adding a section about PlantUML and deleting the Monday.com section as it wasn't needed.
    - We had good feedback on our weekly plans last assessment so decided to keep the formatting the same.

Part B

We updated the roles to fit our team rather than devCharles.

- We felt the need to change this part as we felt our organisation during assessment 1 worked really well within our group, as we found which members were best suited to which roles.
- Another reason for change in this section is that the roles were very particular to devCharles' individual members and also a lot were assessment 1 specific.

Part C

- We made new weekly plans to follow our progress and made sure to explain the key changes between the sprints in bullet points.
    - We felt like this was a necessary change as we felt devCharles failed to explain how their plan changed over time, which is a crucial part to the project to help with progression.
- We made sure to include the initials of who was doing which task. Where there are no initials, this means everyone was involved in the task.
    - This means that we can make sure everyone in the team has a task to complete at any one given time and makes sure no one member is overloaded with tasks.
- We edited how we utilised GitHub's Projects to help us with our organisation as we used it slightly differently compared to devCharles.
    - We made this change as we felt like trying to adhere to their method wasn't necessary, and instead came up with a method that fit our team, as described in the document.
- Added a link to the new gantt charts and to GitHub projects to make it easier for the examiners to find and once again, we added references to the tools used.

Risk assessment and mitigation Document

devCharles' Doc
Updated version

- We assigned our team members with specific risks that they would be responsible for.
- In response to our feedback in Assessment 1, we made sure to research and discuss how to manage the dynamic aspect of risks by linking some risks to specific dates in our project that they were most relevant to.
    - This would highlight to the owner of each risk when the risk would most likely occur.
    - This is also particularly useful as we then know when we are no longer at risk of that particular problem.

We felt most of the document from devCharles' was similar to our risk assessment and mitigation document from Assessment 1 so we didn't change much more. Components of the document we felt were similar to our Assessment 1 document and worked well for us were:

- Similar severity and likelihood ratings and colour-coding
    - These worked well as they showed us which risks we needed to put more effort into prevent ie. ones with a higher likelihood.
- A similar ID system to identify each risk
    - This system didn't need changing as it was the easiest way to identify the risks to discuss them.
- Presentation of the risks in similar tables

Risks added:
We noticed that some of the risks we came up with in assessment 1 were not mentioned by devCharles, therefore we added them.

- R16 Team members may not turn up to meetings.
    - We felt like this was important to add as it could result in team members not doing their fair share of work if they are constantly not able to turn up to meetings.
    - We are aware that sometimes not everyone will be free at the time of the meeting which is why we put this contingency in place.
- R17 A member of the team finds their role difficult and swaps with another team member.
    - We felt like this was important to add because if one member of the team stops doing their assigned tasks then it could result in one task being done twice, and then falling behind on the neglected task.