ENG1 Team Project
Assessment 2

Deliverable 4: Method Selection and Planning

Team 13 – Team Unlucky

LILLY BROWN
ROSCOE GILLATT
ADAM JOHNSON
YUMIS ZYUTYU
BRANDON WEST
AYMAN ZAHIR

Part A

Software Engineering Methods

Our chosen software engineering method is RAD, this was chosen as it allows us to reduce planning time and emphasise on game creation. In the first part of the method, we met with the customer to finalise and get approval for any new requirements and ideas. As we already had a working prototype, the next steps were easier to complete as only modifications of the prototype were required. The project was then taken into testing and all final changes were made in that stage.

We organised frequent team meetings, on average twice a week. These meetings served two key purposes:

- Providing a platform to discuss and agree upon key development decisions, such as progress onto fixing any existing documentation or the implementation of any new requirements or existing issues.
- Allowing each team member to communicate their progress on their assigned tasks, and for new tasks to be assigned when necessary

The frequent team meetings in which we discussed task progression, allowed us to keep each-other accountable for our productivity.

Tools Used

Communication and Collaboration

- Discord - we created a server for general communications throughout the project. This was crucial in allowing team members to ask clarifying questions without having to wait for the next meeting, avoiding any unnecessary obstructions to task progression. The discord was split into different servers allowing the team members to find the correct place to communicate the correct issues, as well as allowing multiple conversations to be held at once.
- GitHub projects - used to organise and keep track of our tasks and issues from the previous assessment throughout the project. Alongside the team meetings, this was crucial in keeping us organised by providing a visual representation of tasks that were completed, in progress and yet to begin as well as keeping people accountable for the incompletion of certain tasks and allowing the possibility of a proper quality control measure.
- Google Drive - used for keeping all the documentation and important information in one place accessible at all times to all team members. The most useful features were the comment feature which made it very easy during the quality control measures and the version control, which allowed us to have only one master file whilst being able to restore any previous versions and see who was done what and at what time.
- GitHub version control - used for version control for the code as each file has a history showing the changes done and the time they were done at. Also, it allowed for a quality check as you can ask other members of the team to review your work.
- OfficeTimeline was used to keep track of the overall progress on the project, and to create weekly snapshots and a final Gantt chart. OfficeTimeline allowed the allocation of team members to tasks and a visual representation of the completeness and priorities of tasks.

Website

- We used GitHub pages to host our website, and HTML/CSS to develop it. This allowed for a straightforward implementation of the website as it did not involve external hosting or server space outside of Github. Using Github pages also accommodates the functionality we want in an easy-to-manage environment containing all of our documentation/code already.

Architecture

- PlantUML in addition to adobe photoshop was used in order to create the concrete and abstract architecture diagrams;
    - PlantUML was used at first in order to create representations of classes, categories of classes and the relationships within each category
    - Adobe photoshop was later used in order to add the inter-category relationships

Implementation

- We chose to use IntelliJ as our IDE. This was due to both the ease of use offered by the tool, along with the fact that several team members had previous experience with the tool: it felt like the ideal choice to avoid unnecessary and time-consuming learning curves.

- We utilised the libGDX game development framework during the implementation of our game. Similarly, our choice of IDE was largely influenced by the previous experience of the team. We were also aware that LibGDX was a popular choice amongst other teams and therefore our use of this framework may make it easier for another team to take over and expand our code.

- We used GitHub Actions as a continuous integration tool, as it allowed us to automate the code, test and deployment. That made the workflow easier and more manageable as the team didn't have to worry about following

Testing

- Junit - the base for the unit tests. This is the library where all the basic test building blocks (like assertEquals) come from.
- LibGDXRunner - used to allow for unit tests on LibGDX programs beyond the basic mathematical functions of Junit.
- Mockito - used to "mock" various parts of the game to allow it to be run in isolation. Junit and LibGDXRunner don't necessarily handle graphical components well so this is there to handle that without refactoring the code. An alternative to this was refactoring the codebase, but due to the flexible development of new features, this proved tough to implement.
- Intellij - All the tests were run in IntelliJ, which has built-in test processing support which gives a pass/fail grade to every test, along with statistics such as code coverage.
- Microsoft Excel and Google Spreadsheet - used for manual tests, tests and results were recorded on a .xlsx file which was started in Microsoft Excel but was later edited mainly in Google Drive spreadsheet tools to keep consistency with the rest of the documentation.

Alternatives considered

- We considered using other IDEs such as Eclipse for software implementation, however, after researching we came to the conclusion that eclipse does not support easy management of Gradle projects and thus we decided to continue using IntelliJ as it has the needed support.
- TeamCity was considered as an alternative to GitHub actions. However, due to the team's unfamiliarity with the software its use would have required more research, which would have cost more time.

Part B

Team Roles

The team was split into three subteams - Documentation, Implementation and Testing split the following way:

- Implementation: Ayman and Roscoe
- Testing: Adam and Lilly
- Documentation: Brandon and Yumis

The team members within each subteam were responsible for both keeping track of the progress and changes, as well as recording any documentation. That made it easier to keep everything organised and nothing was lost in translation; this enabled a smooth and efficient team working process.

At every team meeting, a team member was assigned the task to do the meeting log book.

Yumis was in charge of keeping the Gantt chart up to date and taking weekly snapshots of it, this was then passed on to a specific server that had only the snapshots in it for an easy find.

Task Assignments

Task assignments took place within the individual subteams throughout the whole duration of the assessment process. When the time came to assign new tasks, we tried to keep these assignments in line with each person's particular skill set in order to ensure efficiency as such throughout the project members mainly focussed on the particular aspects which they felt most comfortable with, usually due to either previous experience or as that's what they worked on on the first part of the assessment. We went ahead and allocated tasks during the first meeting as we were already familiar with the skill set of the team due to the first part of the Assessment. Tasks were assigned with a particular workload allocation, set by the assessment lead - which was roughly 15 marks per person. Those were carefully chosen based on the already familiarisation of the team member with the set part of the assessment as they have already worked on it from the previous assessment.

Quality Control

Quality control measures to ensure all the tasks were completed according to the needed standard was set in place. That was split into two parts:

- The first part was the use of GitHub's projects and issues. We created a kanban board on GitHub, including two extra columns there just for quality control - "For review" and "Review approved" that was there to ensure that every issue/document would undergo a review from another team member that has some experience within the area the issues were in. Once the issue was assigned to the reviewer the second part was in action.
- The second part was the reviewer reviewing the issues/document by either manually correcting it or leaving comments on the google document indicating what needs correcting. That assured all the work was checked by two people and nothing was overlooked.

Part C

Plan - Gantt Charts

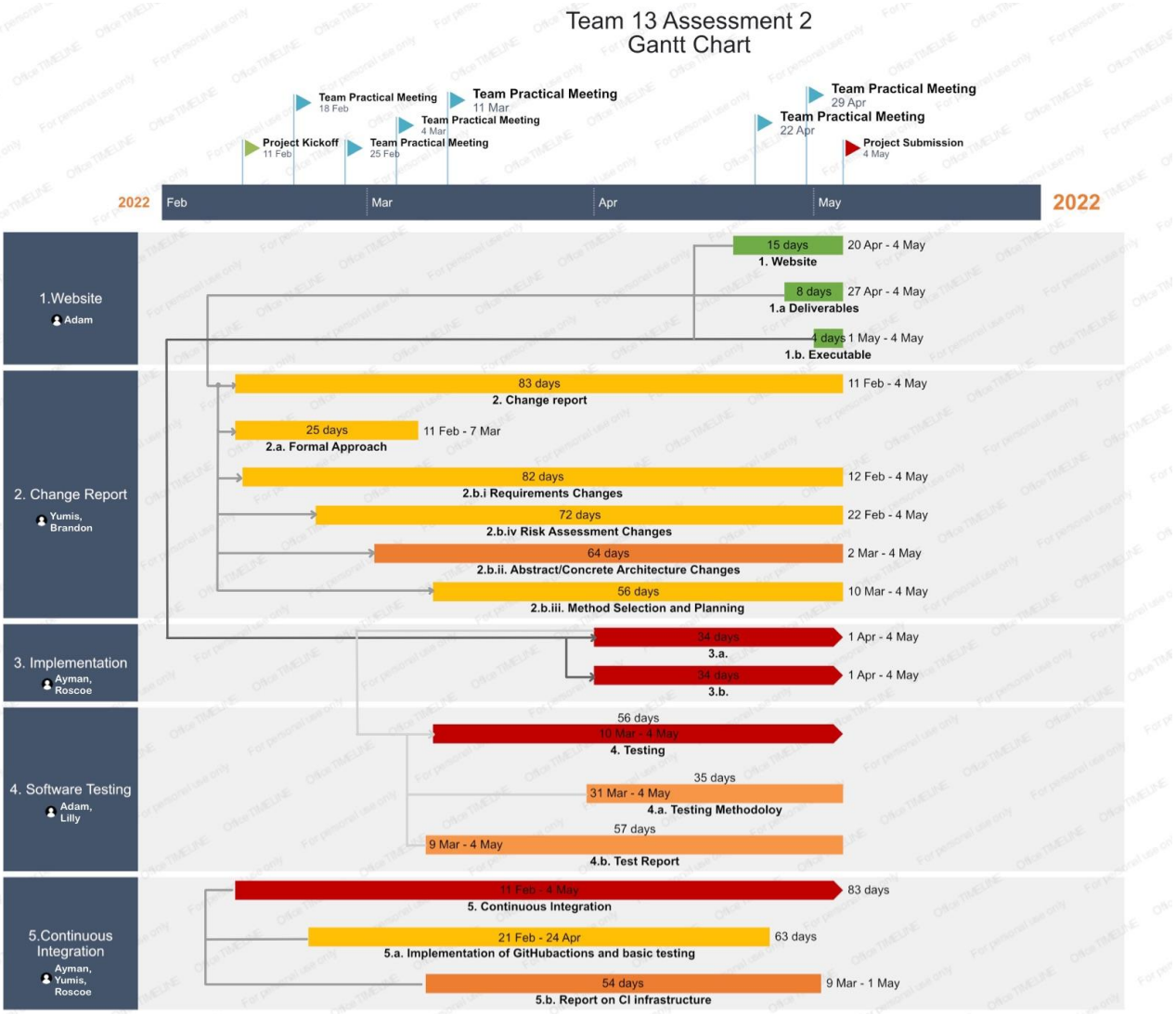

Figure 1.0 Initial Plan Overview

A standardised colour coding prioritising system was set in a place where higher priority tasks were indicated with the colour red, less urgent tasks were in the colour orange, and yellow and no priority was colour green. The priority of the task was determined by how important the task is for the completion of the project. The team kept to the plan expectations set at the beginning of the assessment. Team 3 used this table in order to create an initial Gantt chart to show a theoretical schedule for when each task would be completed for assessment 1(as shown in Figure 1.0)

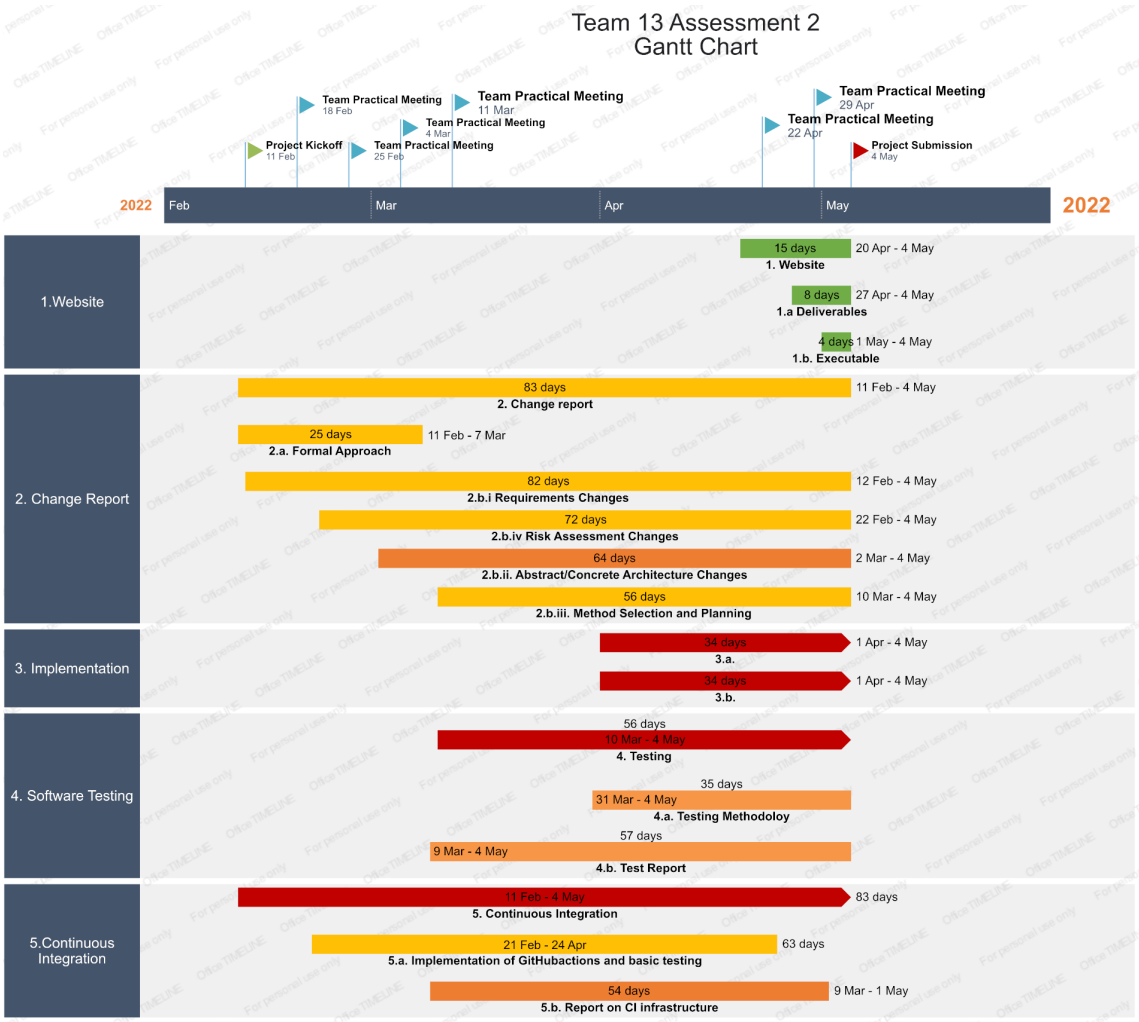

Figure 2.0 Final Gantt Chart

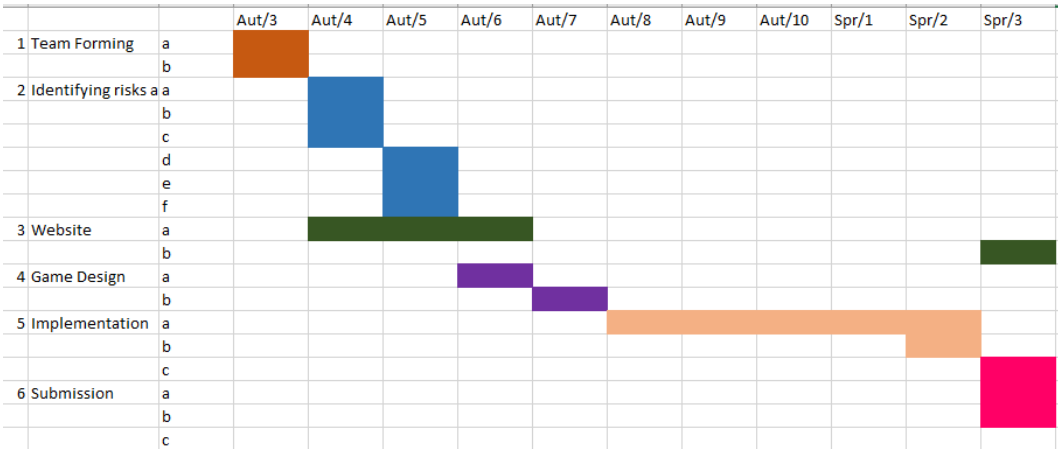

Figure 3.0 Team 3's Plan Overview

At each team meeting team 3's assigned secretary, Jarred, would create meeting notes which he would then upload to our shared google drive. At the end of each week, they then used these notes in order to create a 'snapshot' of their team's progress (shown on the website snapshot page). These snap-shots would be created by taking a condensed version of the task breakdown table and colour coding the tasks which were due to be done at this point in the project. With the following colour connotations: Red – Not started; Orange – In progress; Green – Completed