Method Selection & Planning

Cohort 1, Group 6 - M6

Members:

Mir Baydemir
Adam Fraulo
Azib Hj Abu Akmar
Agata Pittarello
Esther Scanlon
Jazz Stubbins
Sonia Szetela

Method Selection:

As a team, we have adopted the Agile Methodology due to its focus on the client, keeping them at the centre of development. The Agile Methodology allows us to immerse the client in the development process and enables us to respond dynamically to changes in their requirements through iterative development. Iterative development means we develop the software in small stages, with each iteration improving on the previous one. In contrast, a fixed approach involves a single, inflexible plan. As students, we are inexperienced; it is unrealistic for us to plan every stage of our project before we begin. Consequently, an agile approach benefits us, as we can correct our mistakes in real time without having to restructure an entire plan. Furthermore, using an agile approach also allows us to deliver software quickly to our clients, ensuring that we meet their requirements and enable them to see their product frequently. Moreover, using an agile approach allows us as a team to break the project down into smaller, more manageable tasks, helping us stay organised and Structured.

For Assessment 2, we found the Agile methodology particularly useful when working with the inherited code. The iterative approach helped us to understand the existing code, make changes, test them and get feedback before proceeding to the next iteration. We found this essential as a team because we were working with a code we did not originally write.

Since Agile is our overall philosophy, we will use the Extreme Programming (XP) and Scrum frameworks during the development process. We chose to use XP as our main development framework because it provides practical engineering practices such as pair programming, test-driven development, and continuous integration, to ensure that our game is reliable and adaptable. Additionally, XP enables us to maintain a simple implementation, with a focus on frequent releases and iterations. It is much more efficient to work on one feature at a time and have another team member test these while others are being developed. XP promotes these practices and prevents us from developing the entire project before we discover issues with the implementation.

[1] For assessment 2, We adopted some of the XP practices that was particularly useful when working with the inherited code:
Pair Programming: when modifying poorly documented or complex parts of the inherited code, we used pair programming to share knowledge and reduce the chance of getting bugs.
Test-Driven Development: Where possible, we wrote tests before modifying inherited code. This helped us to understand the current behavior of the inherited code before making changes. This decreases risk of breaking functionality.
Refactoring: We spend high amounts of time refactoring the inherited code to improve maintainability and testability. We also made sure that each refactoring was covered by tests.

Using Scrum as our project management framework provides a flexible way to apply agile principles. We can organise our work into sprints, keeping the project manageable as team members can focus on one small feature at a time. Additionally, scrums enable better team collaboration, where everyone knows what they're working on and how their tasks fit into the sprint goal.

Tools

Documentation

For our documentation, we chose to use Google Docs because it is free to use and allows for real-time collaboration, enabling multiple team members to work on the same document simultaneously. Furthermore, with all files stored on the cloud, team members can access documentation anytime, anywhere, and from any device. Additionally, Google Docs automatically saves all changes, eliminating the risk of losing work. Likewise, the version history feature enables us to easily track edits and restore previous versions if needed. Microsoft Word was considered as an alternative way of documentation due to its ease of use and familiarity with all team members. However, we chose Google Docs instead due to its real-time collaboration capabilities, which are essential for an Agile team, as we can see updates instantly and leave comments for discussion. Also, the auto-save feature is crucial for preventing the risk of losing work, which would waste time and reduce productivity.

Communication

As a team, we chose WhatsApp as our primary communication method because all group members were already familiar with the app and had it installed. Furthermore, using a mobile app enabled us to communicate quickly, reducing delays compared to alternative methods such as email. The app also works on tablets and computers, allowing everyone to stay connected wherever they are.

Version Control

We chose GitHub as our version control system because it supports code collaboration through the use of branches. These branches allow multiple people to work on the same project simultaneously, which is essential for our agile approach. Branches will enable us to do this without accidentally overwriting someone else's work. Changes can then be merged smoothly using pull requests. In addition, GitHub maintains a full history of changes to code and documentation, allowing us to review and restore previous versions if necessary. Another version control system considered was Apache Subversion since it is easy to understand and widely used. However, it encourages large commits, which contradicts our development framework of smaller iterative work. Furthermore, branching and merging are more error-prone compared to GitHub, where managing branches is easy. Because we are using an Agile methodology, frequent iterative development is essential; therefore, using Apache Subversion would not be the best approach.

Testing Tools

For assessment 2, we used JUnit 5 as our primary testing framework for automated unit tests. We used Mockito minimally. It is only used for mocking LibGDX’s GL20 context through our custom GdxTestRunner class. For components with complex rendering we choose manual tests over automated testing.

Organisational Approach

At the start of the project, we listed everyone's strengths so we knew who could do what best and could assign tasks accordingly. We decided this was the best approach because it builds teamwork and understanding, reduces confusion later, and helps the project flow more smoothly. Furthermore, it also highlighted gaps in our skills, which we can develop later on by taking on tasks that weren't our initial strengths.

Every week, we meet in person to review our progress on the project and set deadlines for the upcoming week. A Gantt chart is then created for that week, so we can visually see the tasks to be completed. This approach allows all team members to see what needs to be done and by when. Improving team coordination and making it easier to track our progress. Furthermore, it keeps the team accountable and helps us identify when a team member is struggling to meet deadlines, so we can take appropriate steps to address it. We assign multiple people to a task to increase the speed and efficiency with which we complete our set deadlines. Moreover, it improves accuracy and quality by having multiple people review the same work, ensuring our project is of the highest quality. During our meetings, we also use that time to make decisions about our project as a group. We chose this approach over a virtual one because we found it easier to communicate our ideas face-to-face than over a call. In addition, we can ask questions, clarify points, and reach agreements more quickly than during an online call.

All weekly Gantt Charts were included in the website page under the name “Gannt Charts.pdf” with URL: “ https://keybordkat.github.io/websiteGroup6/Gantt%20Charts.pdf ”.

Project Plan:

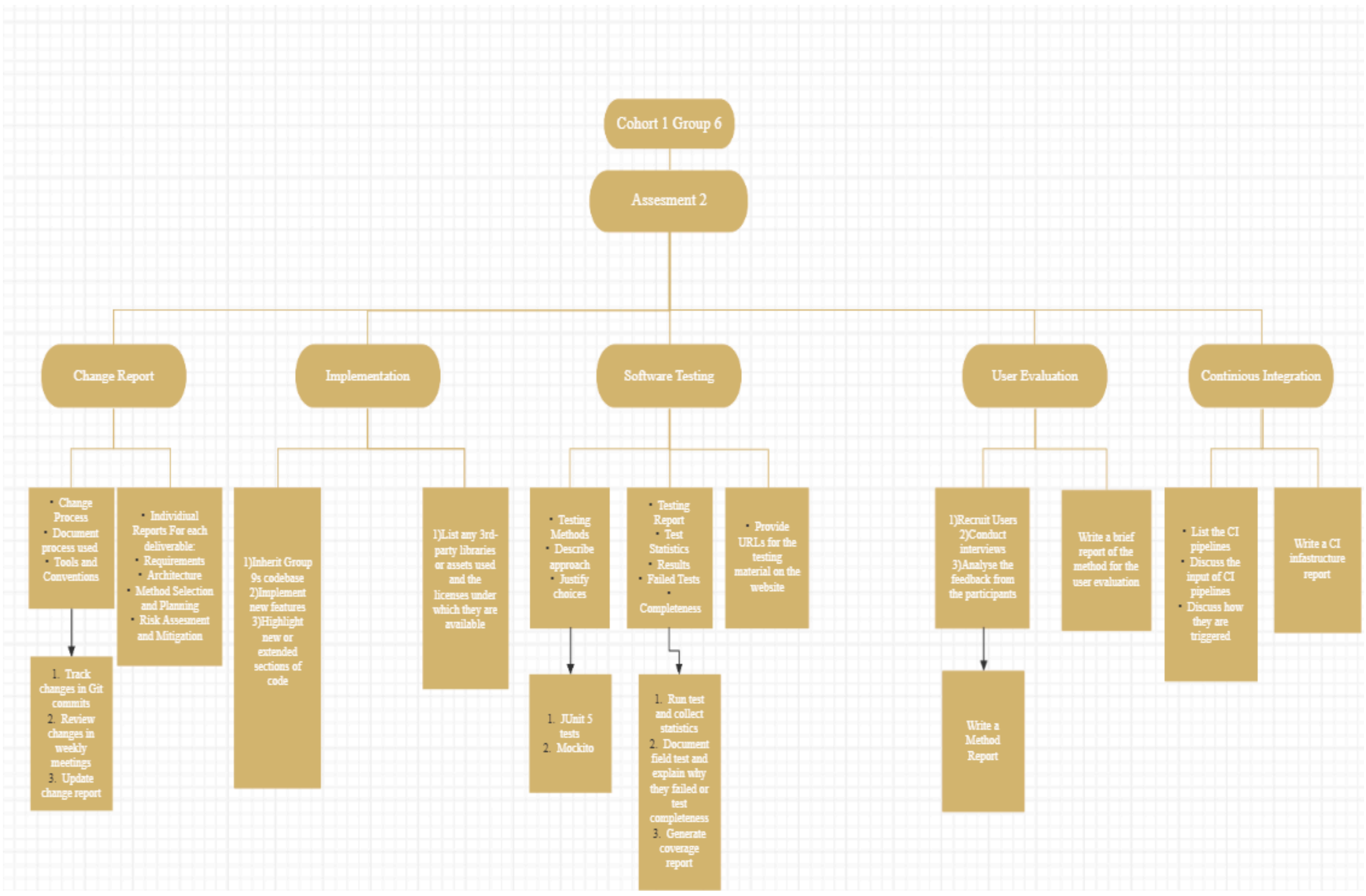
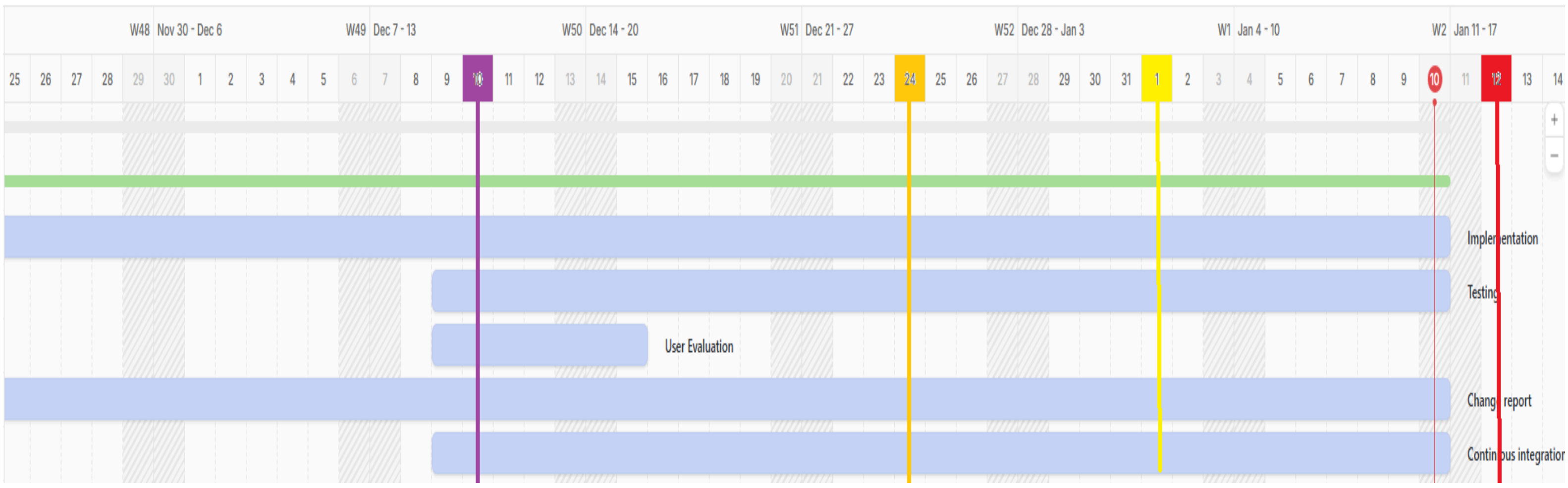

Cohort 1 Group 6
Assesment 2
Change Report
Implementation
Software Testing
User Evaluation
Continious Integration
• Change Process • Document process used • Tools and Conventions
1. Track changes in Git commits 2. Review changes in weekly meetings 3. Update change report
• Individiual Reports For each deliverable: • Requirements • Architecture • Method Selection and Planning • Risk Assesment and Mitigation
1)Inherit Group 9s codebase 2)Implement new features 3)Highlight new or extended sections of code
1)List any 3rd-party libraries or assets used and the licenses under which they are available
• Testing Methods • Describe approach • Justify choices
1. JUnit 5 tests 2. Mockito
• Testing Report • Test Statistics • Results • Failed Tests • Completeness
1. Run test and collect statistics 2. Document field test and explain why they failed or test completeness 3. Generate coverage report
• Provide URLs for the testing material on the website
1)Recruit Users 2)Conduct interviews 3).Analyse the feedback from the participants
Write a Method Report
Write a brief report of the method for the user evaluation
• List the CI pipelines • Discuss the input of CI pipelines • Discuss how they are triggered
Write a CI infastructure report

Diagram Showing Which Team Member(s) are responsible for which deliverables

	Website	Change report	Implementation	Testing	User evalutation	Continuous integration
Sonia	🟦	🟪	🟦	🟩	🟪	🟪
Adam	🟦	🟪	🟩	🟪	🟪	🟪
Agata	🟦	🟩	🟪	🟪	🟪	🟦
Azib	🟦	🟪	🟪	🟪	🟪	🟩
Esther	🟦	🟩	🟪	🟪	🟪	🟪
Jazz	🟦	🟪	🟩	🟩	🟪	🟪
Mir	🟦	🟦	🟪	🟪	🟩	🟪

Key:
Green = Main Person(s)
Blue = Moderate involvement
Purple = overview

Gantt Chart Overall plan



Key:
The purple line shows the time (10 December) where we planned to finish implementation.

The orange line (24 December) shows our adjusted deadline after realizing implementation required more time.

Yellow line shows when we planned to submit (1 January)

Red line at the end shows the submission deadline (12 January)

During Week 7, we analyzed the multiple teams implementations and documentations. Then we held a team vote to select which team's project to work on.

During Week 8, We splitted the work according to team member's strengths. Then we reviewed the inherited code and the documentation.Through the review process, we identified improvement opportunities. Based on our findings, we created an initial plan.

During Week 9, we have updated risk assessment to identify new risks specific to the Brownfield development. Furthermore we added new requirements that were specified in the assessment 2 brief. We also updated method selection and planning documentation. In addition, we also started implementation this week.

During Week 10, We continued implementing new features for Assessment 2 . We also started testing our code to make sure that new features were working correctly. In addition, we started making changes in architecture as the requirements document was completed and new features were implemented. Furthermore, We updated the change report as we made changes in Architecture, Risk Assessment, Method Selection and Planning and Requirements.

During Week 11,We conducted interviews for user evaluation and we have gathered valuable feedback from users about usability and functionality of our game.Based on the user feedback we received , we made some changes in our code to improve usability and address issues identified during the interviews. Additionally, we started designing the CI/CD pipelines.

During the Christmas break, we continued our implementation work and made the final changes to the codebase. We also found that the game implementation was not completed by the time we initially stated; however, we still had plenty of time before the submission deadline. We have also finalised the change report during the Christmas break as we made some small changes in documentation. In addition, We finalized the CI/CD pipeline implementation. Any remaining bugs identified during our testing phase were also fixed.

During Week 12, We prepared our presentation materials for the assessed presentation.

References:

[1]Sommerville,I. (2015). Software Engineering (tenth edition), Pearson Education, 2015-08-20, 72-101, 639-753 .