Method Selection and Planning

Team 8: Ben and Bensons

B&B

CODE. CREATE. CONNECT.

Team Members: Alyssa Skipper, Benjamin Senior, Benson Chow, Chloe Ward, Florian Mengkris, Hannah Thomas, James Ingram, Olivia Spencer

Software Engineering and Collaboration Tools

Software Engineering Methods

We decided to use an agile development methodology [1] for this project because it is a flexible methodology that promotes discussion and team collaboration, as well as helping us deliver the best possible version of the project, and continuing feedback in alignment with the clients needs.

From this, we decided to use Scrum [2] as it is ideal for a small team. Scrum splits the project into ‘sprints’ of work, where product necessities are produced by the end of each sprint. This is helpful for our project because it allows for multiple iterations of the product and gives us flexibility to deal with any problem that may arise. Breaking down the project into weekly sprints also helps streamline the process as everyone should know what they are expected to deliver each week. This is also helpful as it allows us to reflect on and evaluate what work has been completed over the last sprint and by who, which will allow us to keep work distribution even and be aware of any issues that a team member may have.

We have weekly meetings where we will report what work has been completed and what everyone will be working on next week. Any issues will also be discussed in these meetings and strategies will be implemented to resolve them as soon as possible. Furthermore, these meetings will ensure that each phase of the project is progressing at an appropriate rate and will allow us to alter tasks that are taking longer than expected. Meeting agendas and minutes will be recorded so we will be able to remember when has been discussed and any actionable points that we would like to be completed by the following week.

We considered other agile methods, such as Kanban [3] and Extreme Programming (XP) [4]. We decided that Scrum was ideal because we have fixed, short timeframes that lend themselves better to sprints than a continuous flow of work (as Kanban helps manage) [5]. Scrum was preferred as the team has more flexibility over the sequence that the product will be developed in than XP, where a strict priority must be followed. [6] XP also requires that all software be developed via pair programming, which is not feasible for this project.

Development and Collaboration Tools

We used both WhatsApp and Discord to communicate. We tended to use Discord for formal and structured communication about specific parts of the project. Discord was ideal for this purpose as the ability to create channels allowed us to split discussions. WhatsApp was used for quick, general communication, such as asking where a meeting will be held that week. WhatsApp was used for this purpose over the general channel on Discord because it was more natural for members of the team and we found that people responded quicker. We also considered using Slack as it also had the channel functionality, however we decided against this as everyone in the team already had Discord.

We used Google Docs and Google Sheets to write up the deliverables. We chose these because multiple people can access and edit the documents simultaneously and the editing history is easily accessible. This allows us to track how much each person is contributing so we can keep the work divided evenly. We discussed using Microsoft Word and Excel as some members of the team were more familiar with these tools and they are very similar, however Microsoft 365 was costly in comparison to Google Workspace (which is free) and the university also provided Google tools so we decided to utilise that. To store the deliverables and meeting record documents, we used Google Drive as it is included in Google Workspace so we could all easily access this space to find any documents. To create the Gantt Chart, Work Breakdown Diagram, and Architecture Diagrams we utilised PlantUML. PlantUML allowed us to quickly and easily create each diagram and insert it directly into the Google Doc via the PlantUML Extension.

We developed the code using Java 17, as per the assessment guidelines. The framework we used for the project was LibGDX. We considered using other engines, such as JMonkey and Unity, but we decided on LibGDX as it is open source and free. LibGDX is also well tested, java based, and ideal for fast, iterative coding as we are doing. The team used VSCode. Other IDEs, such as IntelliJ and Eclipse, were considered but we selected VSCode because the team was more familiar with it (as we used it last year) and different operating systems can be used- as some members use Windows OS while others use Linux. VSCode was also ideal as it has a range of extensions, such as a debugger and gradle, both of which are highly useful for software development. Furthermore, VSCode has Git integration so the coders can easily commit, pull and merge the changes made.

We used GitHub as our code version control system since multiple team members were familiar with its features. GitHub allowed multiple members of the team to upload their code, despite different operating systems, as well as allowing us to create multiple branches of the code so if different members of the team were coding different elements, they would not conflict with each other once pushed to GitHub. Each feature had its own branch to minimise conflict and then were merged via pull requests.

We used Tiled [7] to design the map, as it was recommended by LibGDX [8]. This was also seen as easier than creating a map fully from scratch and the fact that it is already integrated with LibGDX was appreciated. To make the sprites and tileset for the map, Procreate was used. This was chosen as Alyssa, who made the assets, was already familiar with the software and it is a well respected and versatile digital art software.

These tools directly supported our Scrum workflow: Discord and Google Docs enabled continuous communication and documentation; GitHub and VSCode provided rapid and version control for code and PlantUML, Tiled and Procreate integrated naturally into our design. Their open-source and cross platform compatibility made them ideal for us as students using mixed operating systems.

Team Organisation

As previously stated, we primarily organised using the Scrum framework as it promotes collaboration and openness. We decided to split the Developer role slightly to better define what each member is working on. The roles we have sorted the team into are below [9]:

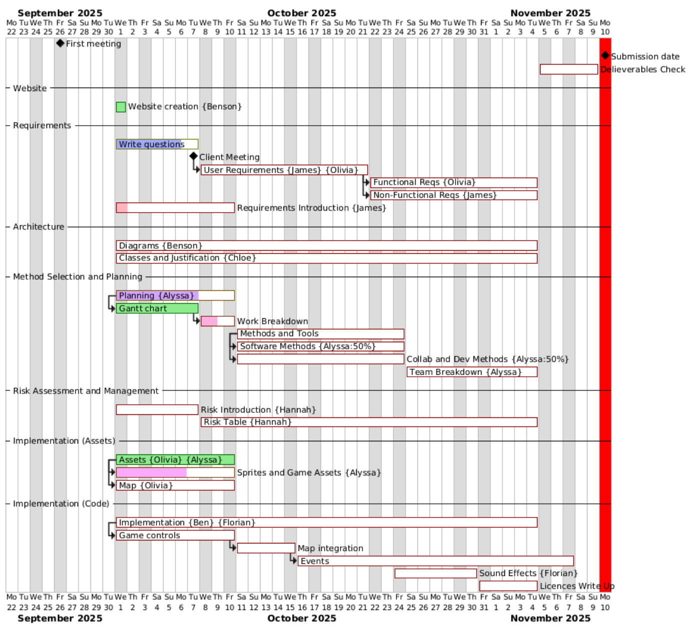
- Product Owner: James. The Product Owner is responsible for ensuring that the product backlog is ordered and well stocked. This is done by organising sprints with the development team as well as collecting feedback from the client.
- Scrum Master: Alyssa. The Scrum Master is responsible for ensuring that the team is performing at a high level. This is done by facilitating meetings and scrum events, and keeping the team focused and productive.
- Code Developers: Ben, Benson, Florian, Chloe. The main role of the developers is to contribute to the sprint goals, while managing the sprint backlog. Code developers mostly focus on creating the code and the architecture document.
- General Developers: Hannah, Olivia. The main role of the developers is to contribute to the sprint goals, while managing the sprint backlog. General developers did not contribute code, instead focusing on the written documentation.

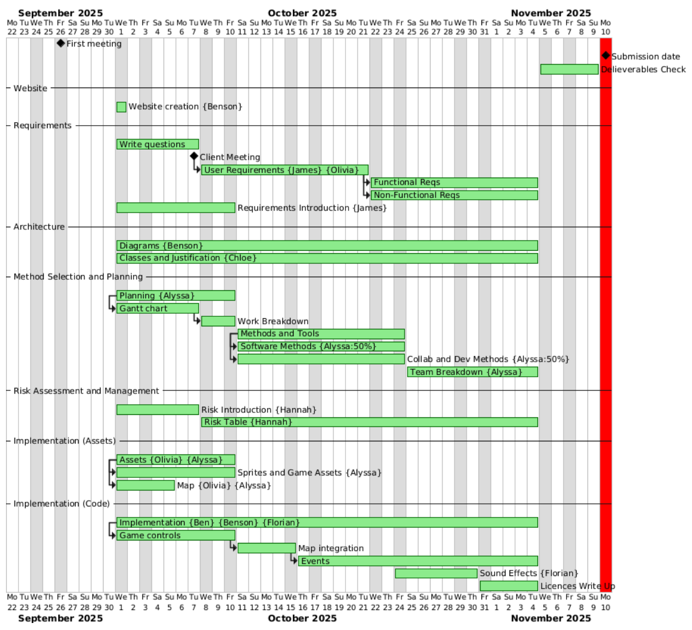
We decided to split each deliverable into teams that would focus primarily on that aspect of the project. This decision was made to ensure that the marks are split evenly between each of us and helped us ensure that every task was receiving equal attention. The teams also meant that risks were mitigated as we had a low bus factor. This approach reflects the Scrum principles of self-organisation and cross-functional teams, while allowing us to tailor roles to the specific skill sets of our members.

Deliverable	**Main Contributor**
Website	Benson Chow
Requirements	James Ingram , Olivia Spencer
Architecture	Chloe Ward, Benson Chow
Method selection and planning	Alyssa Skipper
Risk assessment and mitigation	Hannah Thomas
Implementation	Florian Mengkris , Benjamin Senior

Eventually, we decided to merge the architecture and implementation teams as it was decided that it would be simpler for each person to focus on specific classes- making the diagram and then implementing it themselves, to streamline the process of creating code.
We had two meetings each week, one on Wednesday and one on Friday. On Wednesday we discussed any updates to make sure that everything was proceeding as planned and made a plan for the upcoming week. In the Friday meeting, we discussed how the tasks set on Wednesday were coming along and made any necessary adjustments. Our Wednesday meeting had meeting minutes made by the Scrum Master and minutes were taken so we could look back on what was discussed. Friday meetings were more informal, using the tasks set in the Wednesday meeting as the discussion agenda. Our client (Tommy) was also supervising the Friday meetings, as it was our scheduled time. This meant that we had access to the client so we could ask any questions we had over the week, ensuring that the client is fully involved throughout the project.

Systematic Plan

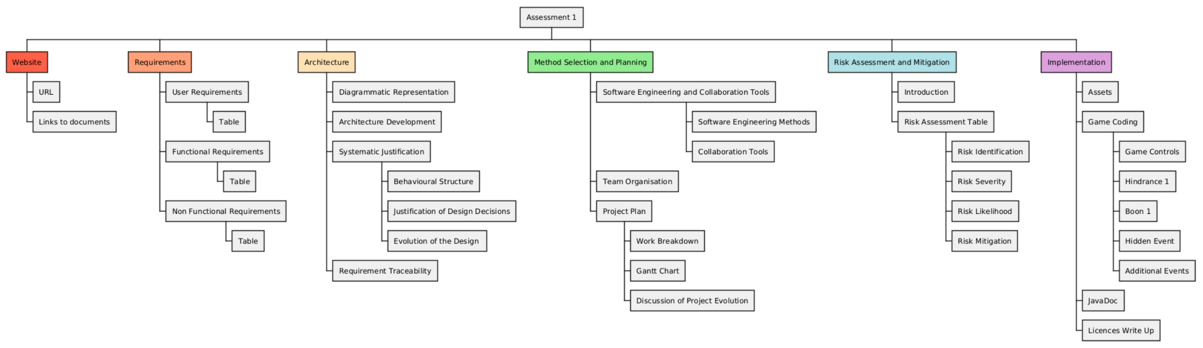




Above is the Gantt chart we used throughout the project- the left is the original chart made in the first week (on 03/10) and the right is the completed chart with various changes made during product development. The weekly charts were uploaded to our website (and can be seen here).

The Gantt charts link closely to our defined SCRUM sprints. Week 1 was focused mainly on creating the websites, Github and deliverable documents, as well as defining the plan. Week 2 was focused on finishing the planning, creating the game assets, and beginning requirements (as the client meeting was this week). Week 3 and 4 called us to continue on with planning, assets and requirements, while beginning risk assessment and implementation of the game. Week 5 was focused mainly on implementation, as many of the other tasks had been completed by this time. Week 6 was the last week, so the tasks were for everyone to complete their deliverables, and check over other documents to ensure everything was completed to the highest expectations.

Some parts of the project were dependent on other tasks being completed prior. For example, the map was required before large sections of code could be worked on (e.g. wall collisions, triggers for the events). The requirements section was also reliant on other sections of requirements being done first, as user requirements could not be done before the client meeting and the functional and non-functional requirements section necessitated that the user requirements section be completed.



Above is the Work Breakdown Structure (WBS), made to give each team a clear idea of what they are expected to do during the project. The diagram breaks down tasks into manageable chunks and was helpful for assigning tasks within the smaller teams. This WBS was deliverable-based (rather than phase-based) [10]. We chose deliverable-based as the end product was very clearly defined.

Evolution of the Project

Over the course of development, various things changed about our project. Some notable instances were when we merged the architecture and implementation teams and when we changed the people creating the assets, from a group to a single person. There also was a lot of collaboration between the previously defined teams. For instance, many members of the team contributed elements to the code.

We had a few problems with members of the team not delivering in the expected time frame, but this was mitigated by other team members helping them with their work. Other problems included group focus on certain parts of the project, which led to some neglect of other project parts. We were able to fix this issue by going through exactly what has been done per deliverable and setting tasks for each team member in our weekly meetings.

We believe that the SCRUM methodology was helpful as the sprint structure allowed us to create each deliverable quickly and to the best of our ability. The sprints were defined above. The reflection aspects of the frequent meetings also ensured that every member of the team was fully involved and aware of what was going on at all times. We found that the concept of a project backlog was very helpful, as it allowed us to prioritise tasks and know what would be worked on next. It also prevented us from neglecting any parts of the project. The fact that the backlog could be updated was helpful as some tasks took longer than expected.

As we went on, we had more serious discussions in our Friday meetings (which were originally casual) and did tasks such as filling in Google Forms with our worries for the project, and changing plans that had been set in the Wednesday meetings.

We have various ideas for how the project could be developed further. We suggested that there could be a hidden event where the player could get back on the bus and go to a bus stop closer to Central Hall. Other suggestions included a crowd of freshers re-routing the character, and including doors that needed to be unlocked via keycard. If we had more time we would've developed a previous part of the map to be Campus East that would be accessible to the player via the bus.

Conclusion

In conclusion, using the SCRUM methodology and various Gantt Charts we were able to complete the project efficiently in the allocated time frame. We also found the tools used to be satisfactory as they were simple to navigate and utilise to bring this game to life. The team organisation changed throughout the project but was overall helpful for keeping us on track. It allowed us to respond to flexibility to challenges and maintain consistency with client needs within the constrained timeframe.

References

[1] K. Beck *et al.*, “Manifesto for agile software development,” *Agile Manifesto*, 2001. https://agilemanifesto.org/
[2]Scrum.org, “What Is Scrum?,” *Scrum.org*, 2020. https://www.scrum.org/resources/what-scrum-module
[3] K. McDonald, “What is Kanban?,” *Agile Alliance | Promoting a more effective, humane, and sustainable way of working*, Apr. 04, 2017. https://agilealliance.org/glossary/kanban/?_gl=1 (accessed Oct. 13, 2025).
[4] Agile Alliance, “What is Extreme Programming (XP)? | Agile Alliance,” *www.agilealliance.org*, Jun. 14, 2017. https://agilealliance.org/glossary/xp/
[5] GeeksforGeeks, “Kanban vs. Scrum : Top Differences You Should Know,” *GeeksforGeeks*, Sep. 26, 2020. https://www.geeksforgeeks.org/blogs/difference-between-scrum-and-kanban/
[6] GeeksforGeeks, “Difference between Scrum and XP,” *GeeksforGeeks*, Jun. 10, 2019. https://www.geeksforgeeks.org/software-engineering/difference-between-scrum-and-xp/
[7]“Tiled,” *Tiled*. https://www.mapeditor.org/
[8] “Tile maps,” *libGDX*, 2024. https://libgdx.com/wiki/graphics/2d/tile-maps
[9] Scrum Alliance, “The Scrum Team Roles and Accountabilities,” *resources.scrumalliance.org*. https://resources.scrumalliance.org/Article/scrum-team
[10] Work Breakdown Structure, “What Is a Work Breakdown Structure,” *Workbreakdownstructure.com*, 2025. https://www.workbreakdownstructure.com/