4 Method selection and planning

4.1 Software engineering method

For the project we are going to use an agile software engineering method called SCRUM.

SCRUM technique and adaptations

SCRUM breaks development of a project into sprints which run for a set amount of time, usually 2-4 weeks. SCRUM uses a product backlog which is a set of user stories with a priority, based on requirements which have yet to be implemented into the software. Before each sprint the product backlog is updated to reflect changes in the project. Then the team has a planning meeting to discuss how many features they think can be implemented and tested that sprint and move the decided on number of user stories from the product backlog into the sprint backlog. During the sprint the team has short, should be no longer than 15 minutes, daily SCRUM meetings where team members share the previous days work, what they will work on that day and identify and problems they have come across. If a team member has finished their current task they will select a new functionality from the sprint backlog to implement based on their expertise. At the end of each sprint the project should be in a deliverable form, at which point the team conducts a sprint review. A sprint review is where the team demonstrates any new features added during that sprint to stakeholders and discuss whether the requirements have been met and receive feedback on the features. Finally the team has a meeting called a sprint retrospective to reflect on the sprint and discuss any possible improvements for future sprints. The process then continues to iterate through sprints until the stakeholders are satisfied that the software meets the requirements.

One adaptation we have made to the SCRUM technique is our sprints will be 1 week long. We have chosen quite a short length of time for our sprints as the timeframe we have for the SEPR assessments is shorter than a typical software development project would be. In SCRUM the technical lead role can be rotated between members of the team each sprint, we have chosen to instead rotate the technical lead role between each of the 4 assessments. We made this decision as we want each of the assessments to have a clear direction and between some of the assessments we will need to swap code with another group. This could require a different technical lead, as it may lead to the use of different tools or techniques that the current technical lead is less familiar with than someone else in the group.

Justification of SCRUM choice

In SEPR we are working in a relatively small group of only ~~7~~ 6 people. This is one of the reasons we chose to use SCRUM, as agile methods work better with smaller teams of developers, because this makes it easier to hold meetings and there is less documentation overhead to manage. We also chose SCRUM as we wanted to be able to adapt quickly to any changes in the direction of the project during development, either from stakeholders changing the requirements of the project or any unforeseen problems. SCRUM lets us do this because of its iteration, short sprints and regular meetings, allowing changes to be brought up and implemented faster than in a plan driven method. The SCRUM methods daily meetings also help team members synchronize and ensure that none of the team members are coding the same features as each other. This would be an inefficient use of resources and would create unnecessary code. Another reason we chose SCRUM is the sprint reviews after each sprint guarantee that we allocate time to reflect on how effectively the group is working and adapt how we are working accordingly. This will be much more efficient than if we waited until the end of the project to review our method.

Development and collaboration tools

For the documentation we are using a shared google drive, so we all have access to the updated documentation at any time in the project. On google drive we are using google docs for most of the documentation and google sheets for the tables, such as requirements and risk assessment tables. Another tool we are using for this project is ~~discord~~ Slack , on ~~discord~~ Slack we have a private server we are using to communicate general messages about the project and organise meetings. We will also use ~~discord~~ Facebook Messenger to have group call meetings, when it is not possible for group members to meet in person, allowing more flexibility with meeting times. The version control system we are going to use is GitHub and its desktop application GitHub Desktop. We are using Github as having a stored version history allows us to rollback to a previous version if there is a problem with the current version. Additionally Github is a decentralised version control system which mitigates the risk of losing versioning as each member of the project would have a local repository with versioning. We used StarUML for the UML diagram as it simplifies the syntax of making a UML diagram, this was helpful for our group as this was the first time we made a UML diagram. For making our runtime model we used draw.io as it allowed multiple people to collaborate on making the model at the same time. Another tool that proved to be extremely useful for task management and division is ~~Trello~~ Jira. This is a website which perfectly matches the nature of the SCRUM development, as it not only allows to create tasks for each sprint, assign them to each member and mark them as 'Done', but it also allows to organise tasks into groups (the sprints) and archive them all at once when completed, showing us which sections of the development process were had not been met during the sprint so that they could easily be transferred to the next. Finally, the team made use of non-copyrighted assets in order to make the game aesthetically pleasing. These were taken from various sources such as daButtonFactory.com and unsplash.com or created bespokely by our asset designer using Procreate 5 in order to maintain the theme of the game and ensure that the map resembles York as the customer prefers.

4.2 Team organisation

As we are using the scrum methodology, we are also going to use the scrum development roles to break down our team. In scrum, there are three main roles: the product owner, the scrum master, and the development team. We decided to further break down the development team using the general software development roles: UX/designer, Technical lead, software testers, documentation writers, risk managers, web designer and developers.

In the scrum method, roles are highly adaptable; therefore we will be letting team members change roles throughout the project. This is due to the structure of scrum- someone could work in the development team for one sprint and easily change to a software tester for the next sprint. Allowing team members to change roles for each sprint is beneficial for this project as it allows everyone to try out the different roles to see where they work the most effectively. It will additionally prevent members from getting too fatigued with a single role and keep motivation high.

We also realise that is more efficient to have the whole team (or at least most of the team) work on a certain aspect, e.g. software testing, at once instead of giving that task to only one or two people. We have therefore given multiple roles to multiple people.

The assignment of these roles can be found in our previous deliverable

~~Michele was designated early on as the scrum master/ project manager. This is because of his strengths in scheduling, communication and organisation. Two other people in our group also showed an interest in leadership; for future assessments we have kept in mind that this role will be flexible. The technical lead is a good fit for Riju given his prior experience making games and programming with Java. He also expressed that he did not want to do a lot of documentation, meaning he can focus more on the coding.~~

~~Given his propensity for leadership as well as his ability to come up with design ideas, we have decided that Dan should be the UX lead. Mic, Martha and Sean will also focus more heavily on UX.~~

~~Due to the amount of work that needs doing, everyone in the group has agreed to do a fair share of both coding and documentation. For now we have placed everyone on the general development team and documentation team on top of any specific roles they may have. Furthermore, everyone is going to participate in the software testing towards the end of Assessment 2.~~

4.3 Project plan

The plan for the remaining assessments is detailed in the following Gantt charts, which are also presented in full on our website. We used Gantt charts as they allow for dependencies and the critical path to be easily seen. ~~We used PlantUML to create the Gantt charts; favouring this software due to its shallow learning curve and a high level of adaptability. The plans for assessments 3 and 4 are still quite simple; the major tasks in these plans have not been broken down like they have in assessment 2. Given that these assessments are still far in the future, any detailed plan we were to make would likely be disregarded as our understanding of the necessary tasks and timescales change. Assessment 2, on the other hand, requires a full plan with a breakdown of the architecture and implementation tasks. We do however understand that the assessment 2 plan is also going to be subject to change.~~