4. Method selection and planning

Group Number: **Team 20**

Group Name: **Gourdo Ramsay**

Group Members:

Lauren Waine
Megan Bishop
Tommy George
Bartek Grudzinski
Davron Imamov
Nathan Sweeney

a)
The software engineering method chosen was Agile; the primary purpose of this method is flexibility. In Agile, the project is executed recursively as opposed to sequentially. The project is still designed, developed, tested and reviewed but this process is repeated many times within the given timeframe. Thereby, the lack of experience in the team with making games and the lengthy time window allowed for exploration and adaptation of the project. Systematically, each team member is given adequate support with their tasks because the tasks are revisited repeatedly. Compared to Waterfall, this gave our team greater success in learning and complying with the project requirements: since an individual was not expected to complete their portion of work within one cycle.

Due to the nature of Agile, our tools were specifically chosen for being dynamic and malleable. We used Google for sharing data such as spreadsheets, written documents and implementation assets (for example images), using Google Drive, Docs and Sheets. For the codebase and version control system, we used Github, which allows for an uncomplicated and reinforced code sharing structure.
To communicate in a rapid environment, we used Discord which allowed for group calls with screen sharing as well brisk file sharing - for ideas.
The software used granted an inclusive ecosystem that was free, fast and moldable to our requirements. It supported communication, sharing of files and developing a plan of the project. The software supported multiple OS's and devices: keeping the team informed regardless of location.

An alternative proposed was Microsoft Office to replace Google Drive and Slack to replace Discord. The advantages of Microsoft Office are security, faster IT support and advanced features within each software package. However these come at the cost of being locked into the Microsoft cloud space, then on using software such as OneDrive and SharePoint. We evaluated the advantages and ruled them redundant in our project. The security of the project is important, though Google is secure enough, and should the project be compromised the consequences would not be dire. The complexity of the project is too little to require IT support and the advanced features either do not apply or would not be utilised by our team.
The advantages of Slack are its business-oriented design. It integrates with other business tools like Trello, giving users the ability to manage different projects and teams. It has powerful search and organisation capabilities between the different projects and teams.
It also has better security and compliance of industry standards, such as GDPR, compared to Discord. Overall, we decided that the advantages support a larger business as opposed to a small university project. The upsides would be cumbersome and difficult to adapt to whereas Discord is well known and secure enough for our project.

We chose the framework LibGDX to develop our game upon. This free, open source software is well optimised, frequently updated and ductile due to being open source. The framework was also supported by the community in the form of forums and videos easily accessible, walking the team through the framework. It is a very popular choice for developing 2D video games in Java.

The proposed alternative to LibGDX was JMonkey engine. After researching this, we discovered that it was more appropriate for creating 3D games, with extensive support for shader technology. As our project plan was to create the product as a 2D game, this was not useful for us. Additionally, JMonkey has much fewer learning resources available online than LibGDX, which would have made our project much more challenging. LibGDX has many useful online resources, such as forum pages, a discord discussion group, and many YouTube tutorials available to us.

We used Eclipse as our IDE. To accommodate the dynamic environment, we chose to all use the same IDE: reducing the strain when sharing code in meetings. Eclipse supported Java and LibGDX better than its competitors at the downside of our team lacking experience in the IDE. To reduce complexity we chose the most common and supported IDE with online tutorials.

One alternative to Eclipse is VS Code. We decided that although VS Code provides many useful features such as built-in live sharing and a library of modifications that is easy to access. It is also a very lightweight IDE, and all group members were familiar with it through using it in previous university modules. However, it is not built with Java in-mind, hence we chose Eclipse to better support our development. The UI of VS Code is arguably better, though the pace of our team was not held back by UI but experience, hence this is not applicable.

Finally, we used PlantUML for creating the Gantt Charts. It is a free software that is quick to use with no account required. This, combined with a similar format to writing code in a programming language, made it a quick to learn process. It was flexible and had plenty of example code blocks to use. The alternative was Wrike and MeisterTask which required registration and/or purchase of the software. They used a UI to create the chart as opposed to code, which gave less flexibility to the diagrams. Therefore, we chose PlantUML as it was more convenient for the task.

The software tools named have a strong fitness to the team’s goals. All tools used are free, adaptable, and suit the needs of each member. We did not have regrets about any decision in choosing software. Our software engineering method also had a strong fitness to the team, due to the incomplete map of the project and its requirements. The nature of recursively improving the project allowed for each team member to advance the project with their expertise.

b)
Our team organisation was a democratic approach through decentralisation with no specific leader. No team member has more authority than the other. Work is to be outlined and volunteered for. Each piece of work has specific requirements that the team came up with along with a priority and start and end date. The advantages of this approach are it allows everyone to use their strongest attributes by picking their work type. It gives a feeling of purposefulness by utilising everyone through equal opportunities. The use of coordinated strict deadlines allows for the project to be completed on time. Most importantly of all, the democratic approach prevents blind faith put onto one person. As the team is inexperienced in game making, a lack of specific leader can ensure that overtime decisions average out in a positive manner. Instead of a volatile system like a specific leader which can fail or complete the project with no inbetween.

Compared to the alternatives this decision felt the most fair to each member of the team. A hierarchical team organisation would create power dynamics that would not be healthy within a University project setting. Since no external variable would drive the team to come together and work with each other unlike an actual professional environment. A matrix team organisation would complicate the ecosystem, being more proficient in a multi-team and multi-project environment.

In conclusion, the degree of autonomy and responsibility is well supported by the project for the team. Each team member can support other team members through our chosen software engineering method, Agile, which further solidifies our choice.

c)
We decided to take a dynamic approach to the project. The first week of the project, we set up communication mediums alongside outlining methods of file sharing and code backups. We laid the groundwork for ourselves without choosing the people to work upon it [07/11/22 - 13/11/22, high priority].

Afterwards, in the second week we began discussing the requirements for the project [14/11/22 - 20/11/22, high priority, dependent on the stakeholder interview in the upcoming week]. We also started preparing for the stakeholder interview [14/11/22 - 20/11/22, high priority].

In the third week we conducted our stakeholder interview. This allowed us to assign the workload to each team member [23/11/22 - 23/11/22, high priority, dependent on the stakeholder interview]. Each segment of the project had what we at the time believed was the appropriate number of people. The plan for the architecture was assigned to Tommy and Nathan [23/11/22 - 30/11/22, low priority]. The draft for the team website was assigned to Davron [23/11/22 - 30/11/22, low priority]. The requirements for the project had to be polished after the interview, this was assigned to Megan, Tommy and Bartek [23/11/22 - 30/11/22, medium priority, dependent on the stakeholder interview]. The risk assessment table was assigned to Megan to begin [23/11/22 - 30/11/22, low priority, dependent on the requirements for the project]. Lauren was assigned to begin the write up for the software justifications [23/11/22 - 30/11/22, low priority, dependent on the stakeholder interview]. Bartek was assigned to manage the weekly Gantt chart [23/11/22 - 01/02/23, medium priority]. All team members were assigned to get familiar with LibGdx [23/11/22 - 07/12/22, low priority, dependent on project requirements].

In the fourth week we expanded on the specifics of our tasks. The requirements for the project were polished, however we wanted them to be condensed. Megan was assigned to the task [28/11/22 - 04/12/22, low priority]. Nathan was assigned to begin the UML diagrams [28/11/22 - 04/12/22, low priority]. Davron was assigned to begin the risk management process [28/11/22 - 04/12/22, low priority].

Week five was spent working on the assigned workload.

Week six was also spent working on the assigned workload.

The next four weeks were holidays and an exam week, so the team decided to not continue work.

After we returned to schedule in week eleven, we began to refine our project. Davron was assigned to assist Lauren with making the implementation by creating a foundation for the game [16/01/23 - 22/01/23, high priority]. Lauren was assigned to create the main menu for the implementation [16/01/23 - 22/01/23, high priority, dependent on the game foundation]. Nathan had continued with the UML diagrams by moving onto the behavioural diagrams [16/01/23 - 22/01/23, low priority]. Bartek was assigned to help with the website. [16/01/23 - 22/01/23, medium priority]. Tommy was assigned to create and/or find assets for the game [16/01/23 - 22/01/23, medium priority]. Megan was assigned to continue work on the write ups for risk management and requirements [16/01/23 - 22/01/23, medium priority, dependent on the game foundation].

In week twelve, the team had expanded the scope of operations further. Tommy and Davron were assigned to work on the game assets. With Tommy drawing the pantry map and ingredient thumbnails and Davron drawing the tile maps [23/01/23 - 29/01/23, medium priority].

Finally, in week thirteen the team focused on finishing any remaining work [23/01/23 - 01/02/23, high priority].