Method selection and planning

Group 12 - T12

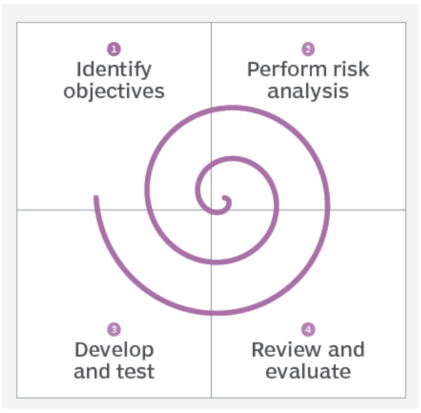
Members:

Ruslan Allahverdiyev (ra1354)
Billy Brudenell (bb1085)
Harry Erskine (hde501)
Usman Khan (muk509)
Adi Laskowski (akl532)
Ben Remmer (br894)
Ollie Stoole (os878)

Contents

Method Selection (Agile Scrum/Spiral)

The method we have selected to use during this project is a Spiral method known as a scrum. This is a repeated cycle of identifying what needs to be done and its risk, followed by some development and then an evaluation of the product so far. This isn't too big of a project, so a plan-driven model is definitely viable. We thought that since it's some of our first times doing a project like this, we should go with an agile method to allow more space for any issues that arise:



1. Identify Objectives

   New requirements are identified in as much detail as possible. This could involve interviewing the "client" or other aspects of the existing system to identify what needs to be done in the next couple of days.

2. Perform Risk analysis

   At this stage, we take a look at the new objectives that we have made and identify which people will be taking on which objective while also looking at each objective and identifying any potential risks that may occur.

3. Develop and test

   This is when we all go and work on our designated parts of the project, we have a group Discord allowing for quick communication and sharing of resources if more than one person is working on a part of the project.

4. Review and evaluate

   We then come together again for a meeting and discuss how our individual tasks have gone, this time allows us to work together and see if anyone else can solve the issues encountered by another group member.

There are a few main reasons why we chose this method:

- This is a good way to break down the projects into smaller, more digestible chunks.
- It's also pretty flexible, as it's a cycle that repeats often; we are able to identify any changes to the requirements quickly and respond appropriately.
- It allows for frequent risk assessment, as each cycle contains a risk assessment segment; we improve the security of our system while also trying to eliminate any risks that may come up.
- If at any point we feel like we are confused about any requirements or want to get more customer input, we can set up customer feedback as this method allows for it.
- We thought it would be a good fit as we were thinking of splitting up the projects into smaller one-week tasks.

Development and Collaborative Tools used

The Java framework we chose to use was libGDX

- Another Java library LWJGL, which gives users access to native audio and graphics APIs, was something we had considered adopting. But as code reuse is a crucial component of software engineering, we opted to employ it since libGDX uses it in its framework.
- As libGDX is built on top of LWJGL, it offers low-level access which enables us to spend more of each Scrum sprint working to resolve problems that are connected to the task that was assigned to us by the client.

For version control, we used Git and stored our repository on GitHub

- Avoids collisions and makes collaboration significantly easier.
- Hosting the repository on GitHub gives us access to GitHub Pages, which makes it simple to view documentation, as well as version control history maintenance.

Our IDE of choice is IntelliJ

- Has the best framework to work with libGDX
- Developing a project is easy using IntelliJ. It has attributes that are absent from VS Code, making it easier to work on large projects with numerous classes. Examples include automatic class refactoring and easy integration with Gradle.

Weekly meetings

- Allows us to properly utilise the spiral method that we have planned to use throughout the project to keep us all updated and aware as to what's going on.

Discord

- A solid messaging platform that allows us all to communicate and share work with each other. We chose this because we're familiar with it, making setup a breeze.
- Meetings can also be held over Discord calls when in-person isn't possible.

Google docs

- An easy way to share and collaborate on various documents.

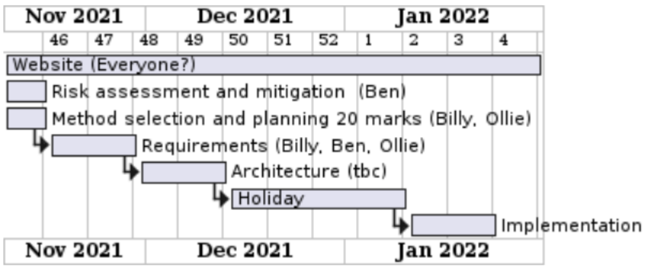
Team Organisation

While we initially considered splitting the team into hard-set groups, we figured it would make more sense for our members to contribute as much as they can to each deliverable, while they still focus on areas they're more familiar with. This allows the work to be done at a decent pace while also not restricting any member to one specific task, letting us assist each other where necessary. Through this method and our larger than average team, we could advance through the work in reasonable time; we'd always have someone working on code, another working on another written deliverable and can be a collaborative effort when necessary. This still allows for weekly scrums in each area, allowing for more collaboration in the scrums.
The final result is an effective and efficient weekly scrum method, alongside regular and prolonged work on all fronts of the project.

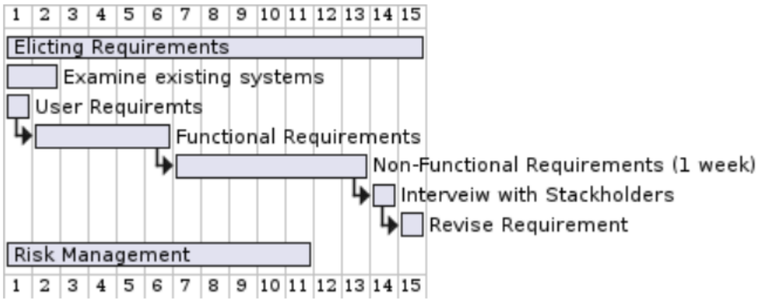
Project Organisation

Project Gantt chart

This was our original project Gantt chart, giving a brief outline of the project. This wasn't detailed enough so that we could express task priority and who should be on each task. We kept it as is for the start as we knew there were going to be changes made to this later down the line.



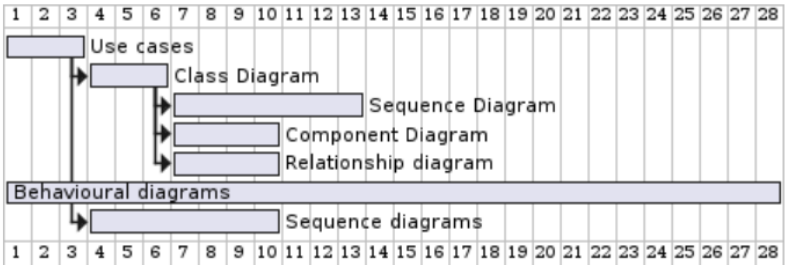
Eliciting Requirements Gantt Chart



This was our timeline for our requirements, which are first extracted from the product brief, then customer interviews and feedback.

Architecture Gantt chart

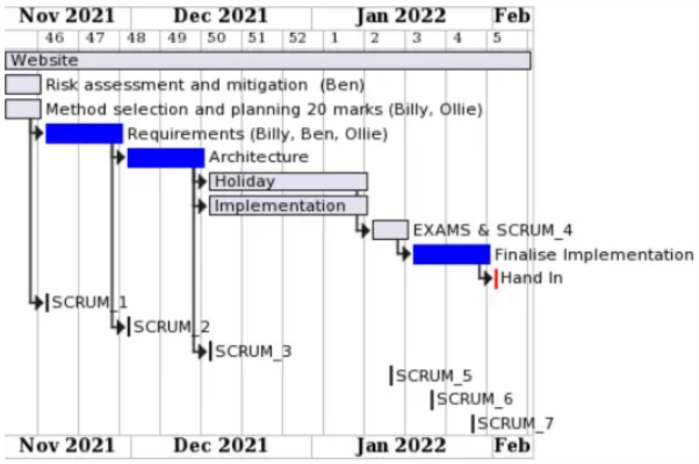
The Architecture Gantt chart is created to allow the team and the client to understand how the implementation is laid out, with a proper representation to be made.



Systematic plan

This plan here is the final plan we used. The parts highlighted in blue were high priority tasks:

- Architecture is at a high priority as without it, the implementation can't begin.
- Our requirements are essential to our architecture.
- Our initial implementation isn't too important as that's where we are likely to find mistakes and whatnot, but doing this earlier allows us to come into the finalising implementation part with a strong foundation, and since the implementation is how we make the project finalising the implementation is very important.



And tasks that are dependent on one another are shown by the arrows:

- The scrums are normally the result of a breakthrough, meaning we need to plan accordingly for the next step of the process.

- To begin with the implementation over the holiday, we need to have completed our requirements which will allow us to complete our architecture and begin with implementation.