Method Selection

Software engineering method

To effectively manage our project, it is important to agree on a software engineering method that will work well with the type of project we have been given, as well as being comfortable to use for our team. It was identified that the development method we picked must be flexible to the customer, with the ability to adapt to changes in the projects end goal requirements. An agile method such as Scrum or Extreme programming would are good candidates for methods to use as they are both simple to implement, and we will be able to manage our project with greater transparency making it easier to work as an effective team.

Extreme programming (XP) is a software development approach that primarily focuses on the ability to handle changing customer requirements quickly whilst still offering high-quality products. To achieve this XP encourages the use of rapid development cycles which gives more space for the customer's requirements to be implemented between the releases. XP suggests using tactics such as pair programming, implementing features only when necessary to simplify the design. XP also stresses that there should be a heavy focus on code review, often through the prior mentioned pair programming strategy.

Scrum is a software development framework that separates work into smaller subtasks (called "sprints") that can be completed in short periods of time, generally around two weeks. Each working day starts with a "scrum" in which all team members engage in a discussion and report the work they have completed previously, the work they aim to accomplish today, and any risks they think might occur. After sprints, there will be a scrum review that looks at the work that has or hasn't been completed and the team then discusses ways to approach the following sprint with greater efficiency.

Ultimately, we decided to use the Scrum development method as we thought that it would be flexible than XP, giving us greater control over our project as we will be able to adapt and grow as a team by reflecting on previous ideas in order to become more effective as the project progresses. By adopting the Scrum framework as an agile method, we will maintain our project whilst allowing us to incorporate any new requirements or changes to previous requirements as requested by the stakeholder. Using agile methods will have a significant benefit considering the stakeholder will have continually changing requirements, therefore an appropriately fast software engineering method will increase our ability to fulfil the requests of the stakeholder; producing high-quality work on time. Furthermore, a team of six members is an ideal size to use the Scrum software engineering method as it will allow us to quickly discuss progress and split tasks effectively.

Tools

To maximise the time spent on the project as efficiently as possible it is essential to use development and collaboration tools that suit the project and team. This will reduce downtime which might occur if poor tools are chosen which might hinder the development process in some way. Being able to create the product without worrying about the tools used allows us to focus fully on each task, resulting in a higher quality product being developed in a shorter time. Using communications tools that are intuitive and easy for all of the team members to use will significantly streamline the project by allowing members to share ideas and update others on the progress of current tasks.

Document file sharing - Google Drive

We decided to use Google Drive to keep track of any documents such as the assessment specifications and scenario we need as well as any document deliverables. Having a universal storage for any admin files in our team allows us to collaborate on creating any deliverables - and makes it easier to refer to any assessment documents. By backing these files from the team drive also decreases the chance of losing data since we can store a copy locally as well as on Google's cloud services.

Project source code & resource management - Git

Using a version control system to manage our source code and any other resources (such as image files or sounds) will make the actual creation of the game much easier. Git gives us the ability to coordinate and work on the project at the same time, increasing the speed at which we can develop. In addition to this, GitHub offers version control and branching, decreasing the risk of data loss. Having access to this also means that if mistakes are made or decisions on development approaches change we will easily be able to revert back to a previously agreed upon version.
We decided that we would use Git over the other suggested VCS, Apache Subversion - SVN, mainly because we are more familiar with Git, as well as having the perk of being decentralised to reduce the chance of data loss risks. Git is also more popular, meaning sharing the project with others will likely be easier than if we chose to use SVN instead.

Task management - monday.com

As a team, we also decided that it would be wise to use a task management tool that would let us visualise and manage current and future tasks. monday.com provides a utility that can create "boards" which can be used to break down tasks into specific groups, and then classify them based on priority, and the current working status. People can then enter that they are working on a task to communicate with other team members on their current work. Using this alongside a messaging service reduces the risk of miscommunication greatly.

Communication - Facebook Messenger

At the very start of the project, we decided to create a group chat on Messenger to initially get to know each other and then begin planning the first steps of the project. Having access to a common communication tool allows us to quickly update each other on project events and can be used to arrange team meetings. Using a group messaging service like Messenger seemed to be the best choice as all of the team members already had accounts.

IDE - IntelliJ IDEA

For our development environment, we decided to use IntelliJ IDEA. This is because IntelliJ is advanced, but still simple to use - sharing similarities with Eclipse used by all team members in our first year. Also, the IDE also supports and has a simple importing process for either of the game libraries we were considering using (LibGDX or LWJGL). IntelliJ also includes Git integration which will is especially useful considering we will be using Git for our version control.

UML diagram production - *StarUML & LucidChart*

We decided to use StarUML to produce the UML diagrams as it is freely downloadable software that you can download onto your personal computer. Since no one has used any UML diagram creation tool prior to the project no one has any preference towards any specific software, and being as this was recommended and we had seen it was used for previous years projects, we decided this was a suitable option. We also did give some testing using other UML software such as draw.io because we saw that it had google drive integration, however actually designing the UML models we found easier in StarUML so we thought this benefit was more important than the group collaboration facilities.

Team organisation approach
To ensure all team members produce the best work we are capable of it is important that we have roles that we are confident and competent in. While people will primarily specialise on tasks based on the specific roles they are assigned, these are not strictly fixed roles. We decided to make this decision to give greater flexibility within our team - as not to restrict creativity or ideas amongst people for any given task. This is especially important considering no member has full specialised experience in one area meaning the cost of having more dynamic roles is minimal.

Team Leader - Charlie Dyson
The role of the Team Leader is core to the success of the project. Overseeing all progress on the project, Charlie will track project development over time to make sure that the everything runs smoothly, and identifying any risks or problems that occur in order to reduce downtime or unnecessary backtracking in the project development.

Graphics Designer & Risk Manager - Shubei Qian
Shubei is in charge of designing any graphics for the project or game assets as well as risk management for the project. The role of Graphics Designer covers designing any art that is used on the website in addition to any assets in the game. The role of the risk manager is to identify track any risks that might occur and then to provide suggestions for potential solutions to these issues.

Head Developer - Ed Gould
The job of the head developer is to handle the creation of the game and be the primary team member to implement features into the game. Ed will work with Charlie and Luke to coordinate certain stages of the game's development to fit into the Scrum work cycle by working out incremental key stages in which additional content can be added each Scrum cycle.

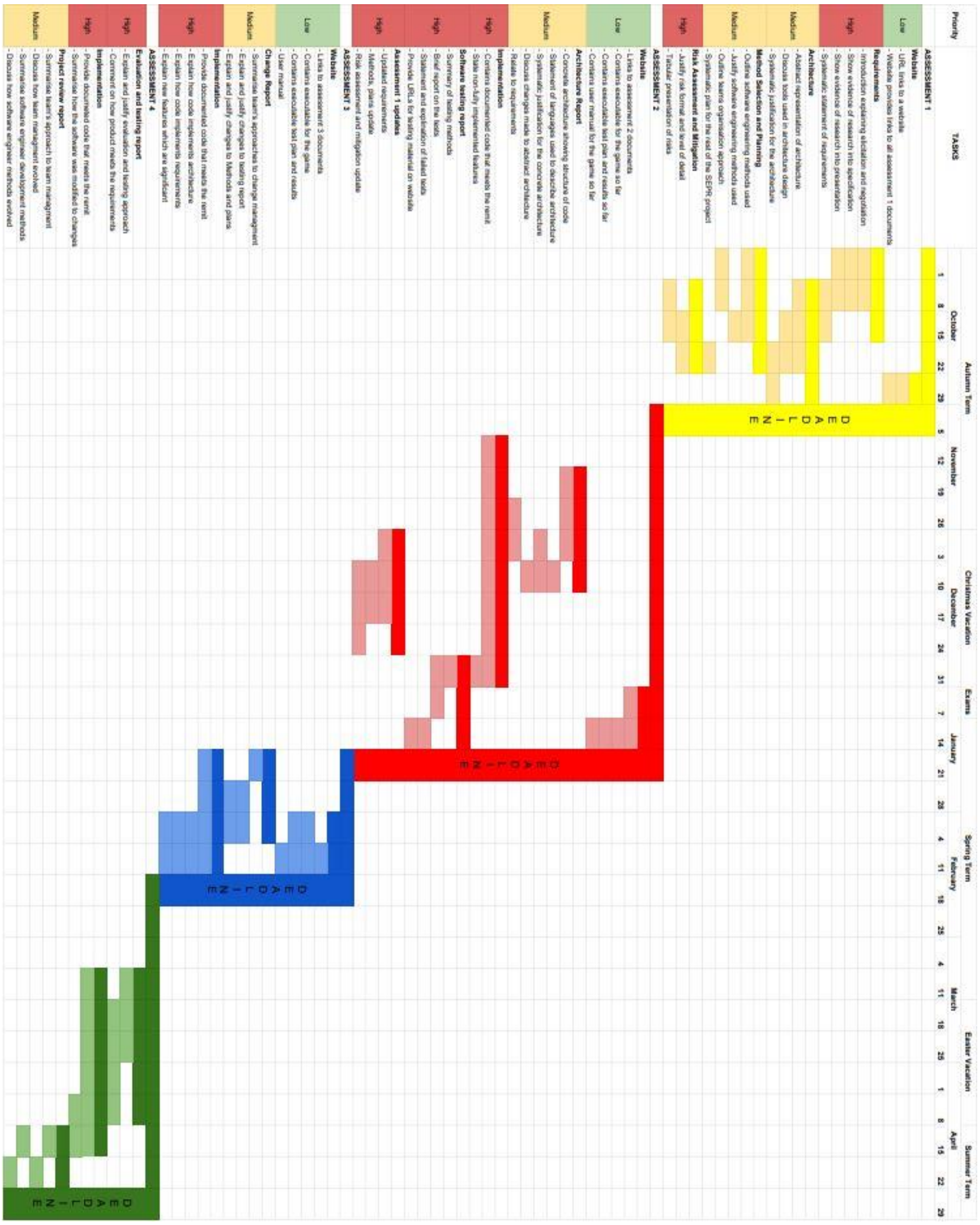
Design Management & Creative Director - Luke Richardson
Luke will mainly be working with Ed and team members that help produce the software for the game by planning the overall design timeline of the game. Working with the team, Luke will manage any design decisions that might change during the development cycle, either due to changing requirements of the stakeholder or agreed upon changes to the design approach during the project.

Web Developer & Interface Manager - Umar Farooqi
Umar was happy to handle the development of the website where our work is presented. He is confident in learning the languages and tools required to produce the website, as well as sorting out web hosting. Umar will also relay information from the stakeholders or other customers to the team.

Secondary Developer & Lead Tester - Andrew Todd
Working mainly with Ed, Andrew will assist in the software development of the game, collaborating through GitHub. With a similar role to Ed, they will both be working on different areas of the game to increase the speed at which it is developed. Andrew will also have a greater focus on testing the software to ensure it functions as intended and runs smoothly and without bugs.

Project Plan

In order to plan the rest of our project, we decided to use a Gantt chart. This allowed us to create a plan for our software engineering project for the entire year and include dates, priority and status to have a visual indication of what needs to be done and by when. The Gantt chart we designed is evidenced below however a more readable version is available in the extra documents section of assessment 1 on our website www.limewire.me


Priority
TASKS
ASSESSMENT 1
Website
- URL links to a website
- Website provides links to all assessment 1 documents
Requirements
- Introduction explaining elicitation and negotiation
- Show evidence of research into specification
- Show evidence of research into presentation
- Systematic statement of requirements
Architecture
- Abstract representation of architecture
- Discuss tools used in architecture design
- Systematic justification for the architecture
Method Selection and Planning
- Outline software engineering methods used
- Justify software engineering methods used
- Outline teams organisation approach
- Systematic plan for the rest of the SEPR project
Risk Assessment and Mitigation
- Justify risk format and level of detail
- Tabular presentation of risks
ASSESSMENT 2
Website
- Links to assessment 2 documents
- Contains executable for the game so far
- Contains executable test plan and results so far
- Contains user manual for the game so far
Architecture Report
- Concrete architecture showing structure of code
- Statement of languages used to describe architecture
- Systematic justification for the concrete architecture
- Discuss changes made to abstract architecture
- Relate to requirements
Implementation
- Contains documented code that meets the remit
- State non-fully implemented features
Software testing report
- Summary of testing methods
- Brief report on the tests
- Statement and explanation of failed tests
- Provide URLs for testing material on website
Assessment 1 updates
- Updated requirements
- Methods, plans update
- Risk assessment and mitigation update
ASSESSMENT 3
Website
- Links to assessment 3 documents
- Contains executable for the game
- Contains executable test plan and results
- User manual
Change Report
- Summarise team's approaches to change management
- Explain and justify changes to testing report
- Explain and justify changes to Methods and plans
Implementation
- Provide documented code that meets the remit
- Explain how code implements architecture
- Explain how code implements requirements
- Explain new features which are significant
ASSESSMENT 4
Evaluation and testing report
- Explain and justify evaluation and testing approach
- Comment on how product meets the requirements
Implementation
- Provide documented code that meets the remit
- Summarise how the software was modified to changes
Project review report
- Summarise team's approach to team management
- Discuss how team management evolved
- Summarise software engineer development methods
- Discuss how software engineer methods evolved
Low
High
Medium
Autumn Term
Christmas Vacation
Exams
Spring Term
Easter Vacation
Summer Term
October
November
December
January
February
March
April
DEADLINE
DEADLINE
DEADLINE
DEADLINE