Method Selection and Planning

Group 1 Cohort 1

Tom Haslam, Christopher Oulton, Charlie Piper, Shirin Sitara, Dillon Anthony, Kevin Thomas, Ella Daramola

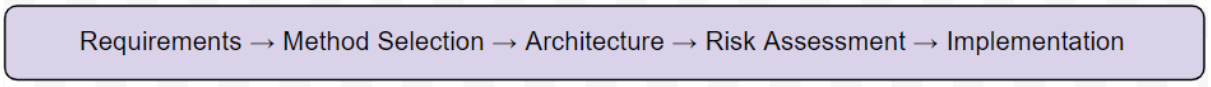
4A.

SOFTWARE ENGINEERING METHODOLOGY:

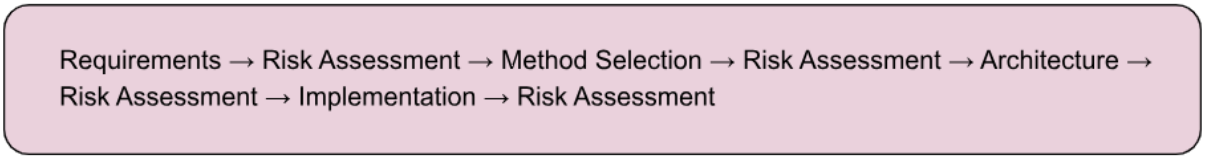
The Agile methodology involves iterative mini-increments to add functionality. It provides flexibility and adaptability by encouraging continuous collaboration between different teams such as designers, developers, testers, analysts, etc. Keeping customer satisfaction a priority, it also requires frequent customer interaction and feedback**.**

The Waterfall methodology has a sequential development approach. It focuses on each phase being completed, before moving on to the next one. This process ensures that each phase is well documented, and provides a guide for subsequent stages. However, this brings in rigidity and limits adaptability of the project.

We followed a combination of the Agile and Waterfall methodologies. This hybrid model integrates the strengths of both processes, which helps achieve our requirements. The first phase of this module focuses on documenting organisation and architecture, and a base layer of implementation. This plays into the strengths of the Waterfall methodology as most phases were dependent on the other, causing it to be implemented in a linear format.



However, to ensure thoroughness, we looked at having an iterative approach to phases, which was the strength of the Agile methodology. For example, phases like ‘Risk Assessment’ were updated regularly after assessing the risks in each stage.



In this manner, we were able to incorporate both methods in our project and the application of the same can be viewed in an upcoming section.

DEVELOPMENT TOOLS:

1) LibGDX -In our quest to develop a 2D game with Java-based technology, LibGDX emerged as the optimal choice among game engines. Its versatility, performance, and extensive feature set make it a standout option for game development projects like ours. LibGDX provides comprehensive support for 2D graphics rendering, input handling, and more, empowering us to create immersive gaming experiences across the web. [1]

2) Tiled - Tiled is our go-to tool for managing tile-based maps in our game development process. With its user-friendly interface and powerful features, it simplifies the

creation and editing of tilesets, tile layers, and object layers, allowing us to design intricate game environments with ease. Its support for various map formats, including JSON and XML, ensures compatibility with our game engine, LibGDX. Also, Tiled offers extensive customization options, enabling us to define properties and behaviours for tiles and objects, such as collision detection and animation triggers. This flexibility is crucial for implementing complex gameplay mechanics and enhancing player interaction within our game world.

3) IntelliJ- For our development environment, we opted for IntelliJ IDEA, primarily because of its robust features tailored for Java development. It offers seamless integration with various tools and frameworks commonly used in Java development which streamlines our workflow and enhances productivity. Additionally, its intuitive user interface and extensive plugin ecosystem provide us with flexibility and customization options to suit our project's specific requirements.

COLLABORATION TOOLS:

1) Communication:
   Our main methods of communication are regular in-person meetings and Whatsapp.

2) Organisation:
   Jira - a ticketing software, helped to break down big deliverables to smaller subtasks. This helps us track the team's progress and organise our utilities in an efficient manner.

   Plant UML - To organise subtasks and visualise our long term plan, we proposed the use of Gantt charts. We looked into using the 'timeline' feature on Jira, but decided against it as it did not match the required format. We chose Plant UML to execute this as this software is open source and has a text based syntax, which makes it easy to learn and update. It is platform independent and is easy to integrate into documents, which further strengthens our ability to collaborate using it.

   Google Drive - For rough diagramming, general planning, documentation and organisation, we relied on the most convenient tool - a shared Google Drive. Being students, we already had access to this resource and were fairly familiarised with the system. Further, it accommodates various types of files, which worked well with our requirements.

3) Version Control:
   For continuous integration and collaboration we chose to use the most popular tool, GitHub. GitHub is one of the most-widely used tools, thus having a well established base to learn from and to acquire help if required. It has a user-friendly interface making it easy for developers of different levels to use. Even though the team was divided on their familiarity with GitHub, seeing its utility and features, we decided on it being the best choice for the project.

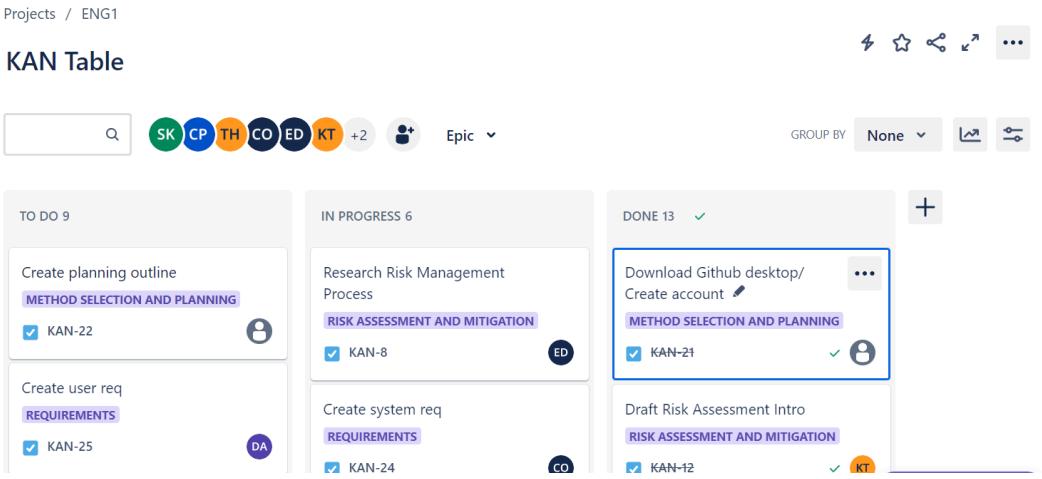
4B.

We first created the base for the website, whilst waiting on client meetings and requirements.

We then organised meeting times. The team is set to have regular weekly meetings every Thursday between 9:30 and 11:30 a.m. Following the methodology, the team discusses progress made in each subfield, and any challenges faced and shares critical feedback on each other's work. Additional meetings take place on Tuesdays with varied timings, depending on the tasks assigned for each week.

At the start of the project, we decided to split up every deliverable task amongst the 7 of us, such that everybody gets the opportunity to learn and contribute to every field. However, we soon realised that it was time-consuming and unproductive to do so. Each task was then divided amongst the members playing to their strengths and choices.

We used Jira to break down big deliverables into smaller subtasks. These sub-tasks are assigned to individual members, who can then place them in either 'to do', 'in progress' or 'completed'. This helps us track the team's progress and efficiently organise our utilities. (Below is a snapshot of our Kanban board on Jira.) We made Gantt charts to organise subtasks and visualise our long-term plan.

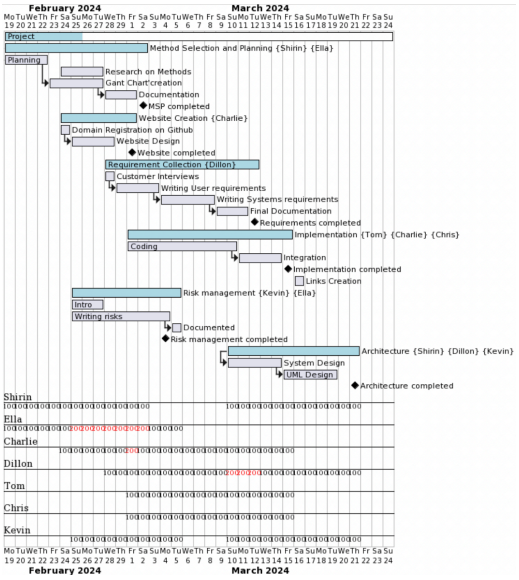


In terms of implementation, we used GitHub for collaboration. Each member of the team worked on their task by creating a branch for themselves and then merged to master as and when required.

In this manner, each of us got a fair share of the workload and responsibility of the project.

4C.
The project commenced with the Method Selection and Planning phase during it's first week. This initial planning involved assigning deliverables to team members based on their interests and strengths. Website and Risk Management tasks were also initiated.

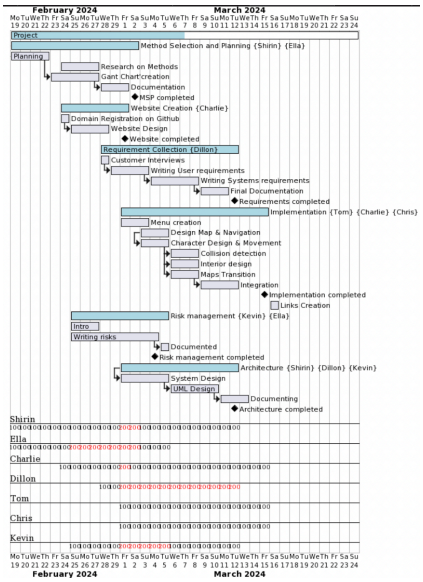
1. Method Selection and Planning (High Priority)
    - Start: 2024-02-19, End: 2024-03-15
    - Includes Research, Planning, Gantt Chart Creation
    - Dependencies: None
2. Website Creation (Medium Priority)
    - Start: 2024-02-24, End: 2024-03-01
    - Includes Domain creation on Github, Website Design
    - Dependencies: None
3. Risk Management (Medium Priority)
    - Start: 2024-02-25, End: 2024-03-15
    - Includes Introduction, Writing Risks, Documentation
    - Dependencies: Product brief



In the second week, the Requirement phase began. Simultaneously, Risk Management progressed from the previous week and Website Creation was completed with the exception of links which would be added later on. After gathering the requirements, the team introduced the software Jira, leading to the addition of crucial sub-tasks.

1. Requirement Collection (High Priority)
    - Start: 2024-03-06, End: 2024-03-19
    - Includes Customer Interviews, Writing User/System Requirements,
    - Dependencies: None
2. Implementation (Highest Priority)
    - Start: 2024-02-28, End: 2024-04-11
    - Includes Menu Creation, Map/Navigation Design, Character Design/Movement, Collision Detection, Interior Design, Map Transitions
    - Dependencies: Requirement Collection

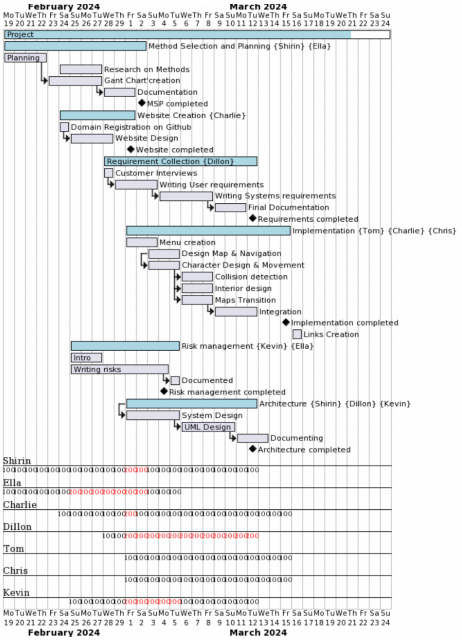
During the third week, the requirement phase was completed. The Implementation phase progressed. However, the "Architecture" phase, initially planned to commence earlier, was slightly delayed due to the priorities of other project components.



During the fourth week, the Implementation phase continued with tasks like integration. The Architecture phase finally commenced.

1. Architecture (High Priority)
    - Start: 2024-03-10, End: 2024-03-21
    - Includes System Design, Database Design
    - Dependencies: None

The fifth and final week was dedicated to addressing any remaining tasks or loose ends. While the core tasks and their order remained largely unchanged, task durations, start and end dates, and resource allocations were adapted based on the project's evolving.



REFERENCES:

[1] : https://libgdx.com/
[2] : https://www.mapeditor.org/
[3] : https://www.atlassian.com/software/jira?&aceid=&adposition=&adgroup=144591217916&campaign=19313277976&creative=652205474105&device=c&keyword=jira&matchtype=e&network=g&placement=&ds_kids=p74602868222&ds_e=GOOGLE&ds_eid=700000001558501&ds_e1=GOOGLE&gad_source=1&gclid=Cj0KCQjw-r-vBhC-ARIsAGgUO2DUwLfx59C5v1Cvn7kMuvDYospLkTSgPb5h-bhOJAeBH0FHLTcEZM4aAoXREALw_wcB&gclsrc=aw.ds