Question A - Q&A

- "What tools did you choose to use?"
  - IntelliJ - Used for editing code , Tiled (Mapeditor.org) - program that provides a GUI to create a tile set for a game , Paint.net - to make the sprites, LibGDX, Gradle - To run LibGDX, compiles packages
- "What made you pick this library?"
  - LibGDX looks like it had the largest online community and support online, so they can have the biggest database of information when developing the game. When researching, Jonathon had realised hes played atleast three games developed using LibGDX, which proved the point more so.
- "How have you split the workload between you two?"
  - The workload is split 50/50. Jonathon has worked on generating the tile map, currently working on collision. Harry working on the movement and creating a chef class. Has created the camera, and sprite patch.
- "What tools are you using to collaborate and share code/information?"
  - Discord , and in person meeting , occasionally snapchat for instant updates. These apps are in use because theyre already used by everyone in daily life, so it is intuitive to continue to use them.
  - Github because it is widely used and extremely popular among developers, when reasearching, Jonathon and Harry found that they could merge code, share data, and upload seemlessly.
- "What forseeable difficulties do you think you might find using this method / already encountered?"
  - Learning how to use Github effectively for teams, initially setting up the shared repository took a large amount of time.
  - A major forseeable difficulty of using Github is using the merge code feature. Due to the workload being shared, there would be a lot of warning errors when merging code. The team found a solution to this problem by solving them either on a voice call or an in person meeting.
- "What alternatives did you consider using for sharing and collaborating on code?"
  - They considered using Google drive, but due to the already present knowledge base of the application Discord, and the ability to access and edit code online with Github, they chose to use this instead.
- "What research tools have you used?"
  - Mainly discuss, Stack overflow, LibGDX documentation, Java Documentation, tutorials on youtube.

In terms of the implementation of the game, the team decided that, due to the structure of the project, two of the members will be coding and the other four will be completing the documentation section - keeping in close communication with the rest of the team at all times to maintain productivity with the most relevant information. Jonathon and Harry will direct the software development aspect.

The team chose to use several tools to support their software development project, including IntelliJ for code editing, Tiled (Mapeditor.org) for creating a tile set for the game, Paint.net for creating sprites, LibGDX and Gradle for running and compiling packages. These programs were chosen due to:

- IntelliJ: An integrated development environment (IDE) that is commonly used for Java programming. It provides a wide range of features for code editing, debugging, and testing.
- Tiled (Mapeditor.org): A program that provides a graphical user interface (GUI) for creating tile sets for games. It allows our team to easily design and edit maps for the game.
- Paint.net: A free image editing software that our team used to create sprites for the game.
- Gradle: This is a build automation tool that the team used to run and compile packages for their game.
- PlantUML: on further research and development, this program has been used by many members of the team , used in Architecture, Requirements, and Method Selection & Planning.

The team ultimately decided on using LibGDX library to develop the game as it has the largest community and support, with an abundance of information and forums available if the team get stuck. Additionally, Jonathon has played three games developed with LibGDX, giving him more hands on experience with what the library is and isn't capable of compared to other libraries.

In terms of the software development, the workload has been split evenly between the two team members. Both Jonathon and Harry work hand-in-hand to develop the game, using Github to merge their work and keep up-to-date with one another's progress. For example, Jonathon started development generating the tile map, whilst Harry worked on the game camera and sprite sheets. However, now Jonathon is working on the collision of the player and game map, whilst Harry is working on the player movement from creating the chef class.

To collaborate and share code and information, our has chosen to use a combination of Discord, in-person meetings, and occasionally Snapchat for instant updates. These apps were chosen because they are already widely used in daily life, making them intuitive to continue using.
We also use Github, which is widely used and popular, to merge code, share data, and upload seamlessly.

The development team has encountered some difficulties when using Github, specifically in learning how to use it effectively together and using the merge code feature. However, we found solutions to these problems by solving them through voice calls and in-person meetings.

As an alternative to Github, the team considered using Google Drive, but ultimately chose to use Github due to its already present knowledge base and the ability to access and edit code online.

The team has used research tools such as Discord, Stack Overflow, LibGDX documentation, Java documentation, and tutorials on YouTube to support the project and the team's working.

To summarise, the team's software engineering methods and the tools they selected were well-suited to their project, and the team was able to effectively collaborate and share code and information.

B. Outline to team organisation

The image above shows how our team is presented during in-person meetings. To help communicate easily and share information about documentation/implementation, we have divided each team to sit on each side of the table. This helps with being able to see each others screens and coordinate with select team members more efficiently, whilst also being able to keep in communication with the rest of the team if there are any questions.

The team has split into two different workloads - coding and documentation. Josh, Calum, Omar, and Rayan have split the documentation workload and distributed the five different subsections equally.

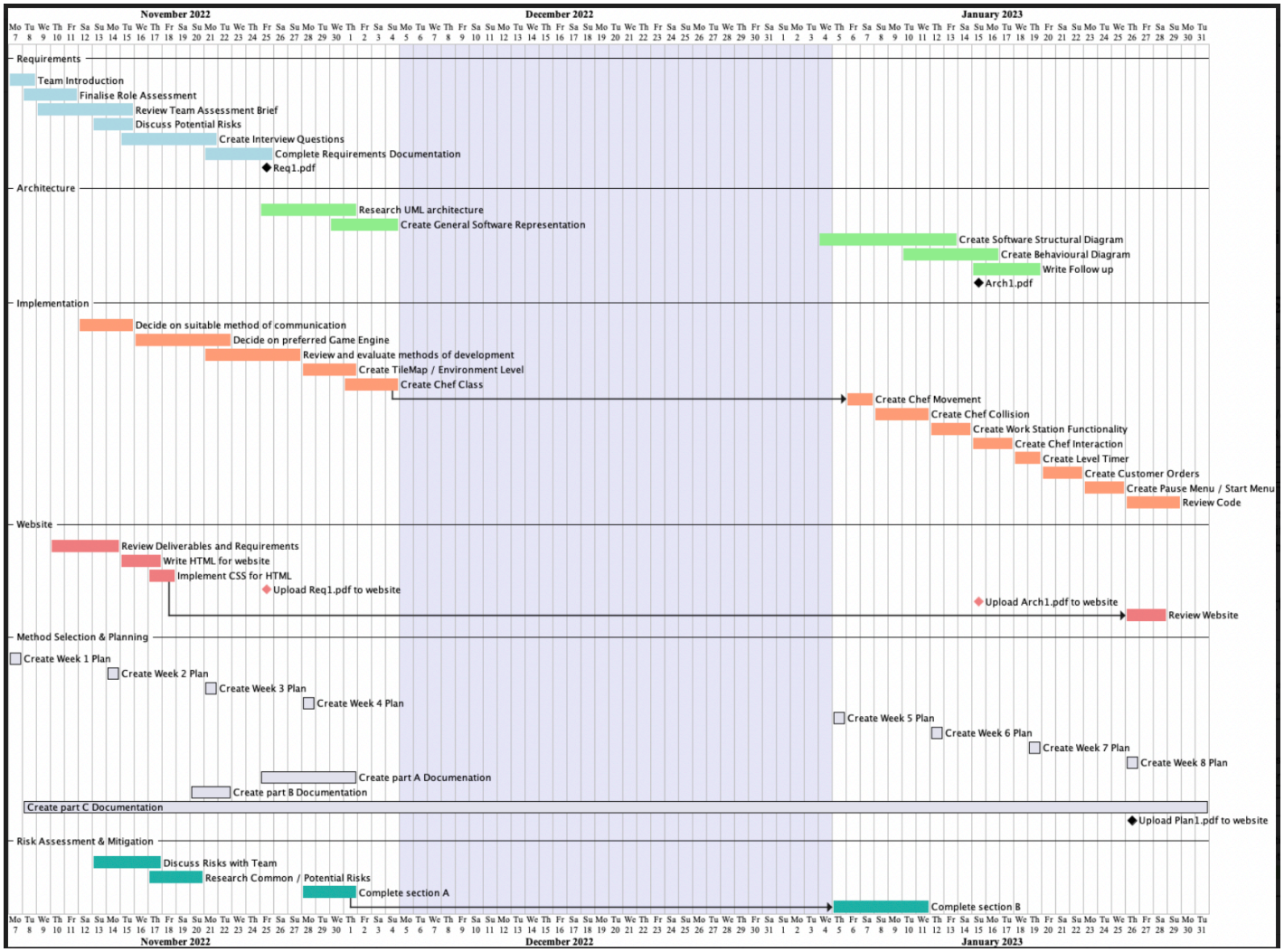
- Josh - Requirements
- Calum - Method selection & Planning, Website
- Omar - Architecture
- Rayan - Risk assessment & Mitigation

In terms of coding, Jonathon and Harry split their workload evenly, working through the implementation together. The workload is split 50/50. Jonathon has worked on generating the tile map, currently working on collision. Harry working on the movement and creating a chef class. Has created the camera, and sprite patch.

We tried to keep each section to one person so that we can use resource allocation to allocate team members to specific tasks. As with all part of the documentation, we also use online and in-person meetings to share information and answer questions about the tasks on hand.

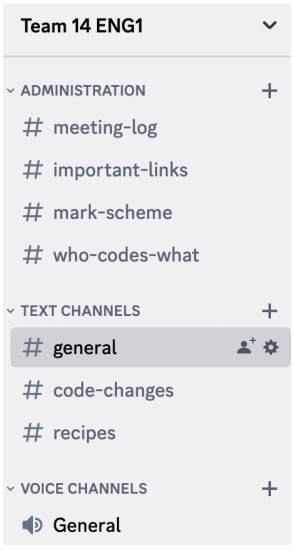
C. **Systematic** Plan

Primarily, we decided that we could section each topic into a role that a team member can work on to split the workload roughly evenly between the entire team, as seen above. From then on, a written down plan was decided to be the best option in order to openly see how and when the project will be progressed. A Gantt chart was the natural option for this, set out as follows:



We found that having a structured plan like this allowed us to clearly see the layout of the project, and the general direction and order in what to work toward. Another positive that was found using this method was that we could see what tasks may need to be allocated the most time from team members , for example the creation of the software structural diagrams.

However, although the Gantt chart was a good over view for the general plan, a week of miscommunication, trial and error, and slow-paced work meant that, as a team, we were completely belated in our work schedule, decreasing the proficiency of the scheduling method. To compensate for this, our team decided that, after the break, we can fit more into our schedule to make up for lost time. One of the methods we used to tackle this problem was to create a task table so that we can focus on current tasks that need to be done before the deadline. Another method that we used to weekly track our progress within the game was to use discord channels to communicate with each other about our progress. On this



Channel, we uploaded screenshots, meeting summaries, code updates, and other general questions. Additionally, we used weekly snapshots to track our progress compared to the initial Gantt chart, and how we could catchup to it

Tasks	Start Date	Expected Finish Date	Actual Finish Date	Assignment	Priority	Dependencies
Define Requirements	Week 2	Week 4	Week 6	Josh	High	-
Customer Interview & Evaluation	Week 3	Week 3	Week 3	All	High	Interview
Create Architecture UML	Week 3	Week 4	Week 6	Omar	Medium	-
Weekly Plans / Snapshot	Week 1	Week 8	Week 8	Calum	Low	-
Risk Assessment & Mitigation	Week 2	Week 4	Week 7	Rayan	Medium	Plan
Website	Week 2	Week 4	Week 8	Calum, Harry, Jonathon	Low	-
Website CSS	Week 2	Week 4	Week 8	Calum, Harry, Jonathon	Low	HTML document
Gantt Plan	Week 1	Week 1	Week 2	Calum, Josh	High	Tasks
Create Tile Map	Week 2	Week 3	Week 4	Harry, Jonathon	Low	-
Create Chef Class	Week 3	Week 3	Week 5	Harry, Jonathon	High	Chef Class Completion
Finish Chef Movement	Week 3	Week 4	Week 5	Harry, Jonathon	Medium	Completion of Tile map
Create Stations	Week 4	Week 4	Week 5	Harry, Jonathon	Medium	Completion of Tile map
Finish Chef Collision Code	Week 4	Week 5	Week 6	Harry, Jonathon	Medium	completion of tile map generation
Create Station Functionality	Week 5	Week 5	Week 6	Harry, Jonathon	Medium	Creation of Station
Finish Chef-Station Interaction	Week 5	Week 6	Week 6	Harry, Jonathon	Medium	Chef Interaction
Create Level Timer	Week 6	Week 7	Week 7	Harry, Jonathon	Medium	-
Create Customer Orders	Week 7	Week 7	Week 7	Harry, Jonathon	Medum	Rest of Game completion
Create Menus (Opening and Pause)	Week 7	Week 7	Week 7	Harry, Jonathon	Low	Rest of Game completion