D4 Method Selection & Planning

Primary: Samuel Knight, Anexa Bijoy
Secondary: Emma Fuller
Tertiary: Hamza Ahmad, Leo Durnion, Wasif Mustafa

D4a - Software engineering method

Our team adopted a Kanban Agile iterative development cycle by working in weekly cycles. Each week we have tasks created and tracked using GitHub projects, weekly meetings to discuss changes and review progress and make adjustments to priorities and internal deadlines. Following this approach allowed us to remain flexible and ensured efficiency when it came to the development process. We also integrated our Kanban board with Gantt charts to graph tasks over time, allowing for better time management of the project as a whole.
Agile development felt like the best for this project as it allows us to be flexible with our deadlines and team structure. We could change our plans as our risks become more prominent to allow for a prompt delivery of the software and documentation.

Outline & Justification of Tools

Tool	**Purpose**	**Justification**	**Alternatives considered**
Git	Version Control	Reliable and supports easy collaboration. It tracks changes, allows us to create and merge branches and gives all team members a copy of the project. Keeps detailed history to help with fixing mistakes and manage code efficiently. Several team members also already had experience with Git reducing time spent learning new tools.	BitBucket
PlantUML	Diagram Creation	Fast and easy tool to create detailed diagrams for task breakdown. Not too difficult to learn and integrate diagrams in Google Docs/GitHub. The text based nature of it makes version control and standardisation much easier than a tool like diagrams.net.	diagrams.net
libGDX	Game Development Framework	Compatible with 2D Java game development and can be deployed to various operating systems. Has built-in tools and documentation to help with quick game implementation. jMonkeyEngine's lack of documentation was a key factor that made us choose libGDX instead.	jMonkeyEngine No game engine
GSuite	Documentation	Multi-user editing and automatic saving helps with quick documentation and collaboration. Ease of use and familiarity took priority while considering other alternatives like Office and Git-Controlled Markdown.	Microsoft Office Git-Controlled Markdown

IntelliJ IDEA	IDE for Java Development	Provides support for Java development through its debugging tools making testing and writing code easier. Integrated frameworks like LibGDX, while using VS code would require extensions.	VS Code
Netlify	Website Hosting Provider	Quick and easy, no requirement for additional setup or complicated configuration. GitHub Pages or similar providers would work	GitHub Pages
Microsoft whiteboard	Brainstorming ideas	Collaborative space with an infinite canvas and real-time editing for visual brainstorming between group members. Easy way to display and break down individual ideas. Considered canva's whiteboard but we had previous experience with microsoft whiteboard.	Canva
VS Code	IDE for Web Development	VS Code is a familiar and user-friendly IDE that supports key web technologies like HTML, CSS and JavaScript, and provides built-in debugging tools to streamline the web development process. Unlike Notepad++, VS Code integrates Git and extensions, which is beneficial to us during development.	Notepad++
Discord	Communication	Discord is an efficient communication method as it's easy to use and provides organisation through the use of separate channels. WhatsApp and Slack, but Discord combined features of both.	WhatsApp Slack
Piskel	Character Design and Pixel Graphic Design	Allows us to create characters by using pixel art so we can make them suitable for our game. The website is intuitive to use and free to use so ideal for this project.	Asprite Pixel Art Make DinoPixel
Paint.net	Room Design and Bitmap Graphic Design	Free and incredibly simple bitmap image editor for creating a wide variety of assets. Particularly useful for room design.	Krita Photoshop MyPaint
Pixabay	Background music	Provides copyright free music that we can use in our game without a license.	Bensound

D4b - Outline of Team Organisation

We organised our team by assigning primary and secondary leaders for each deliverable. This meant that each of us were part of multiple sub-teams. We ensured that each team member was assigned tasks based on their strengths to allow each sub-team to be as effective as possible. This approach ensured that each deliverable benefited from both individual and collaborative engagement to tackle each one to the highest standard.

The primary(s) is in charge of the main part of that deliverable and in delegating tasks to the secondaries to ensure internal deadlines can be met by multiple people working in parallel. The secondary members on each team allowed us to internally review each other's work, provide assistance and feedback easily when needed to reduce the chance of errors and ensure high quality work. This set up allows each member to clearly understand their role and responsibilities leading to seamless documentation and development.

As a team, to stay organised, we wanted to use a **Kanban**-style tool for task tracking, as well as **Gantt** charts for managing time constraints. This allows our sub-teams to communicate with each other and for us to know where the other teams are currently up to. We considered two main options for managing this:

1. Trello for Kanban Boards, PlantUML Gantt Charts
2. GitHub Projects, with GitHub Issues as our equivalent of “Tasks”

Trello Kanban Boards were a tool which we thought was well-suited for the project due to the versatility it would give us by assigning tasks, setting due dates, and creating automations.

PlantUML Gantt charts were also a consideration because of the standardisation of UML. They would facilitate version controlling our Gantt charts, whilst keeping track of tasks within the same *Google Docs* ecosystem used for writing up most deliverables.

We came to the conclusion that GitHub Projects, whilst more complicated to learn at first, integrates both Kanban and UML into one place. This would save time over option 1 since tasks are shared between charts automatically as Issues.

The benefits of using one tool instead of two made the decision to use GitHub Projects a quick one, particularly since there we would not be missing out on any of Trello’s or PlantUMLs benefits by doing this. GitHub already tracks changes to issues on each individual task.
This decision also meant that we avoided the need to regularly produce new charts, and instead had a plan that adapted as we made changes.

D4c - Plan

Game Overview

QUACK? will be a top down, room-based game. The player is a student at the University of York who is being chased by the dean, who is accusing them of stealing the Long Boi statue. The player must find and retrieve the missing *Long Boi* statue somewhere on East Campus in order to get past the dean, who won't let you leave campus until the statue is returned.

To find the statue the player will have to search the University campus, triggering various events which may help or hinder them.

The player will move around the screen whilst the camera remains stationary, but they will be able to move orthogonally between rooms into other adjacent rooms (if they exist). East Campus will be approximated with an intertwining grid of rooms in a maze-like structure, whilst still vaguely approximating the University's campus.

The player will have energy which can be used to run faster, which benefits the player in achieving their goal of finding the statue before the timer runs out. Players may have their energy reduced by the University's geese attacking players who get too close to them. Energy will naturally regenerate, but if it ever hits 0 they will be slowed down until it has fully recovered.

The events which the player may trigger are described in internal deliverable d10.

Key Tasks

Each subdeliverable is a "key task" which has an individual responsible for it, as well as sub-tasks which break down the problem further. Some details of each of these subdeliverables are included below, although full details are on the GitHub Project, available here. This includes dependencies and priorities for tasks.

We also have catalogued snapshots of the state of the GitHub Project (both Kanban and Gantt charts) on the website. These snapshots are available here. The plan evolved week by week in each of our meetings, these changes were reflected in the issues on GitHub and through informal discussion on Discord. For example, we initially planned on implementing the Map screen as our positive event for Assessment 1, but later decided that we would complete the Vending Machine event instead, as this would be easier to implement with the tight time constraints.

Code	Name	Start Date	End Date	Assignee(s)
D1a	Links to Deliverables	Nov 1st	Nov 7th	Leo
D1b	Links to GitHub Repositories	Oct 17th	Oct 22nd	Sam
D1c	Link to Executable JAR	Nov 2nd	Nov 7th	Hamza
D1d	Interim UML Diagrams	Oct 31th	Oct 5th	Sam
D1e	Weekly Snapshots of Plan	Oct 8th	Nov 7th	Sam
D2a	Introduction	Oct 8th	Oct 22nd	Wasif Emma
D2b	Statement of Requirements	Oct 10th	Oct 22nd	Emma Leo
D3a	Statement on Tools & Languages	Nov 4th	Nov 7th	Wasif Leo
D3b	Structural UML Diagrams	Oct 15th	Nov 4th	Hamza
D3c	Behavioural UML Diagrams	Oct 15th	Nov 4th	Leo
D4a	Outline & Justification of Tools	Oct 3rd	Oct 22nd	Anexa Sam
D4b	Outline of Team Organisation	Oct 1st	Oct 22nd	Sam Anexa
D4c	Plan	Oct 6th	Nov 4th	Sam
D5a	Risk Management Process	Oct 5th	Oct 20th	Leo
D5b	Risk Register	Oct 3rd	Oct 20th	Leo Emma
D6a	Documented Code	Oct 17th	Nov 7th	All
D6b	3rd Party Libraries & Assets	Nov 3rd	Nov 6th	Anexa Emma