FIT3077 Sprint 1

Table of Contents

Team Information Document

Team Name: SANdY (d for Development)

Team Membership

	Contact Details	Technical and Professional Strengths	Fun Fact
Selsa	Phone: 0403641670 Email: sson0024@student.monash.edu	Python Java Javascript HTML/CSS Communication Collaboration	I have lived in 3 countries
Annie	Phone: 0406034601 Email: ahoo0034@student.monash.edu	Python Java Javascript HTML/CSS Adaptable, collaboration	I can crochet and bake
Navya	Phone: 0452625236 Email: nbal0016@student.monash.edu	Python, Java, JavaScript, HTML/CSS Active listening, problem solving	
Yokabit	Phone: 0469290048 Email: yfes0001@student.monash.edu	Python, Java, CSS Communication, Creativity	I like to swim

Team Schedule

The team’s regular meeting schedule consists of meetings on Monday at 3:30pm and Friday at 9:30am every week. During these meetings, we aim to discuss project progress, address any challenges and plan upcoming tasks for the specific sprint. In terms of workload distribution and management within the team, we have implemented a system of assigning story points to each task and allowing team members to self-assign tasks in a fair manner based on its story point. This approach ensures that everyone has a clear understanding of their responsibilities and allows for flexibility in task allocation based on individual expertise.

Technology Stack and Justification

Programming language: Java

Our team has experience with Java, making it a natural choice for game development. Java is widely used and provides features for building scalable applications. We aim to utilise the Java Development Kit (JDK) due to the necessary tools and libraries it provides. We also did consider using Python as our programming language as we all were familiar with using classes in Python but we thought creating the UI would be difficult with Python as none of us have created a system like this using Java before, hence we decided to proceed with Java as our programming language.

GUI: JavaFX

JavaFX is a recent, but versatile GUI framework that integrates easily with Java. It provides a range of UI controls, allowing us to create visually appealing and interactive user interfaces. We plan to leverage tools such as ControlsFX for additional UI components and ValidatorFX for input validation, enhancing the usability of our Web App. After considering Swing as an alternative option, we decided to implement FX instead as it is a more modern version with better features.

IDE: IntelliJ

IntelliJ IDEA is a great IDE to support Java game development and all members of the team have used it previously. It has a wide range of features, including code refactoring, debugging tools and can be used in conjunction with Git, which will streamline our development process and enhance productivity. Our team did consider using VSCode as our IDE as it is one we all have used in the past and found it quite useful, but we decided with IntelliJ because it was specific for Java meaning it has in-built Java compilers, easing our coding experience.

Version control: Git

Git provides the opportunity for the team to collaborate, while efficiently managing code changes and tracking project history and progress.

Testing: JUnit

JUnit is a testing framework for Java that will allow the team to write and execute unit tests effectively. By incorporating JUnit into our development process, we can ensure the reliability and quality of our codebase through automated testing.

Documentation: Javadoc

Javadoc is a documentation generation tool for Java code that creates comprehensive and well-structured documentation. The team has used Javadoc before and leveraging it will enhance code readability and facilitate future redevelopments.

User Stories (with Acceptance Criteria)

Epic 1: Initialisation and Game Setup:

1. As a player, I want to be informed about the game rules and mechanics before starting the game so I would understand how to play the game.
    - Have a confirmed set of rules for the game
    - Split the game rules into small sections so the it is easy to understand
    - Display the rules before the game starts
2. As the game developer, I want to set up the game board so that there is a game that can be played.
    - Have a generator to form the board
    - The game board is displayed in the game
    - Validate that each player is assigned a designated starting cave.
3. As a player, I want to start the game with the dragon in the designated cave, so that I have a clear starting point for the race around the volcano.
    - Verify that the dragon is correctly positioned at the designated starting cave for each player.
    - Display the dragon in the cave on the game board
    - Display the colour of dragon cave
    - Display the colour of the dragon
4. As a game developer i want to be able to have a home page so that players have a Landing Page to go to when they first open the program or after they have finished a game
    - Have the name of the game displayed
    - Have images of the game in the background
    - Have buttons - “Play”, “Options”
    - home page is the first screen shown when the game is opened

5. As a game developer I want the dragon cards to be randomised at the beginning of the game so that there is an arbitrary setup of cards so that the player will have to remember the card layout during the game and not before the game starts.
    - Use a randomiser to randomise the cards
    - There should be no predetermined sequence or pattern to the arrangement of dragon cards at the beginning of each game
    - The randomization algorithm must be consistent across game sessions to ensure fairness and consistency in gameplay.

Epic 2: Gameplay

6. As a player, I want my dragon's movement to be based on memory and strategy so that I can plan my moves ahead of time.
    - The tokens should be left in the same place for the duration of the game so that players can apply memory and strategy to win the game
7. As a player, I want to be able to uncover dragon cards and move my dragon according to the quantity shown so that I can advance around the volcano.
    - Players should be able to uncover and recover dragon cards when it's their turn throughout the game
    - The quantity of token on the card should determine the number of spaces the player can move their dragon
    - The token cards are the only way to move the dragon around the volcano
8. As a player, I want the game to have dynamic elements, including challenging dragon cards, so that each game is unpredictable.
    - The game should have a variety of dragon cards to make it challenging and unpredictable
    - Dragon cards should be shuffled after each game to make it fair
9. As a player, I want the interface to be visually appealing and clear so that the gameplay is easy to understand and follow.
    - The graphics and design elements should be clear for all players to understand
    - Important information, such as dragon movement options and the uncovering of dragon cards, should be displayed
    - The interface should be easy to navigate and minimise confusion for players during gameplay

Epic 3: Characters

10. As a player, I want to be able to choose my dragon so that I can identify where i am on the game board
    - Players should be able to select any out of the 4 dragons available given that it is not taken by another player.
    - Players should be able to identify their dragon on the board at all times of the game.
11. As a player, i want to be able to see where my dragon is positioned on the game board so that i know how close i am to winning
    - Players should be able to identify their dragon on the board at all times of the game

    ○

Epic 4: Winning Scenarios

12. As a player, I want the game to end when a player’s dragon travels around the volcano and reaches their cave again so that I can either stop playing the game or start a new game.
    - When a player's dragon successfully travels around the volcano and reaches their cave again, the game should recognize this as the winning condition.
    - Ensure that the game provides clear feedback to the player when the winning condition is achieved, such as displaying a victory animation
13. As a player, I want a winning page to appear so that I know that a player has won the game.
    - The winning page should be displayed automatically when a player has won
    - The winning page should clearly indicate that a player has won the game
14. As a player, i want the winning page to have two button clickable buttons so that i am able to click on the button to be redirected to another page
    - The winning page must contain two distinct clickable buttons.
    - Provide visual feedback (e.g., button highlighting or animation) to confirm button selection when clicked.
    - Both buttons should be prominently displayed on the winning page, ensuring they are easily visible and accessible to the player
15. As a player i want the winning page buttons to say “Play Again” and “Home” so that i can select either buttons to go back to the home page or i can play a new game again
    - The button to play again must be labelled as "Play Again".
    - The button to return to the home page must be labelled as "Home".
    - Both buttons ("Play Again" and "Home") should be clearly visible and accessible on the winning page without the need for scrolling.
    - Clicking on the "Play Again" button should initiate the start of a new game with the same game settings as the previous one.
    - Clicking on the "Home" button should navigate the player back to the homepage of the game.
16. As a player i want the winning page to have the name of the player that won so i know who has won the game
    - Verify that the winning page prominently displays the name of the player who has won the game.
    - Confirm that the displayed player name is accurate and corresponds to the actual winner of the game.
    - Ensure that the player's name is clearly visible and legible, using appropriate font size, colour, and placement on the winning page.
    - Validate that the player's name is dynamically updated based on the winner of each game, reflecting the correct player's name each time a game is won.
    - Test the winning page with multiple players winning games to ensure that the correct player's name is displayed each time.
17. As a player i want the game board to be blurred so that the winning page is my main focal point
    - Verify screen is blurred

    - the blur effect is applied uniformly across the entire game board, making it visually distinct from the winning page.
    - Colours of the board is dull
    - Winning page colours are brighter
    - Confirm that the blur effect does not affect the visibility or functionality of the winning page or any clickable buttons on it.
    - Validate that the blur effect is implemented smoothly and does not cause any performance issues or glitches in the game interface.
    - Ensure that the winning page becomes the main focal point of the player's attention, with the blurred game board serving as a background element.

Epic 5: Multiplayer

18. As a player, I want to be able to choose whether I play the game with 2 players or 4 players so that i can play together with my friends
    - Promot should be provided at the beginning of each game.
    - The prompt should ask the players to select the amount of people playing in the game.
    - Amount of players selected should not be less than 2 or greater than 4.
    - The game should enable each player to select a dragon.
19. As a player, I want to be able to name my character so that I know which dragon is mine.
    -

Epic 6: Extensions

- Potion - one of the cards will have a potion symbol on it so when it is revealed, the dragon moves to the next position with the same symbol on it
    - Lucky potion - when the player reveals the lucky position card, their dragon moves to the last position on the game map with the animal on the volcano card they are currently standing on.
    - Jinxed potion - when the player reveals the jinxed position card, their dragon moves to the first position on the game map with the animal on the volcano card they are currently standing on.
- Survivor mode - where the volcano builds up each turn and then eventually erupts which shuffles the chit cards

20. As a player, I want to be able to pick up cards after they have been reshuffled.
    - …
21. As a player, I want to be able to move on the board according to the results of the potion cards.

3. Domain Model Design

Justification for each chosen domain entity and their relationships. (Annie and Navya, half each)

List entities:

- GameBoard
    - Represents the environment of the game, including locations like Volcanoes and Caves
- Player
    - Represents a participants in the game that interacts with the Gameboard
- TokenCard
    - Represents the type of cards that can be flipped in the middle of the board
- Dragon
    - Players will interact with Dragons so that they can move around the board
- Volcano
    - Part of the extension as it represents a dynamic element on the Gameboard
- Cave
    - Represents the starting and ending points for each dragon
- Main
    - Needs to be run for the game to begin
- GameLogic
    - Defines rules and actions of the game, including movement and card interactions
- Action
    - Includes moving back/forward, into/out of cave, flip card action, volcano eruption action, reshuffle card action

The GameBoard entity is the environment of the game

The Player entity

The TokenCard domain entity represents a parent class of the different card options. The TokenCard entity was chosen since it represents the type of cards that can be flipped in the middle of the board. The child classes of the TokenCard will inherit the attributes and methods from the parent class. The relationship between the classes is inheritance, this was chosen because it promotes code reuse, as common features can be defined in the parent class and reused in multiple child classes without duplication as well as minimise code inconsistencies. Furthermore, having child classes enables the team to add new methods into the class that are specific for the certain cards.

The dragon domain entity represents a parent class of the player's characters. The dragon entity was chosen since it represents the in-game creature, which inturn will display where the players are positioned on the board. Further it was crucial that a dragon entity was created as it defines how the players move and react based on the cards that will be chosen by the player. Given that the dragon entity is the parent class, the child classes will inherit the attributes and methods from that class thus the relationship between the dragon entity and the PinkDragon, PurpleDragon, BlueDragon, GreenDragon will be an inheritance which the four colour dragons shall inherit from the main dragon class. The team has decided to separate the different coloured dragons into different child classes since each player will have different positions on the map and the character will be played by different players. Furthermore, the use of inheritance prevents repetitive code from being created which then will reduce the form of errors. Given that codes would not be copied over for each different character this will encourage consistency in the colour dragon code as well as prevent lines of code from being lost or deleted.

Specific choices made while modelling the domain and why? (include alternatives discarded) (Yokabit)

Assumptions (Yokabit)

Justification for Modelling

Inheritance between TokenCard and other cards, such as DragonCard, establishes that it will inherit properties and behaviours from TokenCard, while also having its own unique characteristics. This allows for code reuse and ensures consistency among different types of cards. Aggregation represents that the Cave entity is considered a part of the Gameboard. The GameBoard contains Caves, but the Caves can also exist independently of the Gameboard conceptually. This relationship allows for flexibility and modular design as Caves can be managed and interacted with separately from the Gameboard if needed. Additionally, most relationships are association to capture interactions with other entities without a hierarchy.

Things to check/change:

- Relationships in domain model
- Difference between aggregation and composition
- Player to Gameboard as well?
- Main should be to gameboard
- Add cardinalities and text to each arrow
- Do we need a positions entity?

4. Basic UI Design Draw

Set up (homepage, selecting how many players, character selection for each player, selecting the mode), flipping a dragon card, moving the dragon, leaving the cave, entering the cave, winning

1. Set up (Navya)
    a. Homepage
    b. Selecting how many players
    c. Name players

d. Character selection for each player
e. Selecting the mode

2. In cave - Selsa
3. Flip card - Selsa
4. Step in front of cave - Selsa
5. Move sideways - Yokabit
6. Move backwards - Yokabit
7. Winning - Annie

Ideas:

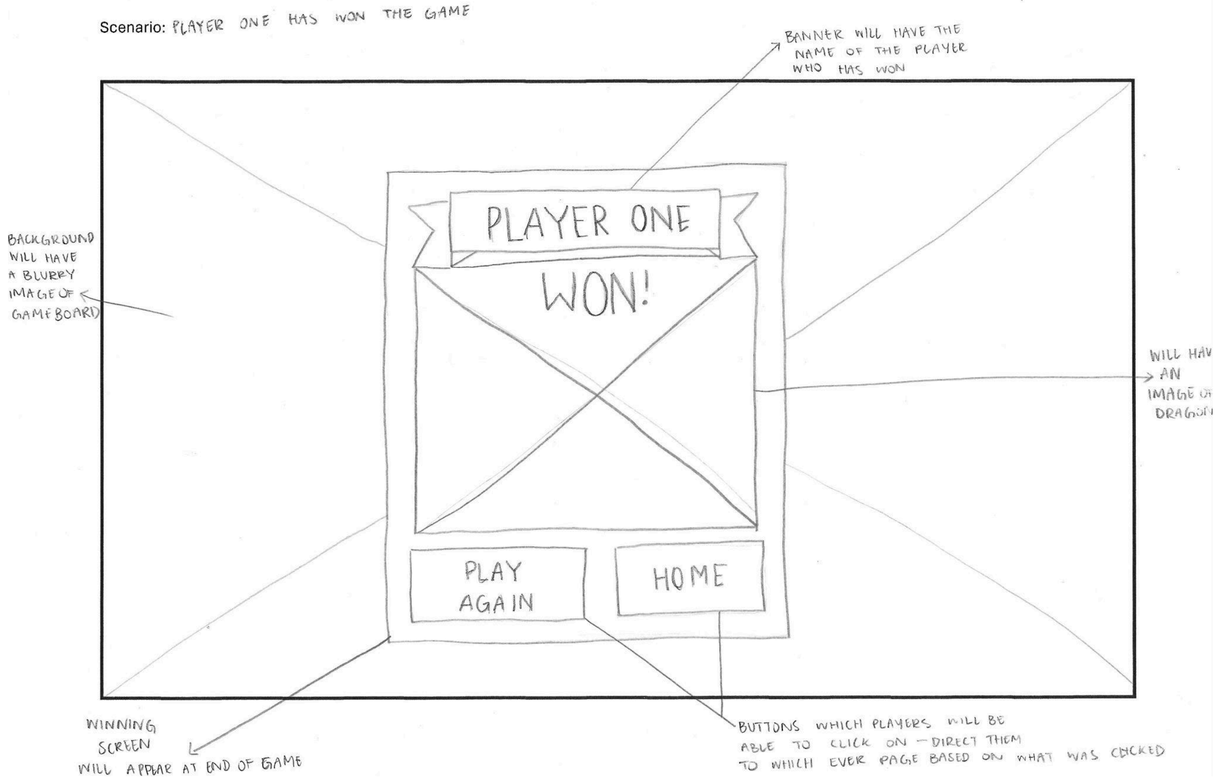
-

-

we recommend that you do some very basic prototyping with your chosen technologies to ensure (i) everybody in the team knows how to use them (not just in theory, also in practice!) and that (ii) you can create an executable - something that will be needed for sprints 2 to 4. Ideally, you test that an executable you have created can actually be run on another computer (that does not have your set-up). You want to get this out of the way so that there will be no surprises for any of the following sprints.

Winning situation:

Scenario: PLAYER ONE HAS WON THE GAME


BANNER WILL HAVE THE NAME OF THE PLAYER WHO HAS WON
BACKGROUND WILL HAVE A BLURRY IMAGE OF GAMEBOARD
PLAYER ONE
WON!
WILL HAV
AN
IMAGE O
DRAGO
PLAY AGAIN
HOME
WINNING SCREEN WILL APPEAR AT END OF GAME
BUTTONS WHICH PLAYERS WILL BE ABLE TO CLICK ON — DIRECT THEM TO WHICH EVER PAGE BASED ON WHAT WAS CLICKED