Sprint 1

Team Information

Team name: MNE Tech ©

Table of Contents

Team Information **1**
Team Membership **3**
Team Schedule **4**
Technology Stack and Justification **5**
Programming Language 5
IDE 6
Java Libraries and APIs 7
User Stories **9**
Basic Architecture **11**
Justification for Change 11
Explanation of Domain Concepts 12
Basic UI Design **14**

Team Membership

Name	Contact Information	Technical and Professional Strengths	Fun fact
Mahesh Suresh	msur0008@student.monash.edu	**Technical skills:** I’ve done an IBL placement where learnt the skills of a Fullstack developer, having worked with both backend and frontend I am comfortable with typical web development using HTML, CSS and jQuery, as well as backend using C# and ASP.NET. **Professional Strengths:** - Agile workflow - Industry environment - Multiple group projects - Able to identify client / management needs and focus on requirements.	I eat malatang I like to dance I love pets I like to swim
Nethara Athukorala	nath0002@student.monash.edu	**Technical skills** - Proficiency in programming languages, such as Python, Java - Web development, including proficiency in HTML and CSS, as well as experience with frameworks such as Django - Mobile app development, with proficiency in Android development tools **Professional strengths** - Time management and organisation skills - Ability to work well in a team and collaborate effectively - Attention to detail and commitment to producing high-quality work	I like to swim

Ethel Lim Jia Yee	elim0050@student.monash.edu	**Experience** I've experienced working for an IT company back in Malaysia which holds a project with a large group of people. I used to work for FrontEnd and QA so I'm highly skilled for creating the UI and testing the program. **Technical skills** - User Interface design - Quality Assurance - Creating Web Application - Advanced in Python, Java , JavaScript , html and css coding languages. - Database Management using MySQL and Firebase **Professional strengths** - High Leadership Skill. Having to be a leader in most projects. - High Time Management Able to prioritise work and keep track of all the project deadlines. - Creativity. Created several UI designs and loved to do so. - High Adaptability.	I like to dance and I love pets.

Team Schedule

Table 1: Team Meeting Schedule

Day	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Time	N/A	N/A	8:00pm-10:30pm	9:00pm-10:30pm	N/A	9:00am-11:00am	N/A

Our team will host meetings three times a week through discord to acknowledge the progress and the time spent on project tasks. Time spent on project tasks is recorded and accumulated manually until the task has been completed. There are no standard working hours for our team but we expect our work to be fully completed at least 2 days before the deadline.

Tasks are allocated randomly and fairly based on the user stories created. The tasks must be completed by each team member within the specified deadline. A team member who runs across a problem is encouraged to let the group know about it using the Discord channel set up for project discussion. Once everyone has finished their own task and in accordance with their individual technical strengths, members can assist one another. Tasks from the prior week will be examined at each weekly meeting so that members who have completed them might be given new ones. As for the individual contribution log, each team member is required to update the status of their work in the wiki contribution page in GitLab in order to keep track of their contribution time for the purpose of recording and reviewing.

All our documentations will be saved in the shared Google drive. GitLab will be primarily used to store all the project code and task allocations of each member. Each member is required to create their own branch with their name in the gitlab repository. Moreover, everyone needs to constantly commit and push their code to their own branch with a meaningful commit message. The weekly meeting will also be used to test the functionality of all the code.

Technology Stack and Justification

Programming Language

The choice of programming language is arguably the most important decision to make because it necessitates careful consideration of the available languages to choose, as to which is the most suitable to be implemented throughout the project. A shortlist of programming languages has been developed that considers all of the languages', their basic characteristics, intended purpose as well as related technologies. Based on these criterias, the following have been shortlisted: Python and Java. The whole team have experience in these two languages and they are found to be highly reputed among developers after a brief search of potential candidates.

Table 2: Comparison between Python and Java

Programming Language	**Python**	**Java**
Language Type	- Interpreted language - High level programming language - Object-oriented programming language	- Compiled language - High level programming language - Object-oriented programming language
Intended Purpose	General purpose language	General purpose language
Compilation Time	Python launches faster and runs slower	Java launches slower and runs faster
Stability	Less stable	Slightly more stable
Speed	Interpreted therefore longer response time	Compiles directly therefore faster response
Syntax	Simple coding syntax	Complex coding syntax
Security	Less secure	More secure
GUI Performance	Unable to build complex GUI	Able to build more complex GUI
Usage	- Web application - Data science - AI/ML projects	- Web application - Game consoles - Mobile applications

Our team has decided to use Java as the main programming language instead of Python for creating the Nine Men's Morris game after comprehensive analysis and detailed discussions. This choice was made after carefully considering the advantages and disadvantages of both Java and Python, particularly with regard to game creation.

One of the primary advantages of Java over Python for game development is its superior performance and speed. Java's ability to compile code allows it to execute game logic much faster than Python, which is interpreted. This performance advantage is especially important for games like Nine Men's Morris, which demand fast-responses and real-time calculations. Another key benefit when creating games is Java's object-oriented programming architecture. It allows developers to create complex game logic and code structures, resulting in more maintainable and extensible codebases (Anderson, 2023). Java's extensive selection of libraries and frameworks can also be used to create unique game engines and toolkits that can be customised to match the particular requirements of the game.

In conclusion, the use of Java as the primary programming language for developing Nine Men's Morris offers significant advantages over Python. Its performance, scalability, and robust support for graphics and multimedia make it an ideal choice for building complex and visually appealing games (Kina, 2023). We are confident that this decision will result in a high-quality, engaging, and enjoyable gaming experience for our users.

IDE

Choosing the right integrated development environment (IDE) is a critical phase in any project. Up until now, we have assessed and contrasted three IDEs - Visual Studio Code (VSC), Eclipse, and Intellij Idea - on different criteria such as cost, performance, language support, usability, and integrated tools.

Table 3: Comparison between Visual Studio Code, Eclipse and Intellij Idea

IDE	**Virtual Studio Code**	**Eclipse**	**Intellij IDEA**
Cost	Free and open-source	Free and open-source	- Offers a free, open-source community edition - Offers a paid Ultimate edition with additional features
Performance	- Fastest among the three - Lightweight and fast performance since it has a built-in Electron framework	- Slowest performance among the three - Can be resource intensive when working with large projects	- Perform code faster than Eclipse - Can be laggy and overly used when working for large codebase projects
Supported Languages	- Support a wide range of languages like Python, Java , C++ , Javascript and more.	- Mainly used for coding Java applications - Support many other languages as well	- Offer strong support on Java, Kotlin, Scala and Groovy - Including other languages from plugins

Ease Of Use	- User friendly interface and is beginner friendly	- Offer a range of customization features to maximise user experience - Not very beginner friendly	- Beginner friendly, providing an intuitive interface that allow user to experience a good time in coding
Integrated Tools	- Offering a wide range of plugins and extensions (for coding, debugging, and testing) - Provides built-in Git integration	- Provides a range of plugins and tools for debugging, version control like git, code analysis and more. - Offer built in web frameworks such as Spring.	- Offers advance features and tools -Built-in tools such as debugging, code analysis, version control like git and subversion, testing framework and more.

Looking into Table 3, it is obvious that the three Integrated Development Environments (IDEs) have different features for different programming languages. As mentioned above, the team has decided on using Java as the main programming language throughout the project. Hence, Visual Studio Code (VSC) will be eliminated from the options since the other two IDEs provide strong support for Java.

In comparison, Eclipse supports a wider range of plugins and tools while Intellij is more user friendly compared to Eclipse. However, the fact that Intellij offers a range of advanced tools and features creates more advantages compared to Eclipse. As this allows a better execution of the code and developers can have a better coding experience.

Additionally, consider that each team member is more familiar with coding Java projects in Intellij. Based on this evaluation, we decided to choose Intellij as the primary IDE to be used throughout the project.

Java Libraries and APIs

In order to build a software project efficiently, developers may need to take in consideration of installing some framework and APIs. Software frameworks are written by a group of professionals and have been maintained properly, which will provide most of the general code or tasks. Thus, it allows the developers to focus more on coding the details or requirements of the software rather than dealing with the lower-level details. Therefore, it greatly saves the developers time and workload , helping them to produce a better software product.

Through discussion and research , there are some famous software frameworks and APIs that are widely used by the developers. For example, Spring Boot , Flask, Ruby On Rails, DJango REST and more (Kaur, 2022). However, since our team chose to program in Java, we only considered the framework for Java coding language.

Table 4: Description of the available options for Framework and APIs

Framework and APIs	Description
Swing	A lightweight GUI toolkit for producing window based applications. It provides a set of components like buttons, labels and text field and also support for event handling and consist of classes that handles the font, colours and icon. It was included in Java standard libraries for a long period and was built on top of the Abstract Windowing Toolkit (Waseem, 2022).
JavaFx	A modern GUI toolkit which contains a set of media and graphic packages for developers to build for a modern, cross-platform desktop and mobile application. JavaFx also makes use of the APIs in Java libraries and supports multimedia, 3D graphics, animation, and a powerful layout system. Moreover, JDK 7 can work for all major desktop platforms. Deploying JavaFx Application can allow multiple modes such as standalone application and web page (Pawlan, 2013).
Maven	A popular building automation tool for Java projects which supports project builds, dependencies and documentation from a central piece of information is known as Project Object Model (POM). It is a software tool hosted by Apache software foundation and created to allow developers to have easier and faster coding experience. It can provide guidelines on the projects and shows out what technical are lacking. It is a Java tool which needs to be installed on an IDE (Contributor, 2019).
JUnit	An open source framework used for executing automated tests. It allows developers to write out all the executable tests for testing the methods created in the class and run the test. It supports not only text-based command lines and also Swing-based graphical test mechanisms. This framework allows the bugs to be detected in the early development stage (Stefan Bechtold, 2022).
libGDX	LibGDX is a cross platform Java game development framework that can work on most of the major desktops. It is beginner friendly, and offers the latest OpenGL graphic technology which also supports audio and graphics in 3D and 2D. It allows games to be developed using multiple platforms with a single codebase and also provides networking capability for building an online leaderboard game (LibGDX, n.d.).

Table 4 was showing the lists of options we have taken into consideration through researching for the framework that works for Java programming languages. The game Nine Men's Morris will be created as a standalone desktop application with a simple and clean user interface.

After further discussion, we came to the conclusion of using JavaFx as the framework to build the software. As JavaFx is easier to learn and provides more traditional UI elements and also consists of API’s and features that are useful to create a more professional look UI.

User Stories

In order to meet the advanced requirement of enabling a single player to play against the computer, with the computer playing a move randomly among all currently valid moves, or based on a selected set of heuristics (advanced requirement C), our team has utilised user story 17. This user story outlines the necessary features and functionality required to support this requirement, and serves as a guide for the implementation of this feature. By implementing user story 17, we can ensure that the advanced requirement is fully met and that the resulting software meets the desired specifications and expectations. All of the following user stories are written such that they follow the INVEST criteria.

Table 5: User Stories For Nine Men's Morris

No.	**User Story**
1	As a game player, I want to be able to enter my name before I start playing so that it is easier to see my own score.
2	As a game player, I want to be able to select my pieces on the board by clicking or tapping on the desired location so that I know which token I have selected.
3	As a game player, I want to be able to move my tokens from one position to another when it's my turn, so that the game can continue.
4	As a game player, I want to be able to arrange my tokens on the game board, so that I can start the game.
5	As a game player, I want to have a mill by forming a straight row of three pieces along one of the board's lines to eliminate one piece of the opponent token.
6	As a game player, I want to win the game by removing the opponent's tokens until there are fewer than three pieces.
7	As a game player, I want to be able to see the remaining pieces for both players so that I can track the progress of the game.
8	As a game player, I want to be able to see the available moves for my piece so that I can plan my strategy.
9	As a game player, I want to be able to resume the game so that I can continue playing the same round.
10	As a game player, I want to be able to resign the game if I feel like I cannot win, so that I do not have to continue playing until the end.
11	As a game rule, each player should be able to start moving their tokens without following the line of the board when they are left with three pieces.

12	As a game rule, players should move their tokens according to the line of the board for each round to ensure no illegal moves were taken.
13	As a game rule, each player should be given 9 pieces of tokens before the start of the game, so that the game can be played fairly.
14	As a game rule, players can start moving the tokens on the board once the 9 pieces of tokens from their side were placed on the board.
15	As a game rule, the player should not be able to take any pieces from a mill of the opponent, so that it allows the game to progress.
16	As a gameboard, I want to ensure that no illegal moves are made so that a fair game can be played.
17	**(Advanced Requirement)** As a game player, I want to be able to play against a computer opponent so that I can play the game against another real player.

Basic Architecture

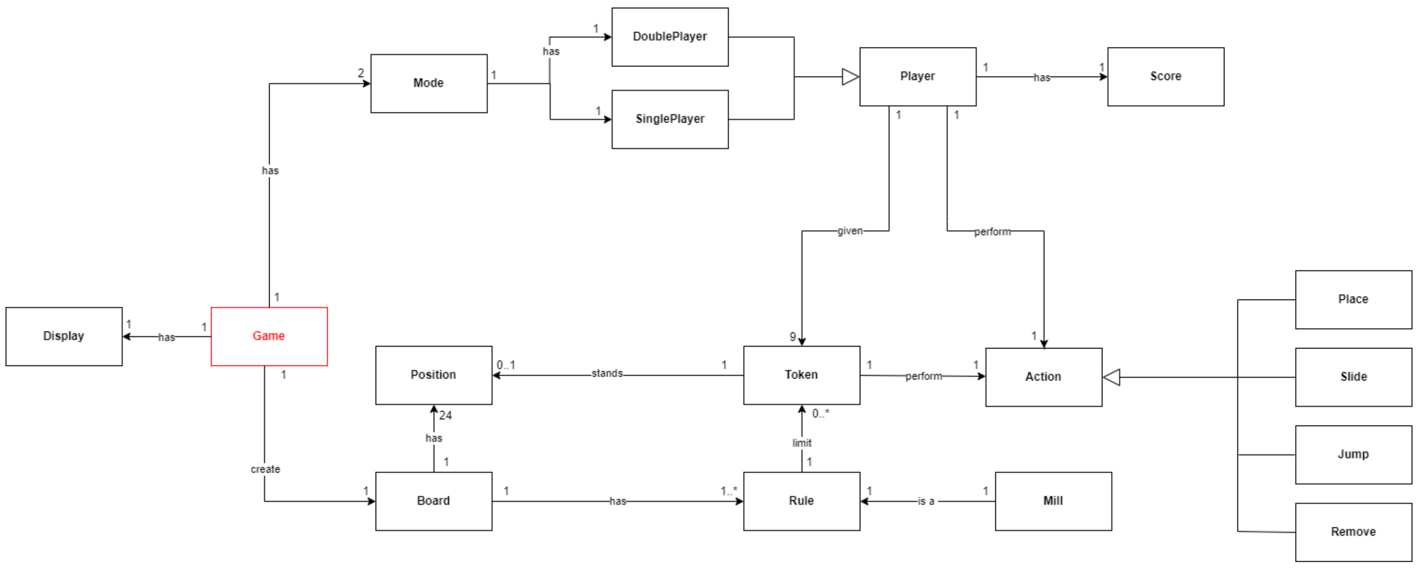


Figure 1: Initial domain model

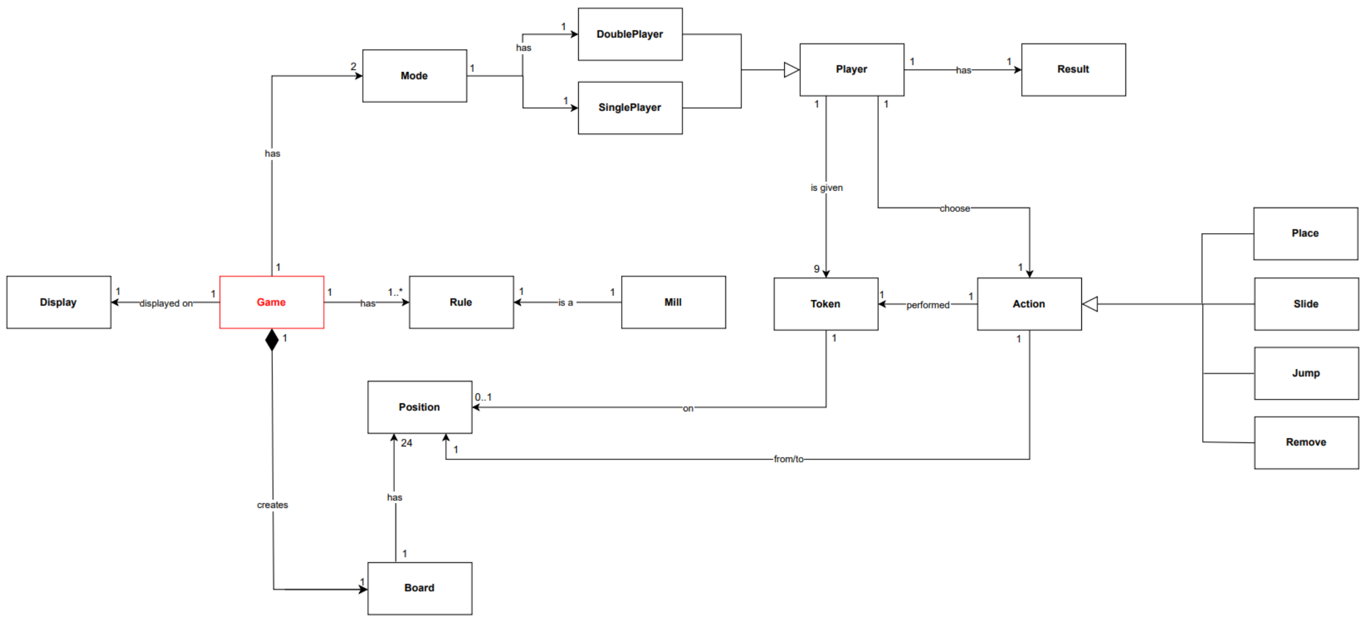


Figure 2 :Final domain model

Justification for Change

We changed our initial domain model because we realised that it didn't make sense to include the rules within the board as that would imply that different boards would have different sets of rules. Furthermore when following the domain model we realised that it would also make more sense for action to know position as when we perform actions they will affect the position of the tokens. Also changed some general explanations of relations, such as action being performed on a token.
Hence, Figure 2 is representing our final domain model structure.

Explanation of Domain Concepts

Display

Display class represents the final display of the Nine Men's Morris game which allows interacting with the user. This class will therefore have a 1 - 1 relationship with the Game class.

Game

The game of Nine Men's Morris has three main components: the board, the rules, and the game mode. Our advanced requirement is to have a player versus computer mode, which will include both single player and two-player games. Since the game can only be played on one board at a time, the board component is one-to-one. The rule component is one-to-many since there can be multiple rules for one game.

Board

The primary purpose of the Board class in the Nine Men's Morris game is to keep track of the position of the tokens. It is designed to manage 24 positions where the tokens might be positioned during the game. So, the Board is associated 1 to 24 of Position.

Position

The Position class represents the 24 lines of intersection of the board. It determines whether a token is present or not at any moment and is utilised by the board to monitor the 18 pieces placed on the board. Instead of using the board to track the piece's position, we chose to use the piece-to-position approach. This enables us to monitor whether a piece exists before relocating it on the board.

Mode

Mode has a 1 - 1 association relationship to both the Single Player and Double Player domains. This means that the game provides two options for the user to choose from: single player mode or double player mode. When the user selects the single player mode, they will compete against the computer, and when they choose the double player mode, two players can compete against each other.

Player

To avoid violating the Open-Closed Principle and the Liskov Substitution Principle, the Player class was designed to have a hierarchical structure with SinglePlayer and DoublePlayer classes as its subclasses. This allows for efficient code management and prevents duplication. The players are distributed with 9 tokens each at the beginning of the game, which creates a 1-to-9 association relationship between the Player and the Token domain. Player has a 1 to 1 relationship with the Result class, since each player will only have one result of either win or lose. Additionally, Player has a 1 to 1 relationship with the Action class , as the player needs to choose an action to move the token.

Result

The Result class in the domain model of the Nine Men's Morris game can be used to keep track of the result of each player during the game. The Player class has a one-to-one relationship with the Result class to show if the player wins or loses.

SinglePlayer

Single Player is created to allow the user to play the game against the computer. This class is derived from the Player class and shares a generalisation relationship with it. The Single Player class inherits the 9 tokens assigned to the Player class, which reduces the dependency between the domains and enhances efficiency in accessing each domain.

DoublePlayer
Double Player is created to allow multiple players to play the game. Similar to the Single Player class, it also has a generalisation relationship with the Player class.

Token
Token class is created to represent each token on the gameboard. It has a one-to-zero-or-one association with Position, indicating that each token can either be placed on a position of the board or not.

Rule
The Rule class plays a crucial role in the game of Nine Men's Morris by setting up the game rules and ensuring that no illegal moves are made by the players. To achieve this, it is associated with the Game class in a 1 to 1..* relationship, as the rules must be applied to every token on the game. It also has a one-to-one relationship with the Mill class.

Action
The Action class is the top-level class that encompasses all the possible actions that can take place in the game. As a parent class, all the actions in the game have a generalisation relationship with the Action class. It has a one-to-one association relationship with the Token class, meaning that each token needs to perform an action among the available moves chosen by the Player. Hence, Action has a 1 - 1 association relationship with the Position class, since each action will lead the token to a different position on the board.

Mill
Mill class is a subclass of the Action class and has a generalisation relationship between the two classes. The purpose of the Mill class is to represent the action of forming a "mill" on the game board, which happens when a player places three of their own tokens in a horizontal or vertical line.

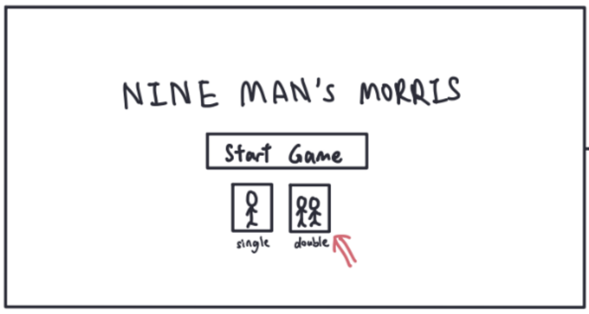
Move
Move class is a subclass of the Action class and has a generalisation relationship between the two classes. The Move class represents the action, when a player “moves” their tokens within the game board.

Place
Place class is a subclass of the Action class and has a generalisation relationship between the two classes. The Place class represents the action, when a token is “placed” on a particular place within the game board.

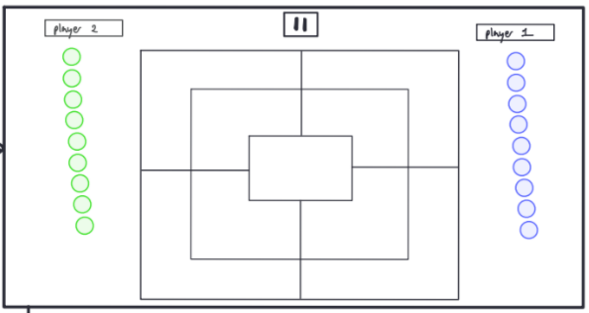
Jump
Jump class is a subclass of the Action class and has a generalisation relationship between the two classes. The Jump class represents the action of jumping a token on the game board. This action can only be performed by a player with three or fewer tokens remaining, and allows the token to move from any point to any other point on the board, regardless of lines or adjacency.

Basic UI Design

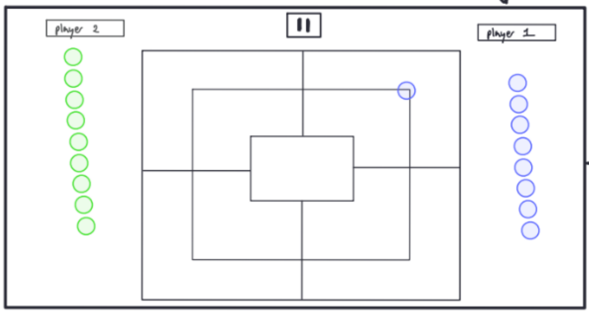
selecting double player



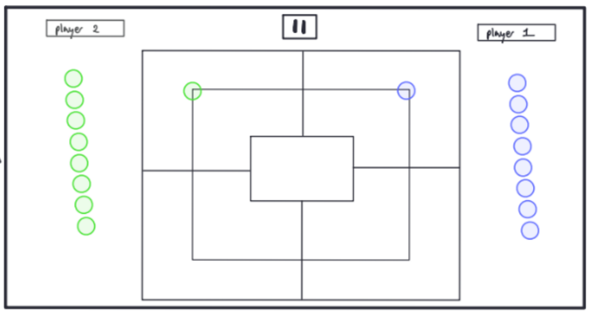
Initial game user-face



Initial status of board game for double player

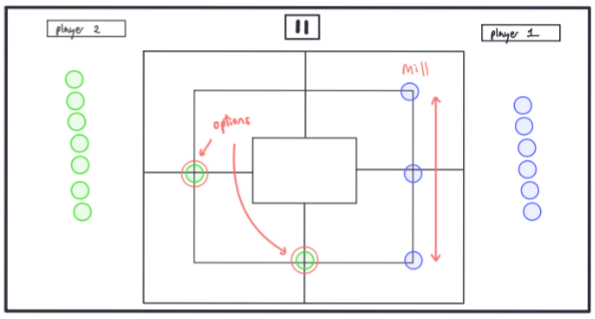


Player 1 start placing a token to the game board

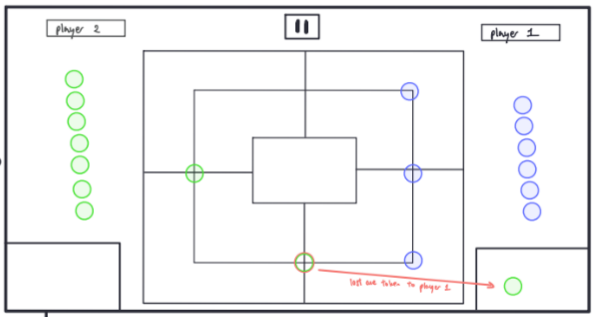


Player 2 placing a token to the game board

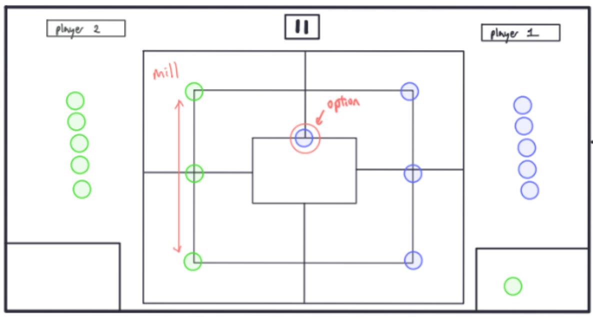
player has a mill



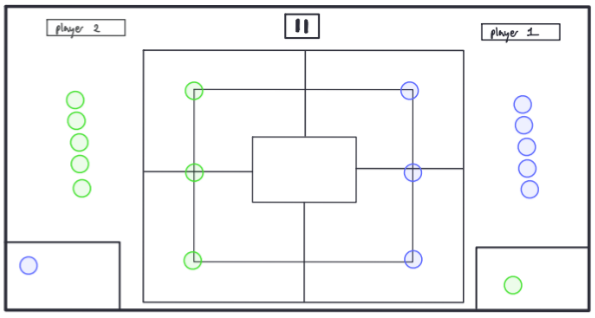
player 1 places three tokens in a row and has a mill



player 1 took a token from player 2

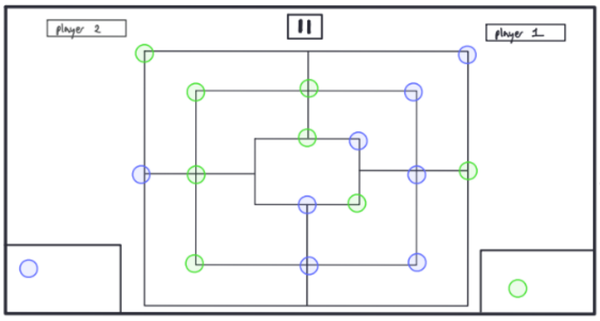


player 2 has a mill can take away a piece from player 1

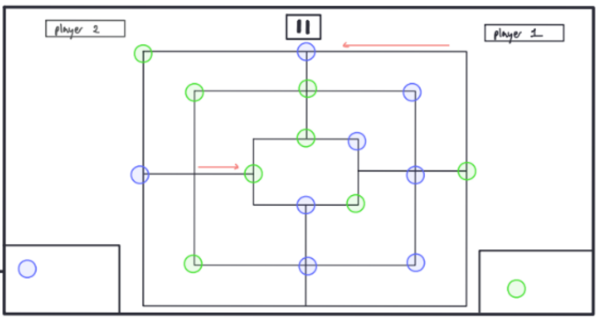


player 2 take away a piece from player 1 that does not has a mill

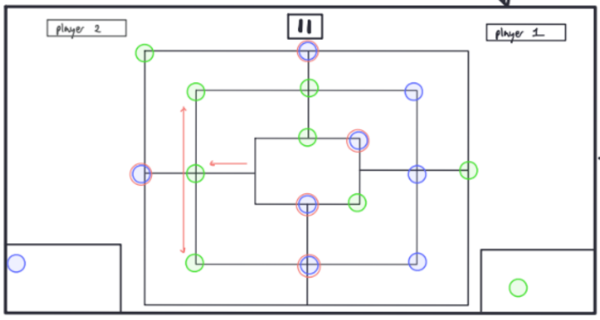
All tokens placed on board



All the tokens from both player were placed on the board

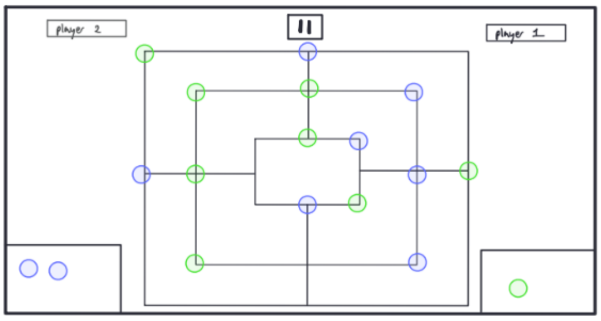


Both players start sliding their tokens along the line of the board



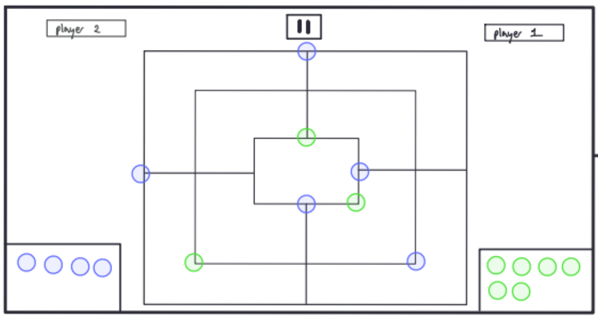
Player 2 has a mill and was prompted with options

to take away a piece from player 1

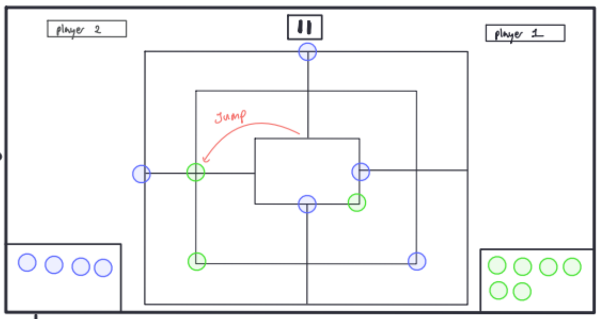


player 2 take away a piece from player 1

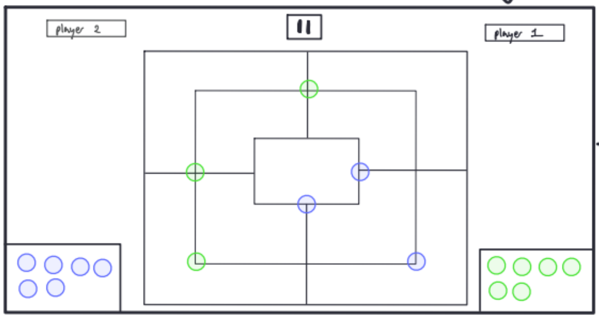
player left 3 tokens on board



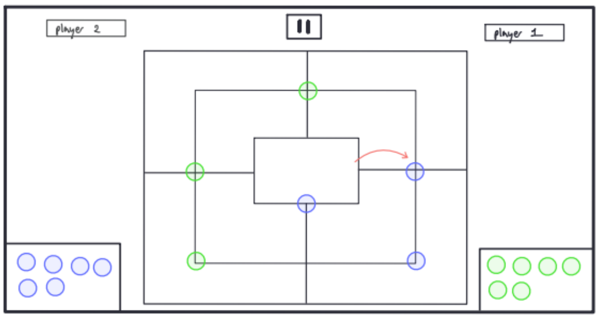
Player 2 left three tokens on the board



Player 2 start jump around with the tokens

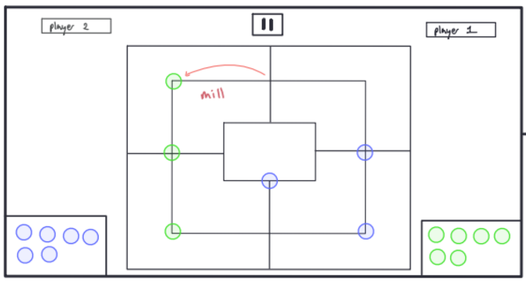


Both player were left with 3 tokens

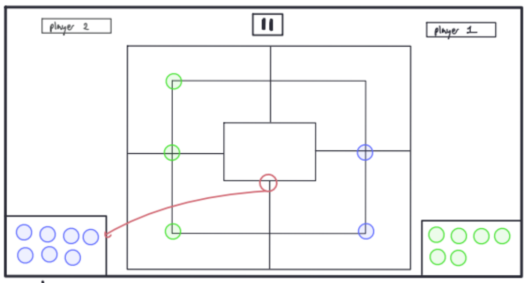


Both player can start jumping around with their tokens.

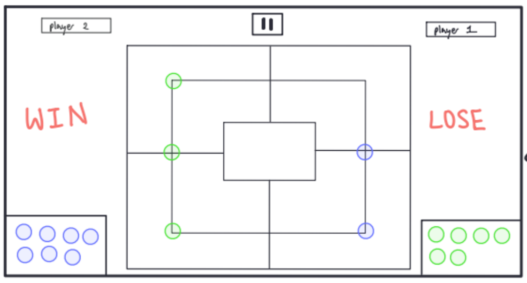
End of game



Player 2 has a mill



Player 2 took one token from player 1

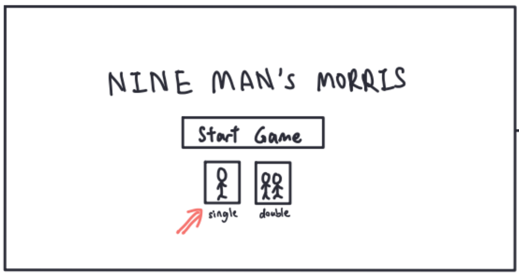


Player 1 was left with 2 tokens on the board and player 2 win the game

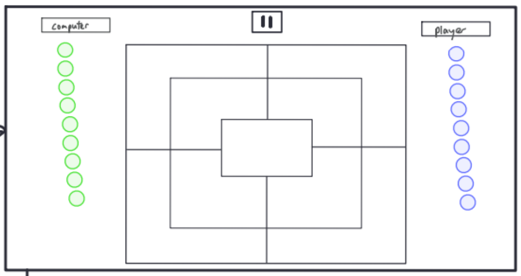
UI Design for Advance Requirement C

A single player may play against the computer, where the computer will randomly play a move among all of the currently valid moves for the computer, or any other set of heuristics of your choice.

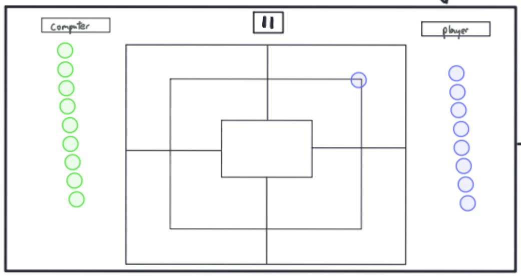
selecting single player (Advance Requirement)



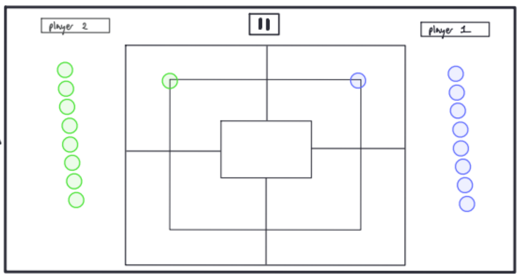
Initial game user-face



Initial status of board game for single player

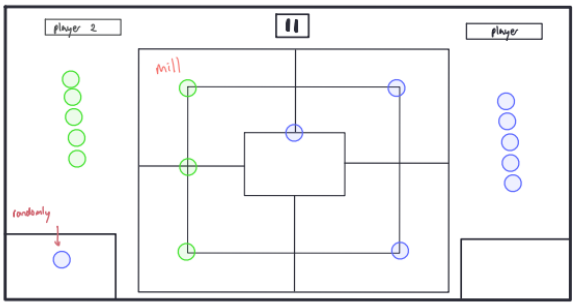


Player start placing a token to the game board



Computer placing the token randomly

Computer has a mill



Computer has a mill randomly take away a piece from player

References List

Kina, J. (2023, January 02). Python vs. Java: Comparing the pros, cons, and use cases. Retrieved March 31, 2023, from https://www.stxnext.com/blog/python-vs-java-comparison/

Anderson, S. (2023, March 28). Python vs Java: A deep dive comparison. Retrieved March 31, 2023, from https://streamsets.com/blog/python-vs-java-comparison/

Team, D. (2022, April 18). Pros and cons of java: Advantages and disadvantages of Java. Retrieved March 31, 2023, from https://data-flair.training/blogs/pros-and-cons-of-java/

Waseem, M. (2022, November 29). Swing in java: Creating GUI using Java Swing. Retrieved April 1, 2023, from https://www.edureka.co/blog/java-swing/

Pawlan, M. (2013, April). Java platform, Standard Edition (Java SE) 8. Retrieved April 1, 2023, from https://docs.oracle.com/javase/8/javase-clienttechnologies.htm

Contributor, T. (2019, August 19). What is Maven? A definition from whatis.com. Retrieved April 1, 2023, from https://www.techtarget.com/searchapparchitecture/definition/Maven

Stefan Bechtold, S. (2022, January 10). JUnit 5 User Guide. Retrieved April 1, 2023, from https://junit.org/junit5/docs/current/user-guide/

libGDX. (n.d.). LibGDX. Retrieved April 1, 2023, from https://libgdx.com/

Kaur, P. (2022, December 2). *10 Most Popular Frameworks For Building RESTful APIs*. 10 Most Popular Frameworks for Building RESTful APIs | Moesif Blog. Retrieved from: https://www.moesif.com/blog/api-product-management/api-analytics/10-Most-Popular-Frameworks-For-Building-RESTful-APIs/