Wordgame – Hangman

[1]Ashutosh Raval, [2]Kruthi Shankar Hegde, [3] Rohan Vasudev Ginde

[1]Software Engineering Systems, Northeastern University
[2]Software Engineering Systems, Northeastern University
[3]Software Engineering Systems, Northeastern University

{raval.as, hedge.kru, ginde.r }@northeastern.edu

***Abstract*— The Wordgame - Hangman project is a fascinating endeavor that aims to create an enjoyable and engaging word game while also testing players' vocabulary and spelling abilities. The game is designed to challenge players to guess a secret word by suggesting letters from the English alphabet, and it features a unique twist - a hangman figure that is gradually assembled with each incorrect guess. In this project, players will have the opportunity to put their word-solving skills to the test and have fun while doing so! The game starts with the player being presented with a set of English alphabets, and the player's first task is to choose a letter to guess. The player's guess is then compared to the letters in the secret word, and if the guessed letter is correct, it is displayed in the correct position of the word. This provides players with clues about the word and helps them narrow down their guesses. However, if the guessed letter is incorrect, a part of the hangman figure is assembled. The hangman figure typically consists of a stick figure being hanged, with different body parts added for each incorrect guess. The game continues until the player correctly guesses the word or until the hangman figure is fully assembled, resulting in a loss. One of the key challenges in developing the Hangman game is creating a diverse and engaging set of words for players to guess. The game can be designed to have different levels of difficulty, with words ranging from easy to challenging. Easy words could include common everyday vocabulary, while challenging words could include complex technical terms or rare words from various domains such as science, literature, or history. The selection of words can be curated to ensure that players are exposed to a wide range of vocabulary, helping them improve their language skills while having fun. Another important aspect of the Hangman game is the user interface and experience. The game should be visually appealing, with clear and intuitive instructions for the players. The game can be designed with appealing graphics, animations, and sound effects to enhance the overall experience. The user interface should allow players to easily select letters from the English alphabet, keep track of their guesses, and view the progress of the hangman figure being assembled.**

Keywords*—*Hangman game, Word game, Algorithm

I. Problem Description

One common issue among language learners is the lack of engaging and interactive language learning tools that can keep them motivated and interested in using their language skills. This can often result in poor language development and discouragement. To address this issue, our team aims to create an exciting and instructive language learning experience through our JavaFX project, "Wordgame - Hangman".

Our project's goal is to develop a word game that is not only entertaining but also challenging, putting players' spelling and vocabulary skills to the test. The game will require players to choose the correct word from a list of English alphabets, and the objective is to keep players engaged while they improve their language skills.

Using JavaFX, we plan to create a visually appealing and user-friendly interface for the game. Players will be able to select different difficulty levels, such as easy, medium, and hard, based on their language proficiency level. The game will provide hints and clues to help players guess the correct word, and they can also track their progress and achievements to stay motivated.

To make the game more interactive, we plan to incorporate features such as time limits, high scores, and leader boards, to encourage healthy competition among players. We also plan to include a variety of word categories, such as common vocabulary, idioms, and phrases, to cater to different interests and language learning goals.

In addition to the gameplay, we aim to make the game a valuable learning tool by providing explanations and definitions of the words used in the game. This will help players improve their understanding of word meanings and usage in context, enhancing their overall language proficiency.

We will also focus on creating an immersive and engaging user experience by incorporating visually appealing graphics, sound effects, and animations. The game will be designed to be intuitive and easy to navigate, ensuring that players can focus on learning and having fun.

Overall, our "Wordgame - Hangman" project aims to create a rewarding and enjoyable language learning experience that will keep language learners motivated and interested as they hone their language skills. By combining entertainment with education, we hope to provide a valuable tool for language learners to improve their vocabulary, spelling, and language proficiency in an engaging and interactive way

II. Analysis (Related Work)

After conducting analysis on earlier studies, these are some of the drawbacks of current technologies and solutions that we discovered while working on this project. Handling user input: One of the main challenges in creating a Hangman word game is handling user input. It is important for developers to ensure that the game can handle different types of user input, including uppercase and lowercase letters, special characters, and spaces. Failure to handle different types of input can lead to a frustrating user experience, as users may be unable to input their desired guesses. One way to handle user input is to use regular expressions to validate user input and ensure that it conforms to the game's requirements. This can help to prevent errors and ensure that the game operates smoothly.

Game logic bugs: Another challenge in creating a Hangman word game is ensuring that the game logic is implemented correctly. Careful testing and debugging is required to ensure that the game logic works as intended and that there are no bugs or errors in the game. Developers must ensure that the game's logic is accurate and that the game behaves as expected in different scenarios. This requires thorough testing and a good understanding of the game's rules and mechanics.

User interface design issues: User interface design is an important aspect of creating any game, and Hangman is no exception. The user interface must be designed in a way that is user-friendly and easy to understand, while also being visually appealing. JavaFx scene builder eases the UI development part, and it is simple to design the GUI of the application, with very little code. However, developers must still take care to ensure that the user interface is designed in a way that meets the needs of the game's players. This may involve conducting user testing to gather feedback on the user interface and making changes based on this feedback.

Performance issue: Another important consideration when creating a Hangman word game is performance. The game must be able to run smoothly and without lag, even on older or slower devices. Concepts such as inheritance and polymorphism can help to remove redundant code, which can improve the game's performance. Developers should also consider other techniques such as lazy loading, which can help to improve the game's loading times and reduce memory usage.

Multi-Threading: Multi-threading is another technique that can be used to improve the performance of a Hangman word game. By using multi-threading, developers can separate the game's logic from the user interface, allowing the game to run more smoothly and efficiently. For example, multi-threading can be used to randomly select words from a list of existing words, which can make the game more interesting and dynamic.

Memory leaks: Memory leaks can be a common problem in Java programs, and Hangman is no exception. Developers must ensure that the game's code is designed in a way that avoids circular references and other issues that can lead to memory leaks. This may involve using tools such as the Java Memory Analyzer to identify and resolve memory leaks and other memory-related issues. By addressing memory leaks, developers can ensure that the game runs smoothly and without issues, even over extended periods of time.

In summary, creating a Hangman word game can present a number of challenges, including handling user input, ensuring accurate game logic, designing a user-friendly interface, addressing performance issues, using multi-threading to improve efficiency, and preventing memory leaks. By taking these challenges into account and implementing appropriate solutions, we have created a Hangman word game that is engaging, enjoyable, and bug-free.

III. System Design

After the first iteration of designing the UML, we started working on the flow of the application. It mainly comprises of the flow from user perspective, that is when a user lands on the home screen of our application what all options and actions user can perform to get the results and make the most of our application. We focused on keeping the flow as simple as possible so that even a layman can use the application easily.

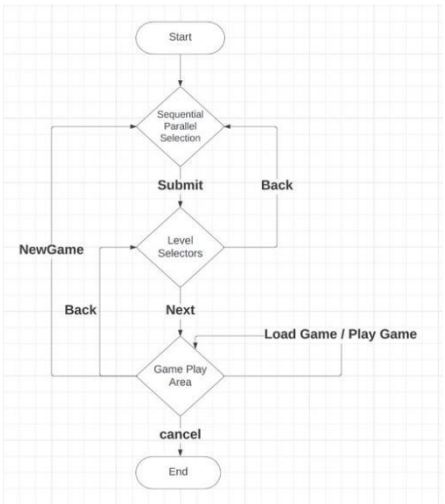


Figure 1. Game Flow Chart

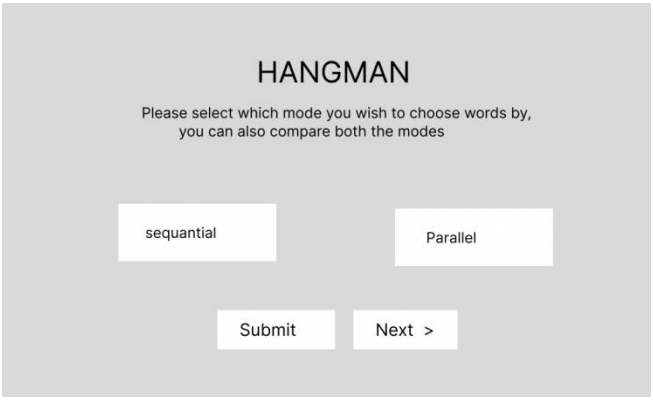


Figure 2. Initial Screen (Wireframe 1)

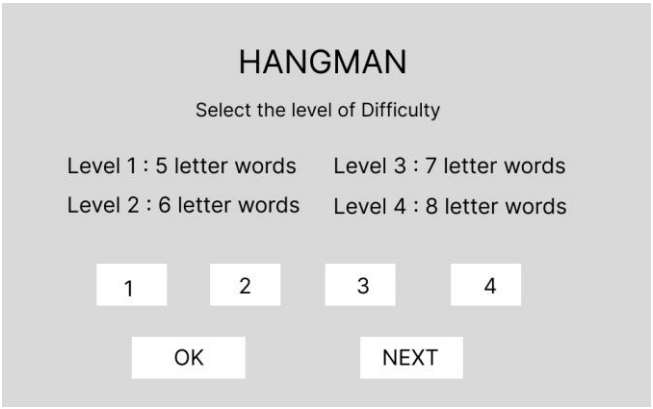


Figure 3: Difficulty level Selection Screen (Wireframe 2)

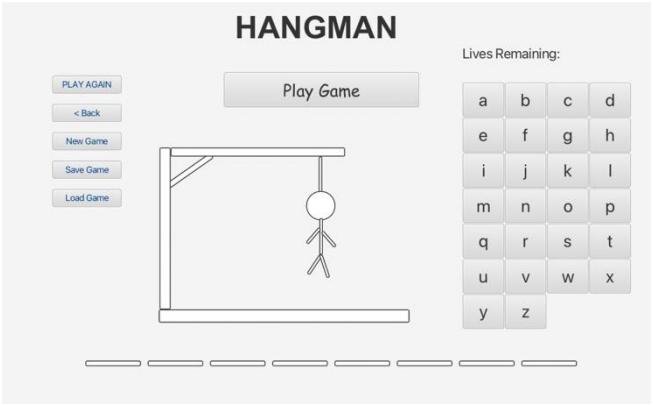


Figure 4: Main Game Screen (Wireframe 3)

IV. IMPLEMENTATION

a. JavaFX

The UI is created using .fxml files and .java controller files. The fxml files are updated using Scene Builder by dragging and dropping various elements. These elements have been assigned an Id and linked to certain controller methods at various events like mouse click. Separate controller files are created corresponding to each page and each .fxml file. The .fxml files are linked to their respective controller files and a .css file to implement the design of the application. Finally, everything is wrapped using a scene and stage. The movement between different screens is achieved by updating the scenes with the root of different .fxml files and setting the stage with the new scene.

b. Class definitions:

We have following classes in the project which we have used to design the game:

1. GameController Class: This class has all the fields representing the fields of type Button, Shape, Label of JavaFx and logical implementation of handling the clicks on the UI.

2. GameHandler Class: This class includes the implementation of the game functionalities.

3. Parallel Class: this class implements the Runnable class and concurrent programming for parallel mode of the hangman game.

4. MiniProject Class: This class contains the main method and is the application execution starts here.

5. SelectWordsController Class: This class is defined to control the modes the of the application.

6. SelectDifficultController Class: This class executes the logic for difficulty selection and methods related to difficulty feature.

7. FileHandling Class: This class does all file handling capabilities and selection of words from different files depending on the game design.

c. Inheritance:

Inheritance is a mechanism in object-oriented programming that allows a class to inherit properties and methods from another class. We have SelectDifficultyController class which extends the DifficultyController class to inherit the properties. In addition to this, we have inherited the Application Class from JavaFX to launch the JavaFX application.

d. Interface:
An interface is a collection of abstract methods that define a contract for how objects of a class should behave. For example, we could define an interface called GameLogicInterface that specifies the methods for managing game state, such as starting a new game, handling guesses, and determining the game

outcome. The WordGame Class could then implement this interface, providing concrete implementations for these methods.

Interfaces used in the project:

Runnable Interface: The Runnable interface is part of Java's concurrency API and is used for creating threads in Java. It defines a single method, run(), which represents the code that will be executed in a separate thread of execution. By implementing the Runnable interface, a class can define the code that will be executed concurrently with other threads, allowing for concurrent execution of tasks in Java applications.

Serializable Interface: The Serializable interface in Java is used for object serialization, which is the process of converting an object's state into a byte stream that can be stored in a file or transmitted over a network. Serializable objects can be deserialized back into objects, allowing for easy storage and transfer of object state. The Serializable interface does not define any methods, but it acts as a marker interface that indicates that an object can be serialized.

Initializable Interface: The Initializable interface is part of JavaFX, which is a Java library for creating graphical user interfaces (GUIs). The Initializable interface is used in JavaFX applications to define controllers for JavaFX UI components. It defines a single method, initialize(), which is called when the UI component is initialized, allowing for setup and initialization of UI elements, event handlers, and other components.

e. Collections:

Collections are data structures that allow you to store and manage groups of objects. For example, you could use a List or an ArrayList to store the list of words, or a Map or a HashMap to store the player's scores and achievements. Collections provide various methods for adding, retrieving, updating, and removing objects, making them useful for managing data in your game.

f. Multi-threading:
Multi-threading is the ability of a program to execute multiple threads (smaller units of a program) concurrently, allowing for parallel processing and improved performance. In the context of the "Wordgame - Hangman" project, multi-threading could be used to handle tasks such as updating the game display, processing user input, or managing game events in separate threads, to prevent the game from freezing or becoming unresponsive during time-consuming operations.

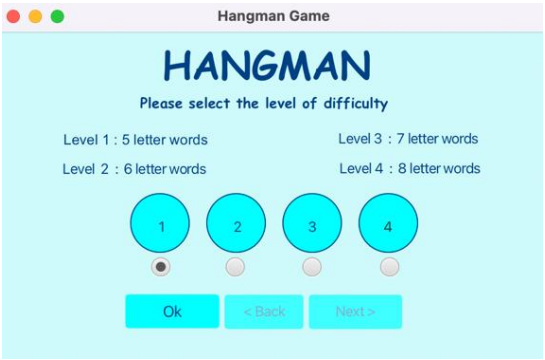
g. Recursion:
Recursion is a programming technique where a function calls itself to solve a problem by breaking it down into smaller subproblems. In the context of the "Wordgame - Hangman" project, recursion could be used, for example, to implement a recursive algorithm for generating hints or clues for the player to guess the correct word based on the partially revealed letters, or for solving other game-related tasks that involve breaking down a problem into smaller steps.

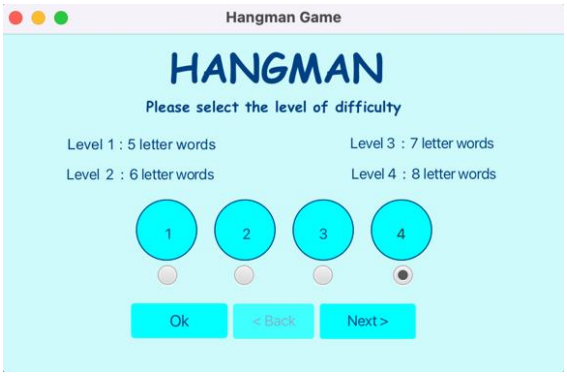
Figure 6. Caption of Figure

V. EVALUATION

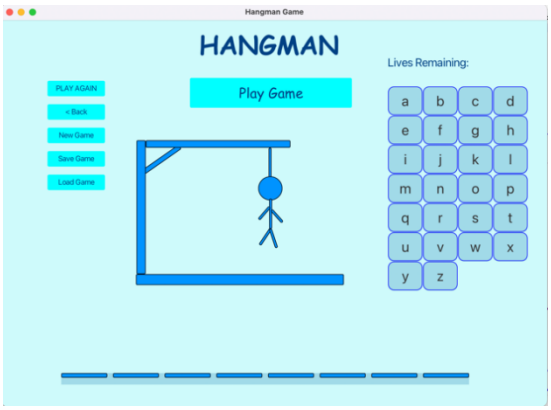
Here are some screen shots of our application in real run time.
Screen 1: Select the mode in which they want to play

Screen 2: Difficulty Level Selection

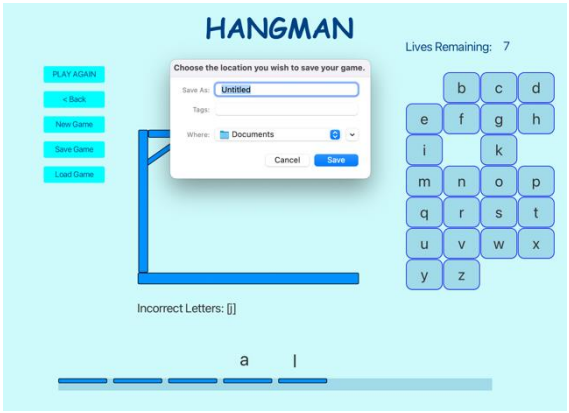


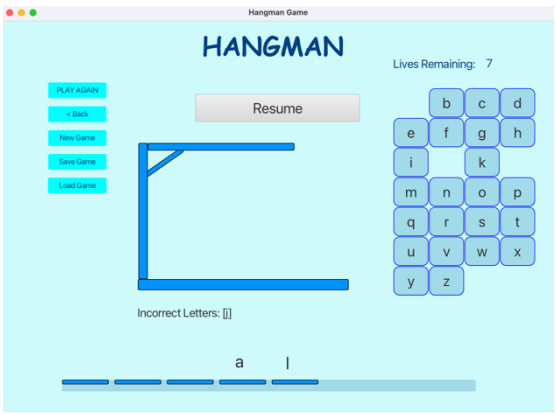


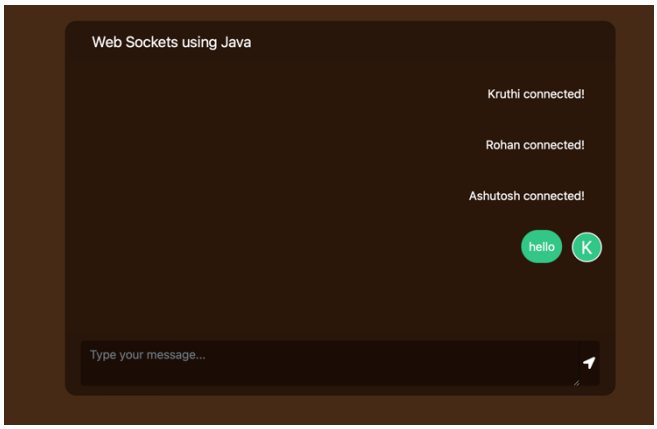
Screen 3: Game play area



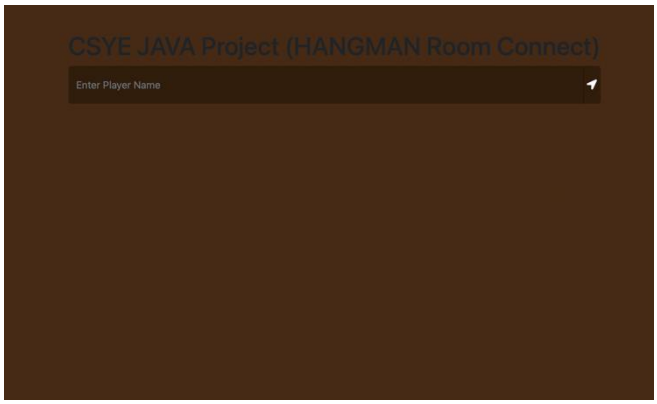
Option to Save the Game and Load It







Chat App Screen 1



Chat App Screen 2

VII. Discussion (Reflection)

a. Gameplay experience

The gameplay experience of Hangman word game can be further improved by incorporating more challenging words and increasing the difficulty levels. This can provide a more challenging experience for advanced players while still maintaining the fun element for beginners. Additionally, the game can be made more engaging by adding animations and sound effects, such as cheering and applause when the player guesses the word correctly.

b. Implementation challenges

One implementation challenge is to designing a user-friendly interface that allows for easy gameplay and interaction with the game. Additionally, implementing the logic behind the game, including the guessing mechanism and tracking of the user's progress, can also be challenging. To overcome this challenge, we used JavaFX. Utilized different types of controls provided by JavaFX, such as buttons, labels, and text fields, to create an intuitive interface.

Changing project requirements: During the development phase, it was crucial to ensure that the project requirements were

fulfilled. This involved maintaining regular communication among the team members and conducting frequent reviews of the project plan. It was imperative to maintain the design structure as per the documented specifications while making necessary revisions to the plan.

Debugging and Troubleshooting: Another challenge that was faced during the project was debugging and troubleshooting the code. As the codebase grew larger and more complex, identifying and fixing errors became increasingly difficult. This required a systematic approach to debugging and a deep understanding of the codebase. Additionally, the use of best coding practices such as code commenting, and documentation helped to mitigate these challenges.

c. Feedback

Furthermore, feedback from users was crucial in improving the Hangman word game. This feedback was collected through surveys and user testing sessions. The feedback was then analyzed and used to make necessary improvements to the game. For instance, we incorporated suggestions for new features such as the chat room and made changes to the gameplay to make it more enjoyable.

In conclusion, the Hangman word game can provide a challenging and fun experience for players. However, it is important to consider the implementation challenges and user feedback to improve the gameplay experience and identify and fix any bugs or glitches. By implementing appropriate solutions and incorporating user feedback, we can create an engaging and enjoyable game that meets the needs and expectations of the target audience.
.

VIII. Conclusions and Future Work

a. User input validation - we can implement various checks to ensure that the user input is in the correct format, including accepting only alphabets, ignoring spaces, and ignoring duplicate entries. This will improve the accuracy of the game and make it more user-friendly.

b. Incorporating a scoring system can add a competitive element to the game, making it more exciting and motivating for players to try and achieve higher scores. The scoring system can be based on the number of attempts made to guess the word, with higher scores being awarded for guessing the word with fewer attempts. This will encourage players to improve their gameplay and compete with others to get the highest score.

c. Sound effects and animations can add to the game's overall user experience, making it more enjoyable and immersive. For example, sound effects can be added for different events, such as when a player makes a correct guess, while animations can be used to show the hangman's progress as the player guesses incorrectly.

d. We have taken a step further in enhancing the gaming experience by implementing a "Chat Room" feature for players. This feature has been implemented using WebSocket technology, which ensures a smooth and real-time communication experience for the players. With this feature, players can interact with each other during the game and share their thoughts and strategies. In the future, we plan to further expand this feature by adding multiplayer functionality. This will enable multiple players to play the game simultaneously and interact with each other in real-time. The chat room will serve as a platform for players to chat and discuss the game, creating a sense of community and engagement. This feature is expected to increase the game's popularity and appeal among players.

In conclusion, there are several potential areas to enhance the Hangman word game, including user input validation, a scoring system, sound effects and animations, multiplayer functionality, and a leader board. By incorporating these features, the game can be made more enjoyable, engaging, and social for players, encouraging them to play and compete with others.

IX. Job Assignment

- Ashutosh Raval: Requirements and Game Design. Game Controlling and handling feature
- Kruthi Shankar Hegde: UI design and development, Wireframes, Scene Builder screens development, Game Loading and Difficulty level Feature
- Rohan Vasudev Ginde: Algorithm design and optimization, Words controller and Chat Room App development

X. HOW TO RUN OUR APP

- Open Eclipse on your computer.
- Open the project file for the Hangman word game.
- Navigate to the "Run Configurations" option by clicking on the "Run" button located in the top menu bar of Eclipse.
- Select the "Java Application" option from the left-hand panel.
- Click on the "New Launch Configuration" button to create a new launch configuration.
- In the "Main" tab, choose the main class for the Hangman word game application.
- In the "Arguments" tab, add the following command line argument: "--add-modules javafx.controls,javafx.fxml". This command allows Eclipse to run the JavaFX module, which is used for the user interface of the game.
- Click on the "Apply" button to save the configuration settings.
- Click on the "Run" button to start the Hangman word game application.

REFERENCES

[1] Java OOP (Object-Oriented Programming)
https://www.w3schools.com/java/java_oop.asp

[2] WebSockets
https://www.tutorialspoint.com/websockets/index.htm

[3] JavaFx Tutorial - Introduction To JavaFx by ProgrammingKnowledge
https://www.youtube.com/watch?v=9YrmON6nlEw&list=PLS1QulWo1RIaUGP446_pWLgTZPiFizEMq

[4] JavaFX 2 Tutorials and Documentation
https://docs.oracle.com/javafx/2/