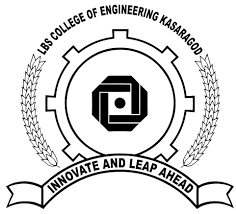
****

**PROJECT REPORT**

**PROJECT NAME : SOUND MATCH GAME**

DEVELOPED BY : GAYATHRI P NAIR-26

HRIDYA S NAIR-30

SOYA DEVASSIA-61

SREYA NAIR V-62

VERSIONS:1.CONSOLE-BASED

2.GUI BASED

LBS COLLEGE OF ENGINEERING KASARGOD

Acknowledgement

We would like to express our heartfelt gratitude to all those who supported and guided us in the successful completion of our project, “Sound Match Game.”

First and foremost, we express our sincere thanks to our subject teacher, Prof. Nayana Murali, for her valuable guidance, encouragement, and constant support throughout the development of this project.

We are deeply grateful to our project mentor, Prof. Prathima, for her continuous motivation, insightful suggestions, and constructive feedback that greatly helped us in improving the quality of our work.

We would also like to extend our gratitude to our Head of the Department, Dr. Manoj, for providing us with the necessary facilities and for his inspiring leadership.

Our sincere thanks to all the faculty members and staff of the Department of Computer Science and Engineering, LBS College of Engineering, Kasaragod, for their support and cooperation.

Finally, we thank our friends and family members for their encouragement, understanding, and unwavering support throughout this project journey.

**Introduction**

The Sound Match Game is an interactive educational and entertainment-based Java application designed to help users improve their listening skills and memory through sound recognition. The game plays a sound, and the player must correctly identify the matching image or option.

Two versions of the game were developed:

**Console Version**: A simple text-based version that runs in the terminal.

**GUI Version**: A visually enhanced version built using JavaFX, which includes images, sound buttons, and a scoring system.

**Console Version – Detailed Working**

Overview

The console version focuses on the essential logic of the game, without graphical elements. It uses Java’s core libraries along with the javazoom.jl.player.Player library for audio playback.

Step-by-Step Execution

**1.Program Start**

Main.java initializes the game by creating an instance of GameManager.

**2.Question Loading**

QuestionBank.java loads a list of questions. Each question contains:

* Sound file name
* List of options
* Correct answer

**3.Playing a Question**

SoundPlayer.java plays the associated MP3 file.

Console displays the options for the question.

4.User Input

The player enters their answer via keyboard input.

**5.Answer Validation**

The game verifies if the input matches the correct answer and updates the score accordingly.

**6.Next Question**

The next question is loaded until all questions are completed.

**7.Game End**

The final score is displayed.

**Core Files Involved**

|  |  |
| --- | --- |
| File Name | Responsibility |
| Main.java | Entry point of the game;runs the main loop. |
| GameManager.java | Controls question flow , scoring and logic. |
| Question.java | Represents a single question object. |
| QuestionBank.java | Stores all questions and loads them for the game. |
| Soundplayer.java | Plays MP3 files from sounds folder using javazoom library. |
| Sounds | Contains the MP3 sound files that correspond to the images.  lib folder (j1.0.1.jar):  Contains the external Java library needed for MP3 playback. |

**Console Version – Example Flow**

Program Execution

s?Welcome to the Sound Match Game!?

Playing sound…..

1. Dog

2. Bird

3. Cat

Your answer (1-3): 2

Correct

Playing sound….

1. Bell

2. Drum

3. Guitar

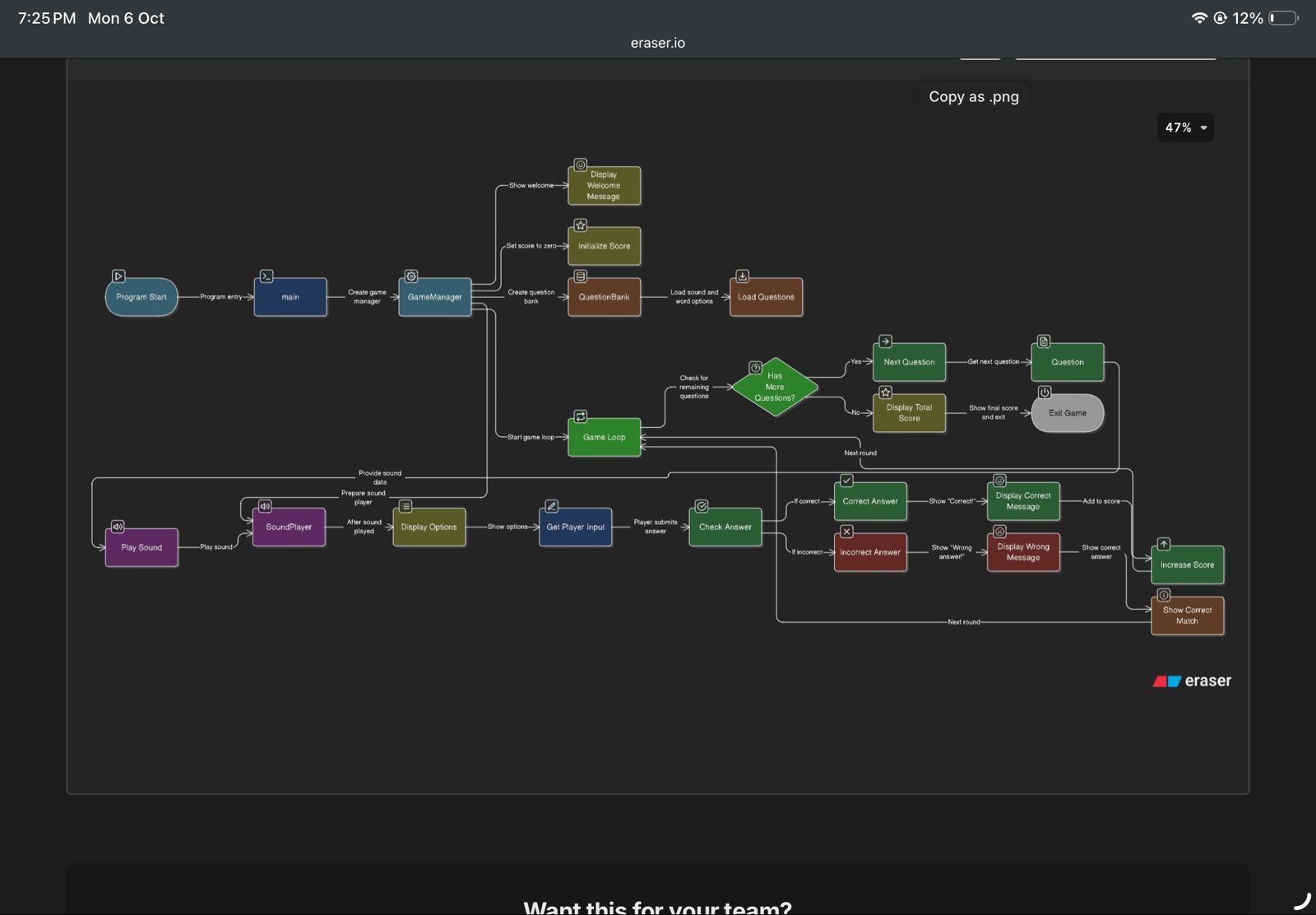
Enter your answer (1-3): 3

?Wrong! Correct answer:Bell

Game Over!

Final score: 1

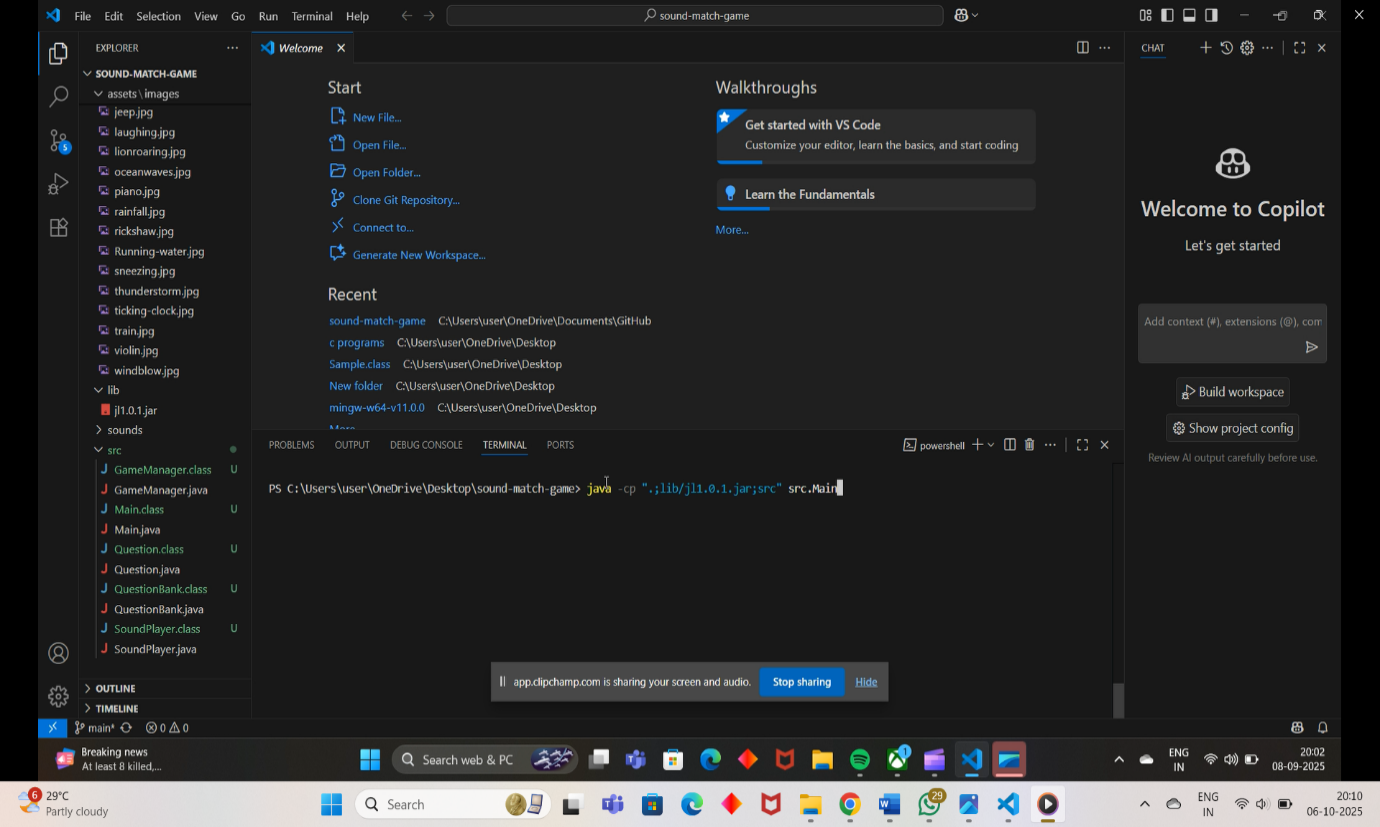
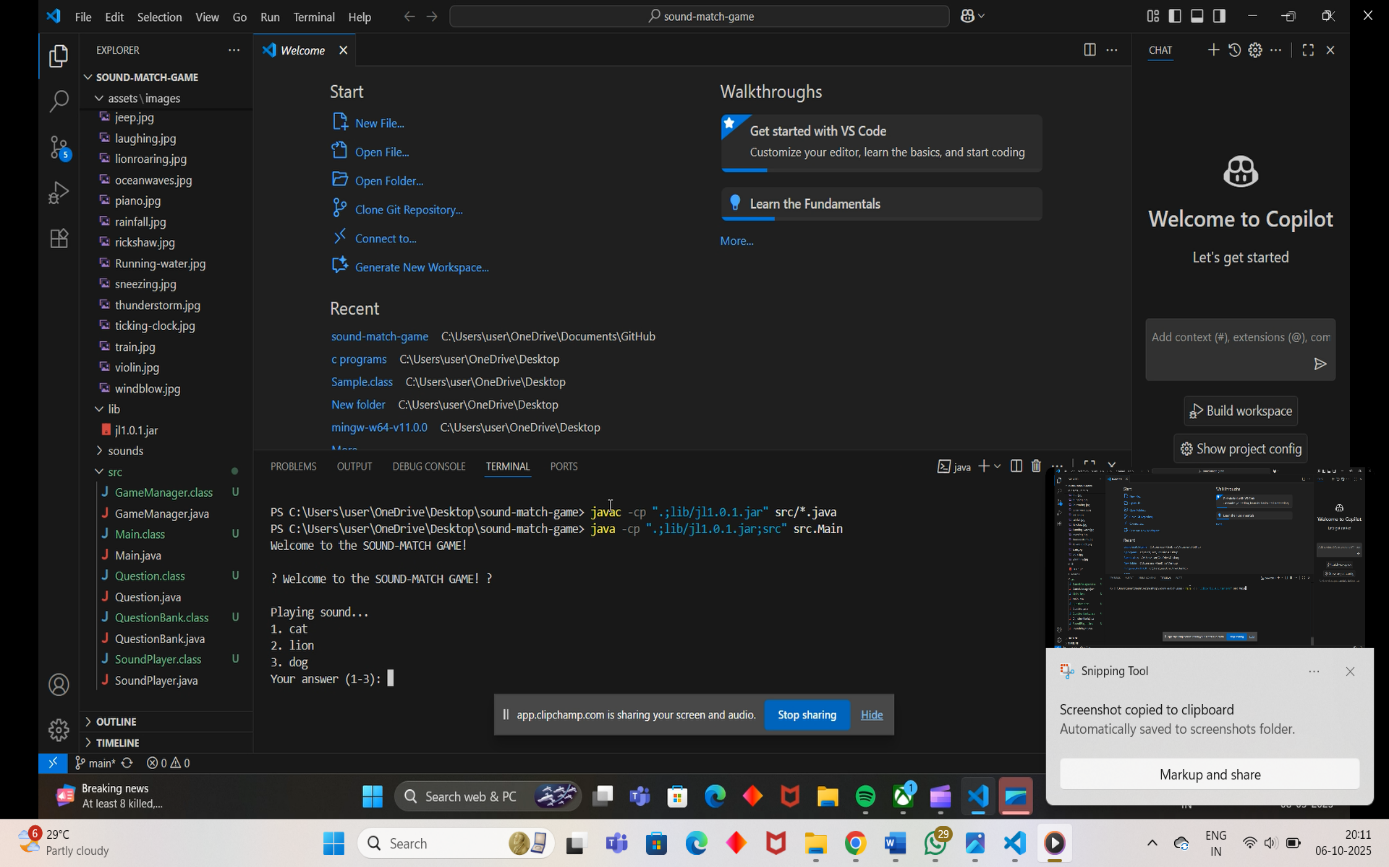
**Flow Diagram Console-version**

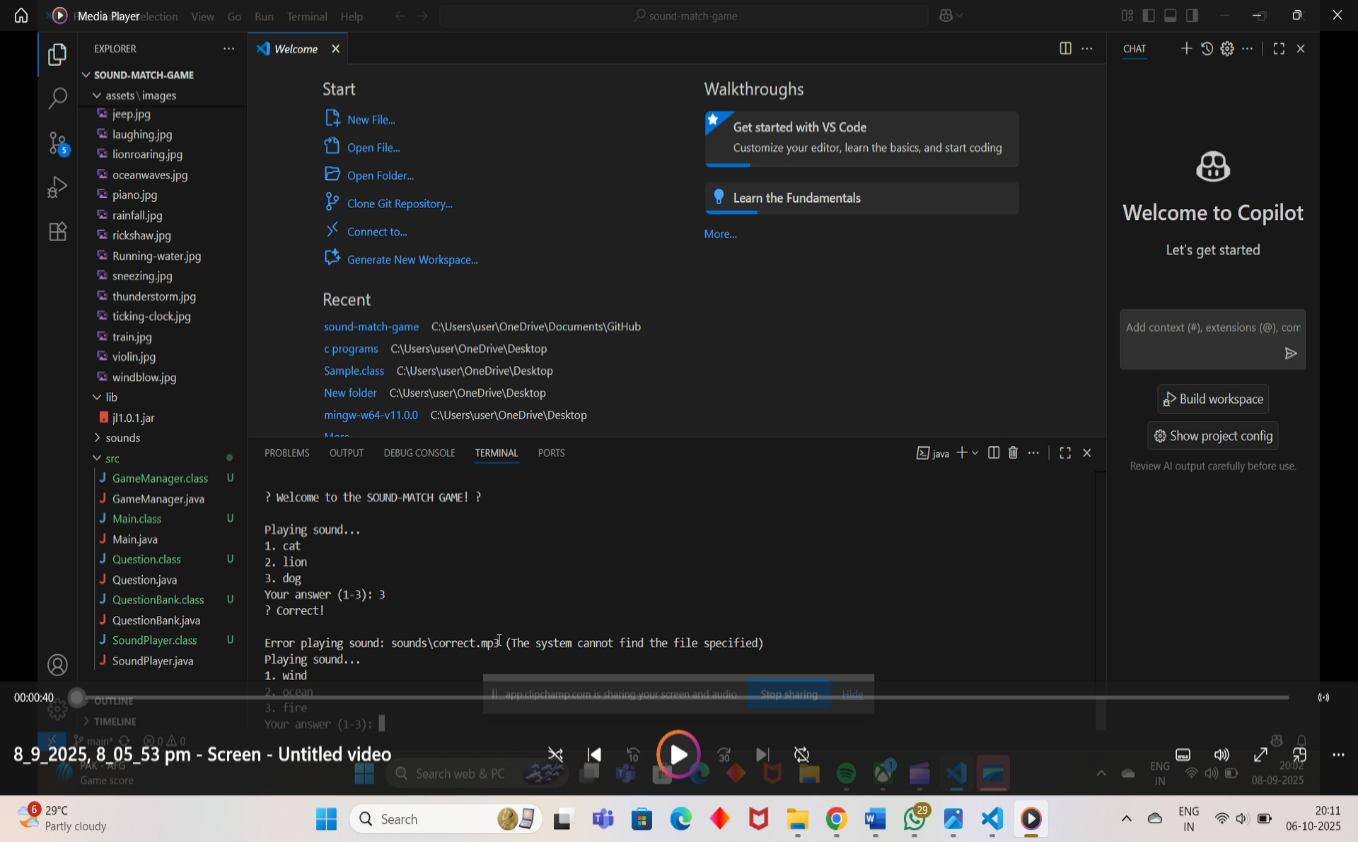
****

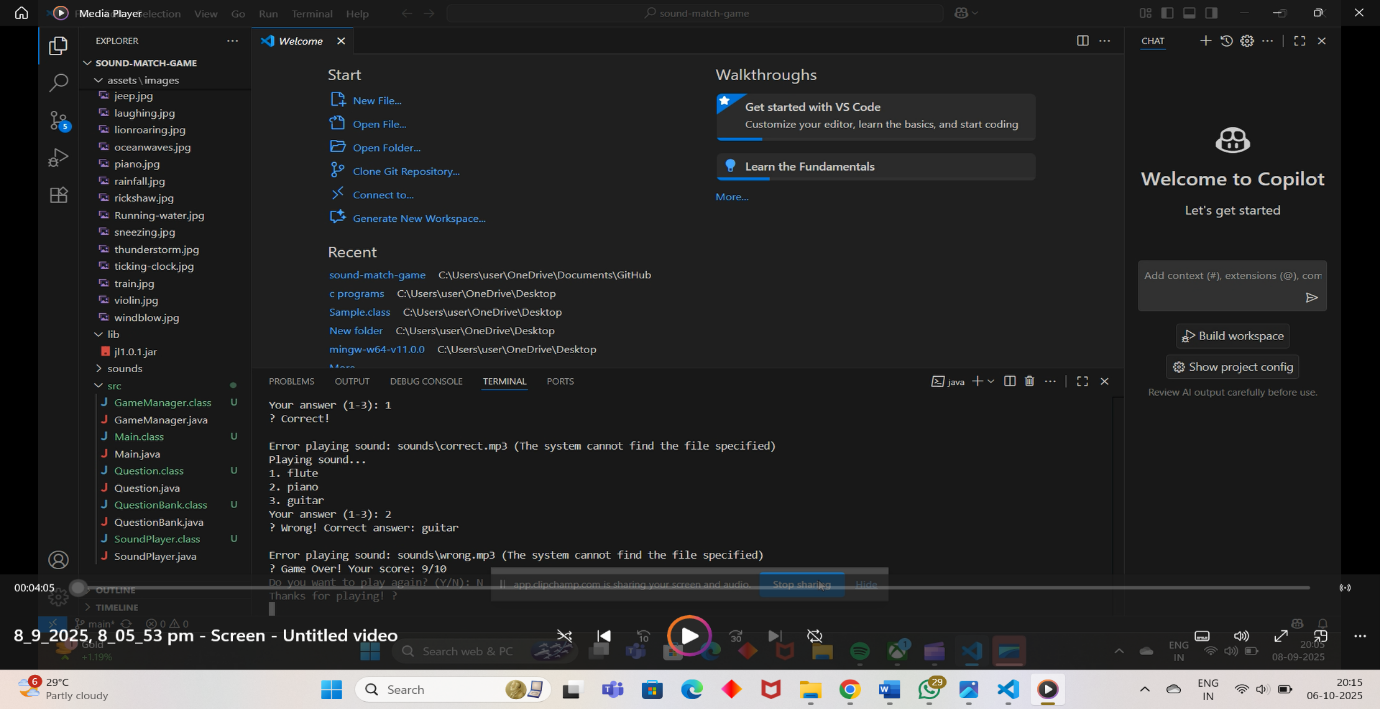
**Test Cases – Console Version**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test case no**. | **Description** | **Input** | **Expected output** |
| 1 | Correct answer | Correct option number | ?Correct! |
| 2 | Incorrect answer | Wrong option number | ?Wrong! Correct answer:option |
| 3 | Invalid Input | Text or out-of-range number | Invalid input please enter a valid input |
| 4 | End of questions | All questions are answered | Game over  Your score is;displays score |

**Screenshots of running console-version**

****

****

****

**GUI Version -Detailed working**

Overview

The Graphical User Interface (GUI) version of the Sound Match Game is a visually interactive application developed using JavaFX.

It challenges players to match a sound with its corresponding image, combining learning and entertainment through sound-image association.

All logic and gameplay flow are managed within SoundMatchApp.java, while SoundPlayer.java handles all sound playback operations.

The project follows a clean folder structure, ensuring organized access to assets, libraries, and styling components.

Step-by-Step Execution

**1.Launching the Game**

The main class SoundMatchApp.java initializes the JavaFX stage and sets up the window using layouts like VBox and GridPane.

**2.Loading Resources**

* Images are loaded from assets/images/.
* Sounds are loaded from the sounds/ folder.
* SoundPlayer.java uses the external library j1.0.1.jar for MP3 playback.

**3.User Interface Setup**

* A speaker button (with image icon) lets users play the current sound.
* Multiple image options are displayed in a grid layout for selection.
* The score label updates dynamically after each question.
* game-styles.css defines button colors, fonts, hover effects, and layout margins for a modern and appealing look.

**4.Gameplay**

* When the user clicks the speaker icon, the corresponding sound is played.
* The user selects one of the displayed images as the answer.
* The system checks for correctness and updates the score.
* The next sound automatically loads after a short delay.

**5.Score Management**

The score variable is updated within SoundMatchApp.java and shown on the screen using:

Java

scoreLabel.setText("Score: " + score);

**6.Game Completion**

When all questions are completed, the final score is displayed with an option to Restart the game.

**Core Files Involved**

|  |  |
| --- | --- |
| **File Name** | **Description** |
| SoundMatchApp.java | Main controller file that handles the user interface, question logic, event handling, and scoring system. |
| SoundPlayer.java | Handles MP3 sound playback using the javazoom library (j1.0.1.jar), located inside the lib folder in the project root. |
| game-styles.css | Defines the GUI appearance—button colors, hover effects, background, text font, and alignment. |
| assets/images | Contains all the image files used as answer options in the game. |
| sounds | Contains the MP3 sound files that correspond to the images.  lib folder (j1.0.1.jar):  Contains the external Java library needed for MP3 playback. |

**GUI-version Example flow**

Step 1: The user clicks the speaker button — the sound “dog.mp3” starts playing.

Step 2: Three image options appear on the screen (dog, cat, bell).

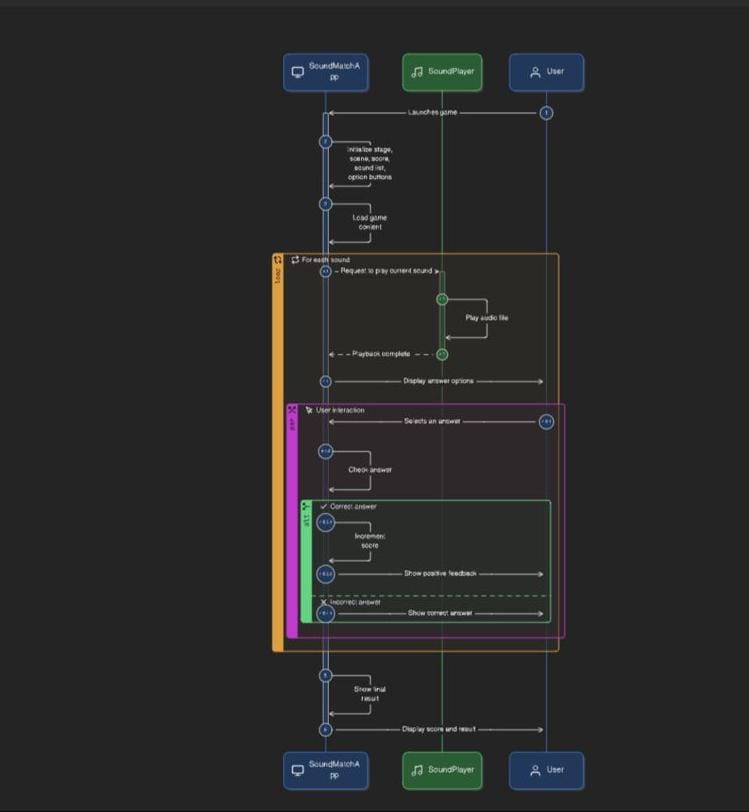
Step 3: The user selects the “dog” image.

Step 4: If the answer is correct, the score increases by 1 and “Correct!” is displayed.

Step 5: The next sound automatically loads, and the game continues until all questions are answered.

Step 6: When finished, the game displays the final score with an option to restart.

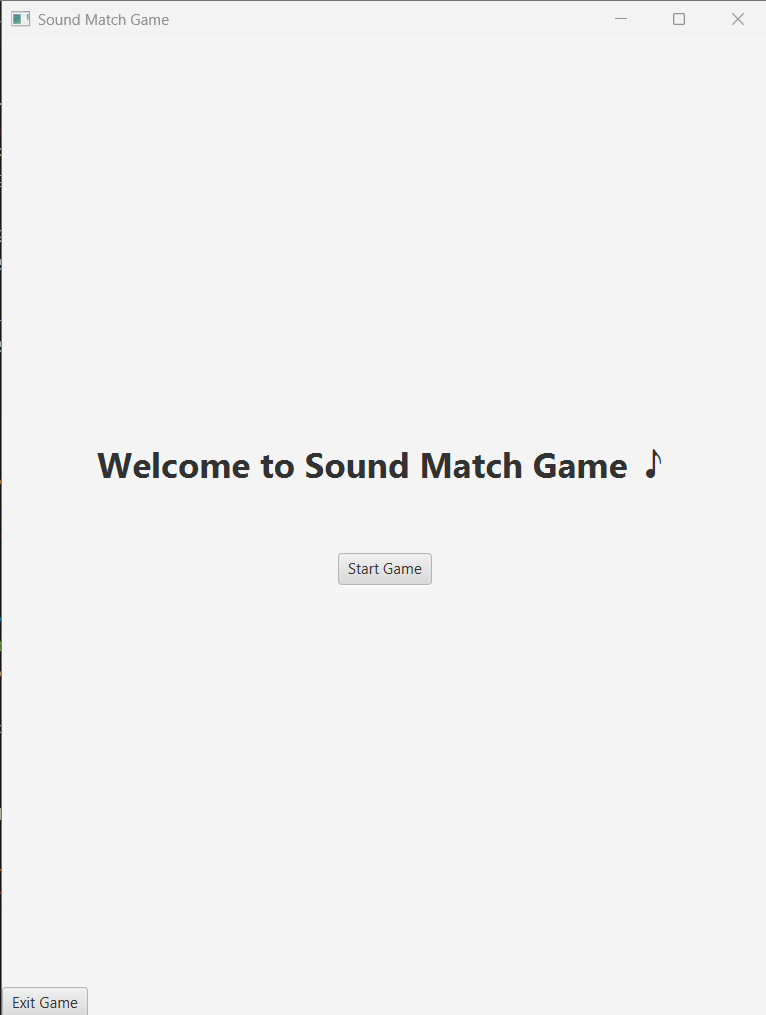
**Flow Diagram GUI-version**

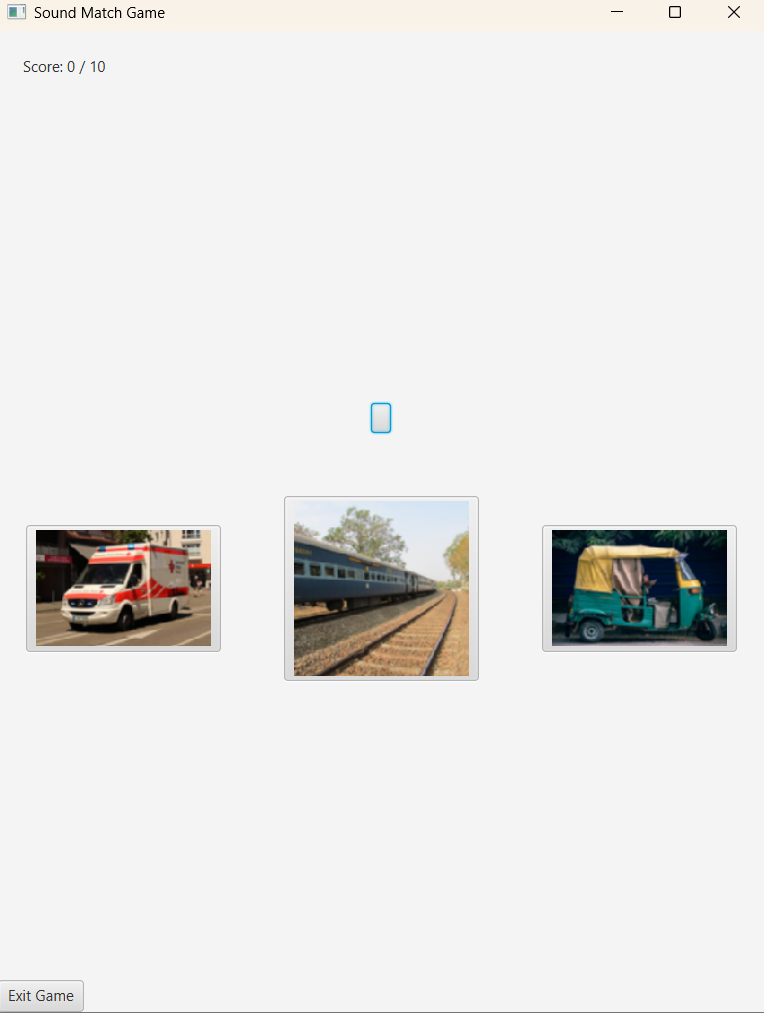
****

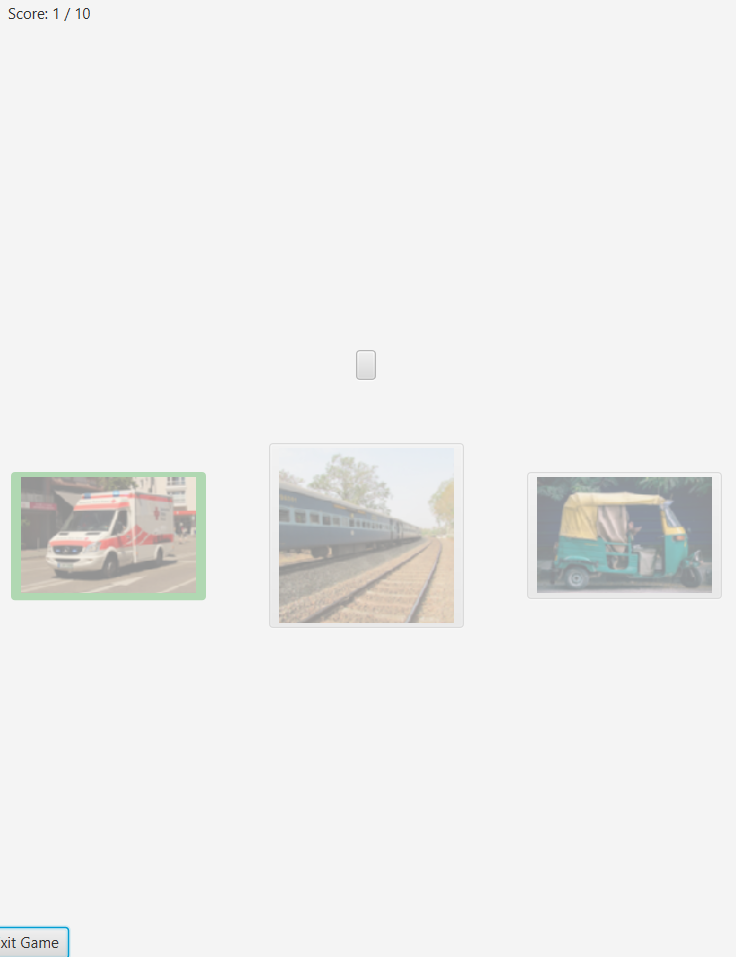
**Test Cases-GUI Version**

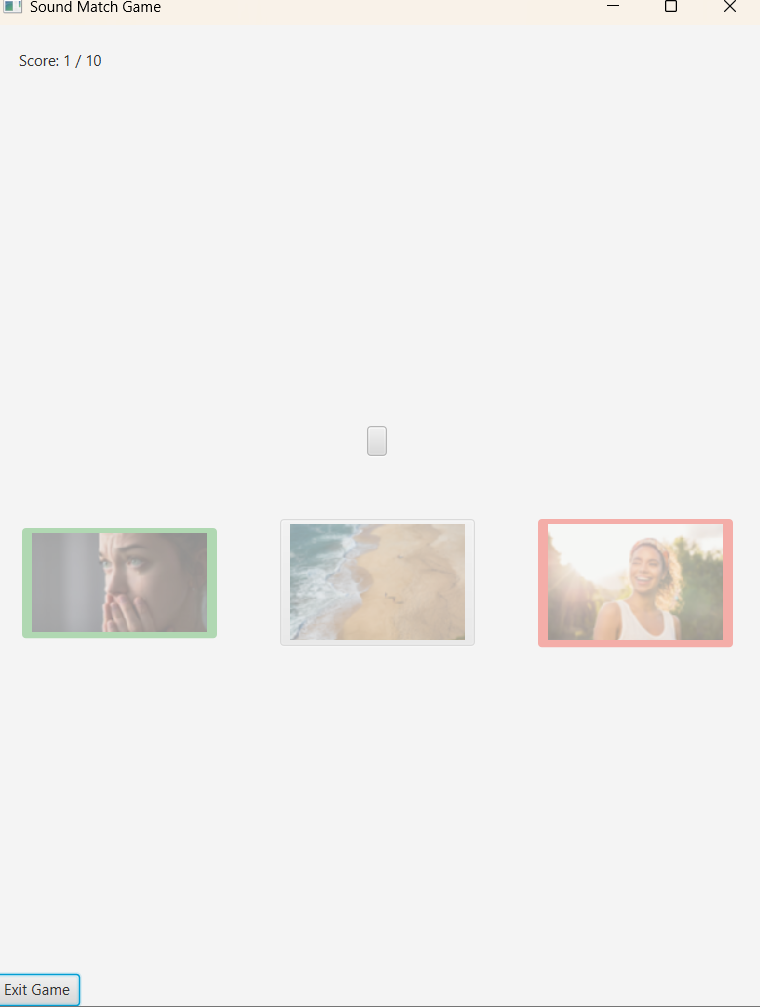
|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Scenario** | **Expected Outcome** | **Result** |
| TC01 | Launch the game. | Game window opens successfully with all UI components visible. | Pass |
| TCO2 | Sound Playes | Game window opens successfully with all UI components visible. | Pass |
| TCO3 | Select the correct image. | Displays “Correct!” and increases the score by 1. | Pass |
| TCO4 | Select an incorrect image. | Displays “wrong” and score remains the same. | Pass |
| TC05 | Load all images. | All images appear from assets/images/ without delay. | Pass |
| TCO6 | CSS styling | Buttons and text follow the format defined in game-styles.css | Pass |
| TC07 | Game completion. | Displays the final score and a restart button. | Pass |

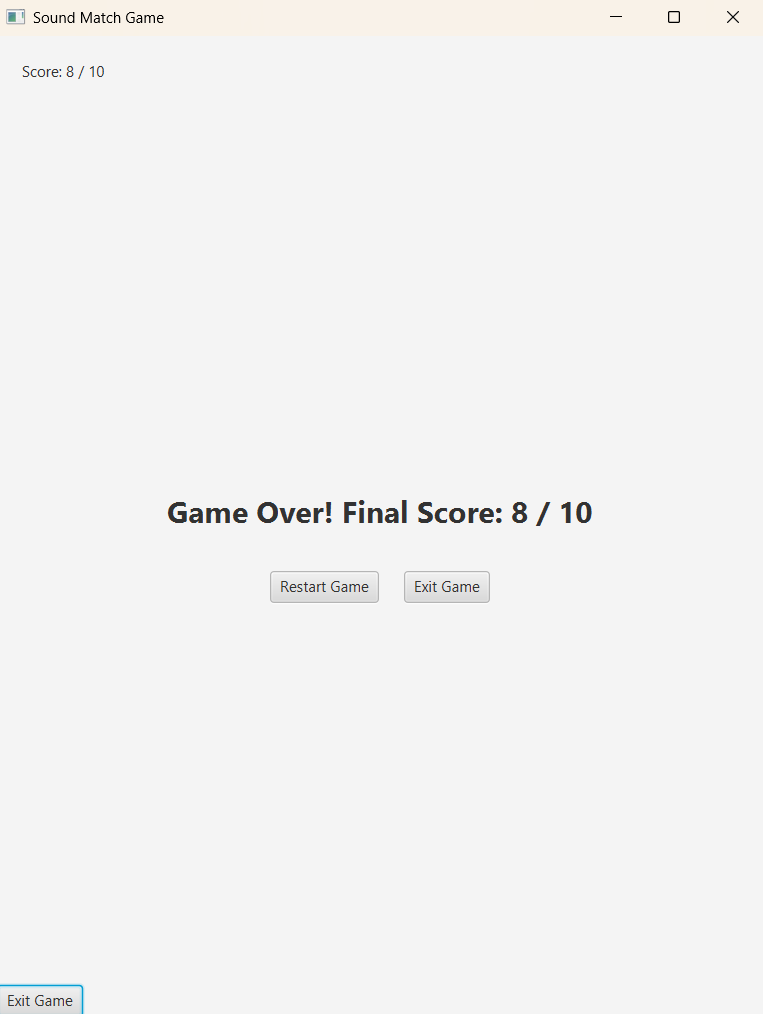
**Screenshots of running GUI-version**

****









**Outcomes and Results**

The Sound Match Game project successfully achieved its core objective of creating an interactive application that enhances auditory recognition and reflex skills. The game demonstrates effective integration of sound playback and matching logic, providing an engaging and user-friendly experience in both console and GUI versions. Through testing, it was confirmed that the system accurately plays sound files and reliably matches user responses within defined time limits. The project also delivered a functional and visually appealing interface, incorporating an asset-based structure for images and sounds, along with smooth navigation and responsive gameplay. Furthermore, the game’s modular design allows easy future expansion, ensuring scalability for features such as multiplayer mode, dynamic sound libraries, and cross-platform support. Overall, the project not only met its functional requirements but also laid a strong foundation for further development in interactive sound-based applications.

**Future Scope**

The Sound Match Game project holds a wide range of possibilities for future development and improvement. In the coming versions, the game can be expanded by introducing a multiplayer mode that allows multiple users to compete in real-time sound recognition challenges. A dynamic sound library can also be implemented so that new sound packs—such as animal sounds, musical instruments, or vehicle noises—can be downloaded or updated automatically, keeping the game engaging and fresh.

Another area for enhancement is the introduction of level-based gameplay, where players can progress through beginner, intermediate, and advanced stages depending on their accuracy and speed. To make the game more adaptive and intelligent, machine learning algorithms can be incorporated to analyze player performance and adjust difficulty levels or recommend sound categories suited to individual skill levels.

Additionally, the inclusion of leaderboards, achievements, and reward systems can help foster competition and motivation among players. The project can also be extended to multiple platforms such as Android, iOS, and web browsers, enabling cross-platform compatibility with synchronized user data.

In the long term, the game can integrate accessibility features for hearing-impaired users through visual or vibration-based feedback, ensuring inclusivity. Lastly, continuous UI and UX enhancements—such as animated transitions, customizable themes, and improved sound effects—will make the game more appealing and interactive for all types of users.

**Conclusion**

The Sound Match Game project successfully demonstrates a creative and interactive approach to improving auditory recognition skills. Both console and GUI versions work efficiently, providing an engaging user experience while maintaining accuracy and responsiveness. The project’s modular design ensures easy scalability, making it a solid foundation for future enhancements such as multiplayer modes, dynamic sound libraries, and cross-platform support. Overall, this project highlights the potential of combining sound processing and interactive design for educational and entertainment purposes

**References**

Java Platform, Standard Edition Documentation. Oracle. https://docs.oracle.com/javase/

JavaFX Official Documentation. Oracle. https://openjfx.io/

JLayer Library for Java MP3 Playback. http://www.javazoom.net/javalayer/javalayer.html

Sound and Audio Processing Concepts. Smith, J.O., Introduction to Digital Audio, W3 Press, 2019.

Fowler, M., UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison-Wesley, 2010.

Git Documentation. Git SCM. https://git-scm.com/doc

W3Schools. HTML, CSS, and JavaFX Tutorials. https://www.w3schools.com