Chapter 1

Introduction

1.1 Introduction

Project Pictionary is a real-time multiplayer game developed using Flutter for the client-side and Node.js with Socket.io and MongoDB for the backend. It enables users to create and join game rooms where players take turns drawing randomly generated words while others guess. The game features interactive doodling, chat, scoring, and leaderboards. This documentation provides a comprehensive guide to set up and understand the project, including the architecture, features, and server-client communication, allowing developers to contribute or customize the game for various applications.

1.2 Motivation

The motivation behind the development of our Pictionary project stems from the desire to modernize and enhance the traditional drawing and guessing game experience. Traditional board games have always been a source of entertainment and social interaction, but they often require physical presence and can be limited in scope. In the digital age, there is a growing need for engaging and accessible gaming experiences that transcend physical boundaries. Pictionary, as a beloved draw and guess game, provided the perfect opportunity to leverage modern technologies like Flutter, MongoDB and Node.js to bring this classic game to a wider audience.

By developing a digital version of Pictionary, we aim to make the game more accessible, interactive, and enjoyable for players regardless of their physical location. This project also serves as an exploration of the capabilities of these technologies, highlighting their potential to transform traditional games into immersive digital experiences. Additionally, the global shift towards online entertainment and social interaction, particularly in light of recent events, underscores the relevance and timeliness of such a project. Ultimately, our motivation is to provide a platform that fosters creativity, connection, and entertainment in a digital age while showcasing the possibilities of modern software development.

1.3 Problem Statement & Objectives

In the realm of online gaming, Pictionary aims to provide a virtual platform where players can participate in a draw and guess game seamlessly. This project addresses the need for accessible and engaging entertainment in the digital age, eliminating the need for physical presence and offering a convenient and fun experience for players. The primary objectives of this project are:

- To develop a digital version of the Pictionary game using modern technologies such as Flutter, MongoDB, and Node.js.
- To create an intuitive and user-friendly interface that allows players to draw, guess, and interact effortlessly.
- To offer a platform for players to enjoy Pictionary from anywhere, fostering creativity and social connections.
- To showcase the potential of these technologies in transforming traditional games into engaging digital experiences.

1.4 Organization of the Report

This report consists of three chapters. The first chapter deals with introduction of the topic, problem statement, motivation behind the topic and objectives. The second chapter is the Literature Survey. It includes all the research work done related to this topic. All information related to study of existing systems as well as learning of new tools is mentioned in this chapter. The third chapter is about the proposed system which is used in this project. The block diagram, techniques used, hardware and software used screenshots of the project are presented in this chapter. All the documents related to development of this project are mentioned in References.

Chapter 2

Literature Survey

This chapter explains the concepts used in this project, study of existing system and contribution of this project

2.1 Survey of Existing System

Our project team conducted a survey of existing digital drawing and guessing games to gain insights into the current landscape of similar applications.

1) Drawasaurus:
   - A collaborative art platform for creative users.
   - Available on Android, iOS, and PC with an intuitive interface.
   - Participants guess drawings to earn points.
   - Users can join or create chat rooms with customizable settings.
   - Offers drawing and creative tools for free.
2) Pictar.io:
   - Web-based gaming site for art lovers.
   - Users can create accounts or play as guests.
   - Features drawing and multiplayer puzzle games with a chat system.
   - Turn-based gameplay where players draw and guess words.
   - Offers drawing tools, private chat rooms, and 2D graphics.
3) Draw & Guess:
   - Provides Pictography games with artistic drawing tools.
   - Players earn points by guessing others' drawings.
   - Offers various game modes, including classic, multi-color, and invisible mode.
   - Allows players to create custom avatars and compete daily.
   - Features different brushes for drawing and a user-friendly interface.

2.2 Limitation of existing system

1) Drawasaurus Limitations:
    - Limited Features: While Drawasaurus offers free drawing tools and a chat room for gameplay, it may lack the complexity and variety of tools that more dedicated graphic design or drawing software provides.
    - Mobile-Centric: It mentions Android and iOS mobile compatibility, but this may not be suitable for users who prefer working on larger screens with more advanced tools.
    - User Base: The success of such platforms heavily depends on the active user base, which can sometimes be limited, affecting the availability of games and players to interact with.
2) Pictar.io Limitations:
    - Limited Features for Guest Users: Guest players on Pictar.io have restricted access to the platform's features. This can be frustrating for users who want to explore the platform before committing to creating an account.
    - Dependency on User-Generated Content: The quality of gameplay and word choices can depend on user-generated content, which may vary in quality and creativity.
    - Potential for Inactive Rooms: If users create private chat rooms and do not actively invite participants, it may result in inactive rooms or limited engagement.
3) Draw & Guess Limitations:
    - Limited Complexity: While it provides a fun and engaging drawing and guessing experience, it may not cater to professional artists or those seeking advanced drawing tools.
    - Dependency on User Participation: The enjoyment of the platform relies on having active players and participants. If there are too few users, it can limit the quality of the gaming experience.
    - Lack of Seriousness: Some users might find the platform's games less suitable for serious art or design work, as it is primarily focused on casual gaming.

2.3 Mini Project Contribution

Our Pictionary project seeks to address these limitations by providing a user-friendly and customizable digital drawing and guessing game. The key contributions of our project include:

- Intuitive User Interface: We have designed a straightforward and user-friendly interface to ensure players of all skill levels can enjoy the game.
- Diverse Gameplay Elements: Our Pictionary game incorporates a range of features and game modes to keep players engaged and entertained.
- Customization Options: We offer players the ability to customize game settings, including difficulty levels and drawing tools, providing a tailored gaming experience

Chapter 3

Proposed System

This chapter consists of detailed description about the methodology used, the hardware and software components, the tools used and also the screenshots of the project.

3.1 Introduction

Flutter :

Flutter is an open-source UI development framework developed by Google to build apps. It enables us to build cross-platform apps using a single codebase. A single codebase can build smooth, high-performance natively compiled applications for iOS and Android quickly.

Flutter provides a simple, powerful, efficient, and straightforward SDK for developing mobile applications in Google's language, Dart. Dart is an open-source, Object-Oriented programming language with C-style syntax used to power Flutter apps.[1]

Node JS:

Node.js is an open-source, cross-platform, JavaScript runtime environment built on the V8 JavaScript engine of Google Chrome. It allows developers to write server-side JavaScript code, which runs outside of the web browser. Node.js provides an event-driven, non-blocking I/O model that makes it efficient and scalable for building network applications. This architecture allows Node.js to handle a large number of concurrent connections with low overhead and high throughput.

Node.js is commonly used for building web applications, real-time chat applications, and server-side APIs. It comes with a large set of built-in modules, such as HTTP, HTTPS, and FileSystem, that make it easy to develop server-side applications. Node.js also has a large and active community of developers who contribute to the ecosystem of modules and tools. One of the advantages of Node.js is its ability to work with the JavaScript language on both the server-side and client-side, providing a unified programming language for the full stack. This allows developers to share code between the front-end and back-end, simplifying development and maintenance.[2]

Socket.io:

Socket.IO is an event-driven library for real-time web applications. It enables real-time, bi-directional communication between web clients and servers. It consists of two components: a client, and a server. Both components have a nearly identical API.

Socket.IO is also a protocol, where different complying implementations of the protocol can communicate with each other. The main implementation consists of two parts: a client that runs in the browser and a server for Node.js. Apart from the main implementation, there are multiple implementations, for example, the official Deno (JavaScript), C++, Java , and Swift servers.

Socket.IO primarily uses the WebSocket protocol with polling as a fallback option, while providing the same interface. Although it can be used simply as a wrapper for WebSockets, it provides many more features, including broadcasting to multiple sockets, storing data associated with each client, and asynchronous I/O.

It can be installed with the Node Package Manager (NPM).[6]

MongoDB:

MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data.[3]

3.2 Architecture

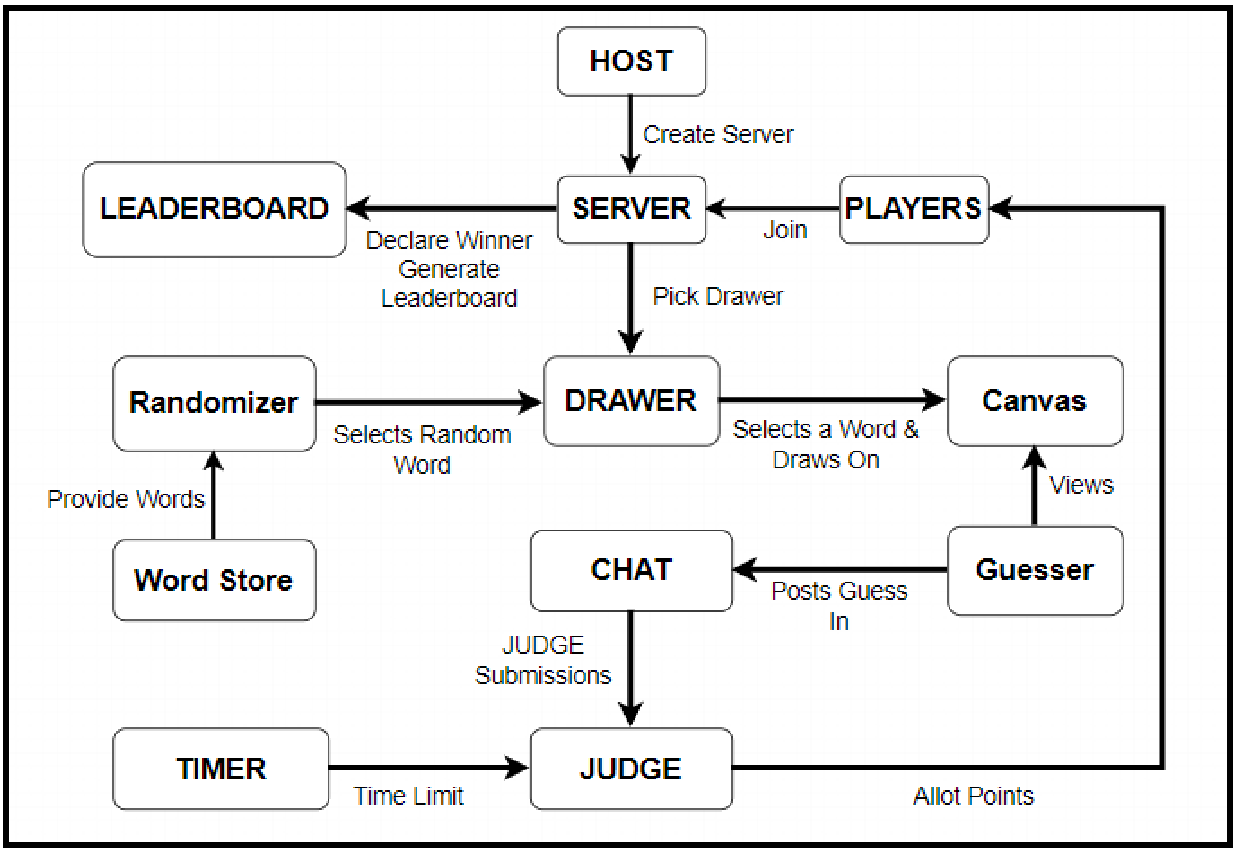


Fig 3.2.1 Block Diagram

The user can create the live server and make room for other players to join and play. The server chooses a sketcher and other players make guesses.

The judge judges the guesses and allot points and manage leaderboard.

3.3 Algorithm and Process Design

The development of the Pictionary project was guided by a structured algorithm and a carefully planned process design to ensure a systematic and efficient development journey. The key phases of this algorithm and process design are outlined below:

1. **Problem Statement Formulation:**

   Our project's inception involved extensive discussions among team members and an in-depth analysis of existing drawing and guessing games. This collaborative effort led to the formulation of a precise problem statement that defined the project's scope and objectives.

2. **Framework and Requirements Assessment:**

   Once the problem statement was established, the team deliberated on selecting the most appropriate technologies and frameworks for our project. We considered the group's familiarity with these technologies and their suitability for fulfilling the project's requirements. Ultimately, we chose to leverage Flutter for the front-end and Node.js for the back-end to deliver a dynamic and cross-platform gaming experience.

3. **Tool and Technology Selection:**

   In addition to technologies and frameworks, we carefully selected the tools and software that would facilitate our development process. Tools like Visual Studio Code and various web browsers for testing and debugging were chosen to ensure a smooth development workflow.

4. **Feature Finalization:**

   With a clear project overview and wireframes in place, our team collectively identified and finalized the features to be integrated into the application. These features were designed to enhance the gameplay and overall user experience of Pictionary.

5. **Development Phase:**

   To expedite development, we divided the project into modules, with each group member taking responsibility for specific components. This approach allowed us to harness the diverse skills and expertise of our team members effectively.

6. **Testing and Issue Resolution:**

   Rigorous testing procedures were implemented to identify and resolve any issues within our modules. Regular communication and collaboration among team members were crucial for successful issue resolution.

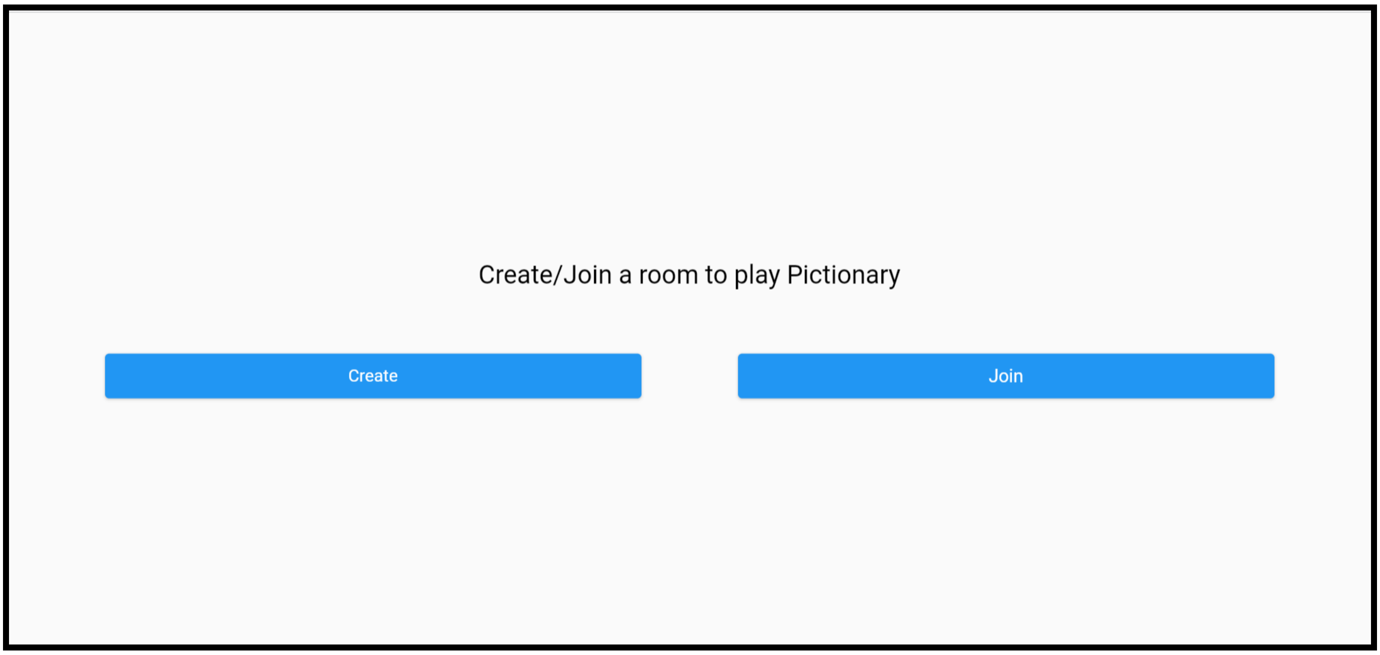
7. **Evaluation and Documentation:**

   In the final stages of the project, we documented every aspect of our development journey. This documentation includes details on the project's development, testing, and outcomes, providing a comprehensive record of our work

3.4 Details of Hardware & Software

Sr. No.	Name of Equipment	Specification	Quantity
1	Desktop with Windows 10 & above	4 GB RAM	1
2	Flutter	3.10.6	1
3	MongoDB	7.0	1
4	Node.js	18.18.0	1
5	Socket.io	4.7.2	1
6	VS Code	1.83.1	1

3.5 Results



Create/Join a room to play Pictionary

Figure 3.5.1 Main Screen

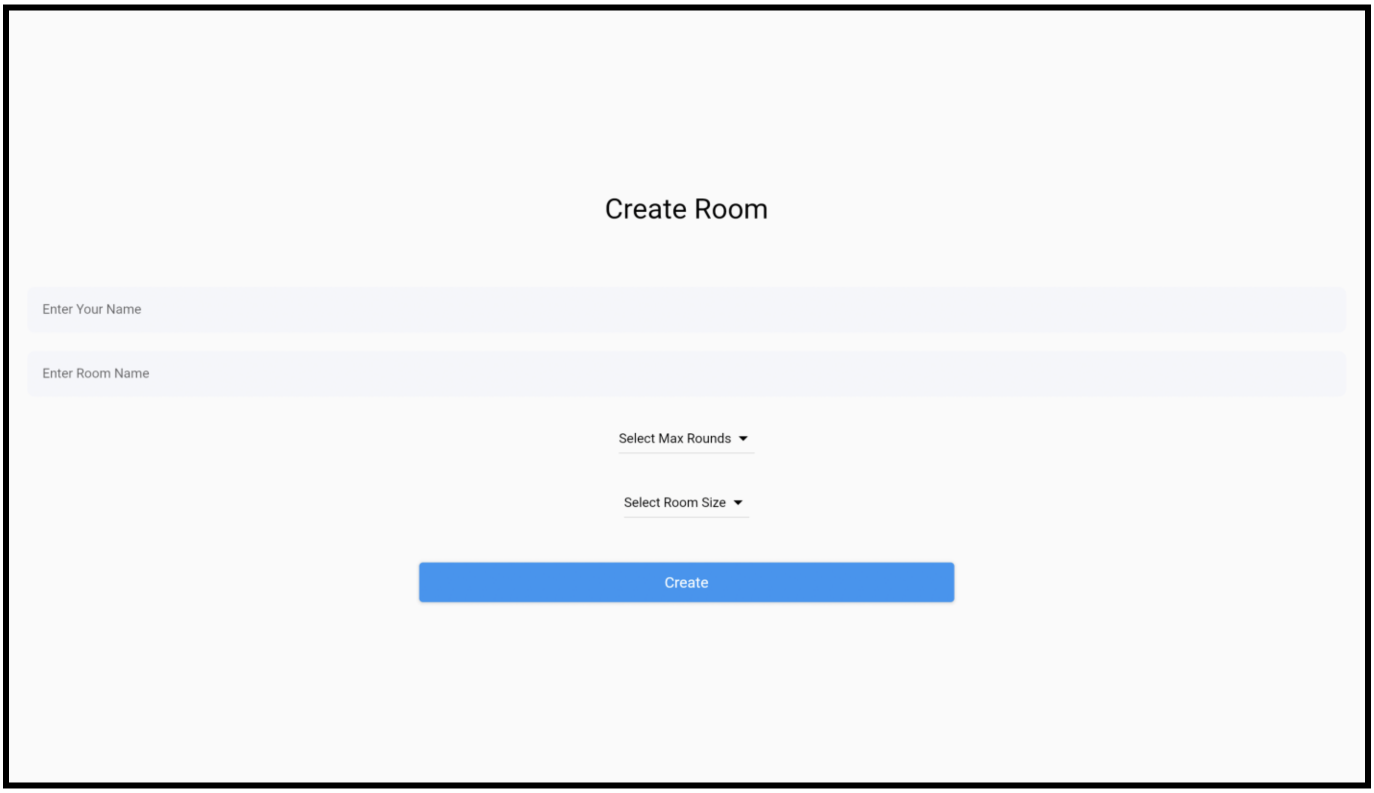


Figure 3.5.2 Create Room Screen

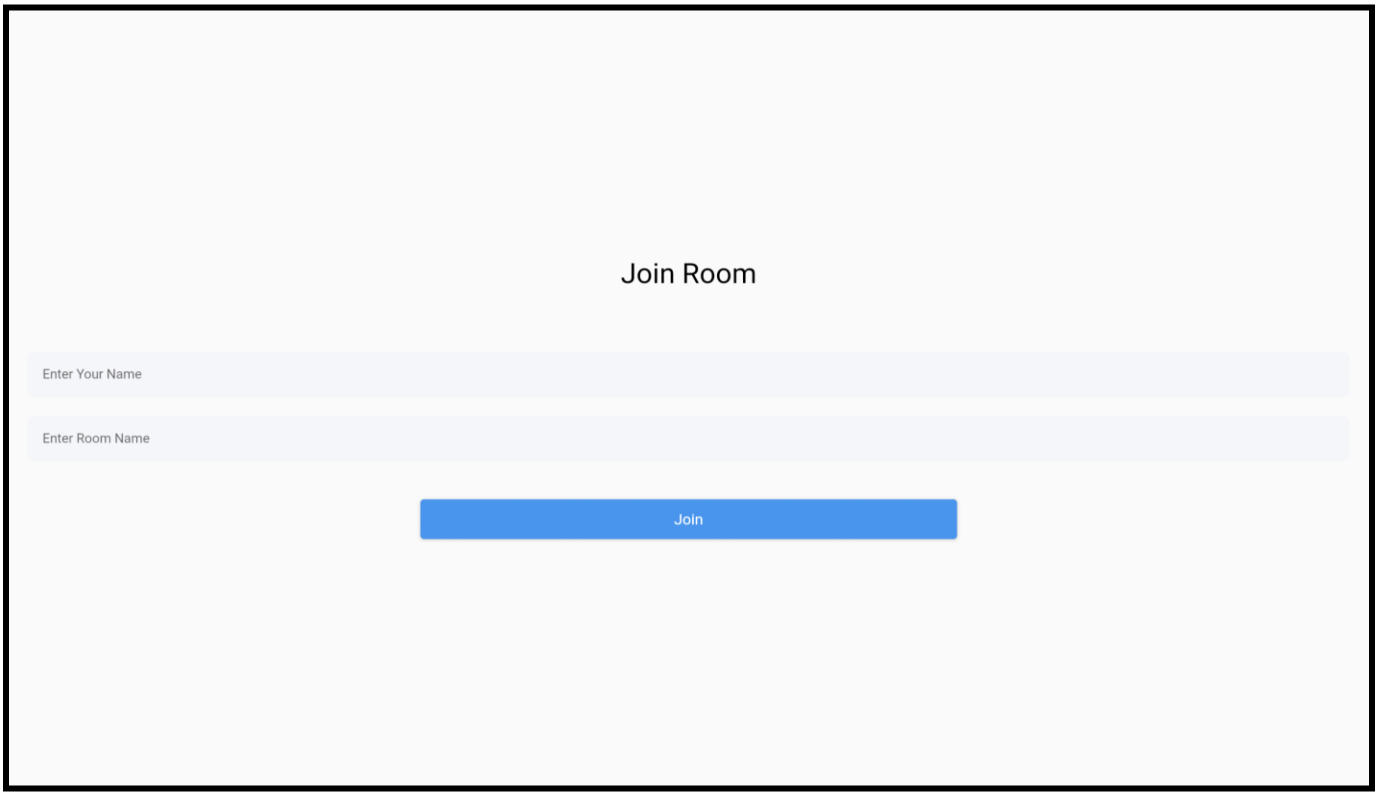
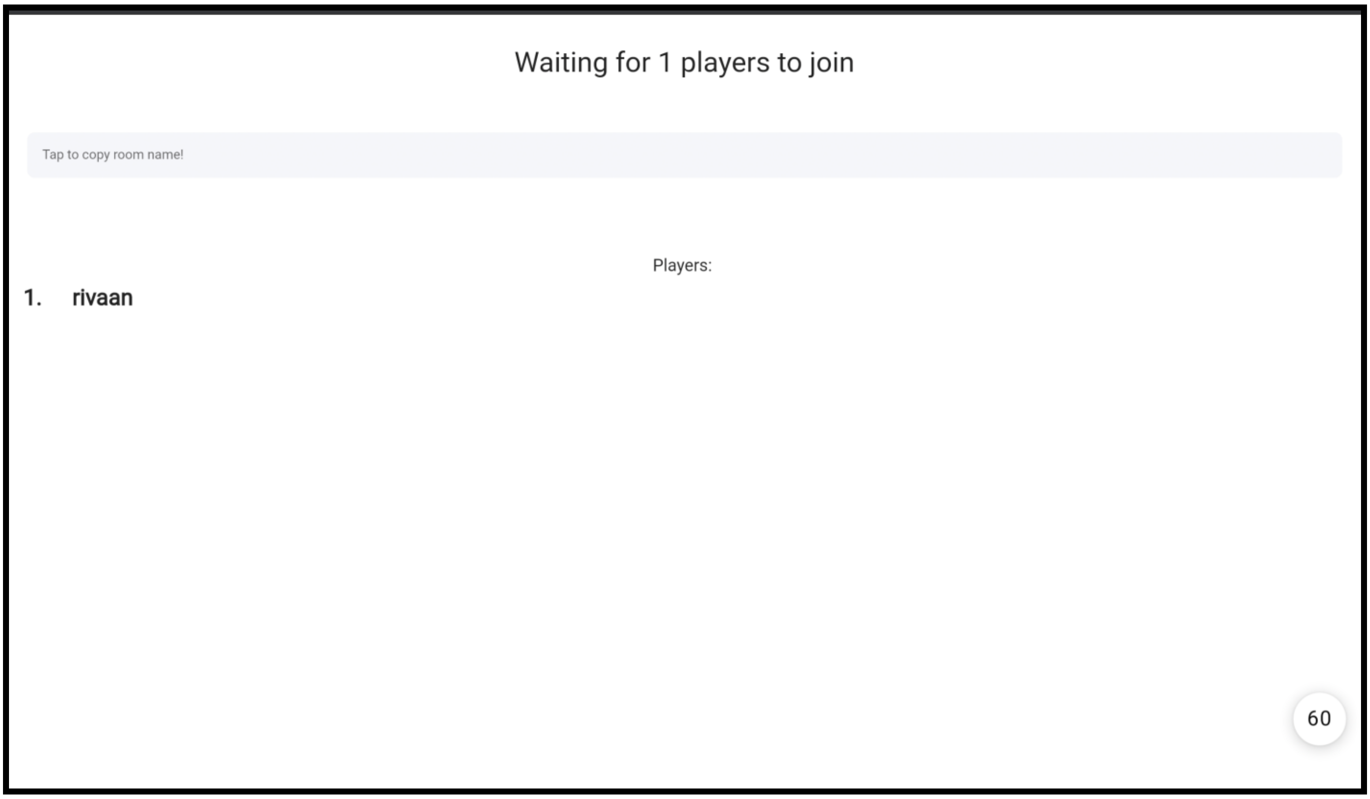


Figure 3.5.3 Join Room Screen



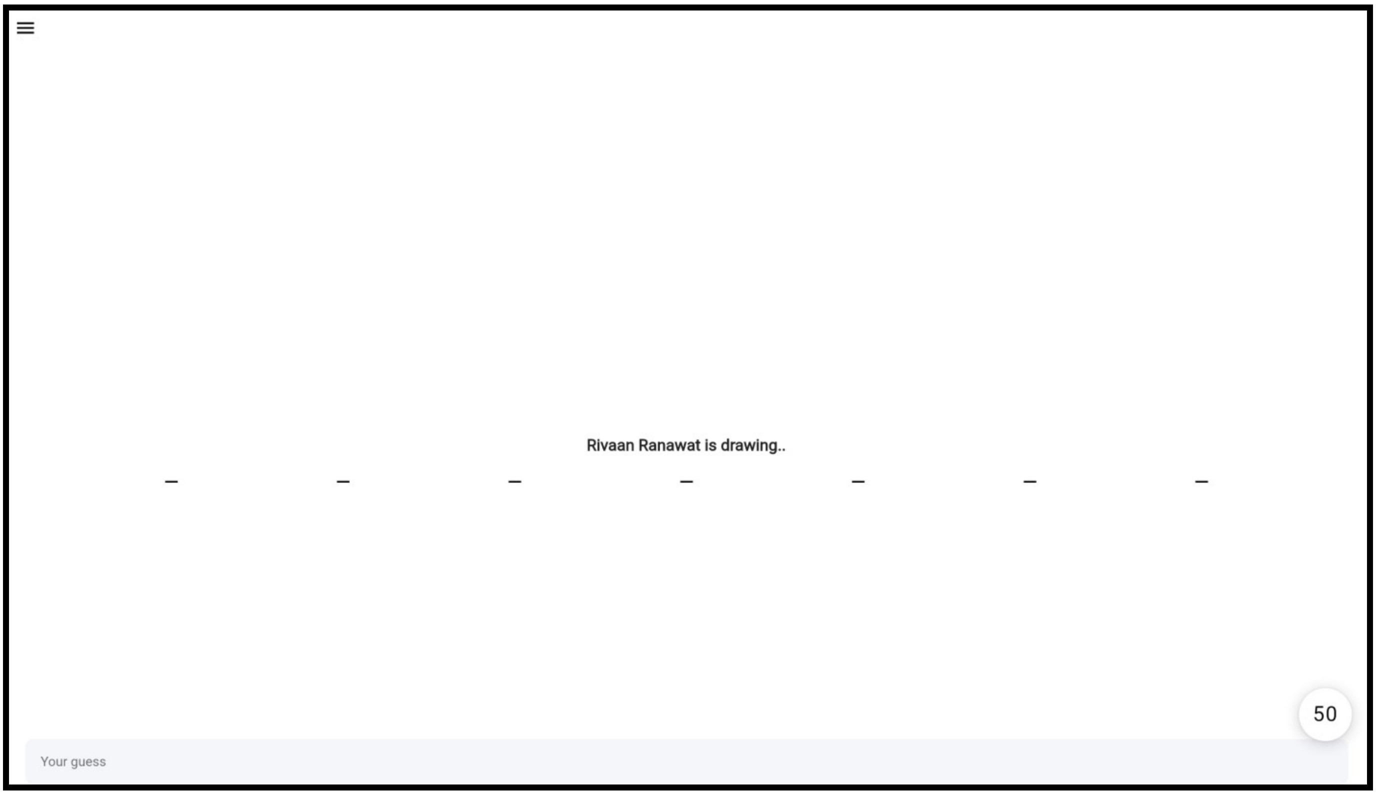
Waiting for 1 players to join

Tap to copy room name!

Players:

1. rivaan

Figure 3.5.4 Waiting Lobby Screen

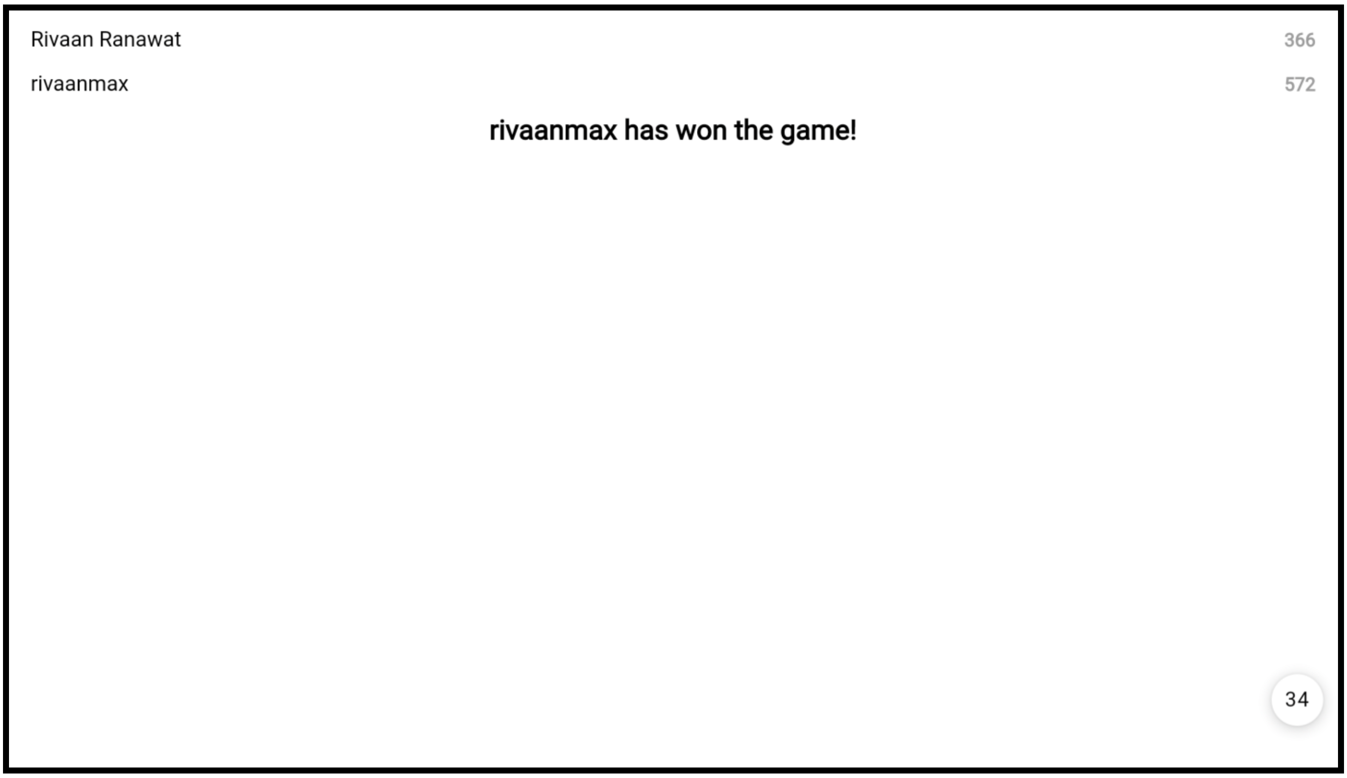


Rivaan Ranawat is drawing..

_ _ _ _ _ _ _

Your guess

Figure 3.5.5 Playing Area Screen



Rivaan Ranawat	366
rivaanmax	572

rivaanmax has won the game!

Figure 3.5.6 Leaderboard Screen

3.6 Conclusion and Future Work

The Pictionary project has successfully realized the vision of creating an engaging and interactive digital drawing and guessing game, providing users with an enjoyable gaming experience.

Our project's core objective was to bridge the gap between the classic board game and the digital world, fostering creativity, teamwork, and fun among players.

Through the collaborative efforts of our team, we have not only developed a captivating game but also gained valuable insights into modern app development, UI/UX design, and teamwork dynamics.

As we conclude the initial phase of our Pictionary project, we envision several exciting avenues for future development and enhancement:

- Multiplatform Accessibility: Expanding the game's compatibility to various platforms, including web browsers and desktop applications, will enable a broader audience to enjoy Pictionary.
- Community and Social Integration: Incorporating social features like player profiles, leaderboards, and real-time multiplayer interactions will enhance the social aspect of the game.

- Advanced Gameplay Features: We can introduce advanced gameplay features, such as themed drawing categories, tournament modes, and interactive challenges, to keep players engaged and entertained.
- User-Generated Content: Allowing users to create and share their custom word lists, drawings, and game modes will foster a vibrant and creative Pictionary community.

References

1. https://flexiple.com/flutter/deep-dive
2. https://nodejs.org/en/about/
3. https://www.geeksforgeeks.org/what-is-mongodb-working-and-features/
4. https://www.mongodb.com/
5. https://socket.io/docs/v4/
6. https://en.wikipedia.org/wiki/Socket.IO