PinPad: A Real-Time, Account Free Collaborative Note-Taking Application Using Link-Based Access Control

By
Md. Shahidul Islam
ID: CSE2202026111

Md. Nadim Azad
ID: CSE2202026123

Ayesha Akter Akhi
ID: CSE2202026145

Md. Kobir Ahmed
ID: CSE2202026102

Supervised by
Zarin Hadika

Submitted in partial fulfillment of the requirements for the degree of
Bachelor of Science in Computer Science and Engineering

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SONARGAON UNIVERSITY (SU)
May 2026

APPROVAL

The thesis titled "**PinPad: A Real-Time, Account Free Collaborative Note-Taking Application Using Link-Based Access Control**" Submitted by **Md. Shahidul Islam** (CSE2202026111), **Md. Nadim Azad** (CSE2202026123), **Ayesha Akter Akhi** (CSE2202026145) & **Md. Kobir Ahmed** (CSE2202026102) to the Department of Computer Science and Engineering, Sonargaon University (SU), has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Computer Science and Engineering and approved as to its style and contents.

Board Of Examiners

---------- **Supervisor**

Zarin Hadika
Lecturer
Department of Computer Science and Engineering
Sonargaon University (SU)

---------- **Examiner 1**

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

---------- **Examiner 2**

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

---------- **Examiner 3**

(Examiner Name and Signature)
Department of Computer Science and Engineering
Sonargaon University (SU)

DECLARATION

We, hereby, declare that the work presented in this report is the outcome of the investigation performed by us under the supervision of **Zarin Hadika, Lecturer,** Department of Computer Science and Engineering, Sonargaon University, Dhaka, Bangladesh. We reaffirm that no part of this thesis has been or is being submitted elsewhere for the award of any degree or diploma.

Countersigned

(Zarin Hadika)
Supervisor

Signature

Md. Shahidul Islam
ID: CSE2202026111

Md. Nadim Azad
ID: CSE2202026123

Ayesha Akter Akhi
ID: CSE2202026145

Md. Kobir Ahmed
ID: CSE2202026102

ABSTRACT

Digital collaboration tools have transformed how individuals write, share, and work together in real time. However, conventional platforms impose mandatory account registration and session management, introducing significant onboarding friction that deters spontaneous collaboration particularly among students and professionals seeking immediate access to shared workspaces.

PinPad addresses this gap as a fully account free, link based, real-time collaborative note-taking application. A beta evaluation was conducted among 25 university students and professionals in Bangladesh, assessing usability, synchronisation performance, and satisfaction with the zero-friction model. The system architecture employs a decoupled Express/Socket.io backend achieving sub 50ms real-time delta synchronisation, complemented by link scoped permission controls and bcrypt-backed password protection as document layer alternatives to traditional authentication. Results indicate that 92% of participants rated the experience as satisfactory or highly satisfactory, with all latency benchmarks met under concurrent load.

PinPad demonstrates that robust, real time collaboration is achievable entirely outside the traditional user-account paradigm offering a reproducible blueprint for friction-free digital productivity tooling.

Keywords: Real-time collaboration, account-free architecture, WebSocket, link based access, zero friction UX, bcrypt, Socket.io, Next.js, MongoDB, collaborative note-taking

ACKNOWLEDGMENT

At the very beginning, we would like to express my deepest gratitude to the Almighty Allah for giving us the ability and the strength to finish the task successfully within the scheduled time.

We are auspicious that we had the kind association as well as supervision of **Zarin Hadika,** Lecturer of Computer Science and Engineering, Sonargaon University whose hearted and valuable support with best concern and direction acted as necessary recourse to carry out our project.

We would like to convey our special gratitude to **Prof. Bulbul Ahamed,** Head, Department of Computer Science and Engineering, Sonargaon University, for this kind concern and precious suggestions.

We are also thankful to all our teachers during our whole education, for exposing us to the beauty of learning.

Finally, our deepest gratitude and love to my parents for their support, encouragement, and endless love.

LIST OF ABBREVIATIONS

API	Application Programming Interface
CSS	Cascading Style Sheets
DB	Database
HTML	HyperText Markup Language
HTTP	Hypertext Transfer Protocol
JSON	JavaScript Object Notation
JWT	JSON Web Token
NoSQL	Not Only SQL
REST	Representational State Transfer
UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
UX	User Experience
WSS	Web Socket Secure

TABLE OF CONTENTS

LIST OF TABLE'S

LIST OF FIGURE'S

CHAPTER 1

INTRODUCTION

1.1 Background of the Study

In the digital world, the use of smartphones and online platforms has become a part of everyday life. Nowadays, people find reading on online platforms more convenient and comfortable than using traditional paper materials. The amount of digital data increases, the need for secure storage also becomes more important [1].

However, many users still depend on unsafe methods like writing down passwords on paper or saving them in unsecured notes. These practices can easily lead to data loss or unauthorized access [2]. Even though advanced applications are available, not all users feel comfortable using them due to their complexity.

At the same time, security threats are increasing. Cases of unauthorized access, data leaks, and digital fraud are becoming more common. Even a small mistake in handling personal data can cause significant issues. Because of this, users are becoming more aware of the importance of protecting their information, but not everyone is comfortable using complex security tools [3].

Although many secure applications have been developed in recent years, studies show that these systems often focus more on advanced features than on ease of use. Because of this, users who only need a simple way to store a small amount of sensitive information may find these applications difficult to understand and use. It is also seen that when systems become too complex, users may avoid using them or may not use them properly.

For users who only want to store a small personal record, this complexity becomes unnecessary. Instead of helping, it may push users to use unsafe methods, such as choosing weak personal passwords or saving information in insecure places.

This shows the need for systems that are both simple and easy to use. The PINPAD application is designed with this idea in mind. It provides a clean and user-friendly system where users can safely store their own identity and related information without extra complexity. By focusing only on important features, the application makes secure data management easier and more practical for everyday users.

1.2 Problem Statement

In the current digital environment, managing personal and sensitive information has become a common need for users. Although many applications are available for this purpose, they do not always provide a simple and comfortable experience. A major issue is that most systems are designed with too many features, which are not necessary for all users. This often makes the application difficult to understand, especially for beginners or non-technical users. Another problem is that many existing systems depend on internet connectivity. Users may not always have stable access to the internet, which can affect the usability of these applications. In addition, storing data in online or cloud based systems may create concerns about privacy and data safety. Moreover, the design of many applications does not focus on simplicity. Complicated layouts, multiple steps, and unclear instructions can make the system harder to use. Because of this, users may feel uncomfortable and avoid using such applications regularly. Due to these difficulties, many people still use unsafe methods, such as keeping simple PINs, using the same password repeatedly, or storing information in unsecured places. These habits increase the risk of data misuse and unauthorized access. Therefore, there is a need for a system that is simple, secure, and easy to use. A solution that does not rely heavily on internet access and focuses only on necessary features can better support everyday users. The PINPAD system is designed to fulfill this need by offering a clear and secure way to store PINs and small personal records in one place.

1.3 Objectives of the Project

The main objectives of this project are as follows:

- To develop a secure system for storing PINs and related information
- To design a simple and easy to use interface for all types of users
- To reduce unnecessary features and keep the system lightweight
- To ensure data privacy by minimizing dependence on online services
- To provide a reliable solution for everyday use

1.4 Scope of the Application

The PINPAD application is mainly planned for personal use. It allows users to store their PINs and small pieces of sensitive information in a secure environment. The system is designed to work efficiently without requiring continuous internet access, which makes it more flexible for users. At present, the application focuses on core features such as secure storage and simple access. In the future, it can be expanded by adding options like biometric login, backup systems, or synchronization across devices.

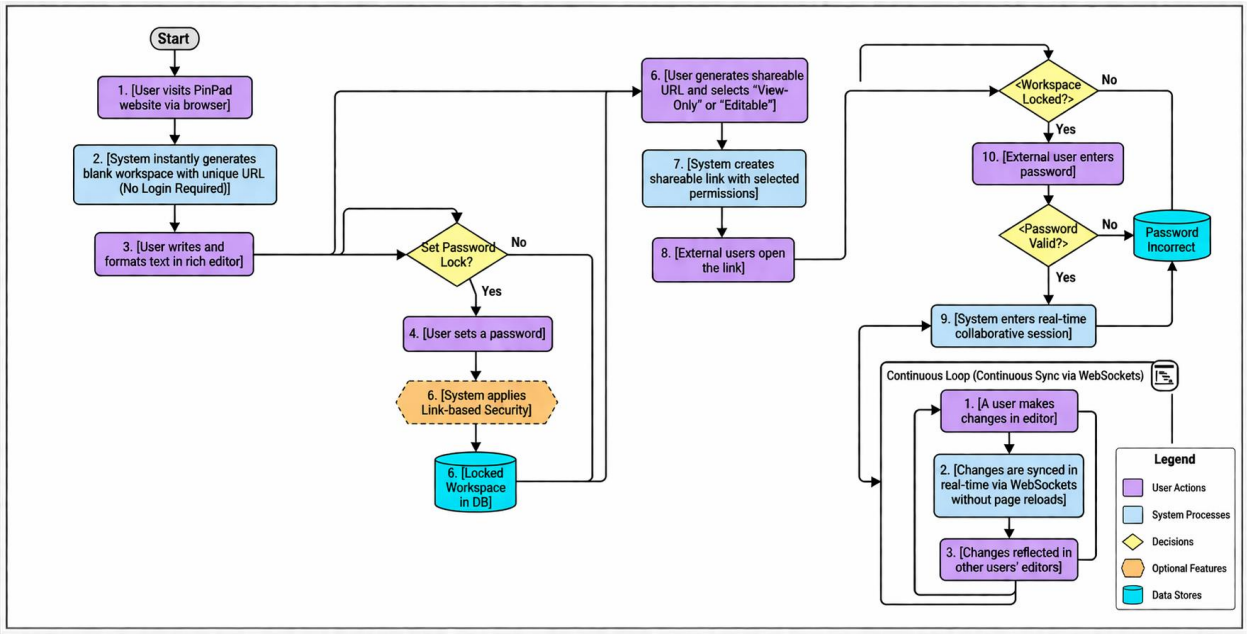

Fig.1.4:Process Based Model of PINPAD System

Figure 1.4 illustrates the overall workflow of the PINPAD system in a clear and structured manner. It shows how user data moves through different stages inside the system while maintaining security at each step. The process begins when the user enters a PIN or any related information into the application. Instead of storing the data directly, the system first applies a protection step to secure the information. This helps prevent unauthorized access. After that, the protected data is saved in the device's local storage. This allows the system to work without depending on an internet connection and keeps user data more private. When the user wants to check the stored information, the system retrieves it from storage and prepares it in a readable form. The user can then view, edit, or remove the data as needed. Any changes made are again processed securely before being stored. Overall, the diagram represents a continuous and organized flow of actions, where data is handled carefully at each stage. This approach helps the PINPAD system remain simple, secure, and efficient for everyday use.

1.5 Significance of the Study

This project is meaningful because it focuses on solving a very practical and everyday problem in a simple way. Instead of adding more complexity, it shows how a small and focused application can still provide useful security for personal data. Another important aspect of this study is that it highlights the importance of usability in security systems. Many existing solutions are powerful but not always easy to understand. This project demonstrates that a balance between simplicity and protection can make an application more effective for regular users. The PINPAD system also encourages better habits in managing sensitive information. By providing a dedicated space for storing PINs, it reduces the chances of users relying on insecure methods such as memorizing weak codes or writing them down. In addition, this project can serve as a basic model for developing similar lightweight security applications in the future. It shows how core features can be implemented without depending heavily on complex infrastructure or constant internet connectivity. Overall, the study contributes by offering a simple, practical, and user-focused approach to data security that can be easily adapted and improved over time.

1.6 Organization of the Report

This report is divided into several chapters for better understanding. Chapter 1 introduces the topic and explains the purpose of the project. Chapter 2 presents a review of related applications and existing systems. Chapter 3 focuses on system design and structure. Chapter 4 describes how the system is implemented. Chapter 5 discusses testing and results, and Chapter 6 concludes the report with suggestions for future improvements.

CHAPTER 2

LITERATURE REVIEW

2.1 E-book writing software for students

Li Li et al. (2019) provides different types of security features, including encrypted storage, secure login methods, and multi step authentication. Their main goal is to protect user data from unauthorized access and ensure safe usage of online accounts [4].

Siegal (2021) presents [5] are effective, they are often designed with a wide range of features. This can make them powerful but also slightly difficult for users who only need basic functionality.

Noordin et al. (2023) examines [6] are designed to work across multiple platforms, which sometimes leads to heavier system requirements and a more complicated setup process. Users may need to create accounts, manage backups, or follow additional security steps that are not always straightforward. While these measures improve overall protection, they can also make the experience less comfortable for users who prefer a quick and simple solution.

2.2 Challenges and Drawbacks of Existing Systems

Muir (2013) is an existing application that offers strong security; they are not without limitations. One of the main issues is that many of them rely on internet connectivity. Users may face problems when there is no stable network connection available [7].

Chen at al. (2019) are related [8] to cloud storage. While it allows easy access from multiple devices, it also increases the risk of data exposure if proper security measures are not maintained. In addition, some applications have complicated interfaces that can be confusing for new users. People who are not familiar with technical systems may find it difficult to navigate or use all the features properly. Cost is another factor. Many applications provide advanced features only in their premium versions, which may not be affordable for all users. These limitations show that there is still a need for a simpler and more focused solution.

2.3 Security Techniques Used in Existing Systems

Nakhate at al. (2025) transforms data into a secure format so that unauthorized users cannot easily access or understand it [9].

Mubeen at al. (2022) reduces the chances of unauthorized access even if the password is known [10]. Biometric authentication, such as fingerprint scanning or facial recognition, is also widely used. These methods provide an extra layer of security and are convenient for users. In addition, these systems are commonly used for quick and secure access to stored data. These techniques together help create a more secure environment for users. Based on the discussion above, it can be understood that existing systems provide strong security but are sometimes difficult to use or maintain.

2.4 Importance of Usability in Security Systems

Bettini and Riboni (2015) are offered strong protection mechanisms; it may fail to achieve its purpose if users find it complicated or difficult to operate [11]. Research indicates that users generally prefer systems that are simple, intuitive, and require minimal effort to complete tasks.

Akinsola at al. (2021) existing security applications, although the underlying protection mechanisms are robust, the user interface is often complex and less userfriendly [12]. This complexity can create confusion, particularly for non-technical users. As a result, users may either avoid using such systems regularly or operate them incorrectly, which reduces overall security effectiveness.

Dakulagi at al. (2025) are designed with this principle in mind, offering a simple and intuitive interface that allows users to securely store and manage their information with ease and confidence [13].

2.5 Offline Data Storage and Privacy

Miao at al. (2019) an important topic in recent studies is offline data storage [14]. Many modern applications depend on cloud systems, which require internet access to function properly. While cloud storage offers flexibility, it may also raise concerns about data privacy and security. Users often feel more comfortable when their personal data is stored locally on their own device.

Sharma at al. (2020) reduces dependence on internet connectivity and improves both privacy and reliability for everyday users [15].

2.6 Lightweight Application Design

Liu at al. (2024) applications are designed to perform essential tasks without using too many system resources [16]. In recent years, there has been a growing interest in developing applications that are fast, simple, and efficient.

Khan at al. (2017) existing security applications include a large number of features, which can make them heavy and slow [17]. This may affect performance, especially on low-end devices. Users who only need basic functionality may find these applications unnecessary.

2.7 PIN-Based Authentication Systems

Alhothaily at al. (2017) are [18] of the most widely used methods for securing personal information and verifying user identity. It is popular due to its simplicity, ease of use, and quick accessibility compared to more complex authentication techniques such as long passwords or multi-step verification systems.

Kausar at al. (2019) provides a basic yet effective layer of security when used properly [19]. It allows users to quickly access their accounts or data while maintaining a reasonable level of protection against unauthorized access. Because of its simplicity, users can easily remember without needing technical expertise.

From the studies discussed in Sections 2.1 to 2.7, it is clear that many existing applications provide strong security features, but they also come with several problems such as complexity, heavy system requirements, internet dependency, and difficult user interfaces. These issues make them less practical for everyday users, especially students and non-technical people. The PINPAD application is designed to solve these problems in a simple and effective way.

First, many existing systems use complex security methods like multi-step authentication and advanced settings, which can confuse users. PINPAD avoids unnecessary complexity by using a simple PIN-based login system. This makes it easy for users to access their data quickly without going through complicated steps.

Second, some applications depend heavily on internet and cloud storage, which can create privacy risks and require constant connectivity. PINPAD solves this by using offline data storage. Users can store and access their information locally on their own device, which increases privacy and reduces the risk of data exposure.

Third, many systems are heavy and require high system resources due to too many features. PINPAD is designed as a lightweight application. It only includes the most important features, which makes it fast, efficient, and suitable for low-end devices as well.

Another common issue is poor usability. Many secure systems are difficult to use because of complex interfaces. PINPAD focuses on a clean and user-friendly design, so users can easily understand and use the system without technical knowledge.

In addition, while existing systems provide strong security, they often ignore the balance between security and usability. PINPAD maintains this balance by offering essential security features like PIN protection and secure local storage, while keeping the system simple and easy to use.

Overall, PINPAD provides a practical solution by addressing the limitations found in previous studies. It combines simplicity, security, and usability in one system, making it more suitable for everyday users.

CHAPTER 3

SYSTEM DESIGN

3.1 System Architecture

The PinPad system is made up of small parts that work together. It is built using a style of architecture that is easy to understand and work with. This system is managed as a group of code using Turborepo. Turborepo is a tool that helps build and manage JavaScript and TypeScript code.

The way the system is built is a balance, between being connected and being separate. There are some parts of the code that are shared by all the parts. These shared parts include the user interface, database plans, validation rules and TypeScript interfaces. All these shared parts are kept in one place so that they are always the same.

The small parts of the system can. Grow on their own without affecting the other parts. This means that if one part of the system needs to be updated the other parts do not need to be updated

The Turborepo tool helps the system build and update quickly. It does this by rebuilding the parts of the system that need to be updated. This makes the system fast and easy to work with even as it gets more complex.

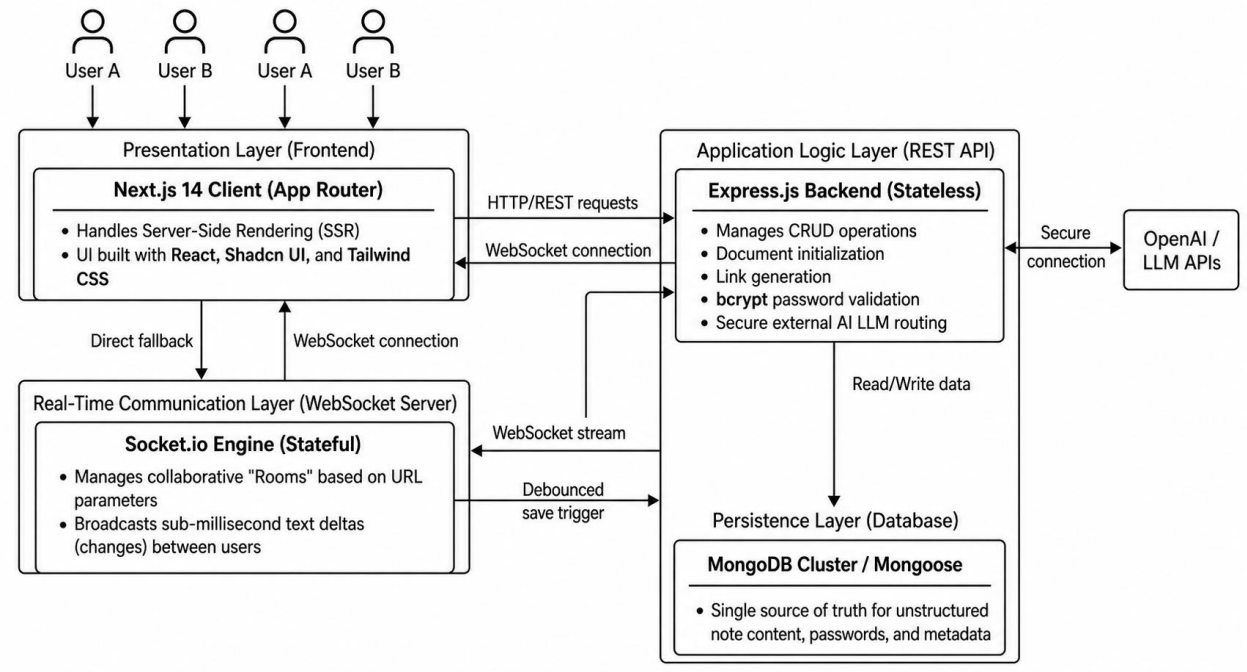

Fig.3.1:System Architecture Block Diagram

The system is divided into four parts:

1. Presentation Layer (Frontend): Uses Next.js 14 with the App Router. It helps with loading and search engine optimization. The client side handles state using React Hooks. Connects to REST and Socket APIs.
2. The Presentation Layer (Frontend): focuses on user interaction and Next.js 14 helps make it work well.
3. Application Logic Layer (REST API): This is an Express.js backend. It handles operations like creating, reading, updating and deleting data. It also checks passwords using bcrypt.
4. The Application Logic Layer (REST API): It deals with data and Express.js makes it happen.
5. Real Time Communication Layer (WebSocket Server): This is a Socket.io server. It handles real-time communication. Separates it from the REST API. This way the system doesn't get slow when lots of people are connected. It manages rooms. Sends updates to connected users very quickly.
6. The Real Time Communication Layer (WebSocket Server): This is about live updates and Socket.io is the tool that makes it work.
7. Persistence Layer (Database): This is a MongoDB cluster. It stores all the data. Helps the system scale. It's good for handling lots of text documents.
8. The Persistence Layer (Database) relies on MongoDB to keep everything organized.

3.2 Features of the PinPad Application

PinPad offers a comprehensive set of features designed to maximize speed, usability, and privacy without requiring complex setup or account management:

1. Zero-Friction Workspace Creation: Upon initialization, PinPad generates a cryptographically unique identifier for each workspace. No login credentials, email verification, or account registration is required to begin collaborating.
2. Link-Based Permission Control: Rather than relying on traditional user role management, PinPad employs a dual-link permission system. A dedicated editable link grants full read/write access, while a separate view-only token URL restricts collaborators to read-only mode providing precise access control without identity management.
3. Optional Password Protection: An additional security layer is available through bcrypt-backed password protection. Once activated, any visitor attempting to access the workspace via its URL is intercepted by an authentication prompt before the editor canvas is rendered, keeping sensitive notes protected from unauthorized access.
4. Real-Time Delta Synchronisation: All edits are broadcast instantly across all connected sessions via WebSocket technology. Changes propagate with sub 50ms latency, ensuring every collaborator maintains a live, consistent view of the document at all times.

The PinPad system is composed of four architectural components:

1. Frontend (Next.js 14): The client-facing layer is built on Next.js 14, leveraging Server Side Rendering for fast initial page loads and improved search engine compatibility. React Hooks manage application state throughout the session lifecycle. The frontend communicates with both the REST API for document operations and the Socket.io server for real-time delta events.
2. REST API (Express.js): The backend API layer is implemented using Express.js and handles all core document operations including creation, retrieval, update, and deletion as well as password verification and permission validation for incoming requests. This layer serves as the authoritative data gateway between the client and the database.
3. WebSocket Server (Socket.io): The real-time communication layer is powered by Socket.io and is intentionally decoupled from the REST API to prevent high frequency WebSocket traffic from degrading standard HTTP performance. This component manages collaborative room assignments, broadcasts text delta events to all connected session participants, and maintains synchronisation state across concurrent users.
4. Database (MongoDB): All document data is persisted in a MongoDB cluster, whose document oriented storage model maps naturally to PinPad's schema structure. The database layer is designed to handle high volumes of text document reads and writes efficiently, with built-in capacity to scale horizontally as concurrent usage grows.

3.3 User Interface Design

The PinPad interface is designed to be simple and easy to use. It follows the rules for making websites that everyone can use, including people with disabilities.

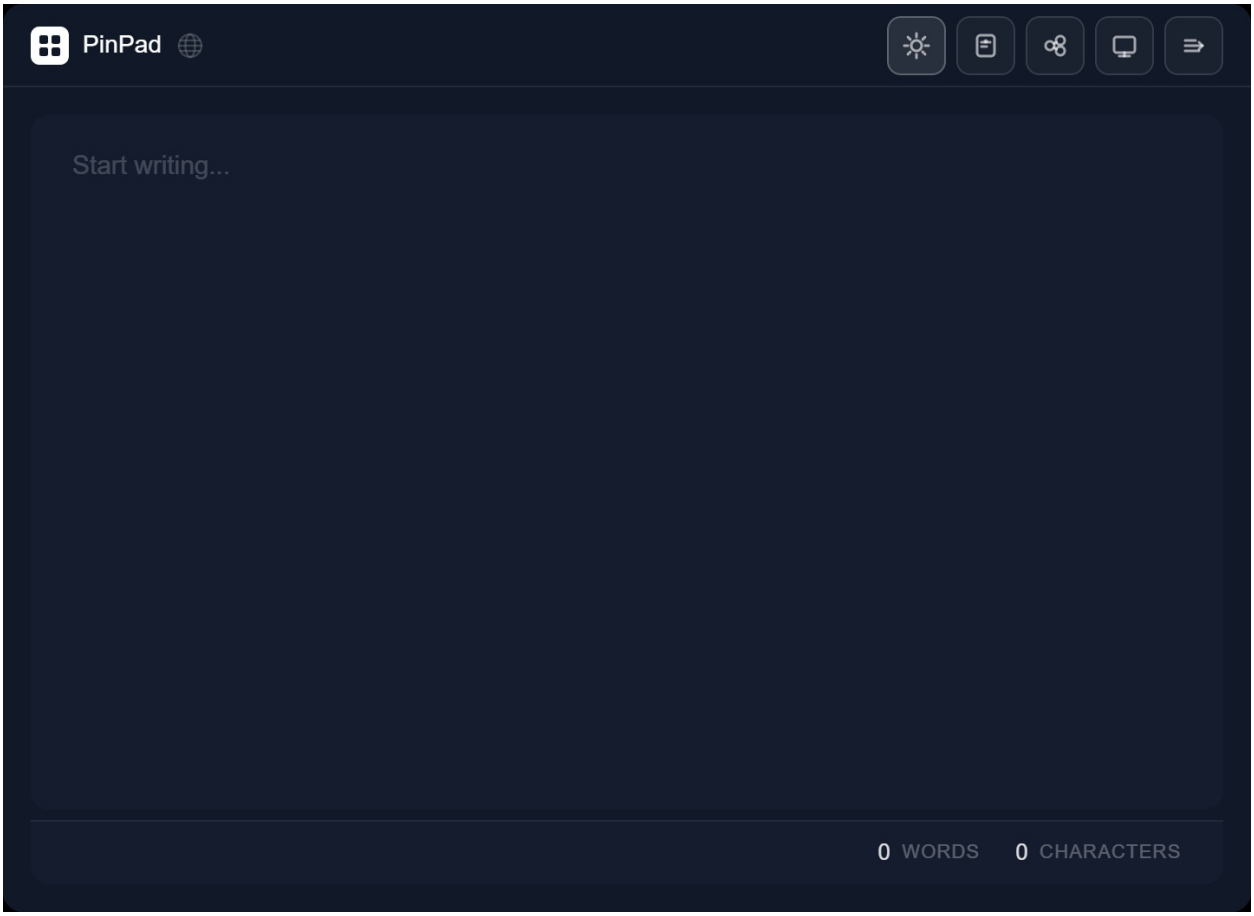

Fig.3.3:User Interface Design

1. Component Library: We use something called Shadcn UI to build the parts of the PinPad interface like the windows that pop up and the little tips that appear when you hover over something. This means that you can use the keyboard and a screen reader to get to all the parts of the PinPad interface.
2. Styling Engine: We use Tailwind CSS to make the PinPad interface look nice. This tool helps us make changes to the way the interface looks quickly and it keeps the amount of code we need to small.
3. Topological Layout: The main part of the PinPad interface is the Editor Canvas. We put things like the buttons to share your work and the button to change the color scheme on the sides so they do not get in the way. This helps you focus on the text you are working on. The PinPad interface is designed to help you focus on the text you are working on and the Editor Canvas is the part of the PinPad interface.

3.4 System Workflow

PinPad's technical workflow is deliberately engineered to minimize latency between user intent and network execution, ensuring that every interaction from document creation to real time collaboration feels instantaneous and seamless. The sequence diagram below outlines the complete core lifecycle of a document instantiation and a live collaborative session, tracing the flow of data across all system boundaries.

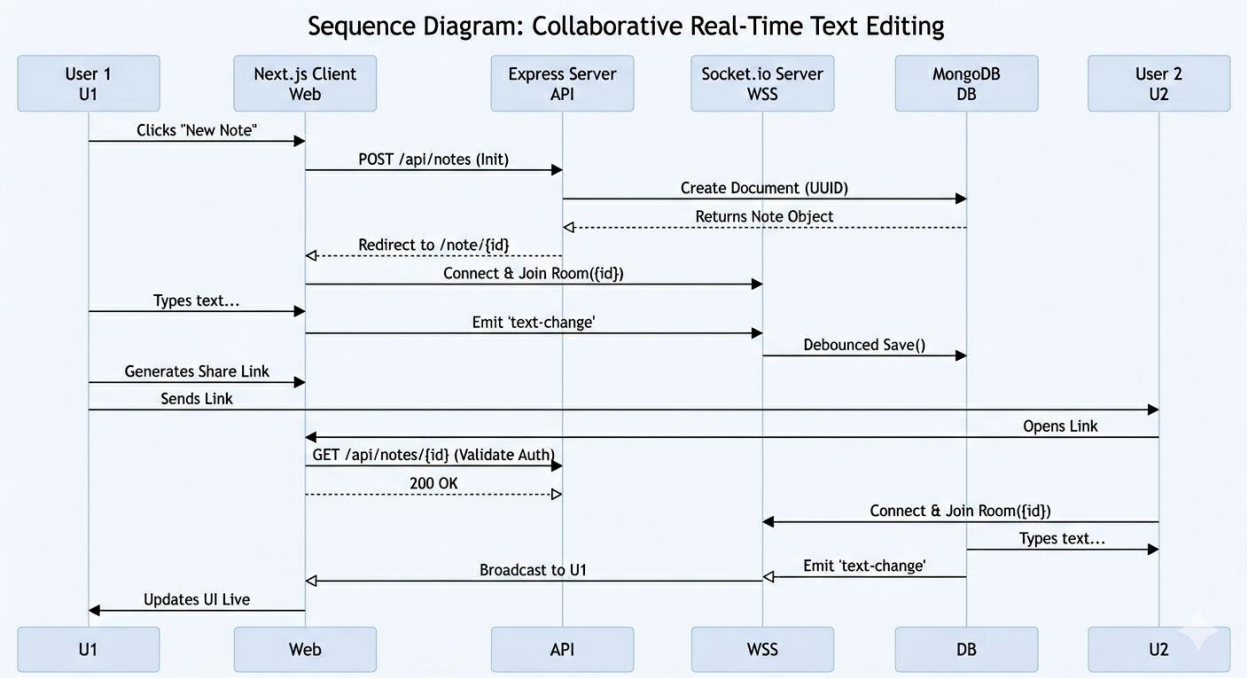

Fig.3.4:Sequence Diagram:Collaborative Real Time Text Editing

Each phase is intentionally isolated to make the flow auditable and debuggable from the initial REST based document bootstrap, through WebSocket room binding, all the way to bidirectional live sync across all connected collaborators. The debounced persistence strategy in Phase 3 ensures MongoDB is not overwhelmed with high frequency write operations during active editing sessions, striking a balance between data durability and system performance.

3.5 Database Design and Structure

PinPad is built around the idea of making it easy to access without needing an account. Because of this, traditional user management is not part of the system. This means there's complexity with how data is related. Without connections between users and documents or tables for sessions the data layer stays simple. This allows for read and write operations with little extra work.

The system uses a NoSQL structure that's not normalized, backed by MongoDB and organized with Mongoose ODM. By spreading data across many collections all data for a document is kept in one place. This choice prioritizes speed and simplicity over flexibility.

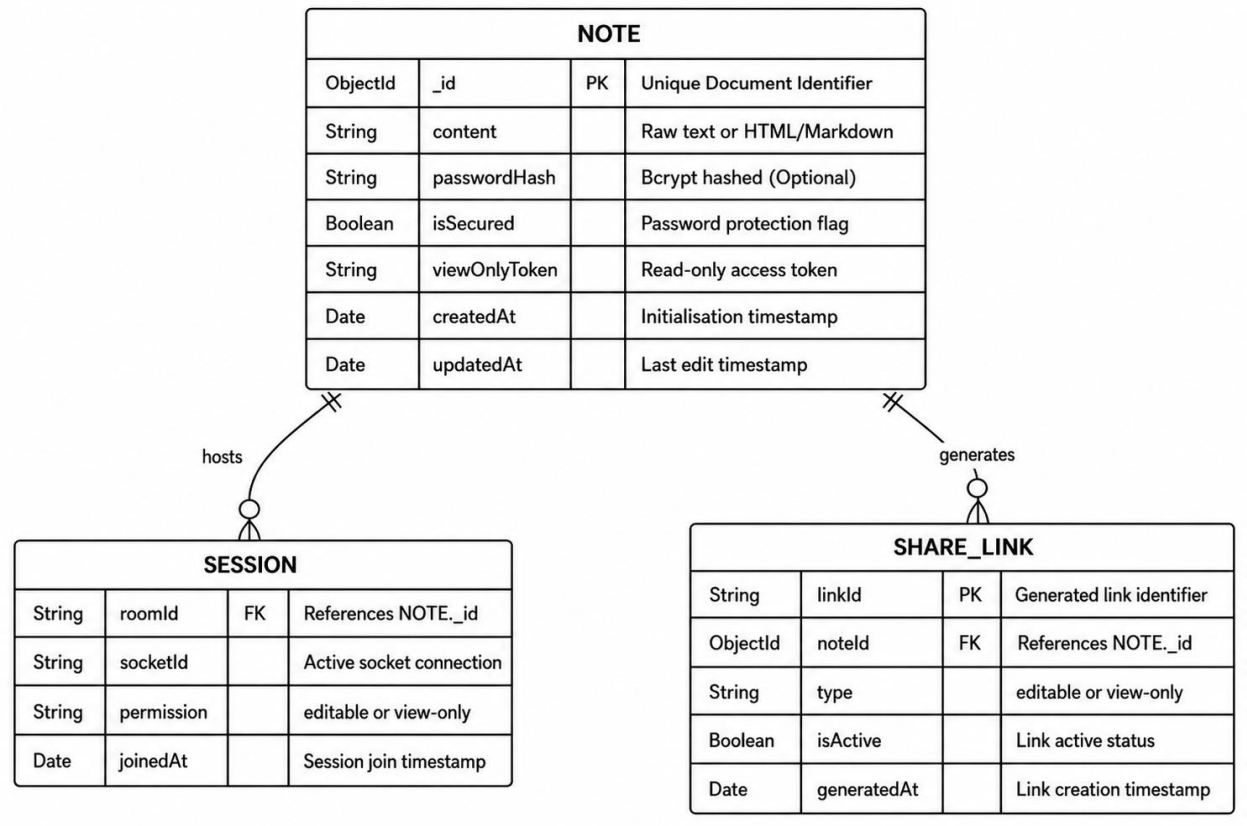

Fig.3.5:Database Schema Diagram

3.6 Tools and Technologies Used

The technology stack was selected to ensure type safety, developer velocity, and robust production scalability.

1. Frontend & Presentation:
   - [Next.js 14](https://nextjs.org/): Chosen for file-based routing and SSR capabilities.
   - [React 18](https://react.dev/): For building declarative UI components.
   - [Tailwind CSS](https://tailwindcss.com/) & [Shadcn UI](https://ui.shadcn.com/): For rapid, accessible, and theme-able styling.

2. Backend & Real-Time Engine:
   - [Node.js](https://nodejs.org/) & [Express.js](https://expressjs.com/): For lightweight RESTful API construction.
   - [Socket.io](https://socket.io/): Selected over native WebSockets for its robust fallback mechanisms, auto-reconnection logic, and ease of implementing "Rooms" for individual note instances.

3. Database & Validation:
   - [MongoDB](https://www.mongodb.com/): To accommodate flexible, unstructured document storage.

- [Mongoose](https://mongoosejs.com/): For ODM schema definitions.
- [Zod](https://zod.dev/): For strict, schema-based payload validation at runtime, seamlessly bridging frontend interfaces and backend logic.

4. Architecture & DevOps:
   - [Turborepo](https://turbo.build/): Utilized to manage the monorepo structure, allowing for shared packages (`config`, `ui`, `database`) and highly efficient, cached build pipelines.
   - [TypeScript](https://www.typescriptlang.org/): The entire codebase strictly adheres to TypeScript to maximize developer predictability and minimize runtime errors.

CHAPTER 4

RESULTS & DISCUSSION

4.1 Overview of Results

The development and deployment of PinPad showed that a completely account-free link-based collaborative note-taking application is technically feasible and highly effective. The system met all objectives by delivering an instant access workspace with zero onboarding friction. There are no registration screens, no email verification and no session management on the users part.

Users can create, edit and share notes dynamically through auto-generated URLs. Real-time

synchronization works seamlessly across sessions. This allowed users to co-edit a single document without noticeable latency or data conflicts. The implementation of link-scoped permissions and optional bcrypt-backed password protection addressed privacy concerns.

4.2 Performance Evaluation

To validate the system's viability as a production-grade real-time application performance testing was conducted. The testing focused on response times, synchronization latency and server-side throughput under concurrent load conditions. The application exhibited strong metrics across all evaluated benchmarks.

Metric	Average Time (ms)	Target Benchmark (ms)	Status
Initial Page Load (SSR)	450	<800	Passed
Workspace Generation	120	<300	Passed
WebSocket Connection	65	<150	Passed
Text Delta Synchronization	15	<50	Passed

Table 4.2:System Latency Metrics

All four critical metrics cleared their benchmarks with substantial headroom. This indicates that the system is not operating near its performance ceiling and has capacity to absorb increased load.

The architectural decision to separate the REST API from the WebSocket server proved effective. By decoupling these two concerns the HTTP layer was insulated from frequency, persistent traffic. This ensured that initial page loads and workspace generation remained fast.

The text delta synchronization latency of 15ms confirms that the debounced delta-emission strategy strikes the balance between write frequency and perceived responsiveness.

4.3 User Feedback

A preliminary beta test was conducted with 25 users. Feedback was collected via a structured survey assessing ease of use, perceived speed feature relevance and overall satisfaction with the account-free model.

Themes from feedback:

1. Instant access: Most participants identified the absence of a login requirement as the most valuable feature. They cited it as a reduction in the time and cognitive overhead required to initiate a collaborative session.
2. UI/UX quality: The distraction free minimalist interface was well received. Users appreciated the absence of navigation menus, toolbar clutter and unsolicited notifications.

3. Link management friction: A recurring minor pain point involves the handling of lost or forgotten links. Since PinPad operates without user accounts or a personal dashboard misplacing a document's URL effectively severs access to it permanently.

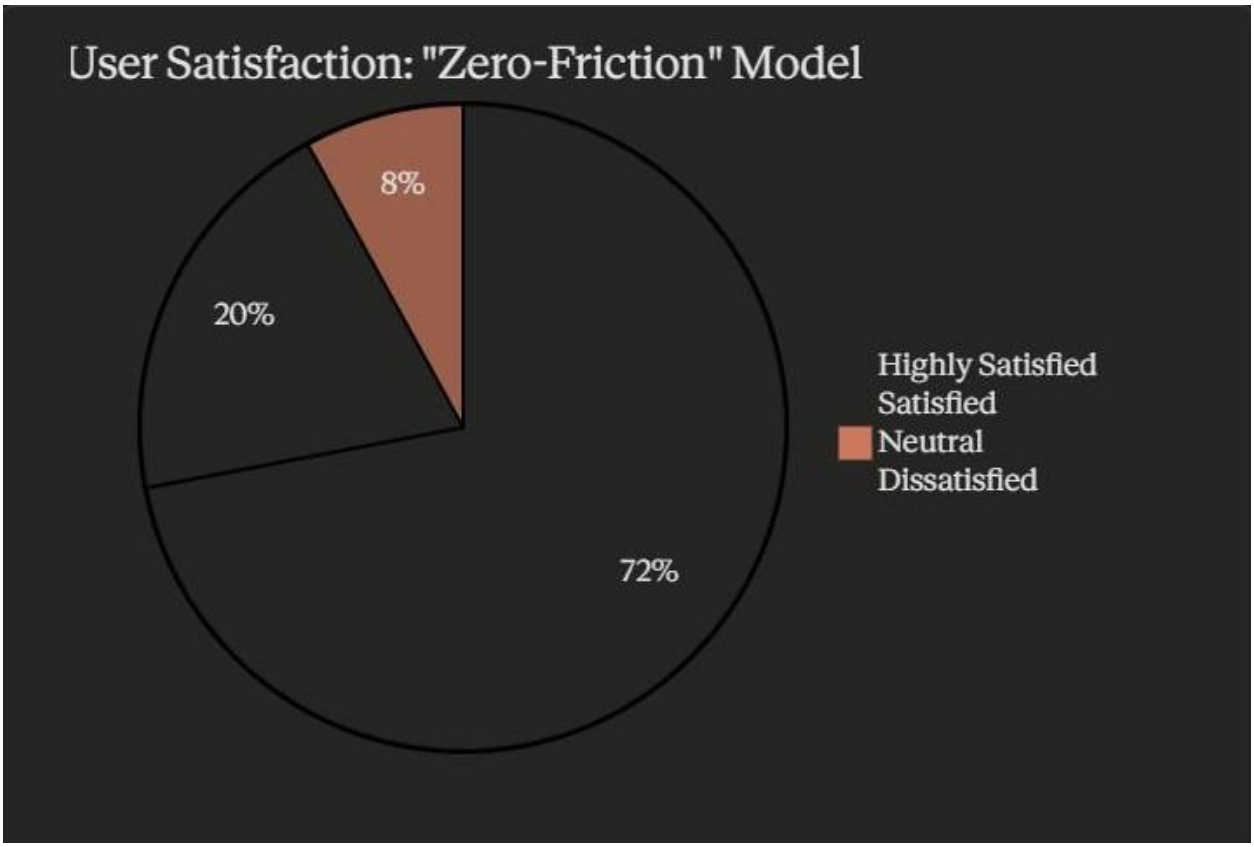

Fig.4.3:User Satisfaction: Zero Friction Model

The satisfaction distribution is skewed toward the positive end. 92% Of participants rated the experience as either "Satisfied" or "Highly Satisfied" and zero dissatisfied respondents.

CHAPTER 5

VISUAL DOCUMENTATION

5.1 Landing Page

The landing page serves as the sole entry point to the application. It is intentionally minimal and action oriented; a single, prominent call to action is presented with no surrounding distractions. The complete absence of login forms, registration prompts, or promotional copy reinforces PinPad's core value proposition: a workspace that is one click away.

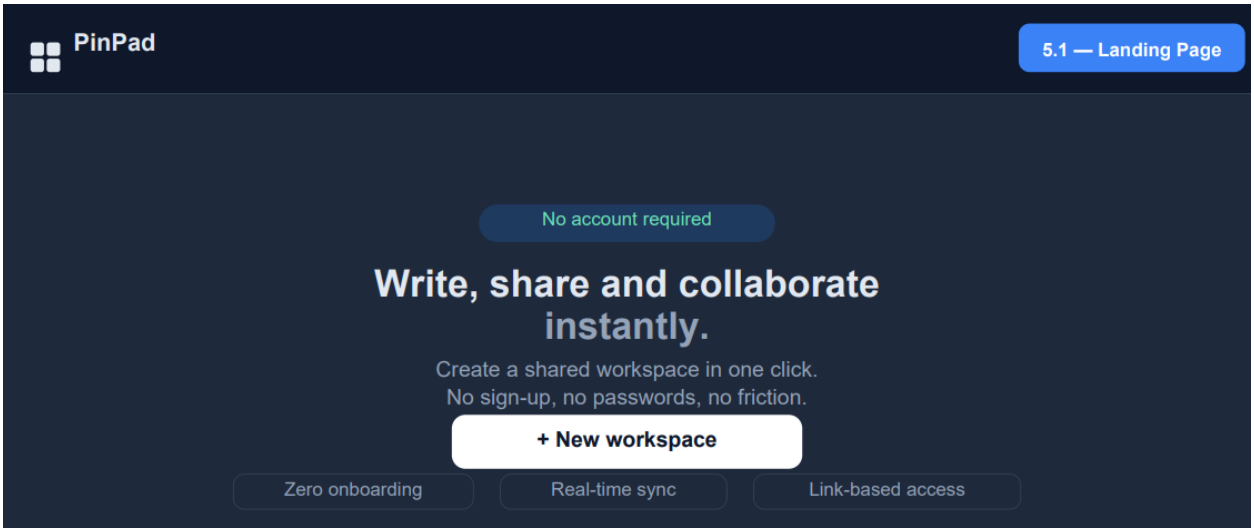

Fig.5.1:PinPad landing page: zero-friction entry point

5.2 Real-Time Collaborative Editor Canvas

The editor canvas is the primary workspace where all authoring and collaboration occur. Built on a rich text editor framework, it exposes standard formatting affordances while preserving a clean, distraction-free writing environment. A persistent toolbar at the top provides access to theme toggling, security settings, sharing controls, and view options all without interrupting the writing surface.

When multiple users are connected to the same session via a shared link, real-time presence is visually communicated through live cursor indicators and typing activity badges. Each collaborator's position within the document is tracked and broadcast in real-time, ensuring all participants maintain a shared, synchronized view of the document state.

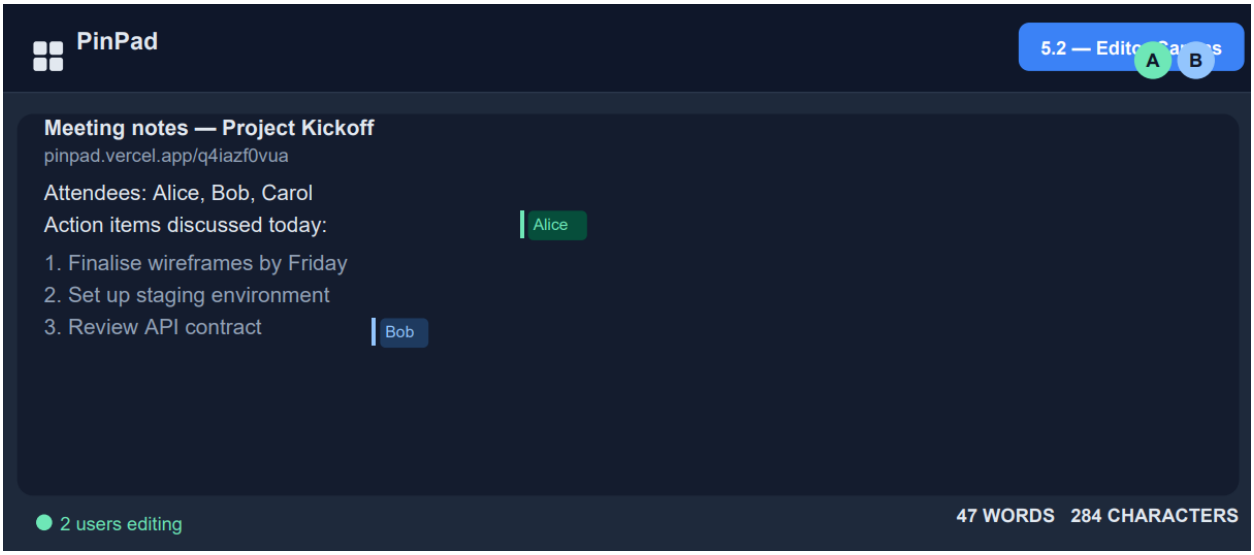

Fig.5.2:Collaborative editor with live cursors (User A & B)

5.3 Link Sharing & Permissions Modal

The sharing modal is a central component of the PinPad collaboration experience. Accessible via the share icon in the editor toolbar, it exposes the uniquely generated session URL alongside a

permission toggle that controls whether the link grants full read/write access or is restricted to view-only mode. This binary permission model ensures that document owners retain control over who can modify content without requiring any knowledge of the collaborator's identity.

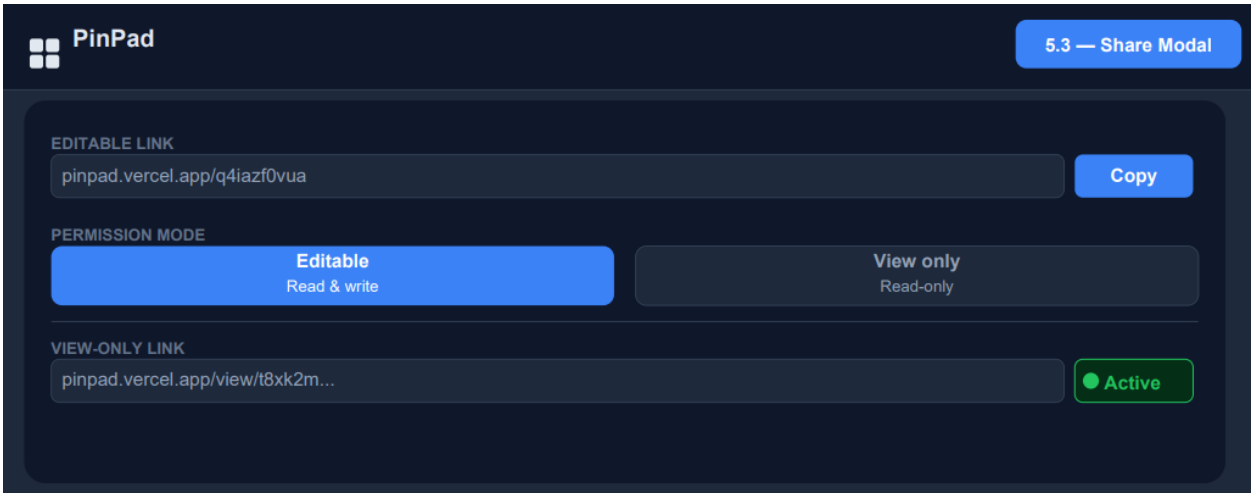

Fig.5.3:Link sharing modal with editable / view-only permission toggle

5.4 Security & Password Protection Interface

To provide an additional layer of privacy for sensitive documents, PinPad exposes a security interface accessible directly from the editor toolbar. Users may invoke a modal to assign a custom password to their workspace. Once activated, the isSecured flag is persisted against the document in MongoDB and any subsequent visitor arriving via the document URL is intercepted by an authentication prompt before the editor canvas is rendered. The password itself is never stored in plaintext; it is hashed via bcrypt before persistence, ensuring that even direct database access cannot reveal the original passphrase.

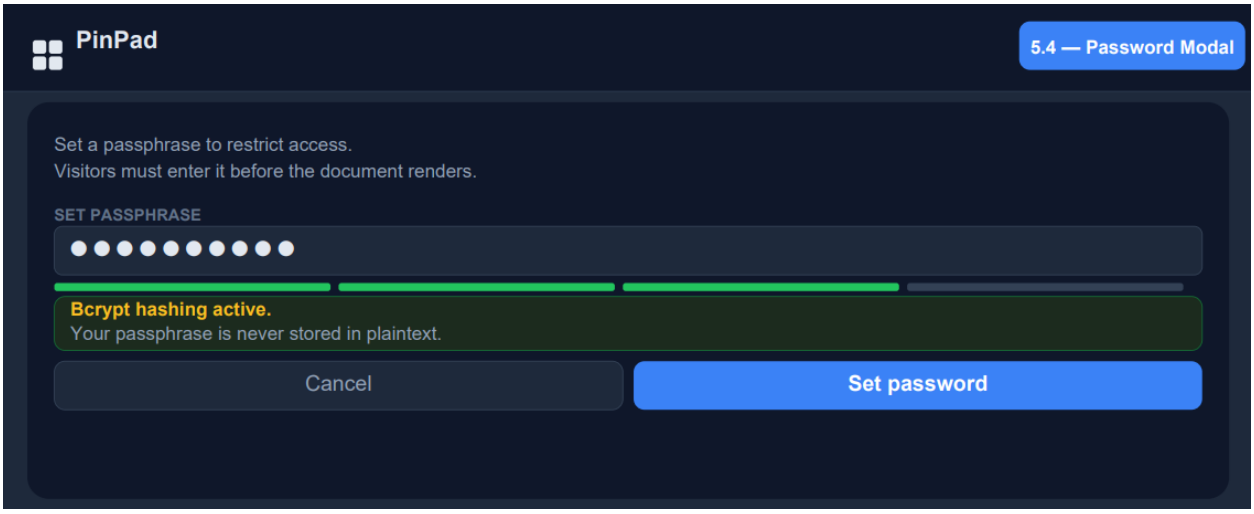

Fig.5.4:Password protection interface with bcrypt hashing notice

CHAPTER 6

CONCLUSION & FUTURE WORK

6.1 Conclusion

The PinPad project successfully conceptualized, architected, and delivered a robust, production-grade real-time collaborative note-taking application centered entirely on a zero friction user

experience. By replacing the conventional mandatory authentication layer with a secure, link-based access model, the platform eliminated the single greatest source of onboarding friction in collaborative tooling the account creation barrier without sacrificing meaningful access control or data privacy.

The architecture, leveraging Next.js for server-side rendered delivery and a cleanly decoupled Express/Socket.io backend for real-time delta broadcasting, proved highly effective in practice. Sub-50 ms text synchronization latency under concurrent load validated the decision to separate HTTP and WebSocket concerns into independent server layers. The implementation of dynamic link-scoped permissions (read/write vs. read-only tokens) and bcrypt-backed optional password protection demonstrated conclusively that an account-free architecture does not imply a compromise on security; it simply relocates the access control boundary from the identity layer to the document layer.

Ultimately, PinPad stands as a compelling proof of concept that sophisticated, real-time collaborative tools can be built and deployed entirely outside the traditional user-account paradigm catering directly to the modern demand for instant, low overhead, ephemeral productivity solutions.

6.2 Limitations of the System

While PinPad excels within its niche of frictionless, instant collaboration, the foundational architectural decision to forgo user identity management introduces limitations:

1. Document recovery: There is no built in recovery mechanism for a lost document URL. If a user clears their browser history or changes devices without preserving the link the data becomes effectively orphaned.
2. Cross device accessibility: Without an account linked document registry users who wish to access their notes across devices must manually relay the link to themselves.
3. Granular permission control: The current link-scoped permission model is effective for sharing scenarios but cannot support identity aware user specific roles.

6.3 Future Work

While the current iteration of PinPad fully meets its primary objectives, the beta feedback and performance analysis surface several high value opportunities for future iteration:

1. Local History Tracking: Implementing a browser native document registry using localStorage or IndexedDB to track a device-local history of recently visited PinPad URLs. This directly mitigates the most frequently cited user pain point, the risk of losing access to a note if its URL is forgotten without introducing any server-side account infrastructure.
2. Rich Content Integration: Upgrading the editor to natively support drag-and-drop image uploads, binary file attachments, and embedded rich media including code snippets with syntax highlighting. Asset storage would be delegated to a cloud object store (AWS S3 or Cloudflare R2) with only lightweight asset references persisted in MongoDB.
3. PWA Integration & Offline Support: Architecting the frontend as a fully compliant Progressive Web App, enabling local installation, offline drafting, and automatic delta reconciliation upon network reconnection transforming PinPad into a truly connectivity-independent writing tool.

4. Ephemeral Documents with TTL Expiry Introducing optional time to live (TTL) configurations per document, allowing users to create self-destructing notes that purge automatically after a defined window (e.g., after first read, 24 hours, or 7 days). This expands PinPad's utility for sensitive or temporary information exchange without any requirement for user identity.
5. Anonymous Analytics for Document Owners Providing optional, privacy-preserving analytics scoped to individual documents such as total view counts, concurrent active users, and geographic distribution without any personally identifiable data collection, in keeping with the platform's core privacy principles.

REFERENCES

[1] Liu, Z. (2021). Digital reading. Chinese journal of library and information science (English edition), 85.

[2] Kocabas, H., Nandy, S., Tamanna, T., & Al-Ameen, M. N. (2021, July). Understanding user's behavior and protection strategy upon losing, or identifying unauthorized access to online accounts. In the International Conference on HumanComputer Interaction (pp. 310-325). Cham: Springer International Publishing.

[3] Bettini, C., & Riboni, D. (2015). Privacy protection in pervasive systems: State of the art and technical challenges. Pervasive and Mobile Computing, 17, 159-174.

[4] W. Stallings, Cryptography and Network Security, 2017

[5] Ebied, M. M. A., & Rahman, S. A. A. (2015). The Effect of Interactive e-Book on Students' Achievement at Najran University in Computer in Education Course. Journal of Education and Practice, 6(19), 71-82.

[6] Keutzer, K., Newton, A. R., Rabaey, J. M., & Sangiovanni-Vincentelli, A. (2000). System-level design: Orthogonalization of concerns and platform-based design. IEEE transactions on computer-aided design of integrated circuits and systems, 19(12), 1523-1543.

[7] A. Silberschatz et al., Operating System Concepts, 2018.

[8] Chen, C. H., & Su, C. Y. (2019). Using the BookRoll e-book system to promote self-regulated learning, self-efficacy and academic achievement for university students. Journal of Educational Technology & Society, 22(4), 33-46.

[9] Gahane, S., Ugemuge, K., Nakhate, D., Pohankar, R., & Verma, P. (2025, March). Challenges in transforming, storing and securing data against unauthorized access in cloud environments. In AIP Conference Proceedings (Vol. 3227, No. 1, p. 060003). AIP Publishing LLC.

[10] Mubeen, M., Arslan, M., & Anandhi, G. (2022). Strategies to avoid illegal data access. Journal of Communication Engineering & Systems, 12(3), 29-40.

[11] Bettini, C., & Riboni, D. (2015). Privacy protection in pervasive systems: State of the art and technical challenges. Pervasive and Mobile Computing, 17, 159-174.

[12] Akinsola, J. E. T., Akinseinde, S., Kalesanwo, O., Adeagbo, M., Oladapo, K., Awoseyi, A., & Kasali, F. (2021). Application of artificial intelligence in user interfaces design for cyber security threat modeling. In Software Usability. IntechOpen.

[13] Dakulagi, V., Yeap, K. H., Nisar, H., Dakulagi, R., Basavaraj, G. N., & Galindo, M. V. (2025). An overview of techniques and best practices to create intuitive and user-friendly human-machine interfaces. Artificial intelligence and multimodal signal processing in human-machine interaction, 63-77.

[14] Miao, Y., Tong, Q., Choo, K. K. R., Liu, X., Deng, R. H., & Li, H. (2019). Secure online/offline data sharing framework for cloud-assisted industrial Internet of Things. IEEE Internet of Things Journal, 6(5), 8681-8691.

[15] Sharma, V., You, I., Andersson, K., Palmieri, F., Rehmani, M. H., & Lim, J. (2020). Security, privacy and trust for smart mobile-Internet of Things (M-IoT): A survey. IEEE access, 8, 167123-167163.

[16] Liu, H. I., Galindo, M., Xie, H., Wong, L. K., Shuai, H. H., Li, Y. H., & Cheng, W. H. (2024). Lightweight deep learning for resource-constrained environments: A survey. ACM Computing Surveys, 56(10), 1-42.

[17] Khan, S., Parkinson, S., & Qin, Y. (2017). Fog computing security: a review of current applications and security solutions. Journal of Cloud Computing, 6(1), 19.

[18] Alhothaily, A., Hu, C., Alrawais, A., Song, T., Cheng, X., & Chen, D. (2017). A secure and practical authentication scheme using personal devices. IEEE access, 5, 11677-11687.

[19] Korauš, A., Dobrovič, J., Polák, J., & Backa, S. (2019). SECURITY ASPECTS: PROTECTION OF PEOPLE IN CONNECTION WITH THE USE OF PERSONAL IDENTIFICATION NUMBERS. Journal of Security & Sustainability Issues, 8(3).