National Textile University NTU

Project for the Subject: Web Application Development

Project Title: Quiver

SUBMITTED BY:

Syed Junaid Jaffery (22-NTU-CS-1167)

Hadiah Iftikhar (22-NTU-CS-1159)

Section:

CS-6-A

SUBMITTED TO:

SIR ZAHID JAVED

SUBMISSION DATE:

####

Department of Computer Science

Table of Contents

Chapter 1: Project Overview ii
1.1 Introduction ii
1.2 Objectives and Goals iii
1.2.1 Primary Objectives: iii
1.2.2 Technical Goals: iii
1.3 Target Audience iv
1.3.1 Target Audience: iv
1.4 Scope of Project v
1.4.1 Core Functionalities: v
1.4.2 Technical Scope: vi
1.4.3 Project Boundaries: vi
Chapter 2: Project Design vii
2.1 Data Model vii
2.1.1 Database Schema Overview vii
2.1.2 Entity Relationship Description vii
2.1.3 ERD viii
2.2 Process Model viii
2.2.1 Activity Diagram ix
2.2.2 Use Case Diagram ix
2.2.3 Sequence Diagram xi
2.2.4 Component Diagram xii
2.2.5 User Flow Diagram xiii
Chapter 3: Implementation Technologies xiv
3.1 Next.js (Routing and Request Handling) xiv
3.2 Authentication with Supabase xiv
3.3 Data Storage with Supabase xv
3.4 TinyMCE Editor API xv
3.5 GROK API xv
3.6 Twilio API xvi
3.7 Tailwind CSS xvi
Chapter 4: Website Features xvii
4.1 Account Management xvii
4.1.1 Signup and Login xvii
4.1.2 Profile Management xvii
4.1.3 Password Management xviii
4.2 Notes xix

4.2.1 Folder Management xix
4.2.2 Note Management xix
4.2.3 AI Enhancements xx
4.2.4 Tagging/Searching System xxi
4.3 Flashcards xxii
4.3.1 Flashcard Deck Management xxii
4.3.2 AI-powered Flashcards Generation xxii
4.3.3 Flashcards Study Interface xxiii
4.4 Pomodoro xxiii
Chapter 5: User Interface xxvi
5.1 Landing Page xxvi
5.2 About Page xxvii
5.3 Signup xxvii
5.4 Login xxviii
5.5 Terms and Conditions xxviii
5.6 Forgot password xxix
5.7 Dashboard xxix
5.8 Accounts xxix
5.9 Settings xxx
5.10 Notes xxx
5.11 [Noteid] xxxi
5.12 Flashcards xxxii
5.13 Create Flashcard Model xxxii
5.14 View/Practice Created Flashcards Deck xxxiii
5.15 Pomodoro xxxiii
Chapter 6: Limitations and Future Scope xxxiv
6.1 Limitations xxxiv
6.1.1 Lack of Note Import Functionality xxxiv
6.1.2 Single Accountability Partner Restriction xxxiv
6.1.3 Limited Manual Flashcard Creation xxxiv
6.1.4 Incomplete Quizzing System xxxiv
6.2 Future Scope xxxv
6.2.1 Comprehensive Import/Export Ecosystem xxxv
6.2.2 Enhanced Collaborative Study Environment xxxv
6.2.3 Advanced Flashcard Management System xxxv
6.2.4 Intelligent Assessment Platform xxxvi
Chapter 7: Conclusion xxxvii

Table of Figures

Figure 1 ERD Diagram of the Database viii
Figure 2 Activity Diagram ix
Figure 3 Use Case Diagram x
Figure 4 Sequence Diagram xi
Figure 5 Component Diagram xii
Figure 6 Use Flow Diagram xiii
Figure 7 Landing Page xxvi
Figure 8 About Page xxvii
Figure 9 SignUp Page xxvii
Figure 10 LogIn Page xxviii
Figure 11 Terms and Conditions Page xxviii
Figure 12 Forgot Password Page xxix
Figure 13 Account Management Page xxix
Figure 14 Settings Page xxx
Figure 15 Notes Page xxx
Figure 16 Inner Note Editing Page xxxi
Figure 17 Flashcards Page xxxii
Figure 18 Flashcard Creation Page xxxii
Figure 19 View and edit Flashcards Page xxxiii
Figure 20 Pomodoro Page xxxiii

Chapter 1: Project Overview

1.1 Introduction

Quiver is a comprehensive web application designed to enhance the academic experience of students through the implementation of modern learning techniques and productivity tools. The name "Quiver" embodies the project's core philosophy - providing students with a versatile collection of tools (like arrows in a quiver) to aim, note, and navigate through their educational journey with precision and effectiveness.

In today's fast-paced academic environment, students face numerous challenges in managing information, staying focused, and effectively retaining knowledge. Traditional note-taking methods often fall short in organizing the vast amount of information students encounter daily. Furthermore, conventional study techniques frequently fail to leverage cognitive science principles that could significantly enhance knowledge retention and understanding.

Quiver addresses these challenges by offering an integrated platform that combines three essential components of effective studying: structured note-taking, active recall through flashcards, and focused study sessions with the Pomodoro technique. These components are not merely juxtaposed but are designed to work in harmony, creating a seamless learning ecosystem.

What sets Quiver apart from other educational tools is its thoughtful integration of artificial intelligence to augment the learning process. Rather than replacing human cognition, Quiver's AI features enhance it by automating mundane tasks, generating valuable insights, and providing personalized study materials. This approach allows students to focus on higher-order thinking and genuine understanding rather than mechanical aspects of information management.

The application features a modern, intuitive interface designed with accessibility and user experience at its forefront. The dark-themed design reduces eye strain during extended study sessions, while the responsive layout ensures functionality across various devices and screen sizes. Color psychology has been carefully considered, with purple tones (represented by #9B87F5 and #5222D0) chosen to evoke creativity and wisdom, complemented by functional accent colors that guide user interaction.

1.2 Objectives and Goals

The development of Quiver is guided by several key objectives that address specific challenges in modern education and leverage technological advancements to enhance learning outcomes.

1.2.1 Primary Objectives:

Streamline Information Management

Create a centralized system for capturing, organizing, and retrieving academic information that eliminates the fragmentation common with traditional note-taking methods. Quiver aims to reduce cognitive load by providing a structured yet flexible framework for knowledge management.

Enhance Knowledge Retention

Implement evidence-based learning techniques, particularly active recall through flashcards, to significantly improve long-term retention compared to passive review methods. The system automatically generates study materials from notes, reinforcing connections between concepts.

Optimize Study Efficiency

Integrate time management tools based on the Pomodoro Technique to combat procrastination and maintain sustained focus during study sessions. The application includes accountability mechanisms to ensure adherence to scheduled study periods.

Leverage Artificial Intelligence Appropriately

Employ AI technologies to augment human learning rather than replace it, focusing on automating mechanical aspects of studying while preserving and enhancing meaningful cognitive engagement with the material.

Provide a Cohesive Learning Environment

Create seamless integration between different study functionalities rather than offering disconnected tools, allowing for natural workflow transitions between note-taking, review, and focused study.

1.2.2 Technical Goals:

Develop a Responsive Web Application

Create a platform accessible across devices with consistent functionality and user experience, using modern web technologies like Next.js and React.

Implement Robust Data Management

Establish secure, efficient data handling using Supabase for backend operations, ensuring user data privacy and reliable performance.

Create an Intuitive User Interface

Design a clean, accessible interface that minimizes learning curve while maximizing functionality, using thoughtful color schemes and layout principles.

Ensure Real-time Collaboration

Implement features that allow for seamless updates and synchronization across devices, with automatic saving to prevent work loss.

Establish AI Integration Framework

Develop a modular system for AI features that can be expanded and refined over time, starting with core functionalities like summary generation and flashcard creation.

1.3 Target Audience

Quiver has been designed with a specific focus on the needs and challenges of the academic community, though its utility extends to various learning contexts. Understanding the target audience has been crucial in shaping both the functionality and interface of the application.

1.3.1 Target Audience:

University and College Students

The core audience for Quiver consists of undergraduate and graduate students across disciplines who:

- Manage large volumes of information from lectures, readings, and research
- Need to retain complex concepts for examinations and practical application
- Struggle with balancing multiple courses and assignments
- Face challenges in maintaining focus in distraction-rich environments
- Seek efficient methods to prepare for assessments and deepen understanding

These students typically have moderate to high digital literacy and are receptive to technological solutions that demonstrably improve their academic outcomes. They value tools that save time while enhancing learning quality.

High School Students

Secondary students in rigorous academic programs who:

- Are developing independent study skills for college preparation
- Need structure in their learning approach
- Benefit from active learning techniques for complex subjects
- Are building time management habits crucial for academic success

Lifelong Learners

Adults engaged in continuous education or skill development who:

- Study independently without formal institutional structure
- Need to fit learning around work and other responsibilities
- Value efficient knowledge retention methods
- Often learn across multiple domains simultaneously

1.4 Scope of Project

The scope of Quiver has been carefully defined to deliver a cohesive, high-quality learning platform while maintaining development feasibility. The project encompasses three integrated core functionalities supported by a robust technical infrastructure.

1.4.1 Core Functionalities:

1. **Notes System**
    - Digital note-taking with organization capabilities
    - Rich text editing and formatting features
    - Folder and tag organization system
2. **Flashcards System**
    - Creation and study of digital flashcards
    - Connection to notes for integrated learning
    - Study progress tracking
3. **Pomodoro Timer**

- Customizable study session timing
- Break management
- Focus accountability features

1.4.2 Technical Scope:

- Web-based application using Next.js and React
- Responsive design for desktop and mobile compatibility
- Supabase backend for authentication and database operations
- User account system with secure authentication

1.4.3 Project Boundaries:

To maintain focus and ensure timely delivery, several potential features have been explicitly excluded from the current project scope:

- Collaborative real-time editing
- Advanced multimedia embedding beyond basic images
- Native mobile applications
- Integration with external learning management systems
- Multiple language support (English-only in first release)

The application architecture has been designed with potential future expansion in mind, but the current implementation will focus solely on delivering a high-quality experience for the three core functionalities.

Chapter 2: Project Design

2.1 Data Model

The data model for Quiver has been designed to efficiently support the application's core functionality while maintaining scalability and performance. The database structure consists of several interconnected tables that store user information, notes, folders, links, and tags.

2.1.1 Database Schema Overview

The database schema consists of five primary entities:

1. **Users**: Stores user account information and preferences
2. **Folders**: Contains organizational folders created by users
3. **Notes**: Stores the actual note content and metadata
4. **Note_Links**: Manages relationships between related notes
5. **Note_Tags**: Handles tagging system for categorization
6. **Flashcards**: stores the front and the back text of the flashcards
7. **Flashcard_decks**: this is like the “folder” where the flashcards are stored

2.1.2 Entity Relationship Description

These entities form a cohesive structure that enables:

- Each user having multiple folders and notes
- Notes belonging to specific folders
- Notes having multiple tags
- Notes linking to other related notes
- Complete user data separation for security and privacy

2.1.3 ERD

The Entity Relationship Diagram illustrates the database schema that powers Quiver. This diagram visualizes how different entities relate to one another within the application's data model. The ERD shows the seven main entities: Users, Folders, Notes, Note_Links, Note_Tags, Flashcard_Decks, and Flashcards. Each entity is represented with its fields and data types, while the connecting lines demonstrate relationships between entities, such as one-to-many relationships between users and folders or notes and flashcards. This diagram serves as the blueprint for database implementation and helps ensure proper data organization, efficient querying, and data integrity throughout the application.

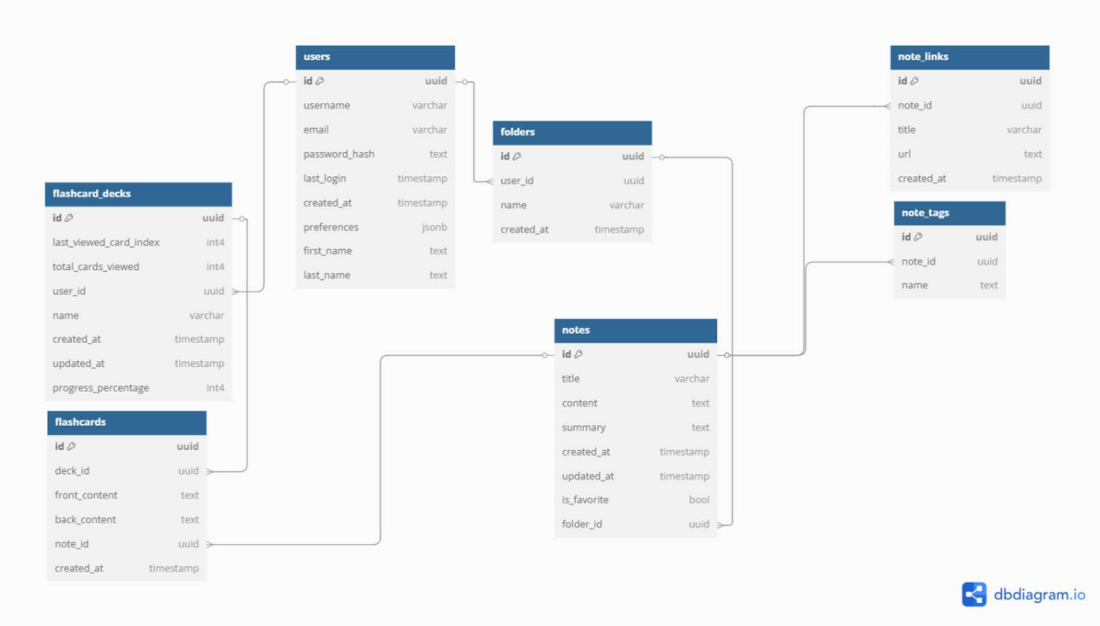

Figure 1 ERD Diagram of the Database

2.2 Process Model

The process model section illustrates how users interact with the Quiver application and how data flows through the system. This section provides various diagrams that represent different aspects of the application's behavior. Each diagram captures a unique perspective on the application's functionality, from user interactions to system components, creating a comprehensive understanding of the system's operation. These visual representations help both developers and stakeholders understand the application's architecture and behavior from multiple viewpoints.

2.2.1 Activity Diagram

The Activity Diagram maps the flow of user actions within the Quiver application, showing the possible paths and decision points users encounter while navigating the system. This diagram illustrates the sequential flow of activities for core features including note management, flashcard creation and study, and Pomodoro timer sessions. Each pathway demonstrates how users interact with specific features, from authentication through to completing tasks within each module. The diagram helps visualize the user experience journey and identify potential bottlenecks or areas for optimization in the workflow design.

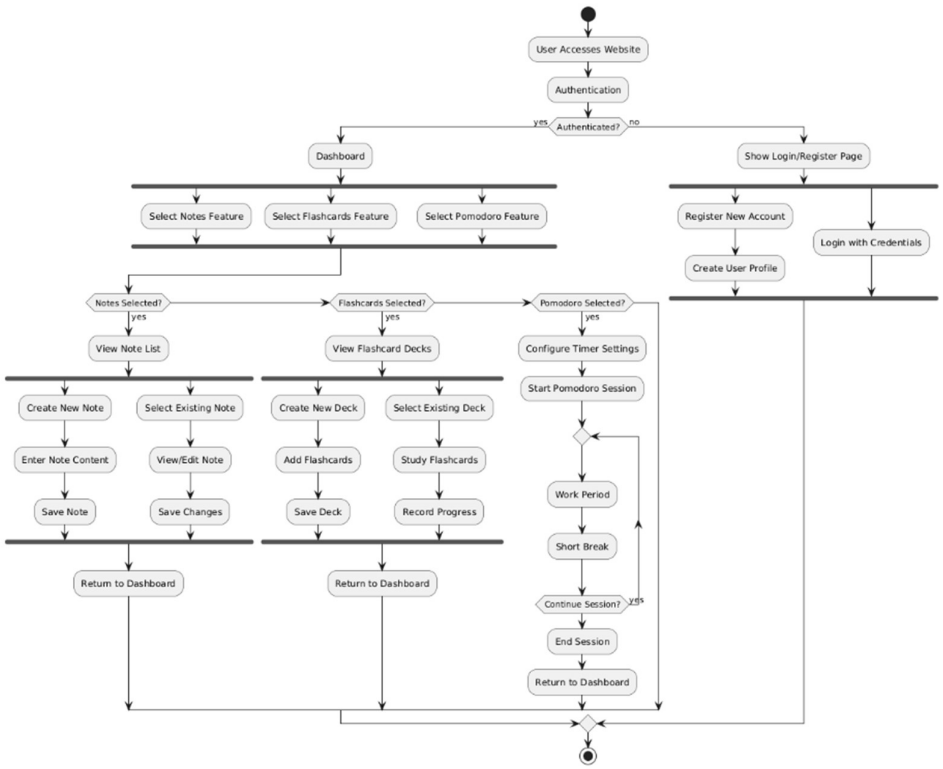

Figure 2 Activity Diagram

2.2.2 Use Case Diagram

The Use Case Diagram provides a high-level view of the functionality provided by Quiver and the different actors who interact with the system. The diagram identifies the primary actor (User) and secondary actor (AI System) and their relationships with various use cases within the application. Each use case represents a discrete function the system performs, such as creating notes, studying flashcards, or running Pomodoro sessions. This diagram serves as an essential communication tool between developers and stakeholders, clearly illustrating the system's capabilities and the value provided to users.

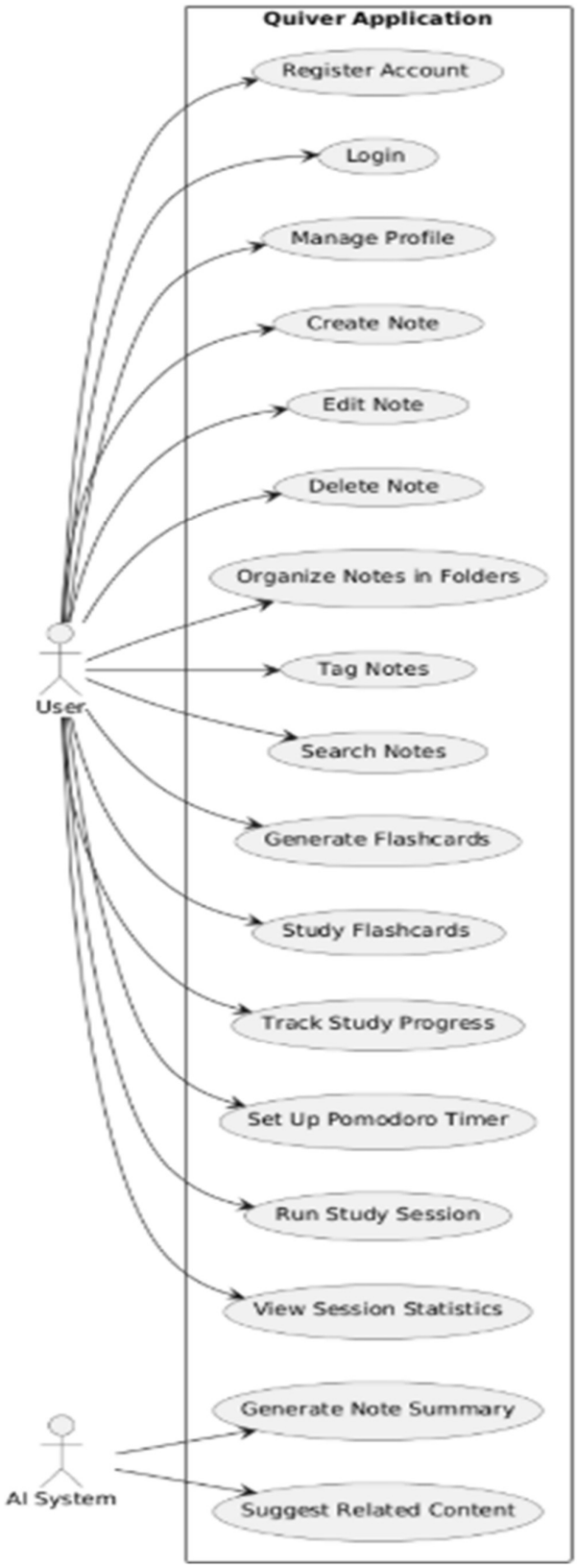

Figure 3 Use Case Diagram

2.2.3 Sequence Diagram

The Sequence Diagram illustrates the temporal interaction between different components of the Quiver system during key processes. This diagram shows the step-by-step sequence of messages exchanged between the User, Frontend UI, Next.js API, and Supabase database during note creation and editing operations. By representing the chronological flow of interactions, this diagram helps developers understand the communication patterns between components and identify potential performance bottlenecks or areas where error handling might be required. The sequence diagram is particularly valuable for understanding the application's behavior during complex operations.

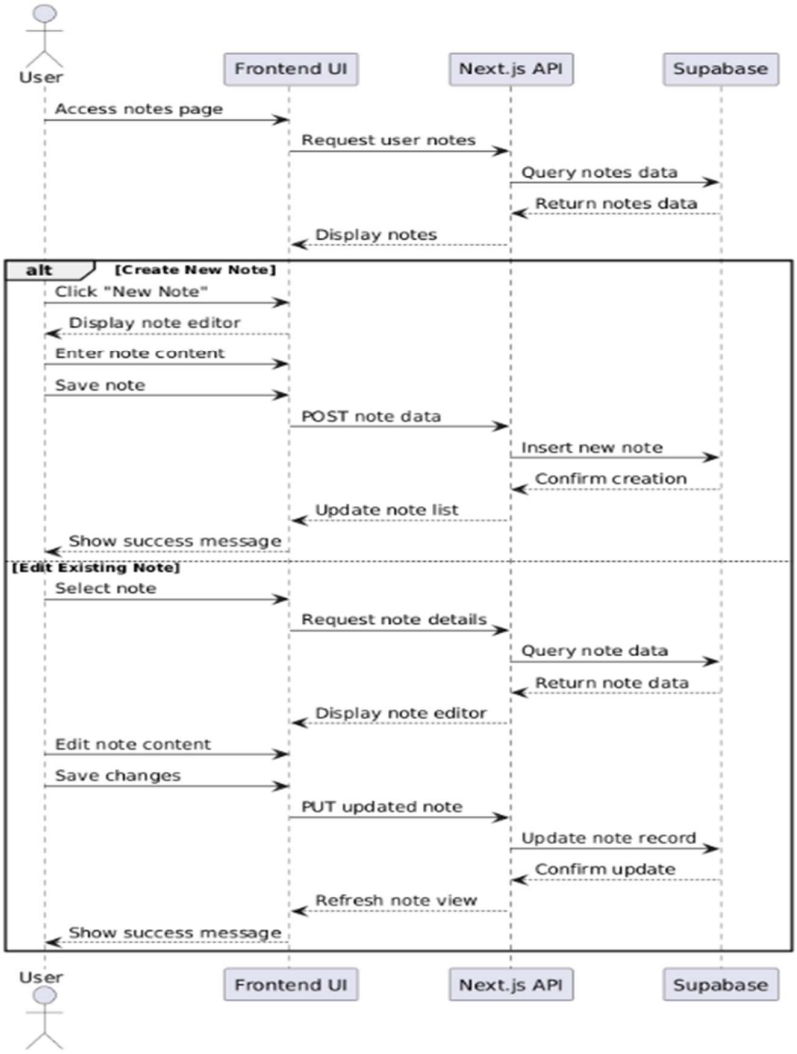

Figure 4 Sequence Diagram

2.2.4 Component Diagram

The Component Diagram depicts the architectural structure of Quiver, showing the major components and their interconnections. This diagram organizes the system into logical packages including Frontend components, API Layer, Data Access layer, and external services. The connections between components illustrate dependencies and communication pathways, helping developers understand how data and control flow through the system. This architectural view is crucial for maintaining a modular design, facilitating future enhancements, and ensuring proper separation of concerns throughout the codebase.

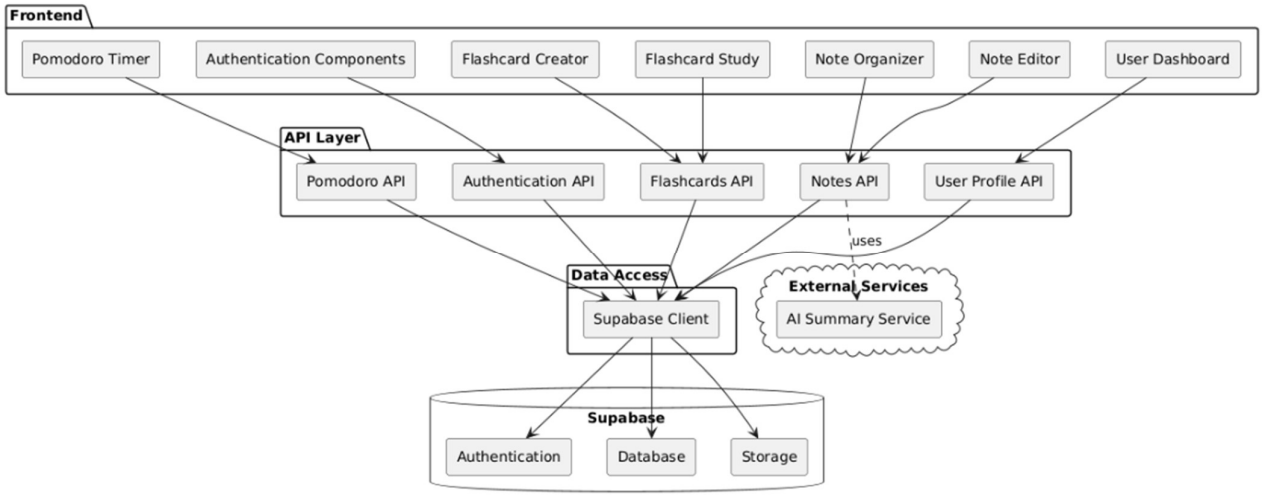

Figure 5 Component Diagram

2.2.5 User Flow Diagram

The User Flow Diagram maps the typical paths users take when navigating through the Quiver application. This diagram visualizes the journey from landing on the homepage through authentication, and the subsequent navigation between different features including notes, flashcards, and the Pomodoro timer. Decision points are clearly marked, showing alternative paths based on user choices or system conditions. This diagram helps designers and developers ensure a logical, intuitive user experience by identifying potential pain points or confusing navigation patterns before implementation.

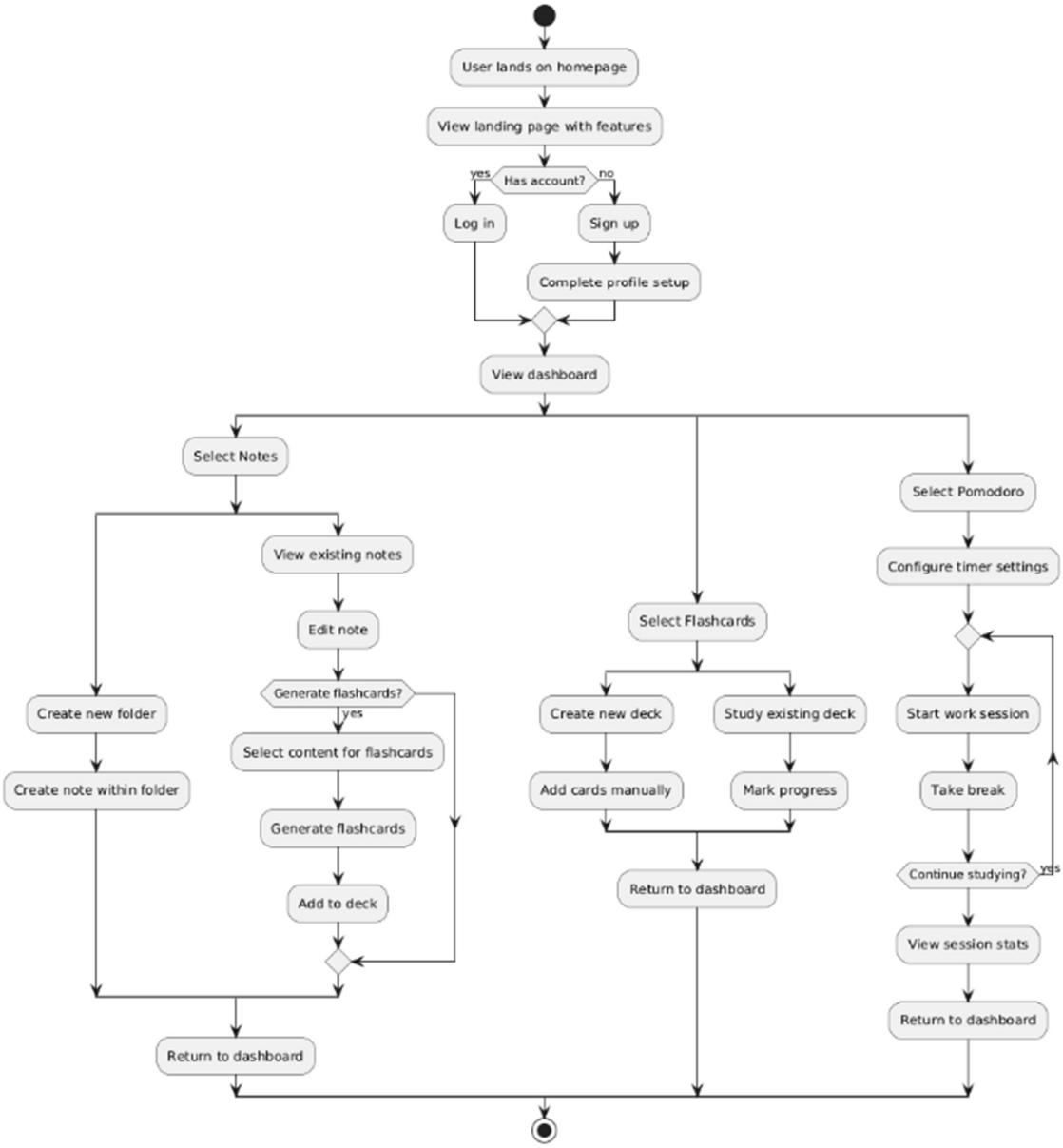

Figure 6 Use Flow Diagram

Chapter 3: Implementation Technologies

3.1 Next.js (Routing and Request Handling)

Next.js serves as the foundational framework for Quiver, providing a robust, React-based architecture that enables server-side rendering, static site generation, and efficient client-side navigation. The implementation leverages Next.js 13's advanced features, particularly its innovative file-based routing system and server components architecture.

The application utilizes the App Router paradigm introduced in Next.js 13, organizing routes based on the filesystem structure. Each folder within the /app directory represents a route segment, with page components rendered at corresponding URLs. This structure creates an intuitive organization that mirrors the application's navigation hierarchy.

We use multiple different routing techniques like static, dynamic, and group routing, for instance, the (auth) and the (misc) directories serve as group routes whereas the [noteid] serves as a dynamic route and dashboard or pomodoro serve as static routes.

For API endpoints, Quiver implements route handlers within the /app/api directory. These handlers export asynchronous functions corresponding to HTTP methods (GET, POST, etc.) that process requests and return responses.

Authentication in Quiver is implemented using Supabase Auth, providing a secure, scalable solution that supports multiple authentication methods while minimizing development overhead. The integration follows best practices for Next.js applications, implementing both client-side and server-side authentication flows.

3.2 Authentication with Supabase

The authentication system is configured through Supabase's dashboard, enabling email/password authentication with robust security features including password strength requirements and email verification. The implementation also includes OAuth providers like Google and GitHub so the user can easily signup or login with a simple ‘continue with google/github’ like how it is in most modern websites.

3.3 Data Storage with Supabase

Quiver leverages Supabase PostgreSQL for data storage, providing a robust, relational database solution with real-time capabilities. The database schema follows the entity-relationship model detailed in the ERD, with tables for users, folders, notes, flashcard decks, flashcards, note tags, and note links.

Integration between the Next.js application and Supabase is facilitated through a dedicated utilities module located in the /utils/supabase directory. This module exports both client-side and server-side Supabase clients

Database operations are implemented using Supabase's JavaScript client, which provides a straightforward interface for CRUD operations. For complex operations involving multiple tables, database functions are implemented using PostgreSQL stored procedures accessible through the Supabase client. This comprehensive data storage implementation provides Quiver with a secure, scalable foundation while minimizing the complexity of database management code within the application.

3.4 TinyMCE Editor API

The note-taking functionality in Quiver is powered by TinyMCE, a robust rich text editor that provides an intuitive interface for creating and editing formatted content. The implementation integrates TinyMCE's React component with custom configurations to meet the specific needs of the application. TinyMCE is integrated using the official @tinymce/tinymce-react package, providing a React component that encapsulates the editor's functionality:

The editor is configured with a carefully selected set of plugins that enhance the note-taking experience: A custom skin is applied to match Quiver's dark-themed design, ensuring visual consistency across the application. To handle content saving, the editor is integrated with Quiver's state management system, automatically saving changes after a brief delay using debounce techniques

3.5 GROK API

The Quiver application enhances its note-taking and flashcard functionalities using Grok's AI capabilities with the Llama3-70b-8192 model. This integration enables intelligent features including automatic note summarization, relevant link generation, and flashcard creation from note content. We are using this API for the following features:

1. **Note Summarization**: Condenses lengthy note content into concise summaries for quick review:
2. **Link Suggestions**: Generates relevant resources for further learning based on note content
3. **Flashcard Generation**: Creates study flashcards from note content to facilitate active recall

Each of these features is implemented as a dedicated API endpoint, ensuring clean separation of concerns and maintainable code. The implementation includes robust error handling, input validation, and response parsing to deliver reliable AI-enhanced functionality to users.

3.6 Twilio API

The Pomodoro Timer feature in Quiver incorporates Twilio API integration to provide accountability through SMS notifications. This feature allows users to designate an accountability partner who receives text messages when study sessions are abandoned, creating external motivation to maintain focus. A dedicated API route handles the sending of SMS messages, encapsulating the Twilio logic and providing a clean interface for the application.

The Twilio WhatsApp integration is particularly valuable for the accountability feature in the Pomodoro timer, as it leverages a communication channel that's widely used and typically has higher engagement rates than regular SMS. When a user fails to respond to a focus check during a Pomodoro session, the system automatically sends a WhatsApp message to their designated accountability partner.

3.7 Tailwind CSS

The application's user interface is built with Tailwind CSS, providing a utility-first approach to styling that enables rapid UI development while maintaining a consistent visual language. The implementation includes custom theme extensions that define the application's color palette, typography, and component styles.

Chapter 4: Website Features

4.1 Account Management

The account management system in Quiver provides comprehensive user authentication and profile management capabilities. This system ensures secure access to the application while offering users flexibility in how they authenticate and manage their accounts.

4.1.1 Signup and Login

Users can create accounts and access Quiver through multiple authentication methods:

1. **Email/Password Authentication**

The primary authentication method allows users to register with an email address and password. The registration form includes validation that ensures:

- Email addresses follow proper formatting and are not already in use
- Passwords meet strength requirements (minimum 8 characters, with both letters and numbers)
- Password confirmation matches the initial entry

After registration, users receive a confirmation email to verify their address before gaining full access to the platform (done using supabase auth).

2. **Social Authentication**

For added convenience, users can authenticate through:

- Google accounts
- GitHub accounts

These methods provide a streamlined login process that requires only a few clicks while still capturing necessary user information. The authentication flow includes proper redirect handling to maintain a smooth user experience.

4.1.2 Profile Management

Once authenticated, users can manage various aspects of their profile through the settings and account interfaces:

1. **Personal Information Updates**

Users can update their first name and last name through a simple form with validation to ensure proper formatting. This information is used throughout the application for personalization.

2. **Username Changes**

The user can change their original username (entered during signup) from the accounts page easily, the username input field accepts alphanumeric input.

3. **Email Address Updates**

Users can update their email address through a secure process where the user must first type the password of the account and only then they can change their email address. This change is done both in the auth table and the users table in supabase.

4. **Password Changes**

Users can update their password through a secure form that requires the current password for verification. The password change form includes validation for password strength and matching confirmation.

5. **Account Deletion**

Users can permanently delete their account and all associated data. This critical action requires explicit confirmation to prevent accidental data loss. Upon confirmation, all user data, including notes, flashcards, and personal information, is permanently removed from the system.

4.1.3 Password Management

The platform includes comprehensive password management features:

1. **Forgot Password**

Users who forget their password can request a password reset from the login page. This sends a secure link to the user's email with a time-limited token for password reset.

2. **Password Reset**

The password reset process verifies the reset token and allows users to set a new password. After successful reset, users are redirected to the login page.

3. **Logout**

Users can securely log out of the application from any page through the navbar. This invalidates their session and requires re-authentication for future access.

All forms in the account management system include comprehensive validation with immediate feedback on input errors, ensuring data integrity while providing a smooth user experience.

4.2 Notes

The Notes feature forms the core of Quiver's knowledge management system, providing a flexible, powerful environment for capturing, organizing, and enhancing academic information. This feature combines rich text editing with AI-powered enhancements to create an efficient note-taking experience.

4.2.1 Folder Management

Notes are organized within a hierarchical folder system that allows users to structure their information according to their preferences:

1. **Folder Creation**

Users can create new folders through a modal interface that prompts for a folder name. The interface is accessible through a prominent "New Folder" button in the sidebar. New folders appear immediately in the folder sidebar after creation.

2. **Folder Renaming**

Existing folders can be renamed through a context menu (accessed by right-clicking) or a dedicated edit button. This allows users to evolve their organizational structure as their needs change.

3. **Folder Deletion**

Users can delete folders, with a confirmation dialog to prevent accidental deletion. The system clearly indicates that deleting a folder will also remove all contained notes.

Within folders, users can create and manage individual notes:

4.2.2 Note Management

1. **Note Creation**

New notes can be created within a selected folder through a "New Note" button. New notes open automatically in the editor with a default "Untitled Note" name for immediate content entry.

2. **Note Editing**

The note editor consists of two main components:

- A title input field for the note heading
- A rich text editor for the note content

The rich text editor provides extensive formatting capabilities:

- Text formatting (bold, italic, underline)
- Paragraph styles and headings
- Ordered and unordered lists
- Hyperlinks

Changes to notes are automatically saved as the user types, eliminating the need for manual saving and reducing the risk of lost work.

3. **Note Deletion**

Notes can be deleted through a context menu or dedicated button. Deletion includes a confirmation dialog to prevent accidental data loss.

4.2.3 AI Enhancements

The notes feature includes AI-powered enhancements using the Llama3-70b-8192 model through the Grok API:

1. **Summary Generation**

Users can generate a concise summary of their note content by clicking the "Generate Summary" button. The system processes the note content and generates a concise overview that captures the main points. The generated summary is displayed in a dedicated panel and can be used for quick review or note organization.

2. **Related Links Generation**

The system can suggest relevant external resources based on note content when users click the "Suggest Links" button. These suggestions appear in a sidebar panel and typically include:

- Academic articles
- Educational websites
- Related Wikipedia entries
- Tutorial resources
- Documentation

Each suggested link includes a title and URL that users can click to open in a new tab. These links provide opportunities for further exploration and learning beyond the note content.

4.2.4 Tagging/Searching System

Notes include a comprehensive tagging system for cross-categorical organization:

1. **Tag Assignment**

Users can add tags to notes through a dedicated input component at the bottom of the page. Multiple tags can be assigned to a single note, creating flexible categorization options that span beyond the folder hierarchy.

2. **Tag Removal**

Tags can be removed by clicking an X icon on the tag chip. This allows easy refinement of categorization as the note content evolves.

3. **Tag-Based Search**

The application includes a search system that allows filtering notes by tags. Users can select a tag to view all notes with that tag, regardless of the folder location. This enables powerful organization and retrieval of notes across folder boundaries.

The Notes feature combines these components into a cohesive system that supports diverse study approaches, from linear note organization to tag-based cross-referencing, all enhanced with AI-powered tools that add value to the captured information.

4.3 Flashcards

The Flashcards feature in Quiver transforms passive notes into active learning tools, leveraging the proven cognitive science principle of active recall to enhance knowledge retention. This feature allows users to create, organize, and study flashcards derived from their notes.

4.3.1 Flashcard Deck Management

Flashcards are organized into decks that represent coherent study units:

1. **Deck Creation**

Users can create new flashcard decks through a modal interface accessible from the Flashcards dashboard. The decks are created with the same name as that of the note of which the deck is of.

2. **Deck Updating**

The deck can be updated and the flashcards in it can be updated as well (we can both delete the flashcards or edit them)

3. **Deck Deletion**

Users can delete flashcard decks when no longer needed through the deck's context menu. The system requests confirmation before deletion to prevent accidental data loss. This deletes all the flashcards in that deck.

4.3.2 AI-powered Flashcards Generation

One of the standout features is the ability to automatically generate flashcards from note content using the Llama3 model:

1. **Generation Process**

Users can generate flashcards from their notes through a dedicated interface:

- Select a note from which to generate flashcards
- Click the "Generate Flashcards" button
- The system analyzes the note content using the Llama3-70b-8192 model via the Grok API
- Question-answer pairs are created based on key concepts in the note

- The generated flashcards are presented for review before saving

2. **Flashcard Review and Editing**

Generated flashcards are displayed in an edit interface where users can:

- Review the AI-generated content
- Edit questions or answers for accuracy
- Delete irrelevant flashcards

4.3.3 Flashcards Study Interface

The study interface provides an intuitive environment for active recall practice:

1. **Card Presentation**

Flashcards are presented in a clean, focused interface that shows one card at a time with the question visible. The design minimizes distractions to help maintain focus on the study material.

2. **Card Flipping**

Users can "flip" the card by clicking on it or pressing a dedicated button to reveal the answer. This implements the active recall process, requiring users to attempt to recall the answer before seeing it.

The Flashcards feature integrates seamlessly with the Notes feature, allowing users to transform their passive study materials into active learning tools. The AI-powered generation capability significantly reduces the effort required to create effective study materials, while the intuitive study interface facilitates consistent practice of active recall.

4.4 Pomodoro

The Pomodoro feature in Quiver implements a scientifically-backed time management technique designed to enhance focus and productivity during study sessions. The implementation includes customizable timer settings and an innovative accountability system that leverages WhatsApp messaging to maintain user commitment.

1. **Timer Setup and Configuration**

Users can configure their Pomodoro sessions according to their preferences:

2. **Timer Duration Settings**

The interface allows customization of three key time periods:

- Focus Duration: The length of concentrated work (default 60 minutes)
- Short Break: Brief rest between focus periods (5 minutes)
- Long Break: Extended rest after multiple focus periods (15 minutes)

These settings are accessible directly in the pomodoro page.

3. **Check-In Configuration**

Users can set the frequency of focus check-ins, which are used to verify continued attention. This feature helps combat distractions by periodically prompting the user to confirm they're still engaged with their work.

4. **Accountability Setup**

Users can provide a WhatsApp number for an accountability partner who will receive notifications. This number is validated to ensure proper formatting and includes the country code.

5. Timer Functionality

The Pomodoro timer implements a comprehensive workflow for focused study:

1. **Timer Initialization and Control**

Users can start, pause, and reset the timer through clearly labeled buttons. The current phase and remaining time are prominently displayed.

2. **Visual Feedback**

The timer provides clear visual feedback about the current phase and remaining time:

- Circular progress indicator showing elapsed/remaining time
- Color coding for different phases (e.g., blue for focus, green for breaks)
- Phase label clearly indicating whether the user should be working or resting

6. Accountability System

The accountability system leverages WhatsApp messaging to create external motivation:

1. **Focus Check-ins**

During focus periods, the system periodically displays a check-in prompt:

- A modal dialog appears according to a simple formula (total time in seconds / number of check-ins + 2)
- Users must respond to this prompt within 20 seconds
- The dialog requires a single click to confirm continued focus
- If no response is received, the system assumes the user is distracted

2. **WhatsApp Notification System**

If a user fails to respond to a check-in, the system sends a WhatsApp message to their accountability partner:

- The message indicates that the user missed a focus check-in
- The actual message that is sent can be edited by the user and the user can essentially send whatever he wishes.

This creates social accountability that helps users maintain focus and commitment to their study goals.

Chapter 5: User Interface

5.1 Landing Page

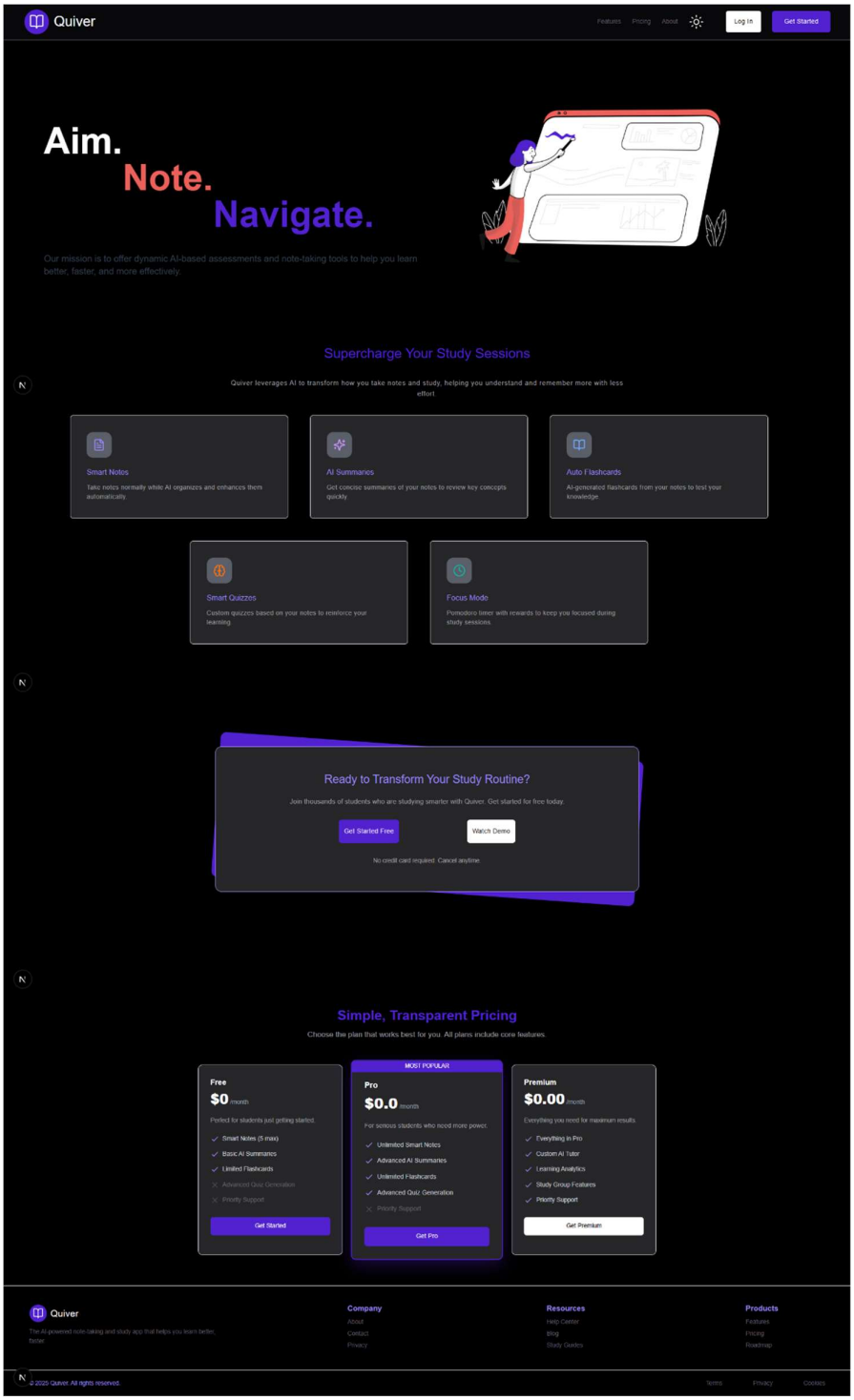

Figure 7 Landing Page

5.2 About Page

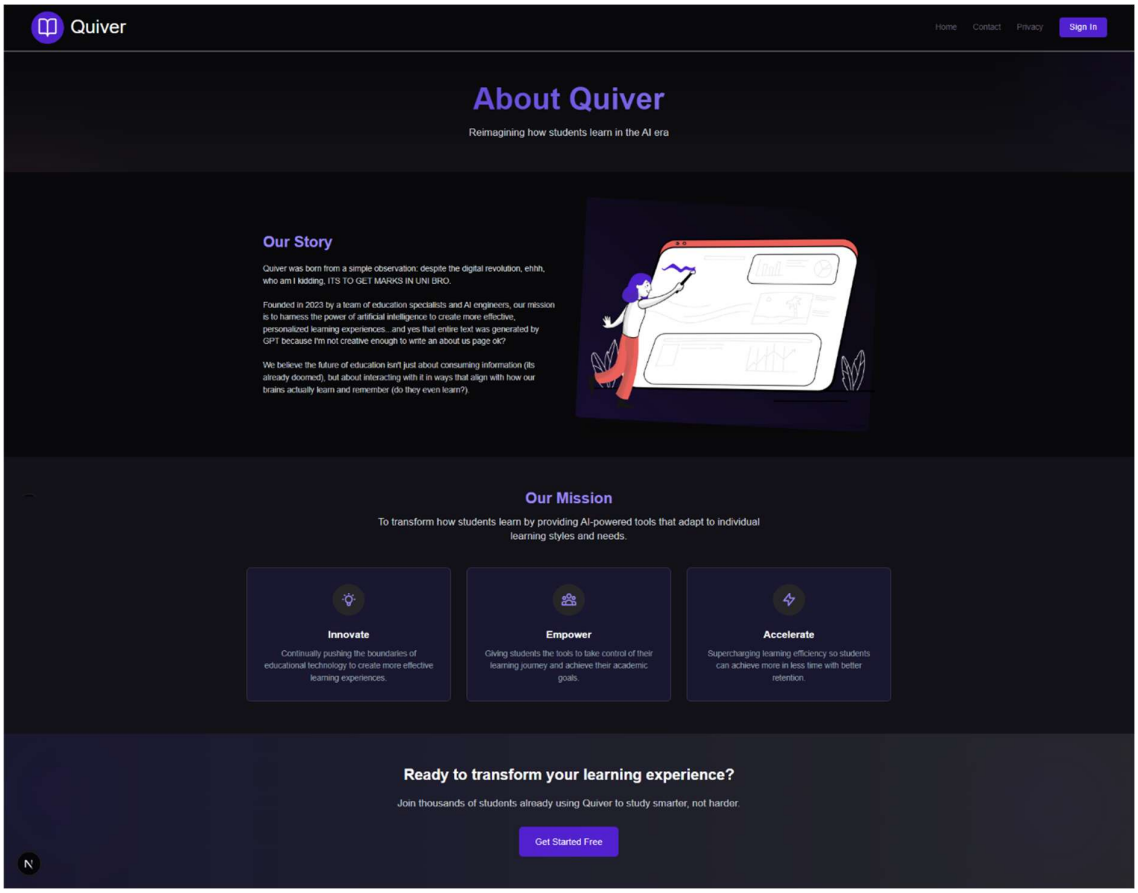

Figure 8 About Page

5.3 Signup

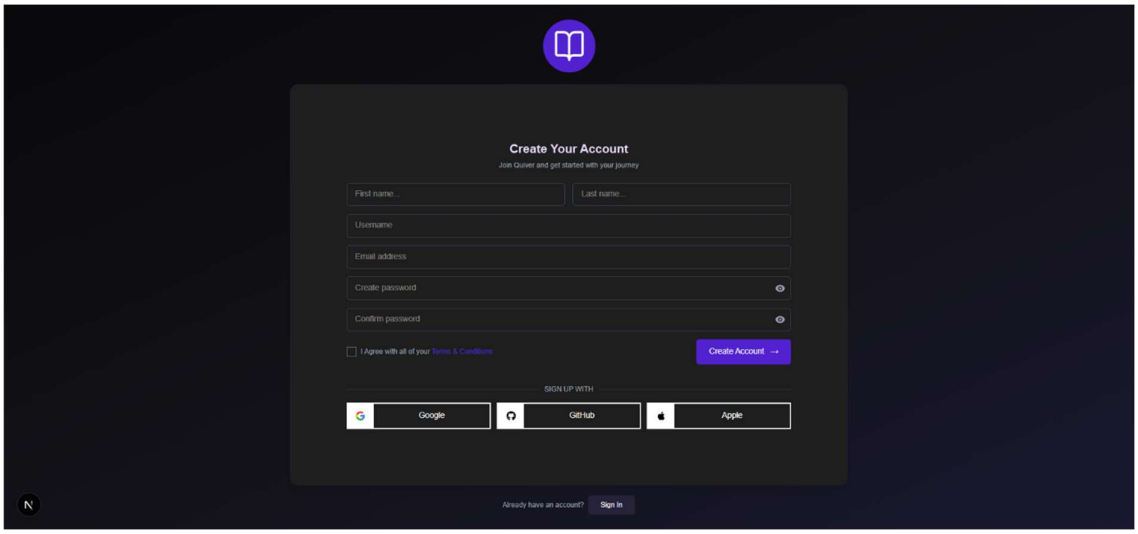

Figure 9 SignUp Page

5.4 Login

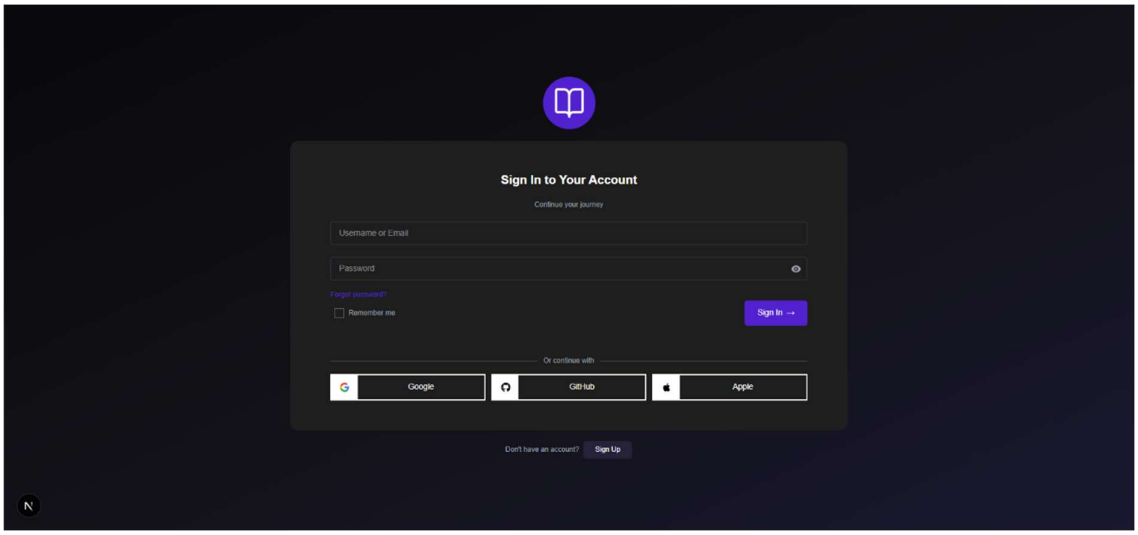

Figure 10 LogIn Page

5.5 Terms and Conditions

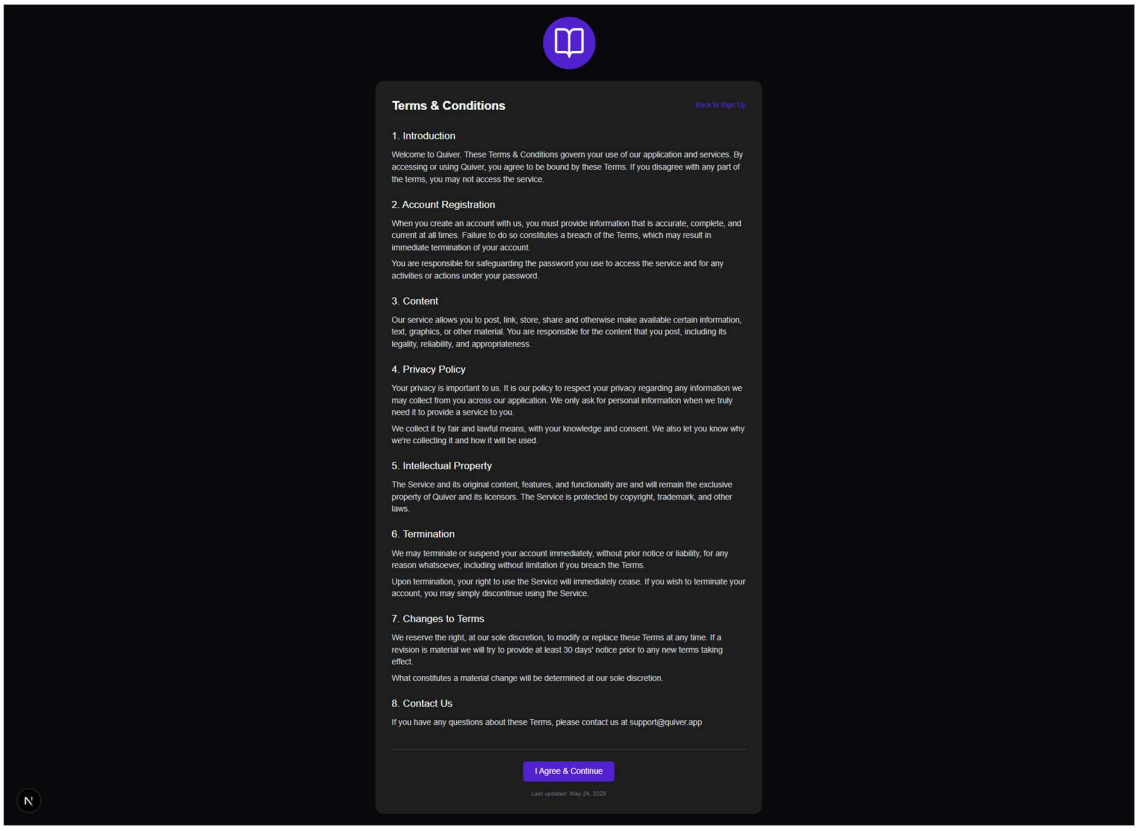

Figure 11 Terms and Conditions Page

5.6 Forgot password

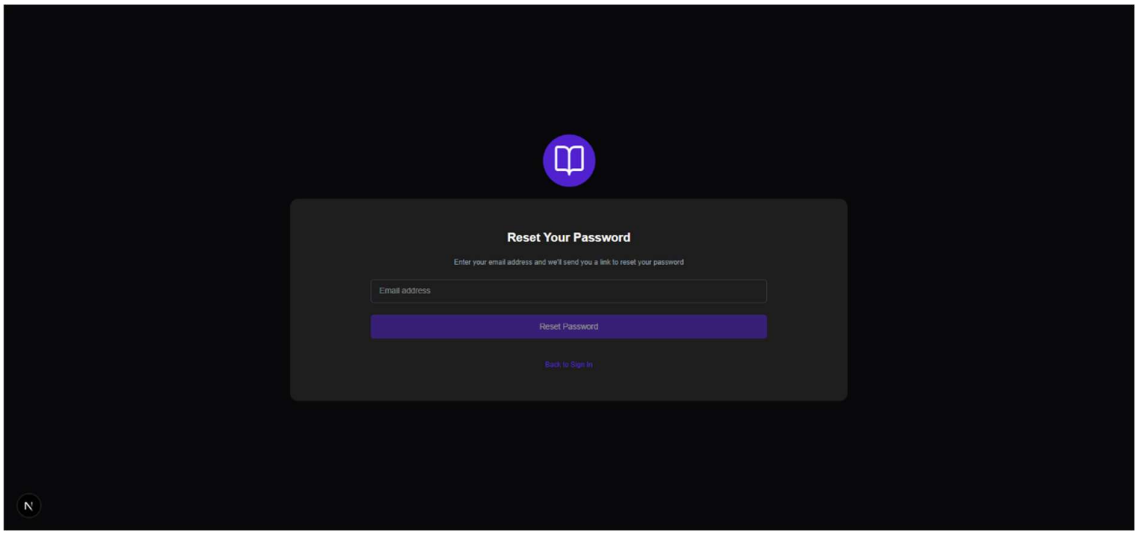

Figure 12 Forgot Password Page

5.7 Dashboard

5.8 Accounts

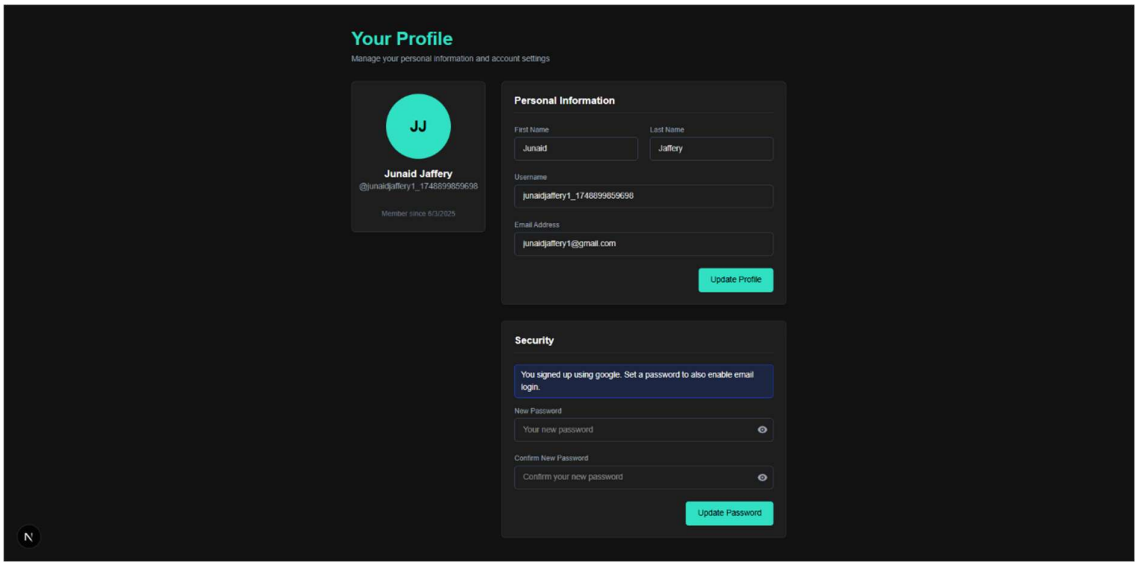

Figure 13 Account Management Page

5.9 Settings

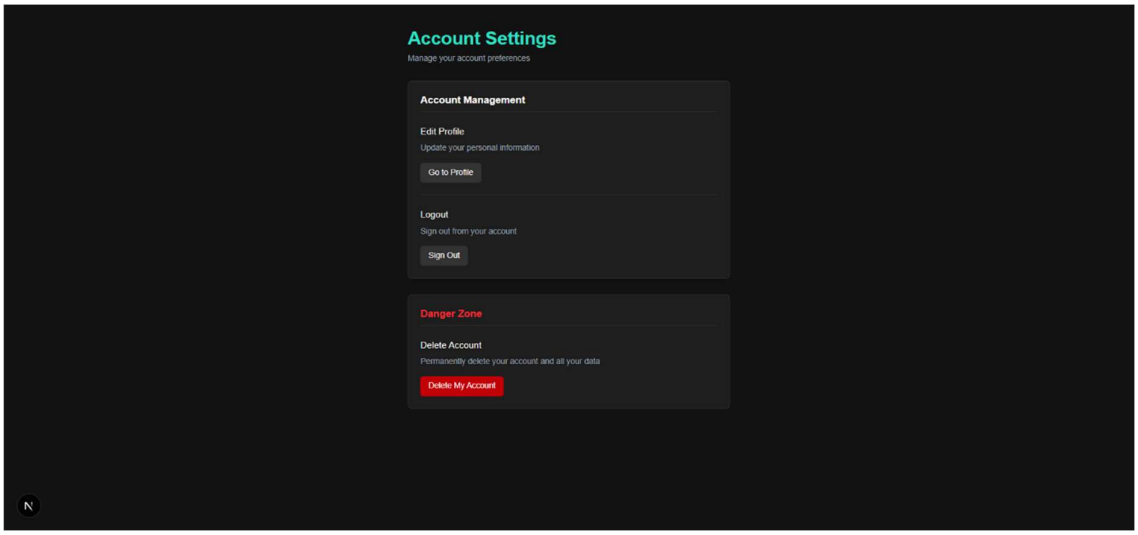

Figure 14 Settings Page

5.10 Notes

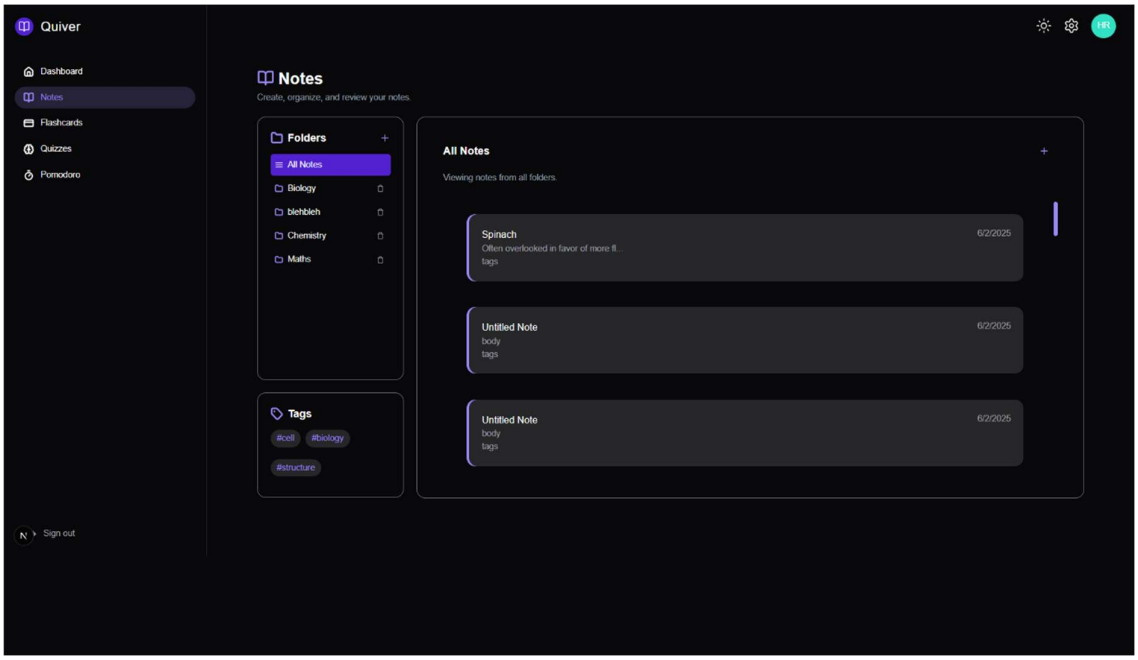

Figure 15 Notes Page

5.11 [Noteid]

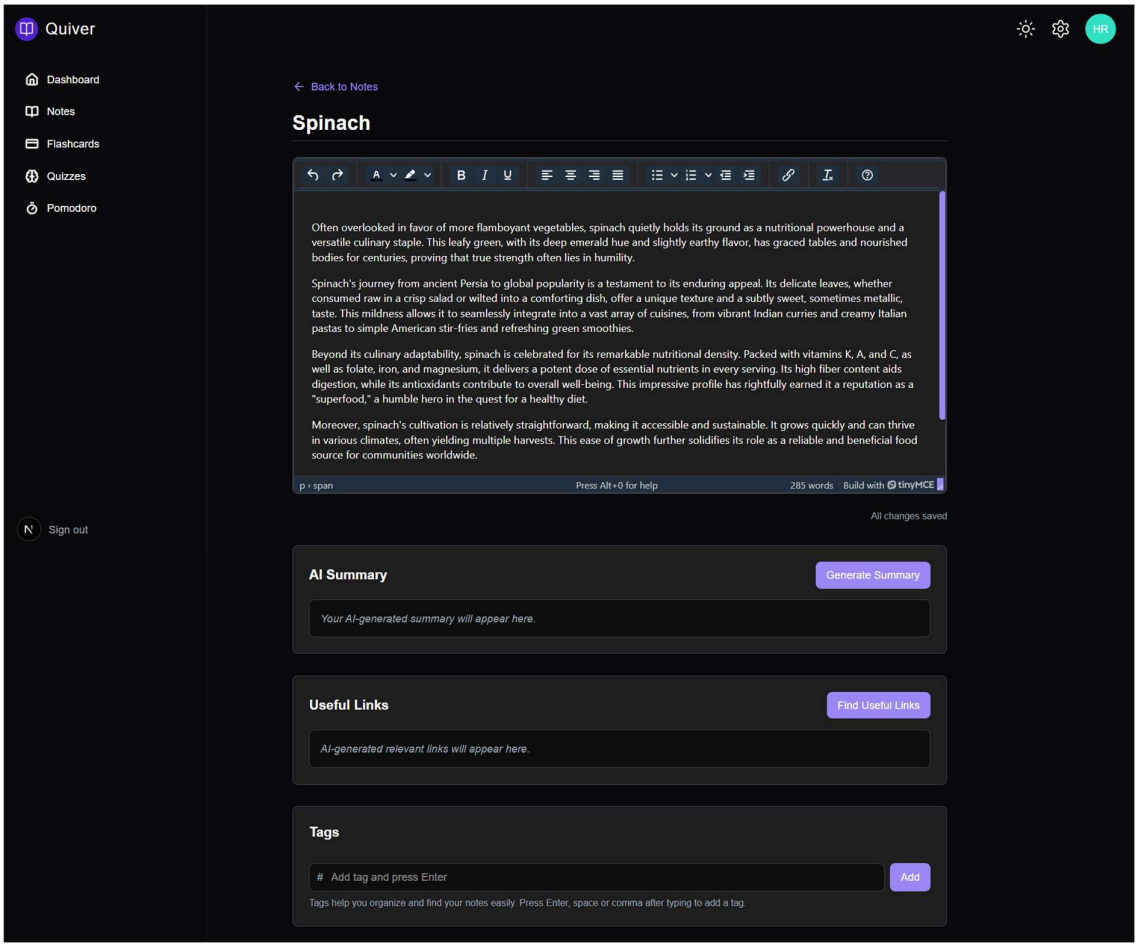

Figure 16 Inner Note Editing Page

5.12 Flashcards

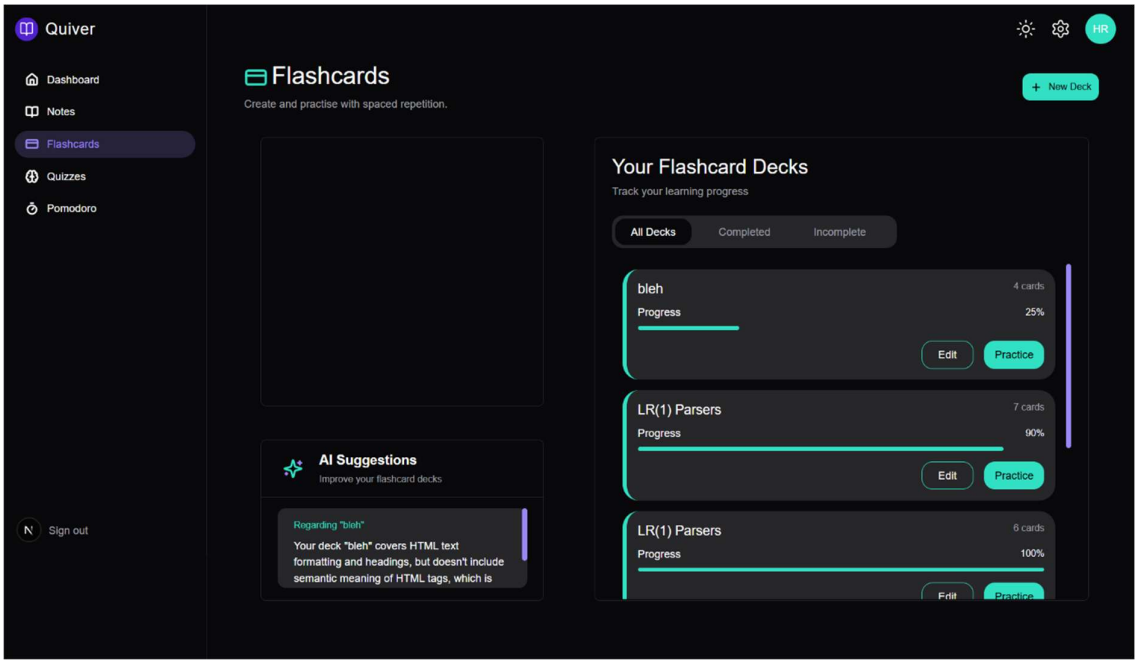

Figure 17 Flashcards Page

5.13 Create Flashcard Model

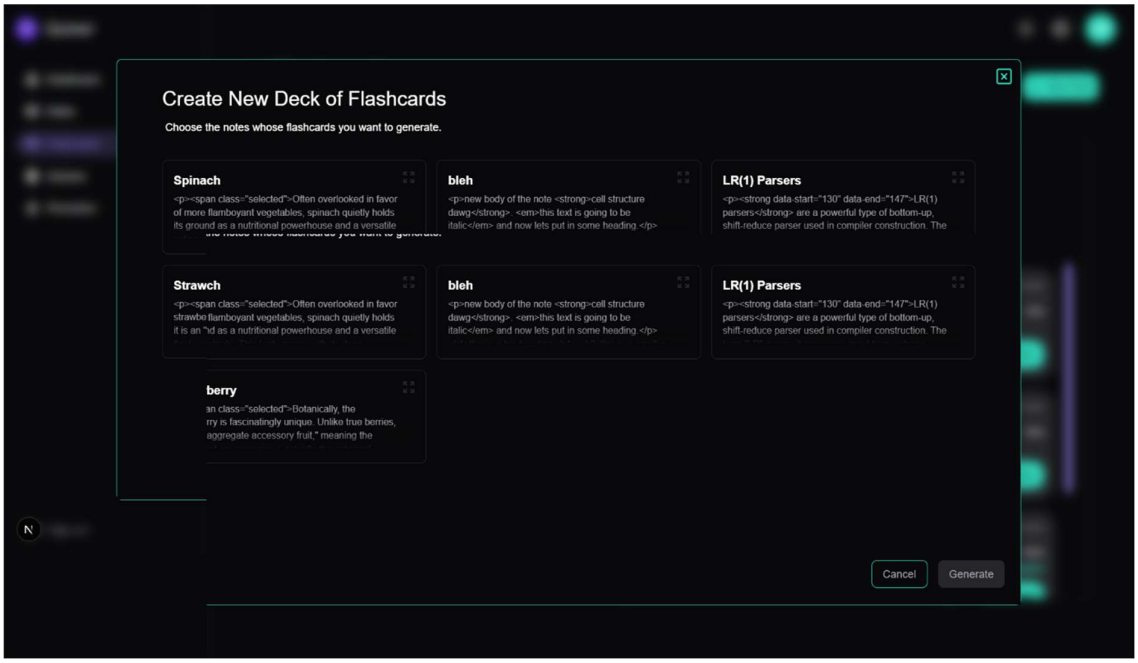

Figure 18 Flashcard Creation Page

5.14 View/Practice Created Flashcards Deck

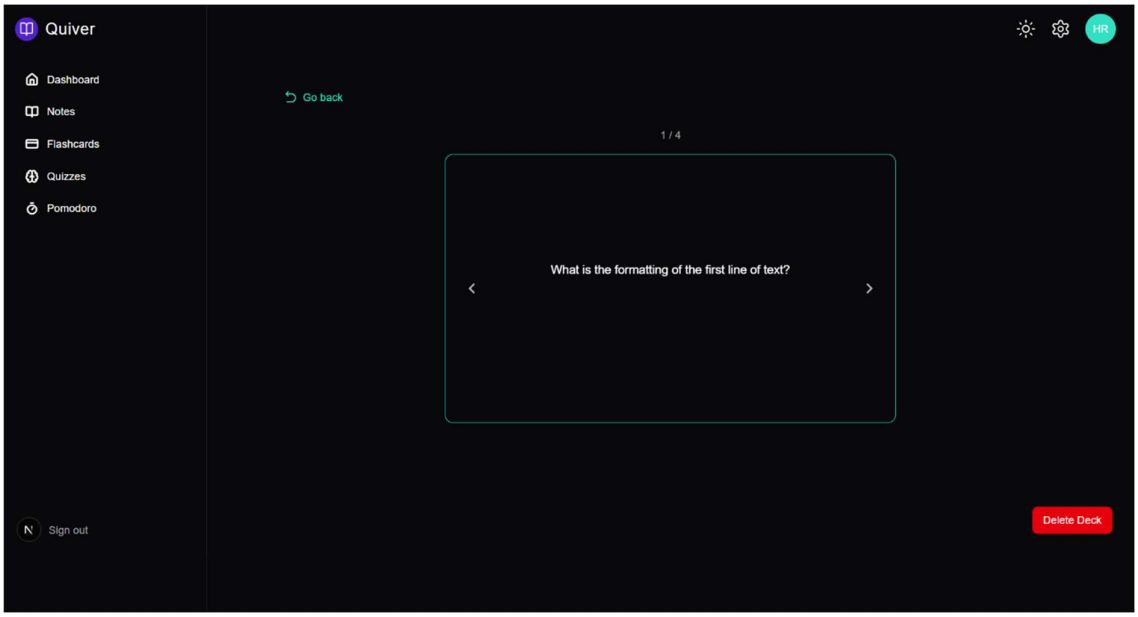

Figure 19 View and edit Flashcards Page

5.15 Pomodoro

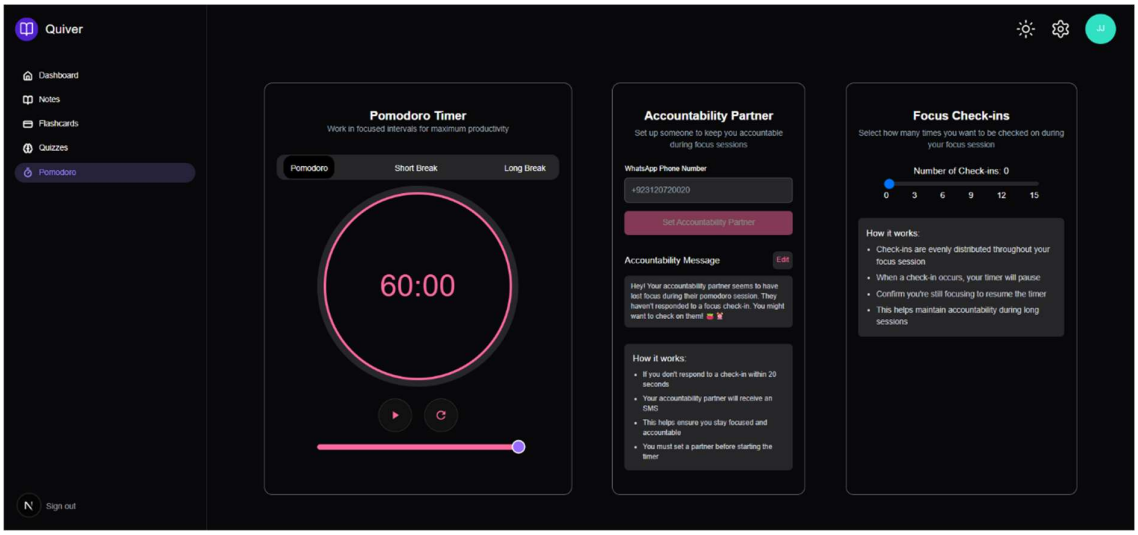

Figure 20 Pomodoro Page

Chapter 6: Limitations and Future Scope

6.1 Limitations

6.1.1 Lack of Note Import Functionality

Quiver currently limits users to creating notes directly within the application's editor. The platform does not support importing content from external file formats such as PDFs, PowerPoint presentations, Word documents, or even plain text files. This limitation requires users to manually transfer information from existing study materials, which can be time-consuming and inefficient. For students with extensive pre-existing notes or course materials, this represents a significant barrier to adoption as it necessitates duplicating content rather than leveraging existing resources.

6.1.2 Single Accountability Partner Restriction

The Pomodoro feature's accountability system only permits users to designate one accountability partner at a time. This restriction limits the flexibility of the accountability network, especially for users who might benefit from different accountability partners for different subjects or time periods. Additionally, the current implementation does not allow for group accountability, which could provide enhanced motivation through peer groups or study circles. The single-partner model may not adequately address varying accountability needs across different study contexts.

6.1.3 Limited Manual Flashcard Creation

While Quiver excels at generating flashcards through AI analysis of notes, the system does not currently support robust manual flashcard creation. Users cannot easily create custom flashcards from scratch with fully customizable content, formatting, and media. This limitation constrains users who prefer to create precisely tailored study materials or who wish to incorporate specialized content that may not be effectively captured through the AI generation process. Additionally, the lack of manual creation options restricts the types of knowledge that can be effectively represented in the flashcard format.

6.1.4 Incomplete Quizzing System

The application's quizzing functionality remains in early development stages and lacks comprehensive features for effective knowledge assessment. Currently, the system does not offer varied question types (multiple choice, short answer, matching, etc.), customizable quiz parameters, or detailed performance analytics. Without these features, users cannot fully leverage spaced repetition principles or receive detailed feedback on their knowledge gaps. This limitation affects the application's ability to serve as a complete study solution, particularly for test preparation scenarios.

6.2 Future Scope

6.2.1 Comprehensive Import/Export Ecosystem

Future development will focus on creating a robust import/export ecosystem that allows seamless integration with common educational file formats. This enhancement will include:

- PDF import with intelligent structure recognition to preserve headings and formatting
- PowerPoint slide import with automatic note generation from slide content
- Word document import with formatting preservation
- Image-to-text functionality for handwritten notes using OCR technology
- Batch import capabilities for processing multiple files simultaneously
- Export options in various formats (PDF, DOCX, HTML) for offline study or sharing

This expansion will significantly reduce the friction of adopting Quiver within existing educational workflows by eliminating manual content transfer.

6.2.2 Enhanced Collaborative Study Environment

The future version of Quiver will transform from a primarily individual tool to a collaborative learning platform by implementing:

- Multi-partner accountability systems in the Pomodoro feature, allowing users to create accountability circles
- Real-time collaborative note-taking with multiple users editing simultaneously
- Note sharing capabilities with granular permission settings
- Group flashcard deck creation and collaborative study sessions
- Discussion threads attached to specific notes or flashcards
- Integration with video conferencing tools for virtual study groups

These features will leverage the social aspects of learning, which research consistently shows enhances knowledge retention and understanding.

6.2.3 Advanced Flashcard Management System

To address current limitations in the flashcard system, future development will create an advanced flashcard management system including:

- Comprehensive manual flashcard creation with rich formatting options
- Multimedia flashcard support (images, audio, video)
- Custom flashcard templates for different knowledge domains
- Advanced spaced repetition algorithms that adapt to individual learning patterns
- Difficulty ratings and confidence scoring for personalized review scheduling
- Flashcard organization by tags, subjects, and knowledge clusters
- Bulk editing and management tools for large flashcard collections

These enhancements will make the flashcard system more flexible and powerful for diverse learning needs.

6.2.4 Intelligent Assessment Platform

Building upon the current quizzing capabilities, Quiver will develop a comprehensive assessment platform featuring:

- Diverse question types (multiple-choice, fill-in-the-blank, matching, ordering)
- AI-generated practice tests from notes with difficulty customization
- Performance analytics with knowledge gap identification
- Adaptive quizzing that focuses on areas needing improvement
- Timed test simulations for exam preparation
- Integration with academic learning objectives and standards
- Peer comparison options (anonymized) for benchmark assessment

This system will provide actionable insights into learning progress and effectively prepare users for formal assessments.

This expansion will ensure that Quiver's functionality is accessible regardless of device or location, supporting flexible study habits.

The roadmap for Quiver's evolution focuses on addressing current limitations while expanding into new areas that enhance the learning experience. By building upon the solid foundation of the current implementation, future versions will provide an increasingly comprehensive tool that adapts to diverse educational needs and technological environments. The development priorities will be informed by user feedback, educational research, and emerging technologies to ensure Quiver remains relevant and effective in supporting academic success.

Chapter 7: Conclusion

Quiver represents a significant advancement in educational technology, combining cognitive science principles with modern web development to create a comprehensive learning platform. Throughout this documentation, we have explored how the application integrates three core functionalities—structured note-taking, active recall through flashcards, and focused study sessions with the Pomodoro technique—to address the multifaceted challenges faced by students in today's information-dense academic environment. By leveraging AI capabilities through the Llama3 model, Quiver transforms passive information consumption into active knowledge construction, helping students not just capture but truly understand and retain critical concepts. The application's thoughtful design, with its dark-themed interface and intuitive navigation, demonstrates our commitment to creating a learning environment that reduces cognitive load while maximizing productivity.

The development process of Quiver has exemplified the power of integrating robust technologies like Next.js, Supabase, TinyMCE, and Groq to create a seamless user experience. The project's architecture prioritizes both performance and security, ensuring that students can focus on learning rather than wrestling with technical limitations. The implementation of features like automatic note summarization, AI-generated flashcards, and accountability through WhatsApp notifications showcases how technology can be harnessed to enhance rather than distract from the learning process. Each component of Quiver has been designed with user needs at the forefront, resulting in a platform that adapts to diverse learning styles while maintaining cohesion across functionalities. The application succeeds not just as a collection of tools but as an integrated ecosystem that guides users through the entire study process.

While Quiver has achieved its primary objectives, the journey of development has illuminated both current limitations and exciting possibilities for future enhancement. The identified areas for growth—including expanded import capabilities, collaborative features, advanced flashcard systems, and comprehensive assessment tools—point toward a vision of Quiver as an ever-evolving platform that responds to emerging educational needs and technologies. As education continues to transform in the digital age, Quiver stands poised to evolve alongside it, maintaining its core mission of empowering students while embracing new opportunities to enhance the learning experience. The success of this project demonstrates that thoughtful application of technology, grounded in educational principles, can create meaningful tools that make a tangible difference in academic achievement and lifelong learning.