PAPER NAME

Node App Saakshi MAD Report B033.docx

AUTHOR

Saakshi Jain

WORD COUNT

2198 Words

CHARACTER COUNT

14898 Characters

PAGE COUNT

27 Pages

FILE SIZE

3.3MB

SUBMISSION DATE

Nov 6, 2024 10:21 PM GMT+5:30

REPORT DATE

Nov 6, 2024 10:22 PM GMT+5:30

9% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

- 3% Internet database
- 0% Publications database
- Crossref database
- Crossref Posted Content database
- 9% Submitted Works database

Excluded from Similarity Report

- Bibliographic material
- Quoted material
- Cited material
- Small Matches (Less then 14 words)

DEPARTMENT OF COMPUTER ENGINEERING

Mukesh Patel College Of Engineering
A.Y. 2024-25

MOBILE APPLICATION DEVELOPMENT PROJECT REPORT ON

MindMap Notes: Intelligent Note-Taking with Firebase and Gemini API Integration.

Submitted By

Roll No: – B033
Name - Saakshi Jain

Under The Guidance of

Professor: Israt Ali

TABLE OF CONTENT

INTRODUCTION

The **Notes Application** has been designed working with accessibility, security and comprehensiveness-a true platform from which users can easily create, manage and access personal notes. Being deeply in line with Firebase Firestore for effective data management, RecyclerView for the dynamic presentation of the notes and Gemini API for interesting knowledge retrieval, this application will be much more powerful than ordinary note-taking software that comes along with features like quiz generation and resource recommendations, in real effect.

Project Overview

- **Goal**: Create a user-friendly platform for creating, saving, editing, deleting, and sharing notes with enhanced security and AI-driven features.
- **Target Users**: Ideal for students, professionals, and anyone needing organized information management.
- **Key Objectives**:
    a. **Seamless Note Management**: Capture, update, and organize notes easily.
    b. **Enhanced Interactivity**: AI-driven quizzes and learning recommendations from notes.
    c. **User-Focused Design**: Prioritizes accessibility, ease of use, and data security.

Key Features

- ➔ **Save Notes with Timestamp**
  - ◆ **Description**: Saves notes with the current date/time for chronological organization.
  - ◆ **Implementation**: Firebase Firestore timestamps notes for sorting/searching.
- ➔ **Edit, Delete, and Share Notes**
  - ◆ **Description**: Users can update, remove, or share notes via external apps.
  - ◆ **Implementation**:
    - **Edit**: Open notes in editor.
    - **Delete**: Remove notes from Firestore.
    - **Share**: Share notes with Android's `Intent` system.
  - ◆ **Benefits**: Enhances flexibility and control over notes.
- ➔ **Authentication & Security**
  - ◆ **Description**: Email-based authentication with Firebase ensures secure access.
  - ◆ **Implementation**: Firebase Authentication handles sign-in, sign-up, password reset, and email verification.
  - ◆ **Benefits**: Secures notes and builds user trust.
- ➔ **Knowledge Retrieval with Gemini API**
  - ◆ **Description**: Integrates with Gemini API to generate quizzes and resources based on note content.
  - ◆ **Implementation**:
    - The app sends notes to Gemini API for quiz generation and study resources.

◆ **Benefits**: Enables active learning by providing insights and reinforcing knowledge.

Additional Project Details

- **Database Choice**: **Firebase Firestore** – Ideal for real-time updates, hierarchical data storage, and scalability.
- **User Interface**: **RecyclerView** – Dynamic display of notes with efficient data binding.

This feature-rich Notes App functions as a secure, interactive digital notebook for capturing, reviewing, and learning from notes.

Figure 1: Notes App

Database Selection: Cloud Firestore vs. Realtime Database

Cloud Firestore

1. **Usage**: Best for apps needing scalability, real-time updates, and complex data structures (e.g., chat, social networking).
2. **Advantages**:
    - **Document-Oriented Model**: Organizes data in documents within collections, ideal for complex data.
    - **Real-time Updates**: Instant data syncing across devices.
    - **Scalability**: Automatically scales to handle growing user bases and datasets.
    - **Offline Support**: Access it offline and synch automatically once the connection is obtained.
    - It has **strong querying** and supports advanced queries such as filtering and sorting.
3. **Drawbacks**: More for voluminous operations; too heavy for simple data requirements.
    - Use cases:
    - Real-time Collaboration Apps: Messaging, document editing, multiplayer games.
    - Content Management Systems: Flexible data models for changing content.
    - E-commerce platforms include product listings, orders, and inventory updates.

- Social Apps: Feeds, notifications, and interactions with real-time updates.

Realtime Database

Suitable for simpler data structures and fast data updates, for instance in webshops or real-time dashboards.

Benefits:

- **Real-time synchronization**: automatic update on other connected devices.
- **Hierarchical JSON Structure**: This is an easy data model to work with for smaller projects.
- **Speed:** Optimized for low latency, so very responsive.
- **Simplicity:** Smoother setup of projects that involve less complex data.
- **Less support for complex queries:** slower syncing for large volumes of data.

Use Cases:

- Chat Applications Simple **real time messaging** using simple data structures.
- **Live Dashboards** Very effective apps for live monitoring data display.
- **Location tracking apps:** Intelligent location management in real time.
- **Simple Multiplayer Games:** manages player score, leaderboard, and game states in real time.

Each of the databases serves different needs, with Firestore offering more robust functions for scalable applications with data intensives and Realtime Database being optimal for lightweight real-time updates.

RecyclerView Implementation

➔ **Overview**: RecyclerView is used to display notes in a scrollable list format, providing efficient memory management and a better user experience.

➔ **Core Methods**:

- ◆ **`OnCreateViewHolder:`** `Creates all the view holders.`
- ◆ **`onBindViewholder:`** `It binds the data to view holders for a list of items.`
- ◆ **`getItemcount :`** `Returns the number of total items, to be used for defining the RecyclerView size dynamically.`

➔ **Advantages of RecyclerView**: Efficient performance for large lists, flexible layouts, optimized memory usage.

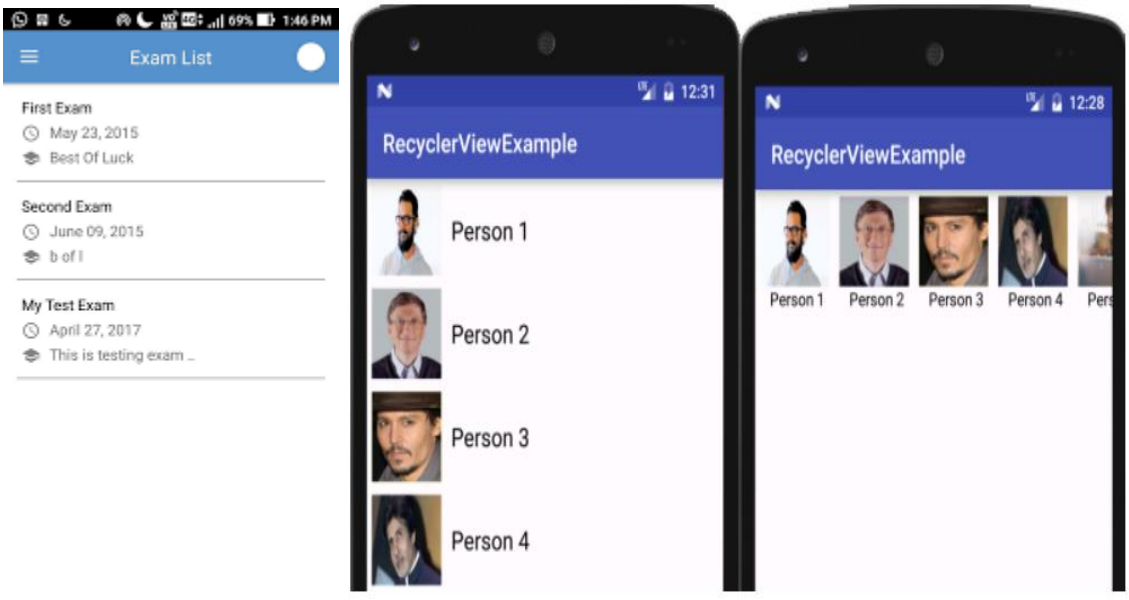


Figure 2: Recycler View

Adapter and ViewHolder Pattern

- **Adapter**: Acts as a bridge between the data source (Firestore) and the RecyclerView. Handles data population and updates.
- **ViewHolder**: Manages and recycles views to minimize repeated layout inflation, optimizing performance.
- **Customization**: Custom adapter setup for notes, including layout binding and handling actions (edit, delete, share).

Figure 3: Adapter

Integration with Gemini API

- **Objective**: Provides users with a knowledge engine by retrieving answers and related information to user queries.
- **Implementation Overview**: Sending requests to the Gemini API with user queries and displaying responses within the app.
- **Benefits**: Enhances user experience by providing a mini AI assistant directly in the notes app.

Steps are

1. API Key from Google AI Studio: Get Your API Key : https://aistudio.google.com/app/apikey .

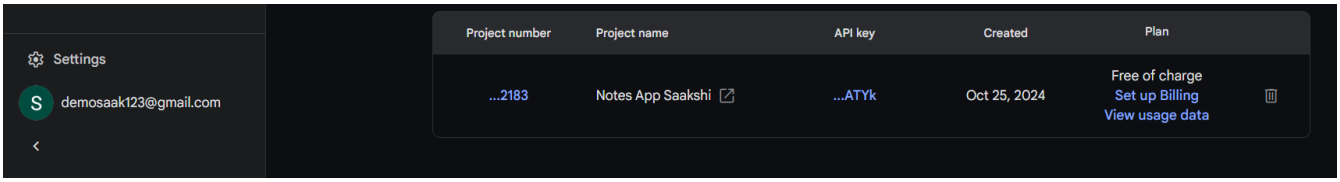


Figure 4: API Key Generation

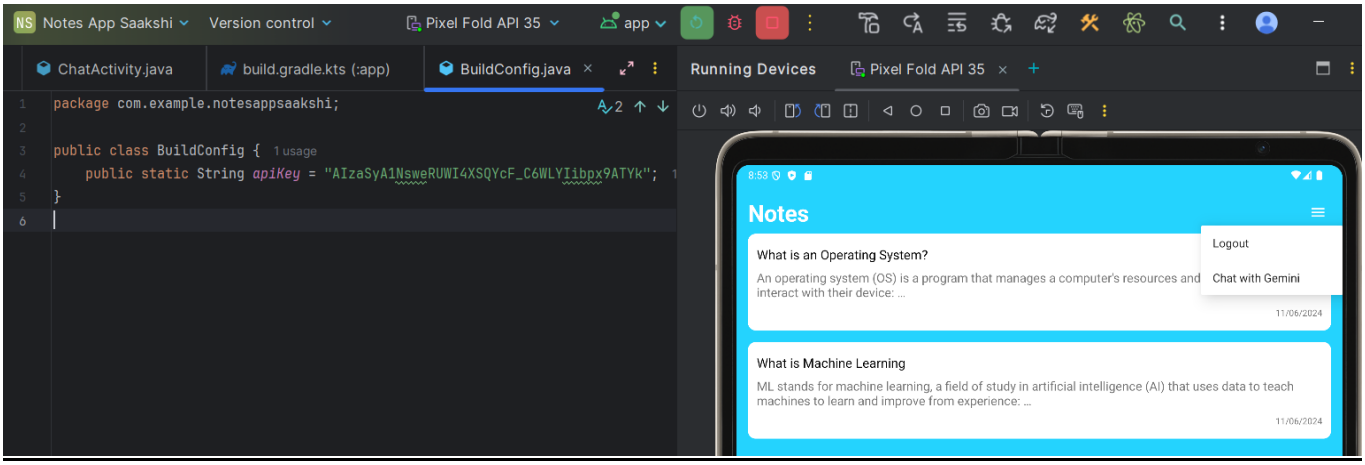


Figure 5: API Key Integration

2. Latest preview version of Android Studio Canary Build : https://developer.android.com/studio/preview .

3. Google's Documentation for Gemini API: https://ai.google.dev/gemini-api/docs/quickstart?lang=android#java_1

4. Adding Dependencies:

Figure 6: Dependencies Integration

Activity Design

- Design a responsive UI with ScrollView.
- Integrate TextView, TextInputLayout, Button, ProgressBar, etc.
- Customize UI elements for a visually appealing app.

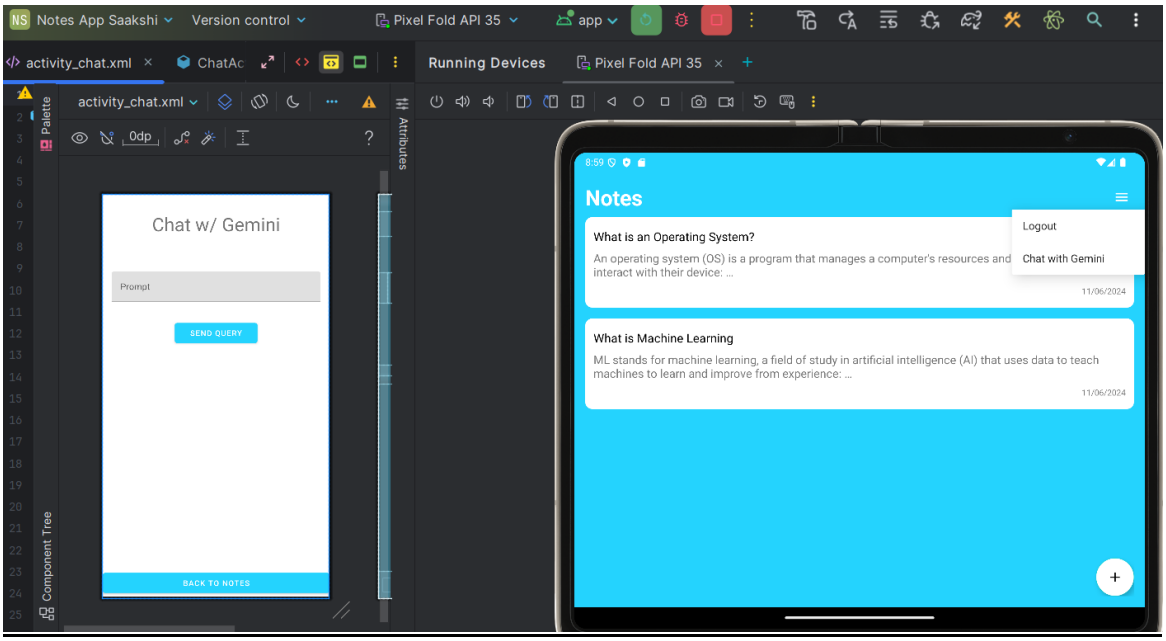


Figure 7: Activity Design

The Java Coding Steps

- Use an explicit intent to navigate from MainActivity to ChatActivity, enabling the chat interface when the "Chat with Gemini" menu item is selected.
- Instantiate views in the ChatActivity class.
- Create the GeminiPro class for Gemini model interactions.
- Explore the getModel and getResponse functions.
- Handle responses asynchronously using the ResponseCallback interface.

ChatAcitivity.java

```
        backbtn.setOnClickListener(new View.OnClickListener() {

            @Override

            public void onClick(View v) {

                Toast.makeText(ChatActivity.this, "Loading to Notes App",
Toast.LENGTH_SHORT).show();

                Intent intent = new Intent(getApplicationContext(),MainActivity.class);

                startActivity(intent);

                finish();

            }

        });

        sendQueryButton.setOnClickListener(v -> {

            GeminiPro model = new GeminiPro();

            String query = queryEditText.getText().toString();

            progressBar.setVisibility(View.VISIBLE);

            responseTextView.setText("");

            queryEditText.setText("");

            model.getResponse(query, new ResponseCallback() {

                @Override

                public void onResponse(String response) {

                    responseTextView.setText(response);

                    progressBar.setVisibility(View.GONE); }

                @Override

                public void onError(Throwable throwable) {
```

```
                    Toast.makeText(ChatActivity.this, "Error: " + throwable.getMessage(),
Toast.LENGTH_SHORT).show();

                    progressBar.setVisibility(View.GONE);

                }});});}}
```

GeminiPro.java

```
public class GeminiPro {

    public void getResponse(String query, ResponseCallback callback) {

        GenerativeModelFutures model = getModel();

      Content content = new Content.Builder().addText(query).build();

      Executor executor = Runnable::run;

        ListenableFuture<GenerateContentResponse> response = model.generateContent(content);

        Futures.addCallback(response, new FutureCallback<GenerateContentResponse>() {

            @Override

            public void onSuccess(GenerateContentResponse result) {

                String resultText = result.getText();

                callback.onResponse(resultText); }

            @Override

            public void onFailure(Throwable throwable) {

                throwable.printStackTrace();

                callback.onError(throwable); } }, executor); }

    private GenerativeModelFutures getModel() {

        String apiKey = BuildConfig.apiKey;

        SafetySetting harassmentSafety = new SafetySetting(HarmCategory.HARASSMENT,
```

```
            BlockThreshold.ONLY_HIGH);

    GenerationConfig.Builder configBuilder = new GenerationConfig.Builder();

    configBuilder.temperature = 0.9f;

    configBuilder.topK = 16;

    configBuilder.topP = 0.1f;

    GenerationConfig generationConfig = configBuilder.build();

    GenerativeModel gm = new GenerativeModel(

            "gemini-pro",

            apiKey,

            generationConfig,

            Collections.singletonList(harassmentSafety)  );

    return GenerativeModelFutures.from(gm);  } }
```

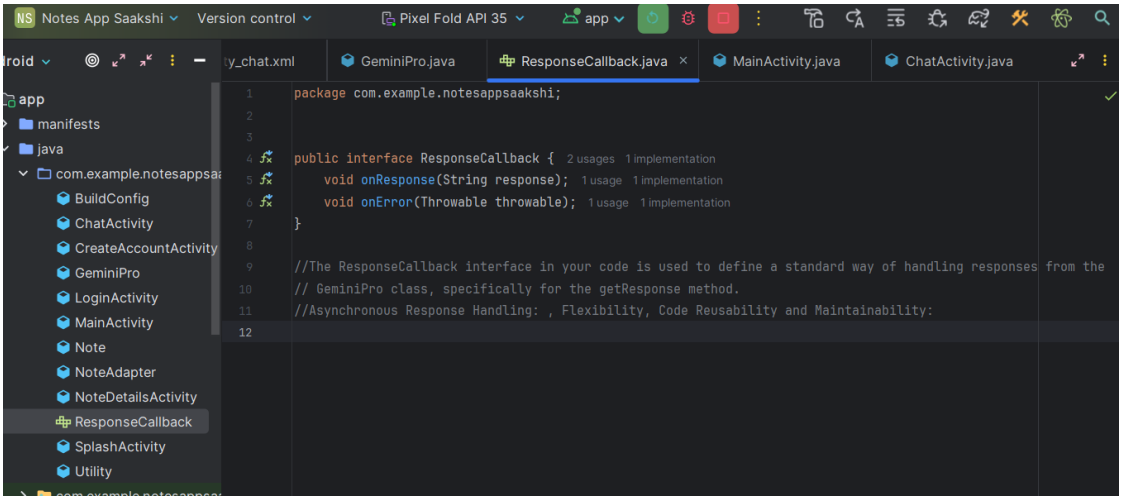


Figure 7: ResponseCallBack.java Code

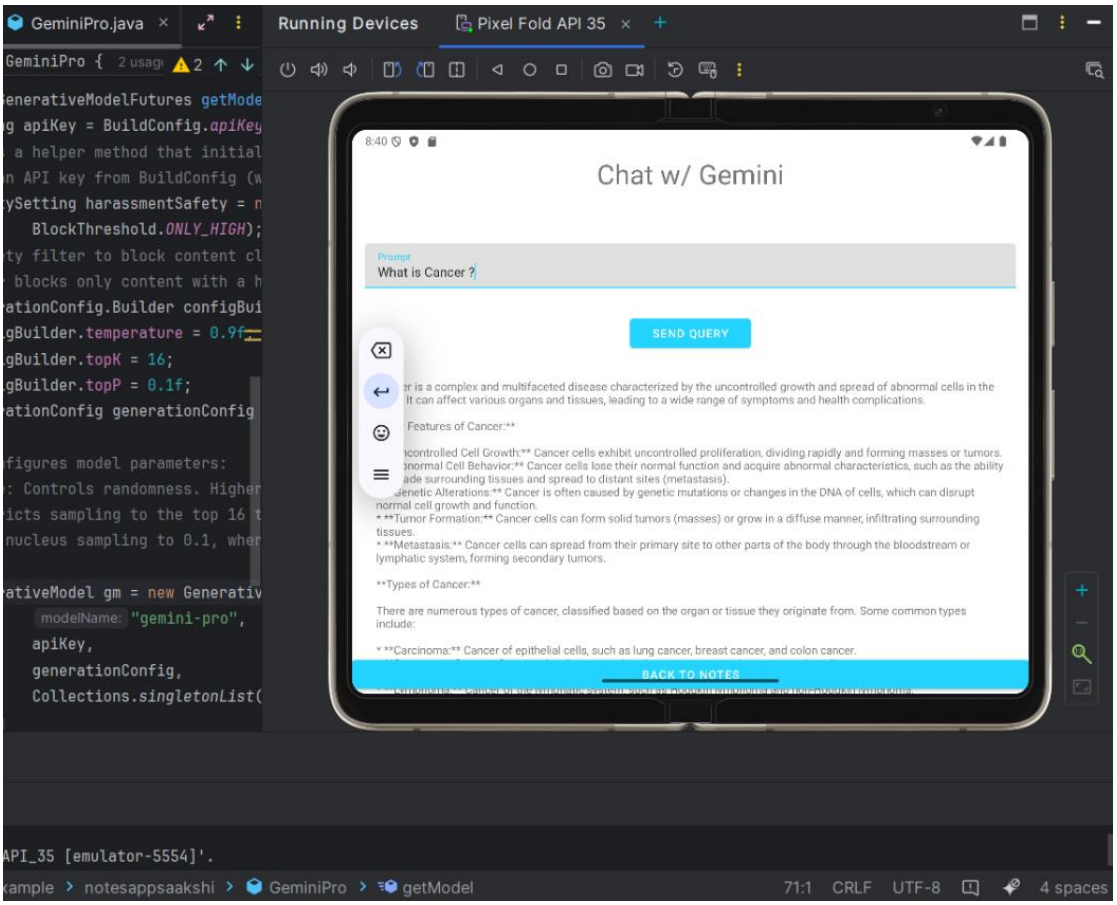


Figure 8: User Query Output

Email Authentication with Firebase

- **Purpose**: Protects user data by allowing only authenticated users to access and manage their notes.

- **Features**:
    1. **Email Verification**: Ensures registered users have valid email accounts.
    2. **Password Management**: Options for reset and retrieval through Firebase Authentication.
    3. **Strong Password Enforcement:** Requires users to create strong passwords that meet security standards, ensuring better protection for user accounts.

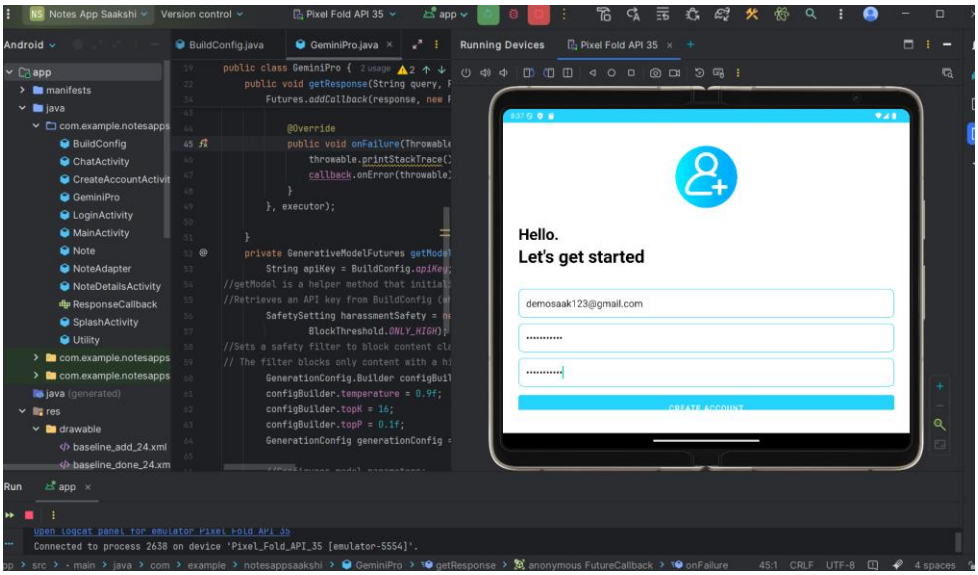


Figure 9: Create New User

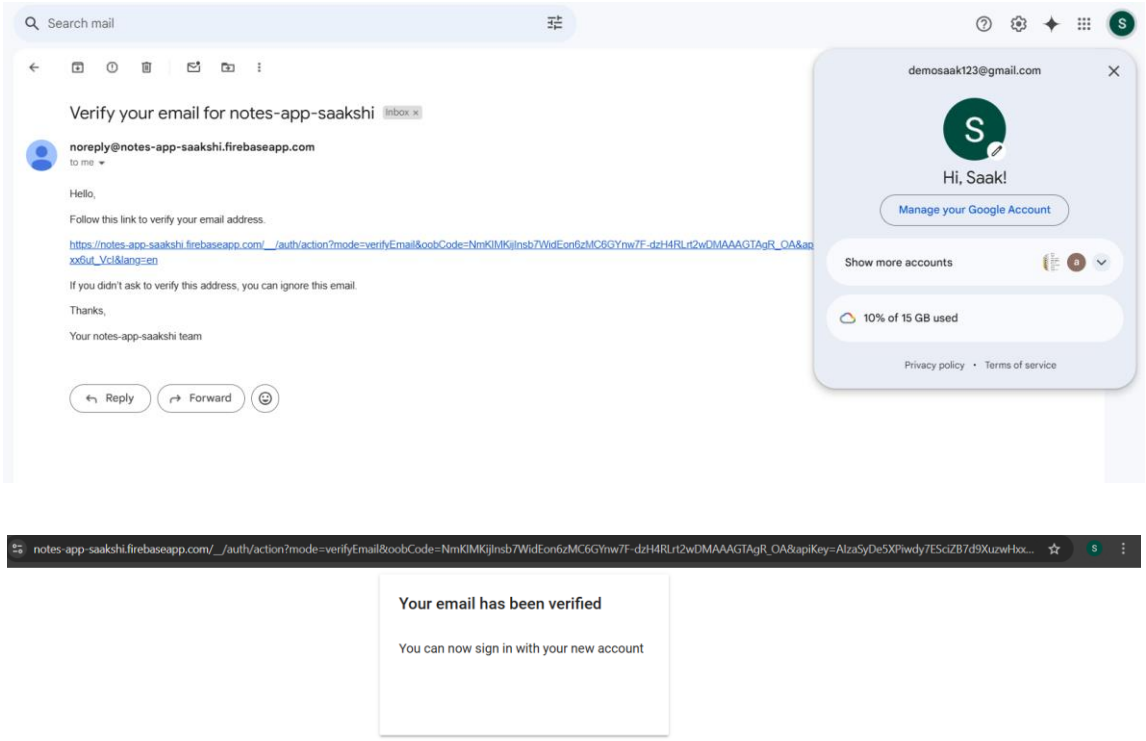


Figure 10: Email Verification

Code

```
public void onComplete(@NonNull Task<AuthResult> task) {
    changeInProgress(false);
    if(task.isSuccessful()){
        if(firebaseAuth.getCurrentUser().isEmailVerified()){
      startActivity(new Intent(LoginActivity.this,MainActivity.class));
            finish();
        }else{
            Toast.makeText(LoginActivity.this, "Email not verified, Please verify your email.", Toast.LENGTH_SHORT).show();}
 }else{  Toast.makeText(LoginActivity.this, task.getException().getLocalizedMessage(), Toast.LENGTH_SHORT).show();
```

```
}}}); }

 void changeInProgress(boolean inProgress){

     if(inProgress){

         progressBar.setVisibility(View.VISIBLE);

         loginBtn.setVisibility(View.GONE);

     }else{

         progressBar.setVisibility(View.GONE);

         loginBtn.setVisibility(View.VISIBLE);}}

 boolean validateData(String email,String password){

if(!Patterns.EMAIL_ADDRESS.matcher(email).matches()){

         emailEditText.setError("Email is invalid");

         return false;

     }

     if(password.length()<6){

         passwordEditText.setError("Password length is invalid");

         return false;

     }

     return true;

 }
```

User Interface & UX

- **Layout Files**: Overview of XML layouts for the main screen, note editing, authentication, and user settings.
- **Design Highlights**:
    a. Clean, minimalistic design with focus on usability.
    b. Consistent color themes and intuitive iconography for seamless navigation.

Data Flow and User Journey

- **Data Flow Diagram**: Visual representation of interactions between the user, RecyclerView, Firestore database, and Gemini API.
- **User Journey**: A walkthrough from user authentication, note creation, storage, and retrieval to querying knowledge.

Testing and Debugging

- **Testing Strategy**:
    a. **Unit Testing**: For methods like `onBindViewHolder` to check data binding.
    b. **Integration Testing**: Testing Firestore connectivity, authentication, and Gemini API integration.
- **Error Handling**: Ensuring error messages are user-friendly, and logging critical issues in Firebase.

PROBLEM STATEMENT

The world is now becoming very fast-paced and information-driven. So, note-taking tools must not only support efficient note-taking but also improve knowledge management with intelligent features. Conventional note-taking apps might be able to create and organize basic notes, but they are mostly deprived of good data security, **real-time data synchronization**, and advanced knowledge retrieval capabilities. Students and professionals, finally, require something more than a passive repository of notes-and tools that provide interactive, AI-driven insights that can deepen learning and build long-term retention.

MindMap Notes Application answers all these needs with a safe, accessible, and intelligent note-taking platform. Data management with Firebase Firestore will be secure and scalable, while the **Gemini API** brings forth real-time information retrieval and AI-powered content. Such an application empowers users to create, edit, organize, and share their notes as seamlessly as possible. Its features, including timestamp-based organization, **email-based authentication**, and knowledge retrieval, bring an **advanced level of functionality and personalization into the user experience.**

The challenge of the design lies in achieving an application that balances the accessibility of users, the security of data, and enhanced interactivity so that the app is not just a reliable digital notebook but also an intelligent learning assistant. This project will create a powerful tool for efficient note management and active learning that can be very **beneficial for students, professionals, and anyone looking for organized, AI-assisted information management.**

PROPOSED SYSTEM

Proposed System: Notes App with Firebase Firestore and RecyclerView

1. **Note Management**:
   - **Create, Edit, Delete, and Save Notes**: Users can create and manage notes, which are saved with timestamps for easy organization.
   - **Share Notes**: Users can share notes through external apps (e.g., email, messaging) using Android's `Intent` system.
2. **Database and Authentication**:
   - **Firebase Firestore**: Chosen for real-time data updates, complex data storage, and scalability. Used to store, update, and retrieve notes.
   - **Firebase Authentication**: Email-based authentication ensures secure access, with email verification for user safety.
3. **Enhanced Interactivity**:
   - **The Gemini API Integration** generates data and recommendations based on note content; thus, making the application a learning tool.
   - **Users may ask for quizzes based on their notes to drill in more learning.**
4. **UI Components**:
   - **RecyclerView for Display:** The system presents the notes as a scrollable, organized list by employing an adapter for efficient data binding and item layout control.
   - Adapter Class. Manages **onCreateViewHolder(), onBindViewHolder(), and getItemCount() methods** to improve RecyclerView performance.

5. **Security**:
    - **User Authentication and Email Verification:** The Firebase application guarantees safe access to notes that protect user information.
6. **Additional Functionalities**:
    - **Timestamping:** It notes the date and time on creation or edit, helping to track note history.
    - **Offline Support: This supports partial offline and the syncing of data on reconnection.**

It is designed for straightforward note-taking, enhanced data security, and a more interactive learning process through the power of artificial intelligence.

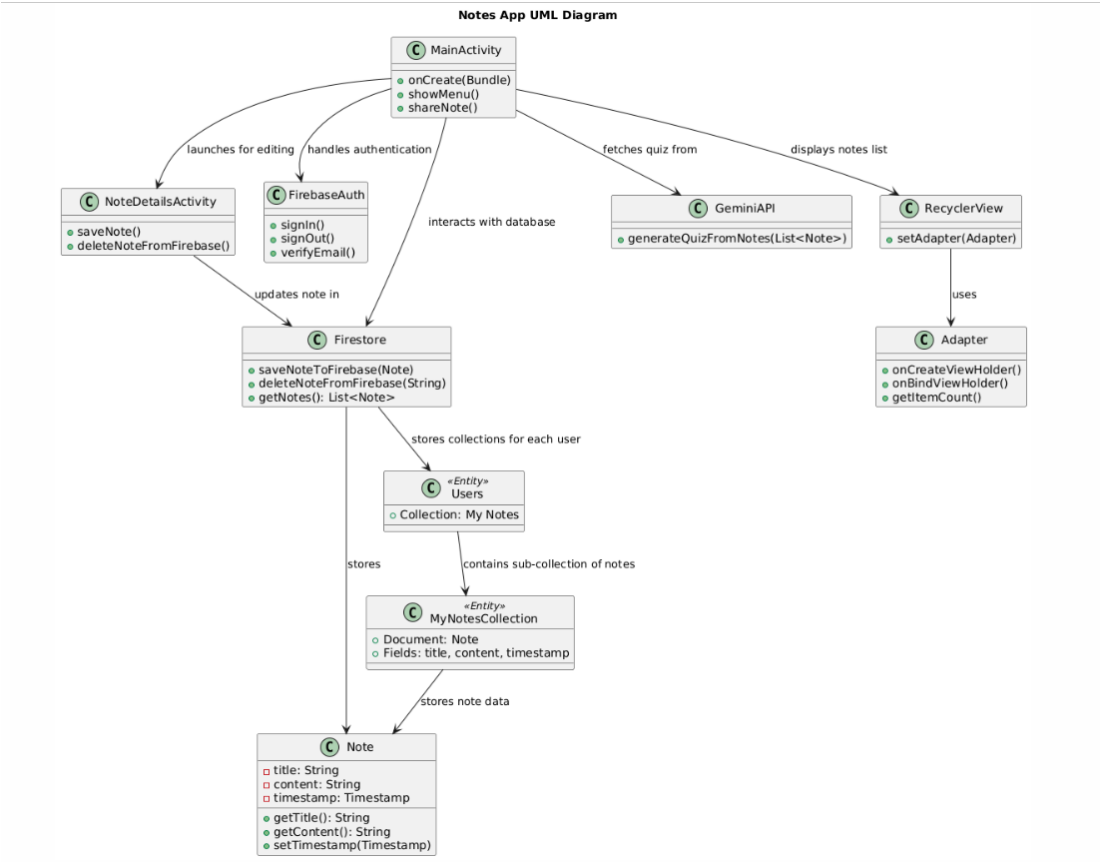
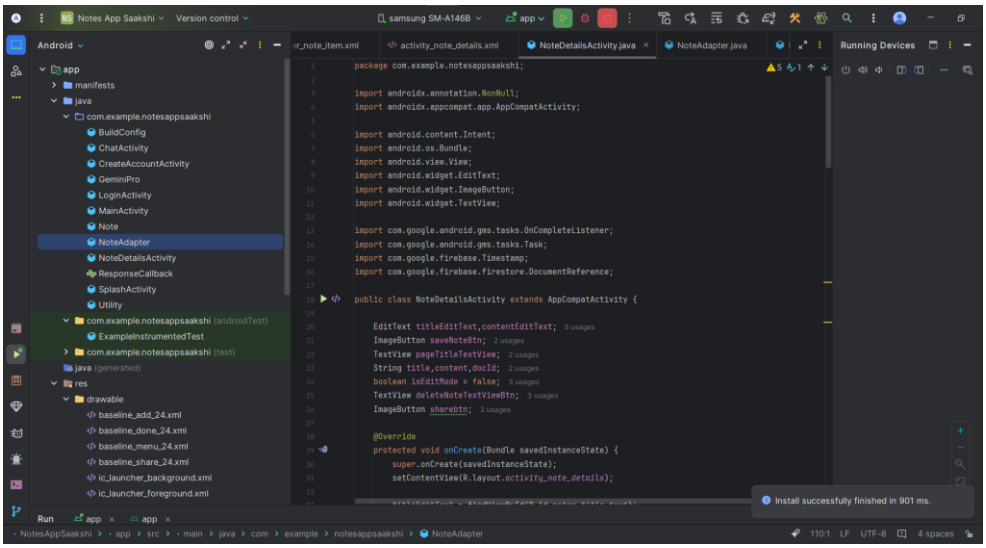
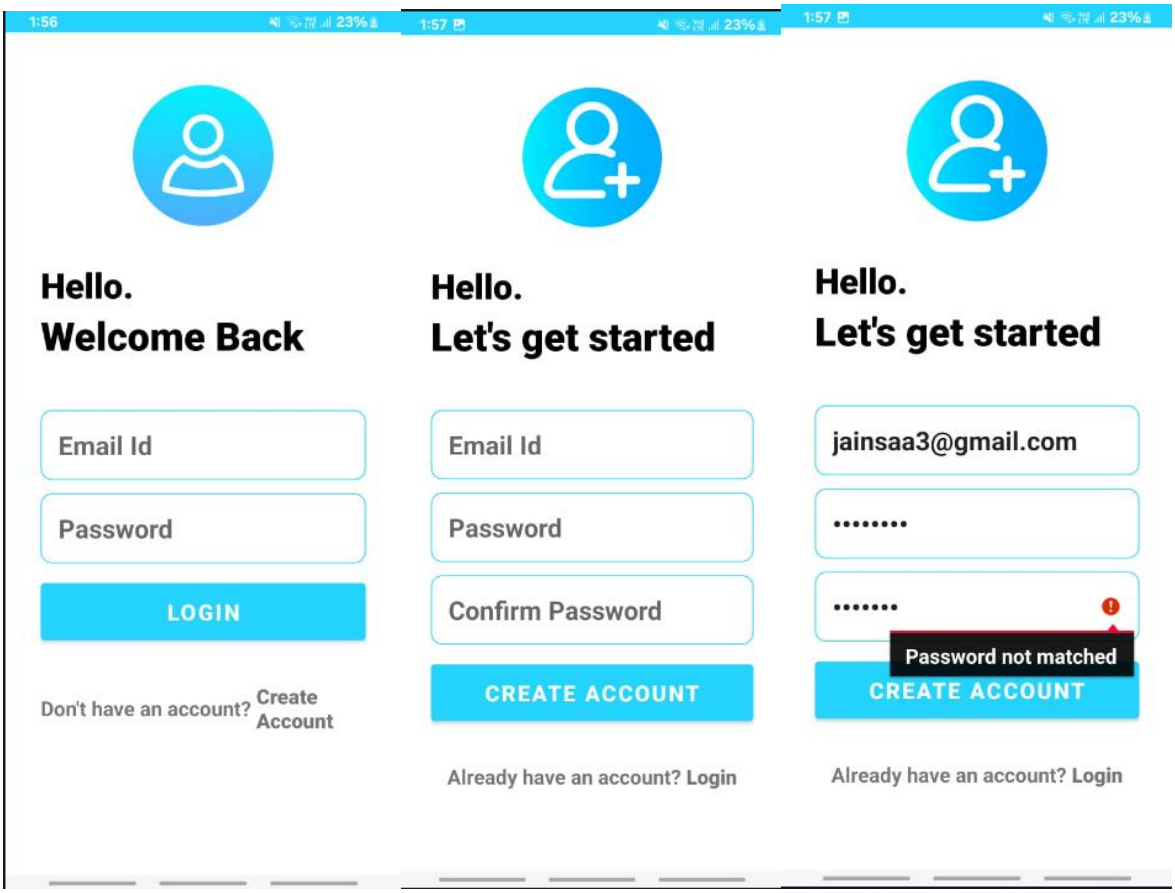
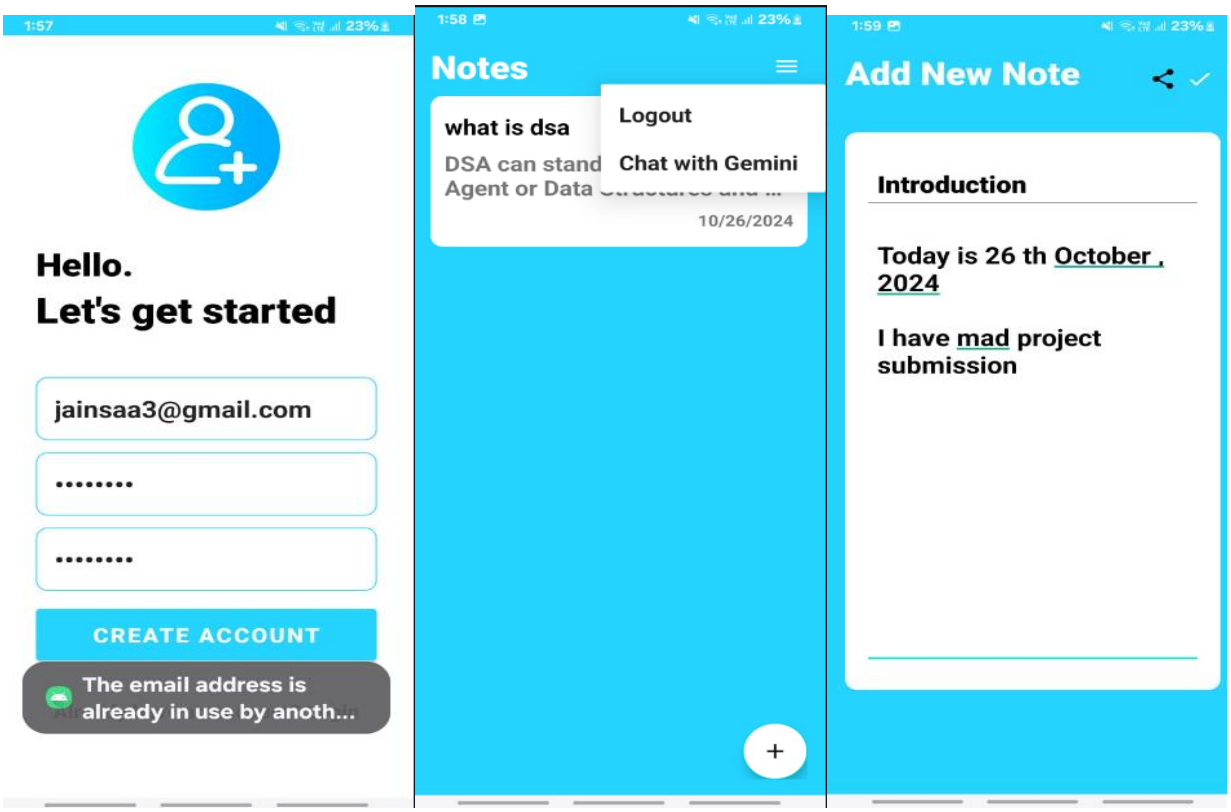


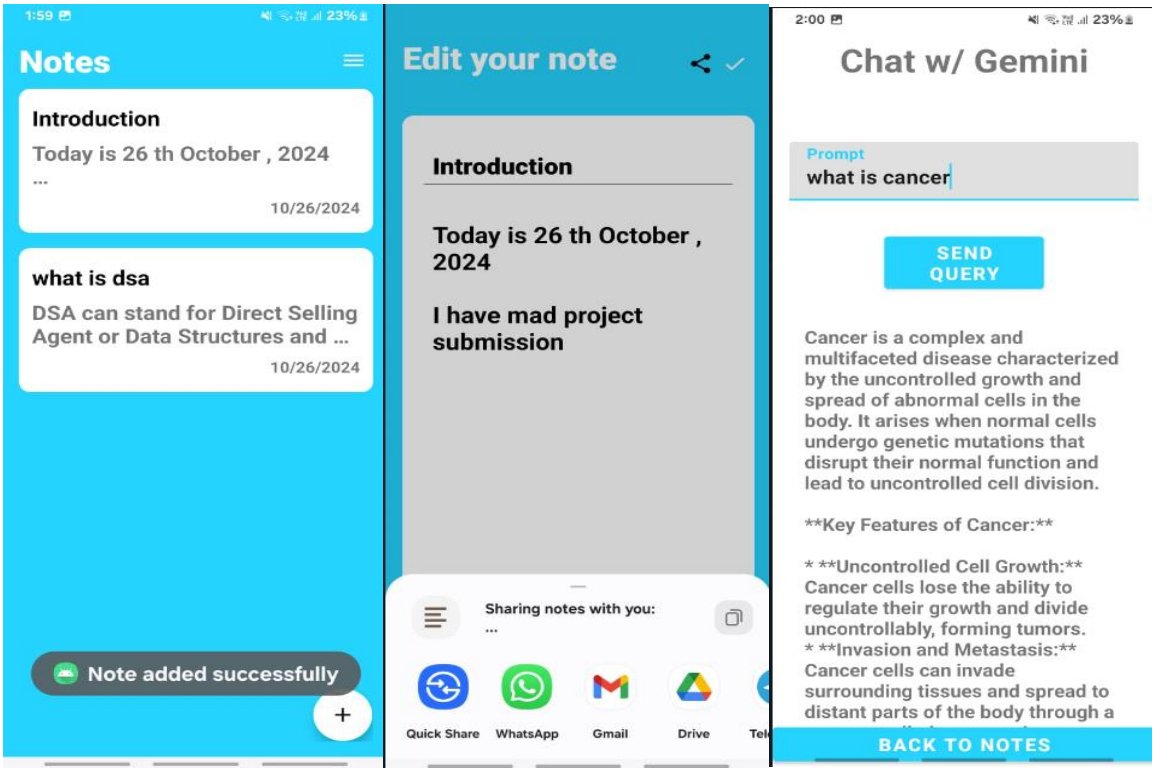
Figure 3: Pictorial Representation of the Architecture

IMPLEMENTATION






Hello.
Let's get started
jainsaa3@gmail.com
CREATE ACCOUNT
The email address is already in use by anoth...
Notes
what is dsa
DSA can stand
Agent or Data
10/26/2024
Logout
Chat with Gemini
Add New Note
Introduction
Today is 26 th October , 2024
I have mad project submission


Notes
Introduction
Today is 26 th October , 2024
...
10/26/2024
what is dsa
DSA can stand for Direct Selling Agent or Data Structures and ...
10/26/2024
Note added successfully
Edit your note
Introduction
Today is 26 th October , 2024
I have mad project submission
Sharing notes with you:
...
Quick Share
WhatsApp
Gmail
Drive
Chat w/ Gemini
Prompt
what is cancer
SEND QUERY
Cancer is a complex and multifaceted disease characterized by the uncontrolled growth and spread of abnormal cells in the body. It arises when normal cells undergo genetic mutations that disrupt their normal function and lead to uncontrolled cell division.
Key Features of Cancer:
* **Uncontrolled Cell Growth:** Cancer cells lose the ability to regulate their growth and divide uncontrollably, forming tumors.
* **Invasion and Metastasis:** Cancer cells can invade surrounding tissues and spread to distant parts of the body through a
BACK TO NOTES

Verify your email for notes-app-saakshi Inbox ×

noreply@notes-app-saakshi.firebaseapp.com
to me

Hello,

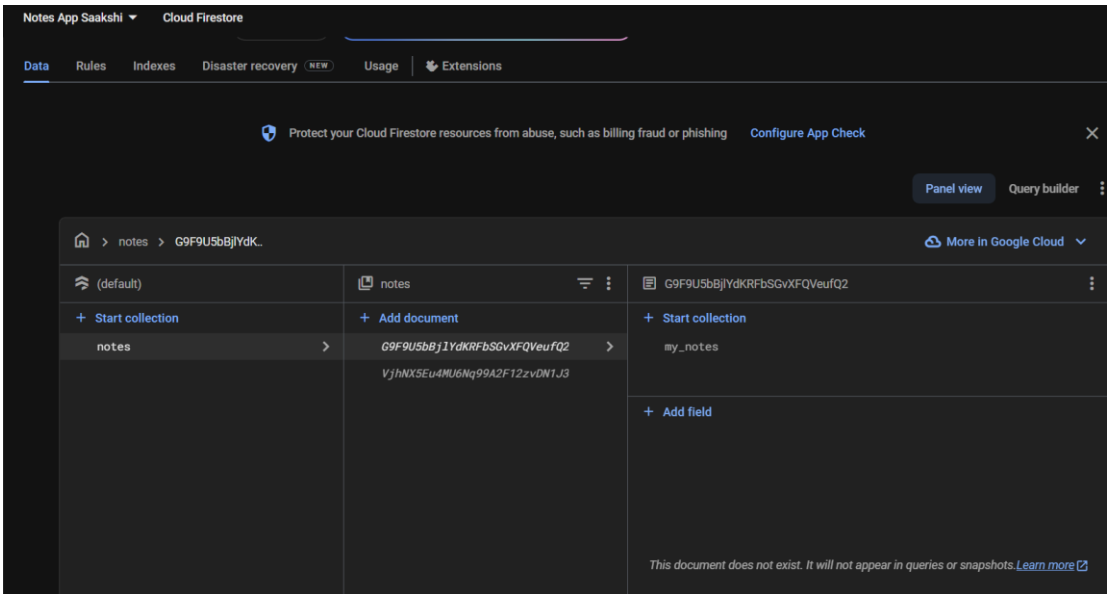
Follow this link to verify your email address.

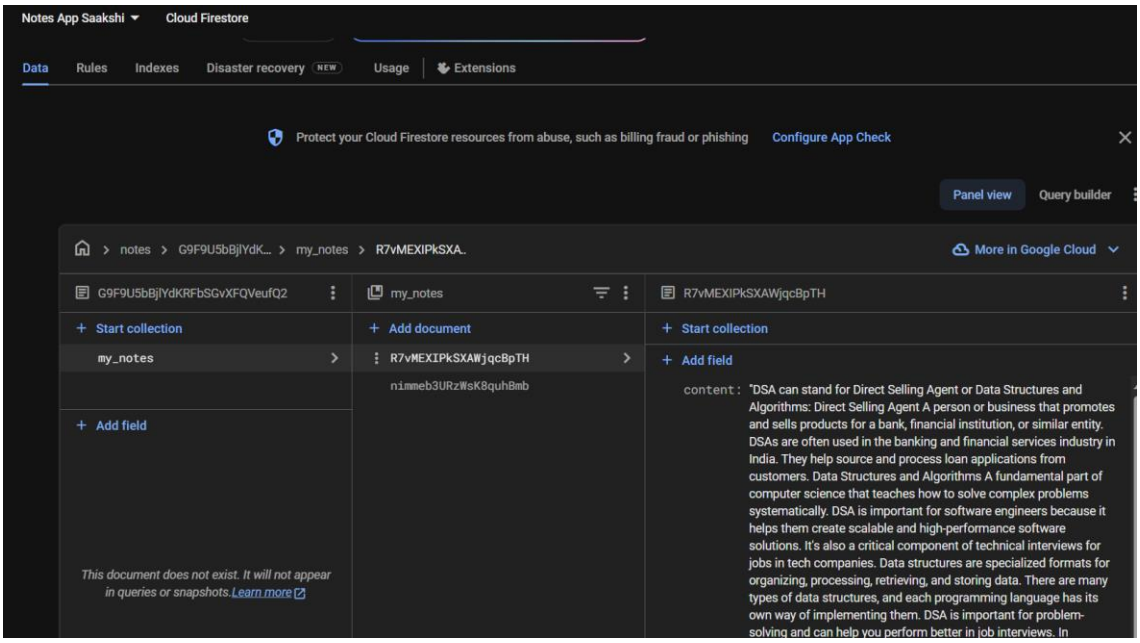
https://notes-app-saakshi.firebaseapp.com/__/auth/action?mode=verifyEmail&oobCode=siA2BkE2F5KE3XiuoGVqVSIquF
AIzaSyDe5XPiwdy7ESciZB7d9XuzwHxx6ut_Vcl&lang=en

If you didn't ask to verify this address, you can ignore this email.

Thanks,

Your notes-app-saakshi team





Conclusion

The Notes App merges the fundamental features of the note-taking apps with advanced features like **real-time syncing, secure access, and AI-driven interactivity.** The app combines Firebase Firestore and Authentication with the users' data; therefore, it ensures they have secure and easy access to their data. Other impressive features include being able to add the Gemini API, which can translate the user's notes into quizzes they can then interact with; hence, increasing learning and retention.

It is especially **suited for students, professionals, and all others who need a dependable, structured note-taking application** that can give much more than the standard applications. It is designed intuitively keeping the usability, security of data, and flexibility in mind and hence becomes an asset for personal as well as educational purposes. **This application is an innovative answer in the current digital knowledge management environment in the service of enhancing productivity and providing support for lifetime learning.**

Summary: The app is perfectly capable of note management while providing elevated smartness through the Gemini API.

Future Improvements:

- Extend AI-driven features to boost query abilities.
- Allow access to notes without the internet.
- Add note tagging and reminders to enhance functionality further.

9% Overall Similarity

Top sources found in the following databases:

- 3% Internet database
- Crossref database
- 9% Submitted Works database
- 0% Publications database
- Crossref Posted Content database

TOP SOURCES

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	**Teaching and Learning with Technology on 2024-09-28** Submitted works	4%
2	**Franconian International School eV on 2024-02-15** Submitted works	2%
3	**Nottingham Trent University on 2024-03-08** Submitted works	2%
4	**Nottingham Trent University on 2013-04-09** Submitted works	1%