



**MIDDLE EAST TECHNICAL UNIVERSITY  
NORTHERN CYPRUS CAMPUS**

**Computer Engineering Program**

FALL 2024-2025

CNG 495 CLOUD COMPUTING

TERM PROJECT PROGRESS  
REPORT 1

“CloudNote”

İLAYDA YAĞMUR KARADAĞ 2315364

## Project Description

CloudNote is a cloud-based note-taking application designed to provide users with a simple and efficient way to create, manage, and share notes across multiple devices. Utilizing Firebase, a powerful cloud platform, the application ensures that user data is securely stored and easily accessible from anywhere. Firebase's real-time database capabilities allow for seamless synchronization of notes across devices, enabling users to capture their thoughts, ideas, and tasks in an organized manner.

**Objectives:** The primary objective of CloudNote is to enhance users' productivity by offering a platform where they can efficiently manage their notes without the fear of data loss. The application aims to provide:

- A user-friendly interface for creating and editing notes.
- Real-time synchronization across devices, ensuring users have access to their notes at all times, powered by Firebase's Firestore.
- Secure storage of notes in the cloud, utilizing modern cloud services to protect user data.
- Easy sharing options, allowing users to collaborate on notes with others.

## Cloud Delivery Models

In the context of the CloudNote project, we are leveraging a combination of cloud delivery models to build and deploy the application. Below is a detailed explanation of how SaaS, PaaS, and IaaS are utilized in this implementation:

### 1. SaaS (Software as a Service):

CloudNote is delivered as a SaaS application. Users can access the note-taking functionality directly through a web interface without worrying about the underlying infrastructure. They can create, edit, and manage their notes via an intuitive user interface, making it easy for them to focus on their tasks.

The application is hosted on the cloud, allowing users to access their notes from any device with internet connectivity. This eliminates the need for local installations and provides seamless updates to the application.

## **2. PaaS (Platform as a Service):**

Firebase serves as the PaaS for this project. It abstracts away the complexities of server management and provides a ready-made platform for developing, deploying, and managing applications.

Using Firebase, we can utilize services such as Firestore for real-time data storage and Firebase Authentication for user management. This allows us to focus on the application development rather than the underlying infrastructure, streamlining the deployment process.

## **3. IaaS (Infrastructure as a Service):**

While we do not directly interact with the underlying infrastructure, IaaS is implicitly part of the overall architecture provided by Firebase. Firebase's services are built on top of infrastructure provided by cloud providers (e.g., Google Cloud Platform).

Although Firebase handles the infrastructure aspects for us, it operates on servers, storage, and networking components managed by cloud providers. This allows us to leverage scalable infrastructure without the need for direct management or interaction.

In summary, CloudNote uses the SaaS model to provide the note-taking application. It uses Firebase as a PaaS, which makes it easier to develop and deploy the app. IaaS is the basic infrastructure that supports everything. This combination helps us develop the app smoothly, allowing us to focus on creating a strong and user-friendly application while taking advantage of cloud technology.

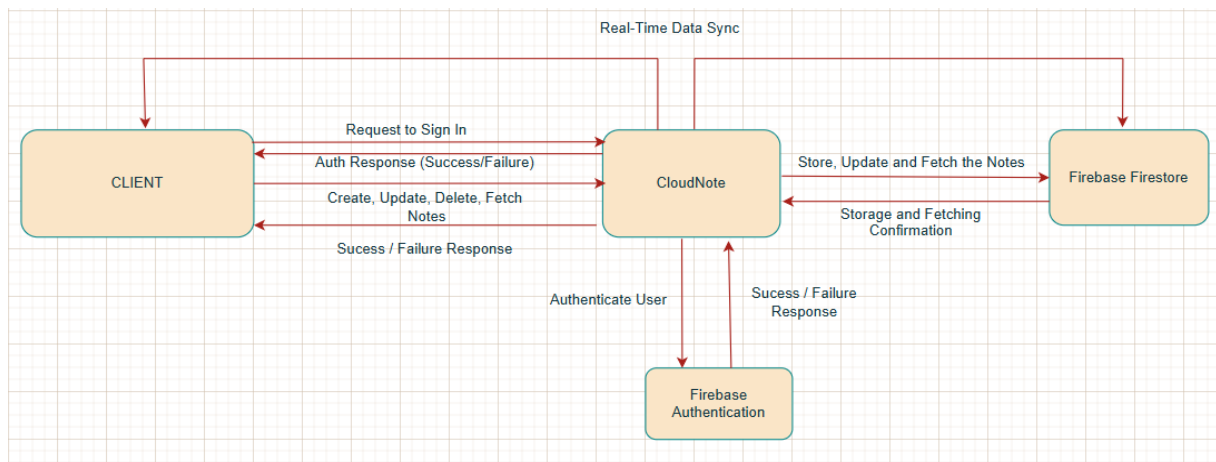
## **Diagrams / Figures:**

### **1) Client-Service-Interaction:**

In the CloudNote application, the Client-Service Interaction diagram illustrates the flow of data between the user's device and the backend services hosted in Firebase. The process starts with the user logging into the app, initiating an authentication request sent from the client device to Firebase Authentication via CloudNote's backend. Once authenticated, users can create, retrieve, and manage notes seamlessly.

For each note-related action, requests from the client are routed through the CloudNote backend to Firebase Firestore, where notes are securely stored. Additionally, To ensure users

have access to up-to-date information, CloudNote utilizes Firebase Firestore's real-time syncing capabilities. Changes made to notes on any device are instantly updated. This feature is implemented as a two-way data sync between the client and Firebase Firestore, mediated through CloudNote, which manages the data flow and processes user requests efficiently. This setup enables users to have a seamless note-taking experience, as their notes are automatically saved and synchronized across devices.



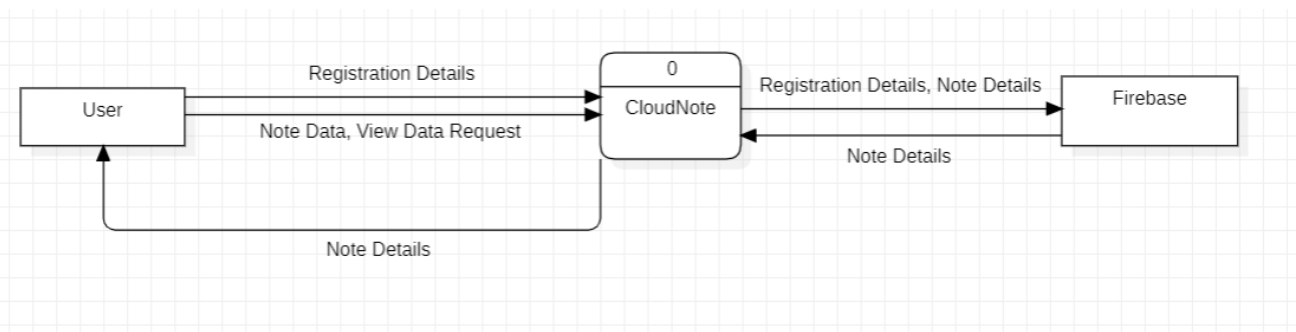
*Figure 1*

*Figure 1 shows the client-service interactions in the system.*

## 2) Data Flow Diagram:

The CloudNote data flow diagram illustrates the flow of information within the system. At Level 0, the CloudNote application interacts with the User and Firebase Firestore to manage note creation, and retrieval. In Level 1, CloudNote is broken down into three main processes registration, note creation and editing, viewing. Each process interacts with Firebase Firestore.

*Level 0:*



*Figure 2*

## Level 1:

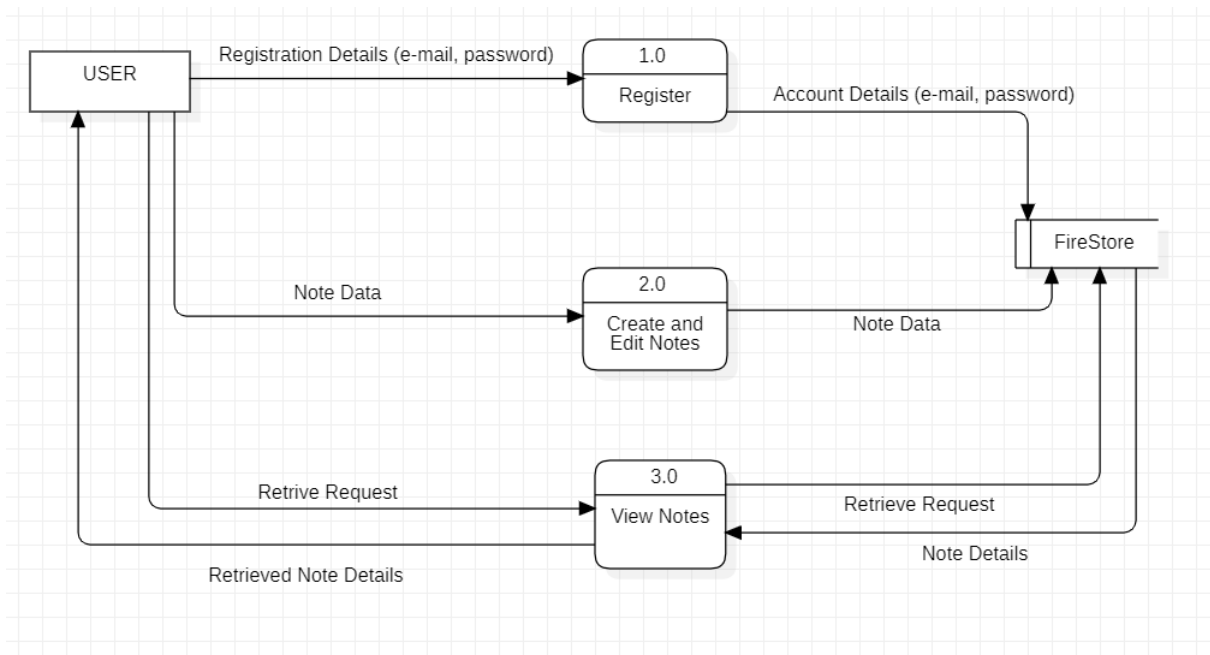


Figure 3

## Data Types in CloudNote

### 1) Binary Data

CloudNote allows users to attach multimedia files to their notes. These files, such as images or documents, are stored as binary data in Firebase Firestore. When a user uploads an attachment, it is converted into a binary format for efficient storage and retrieval.

### 2) Text Data

CloudNote is an app for creating, editing, and saving notes made mostly of text. Users can store their notes as strings in Firestore, which allows them to search through notes easily. The app may also let users use basic text formatting options, like bold or italic, to make their notes look better. Each note has:

- Title: A short name for the note (string).
- Content: The main text of the note where users write their ideas (string). The app may also support some text formatting, like bullet points or simple Markdown.

### 3) Structured Data

Structured data is organized in a specific format, like a table. In CloudNote, notes are stored as structured data in Firestore documents. Each note typically has the following fields:

- Note ID: A unique number or string for each note (string).
- User ID: A number or string that shows who created the note (string).
- Created Timestamp: The date and time when the note was created (timestamp).
- Updated Timestamp: The date and time when the note was last changed (timestamp).
- Attachments: Links or references to any files attached to the note (array of strings).

### **Computation in CloudNote**

- In the CloudNote application, several computing processes work together to help users create, manage, and share their notes effectively. When a user logs in, their email and password are sent to Firebase for verification. Firebase checks if the information matches its records. If it does, a session token is created, allowing the user to securely access their notes.
- Users can easily create new notes or edit existing ones. When they submit a note, the application collects the title, content, and any attached files. Before saving, the app checks that the title and content are not empty. If everything is correct, the note is sent to Firebase Firestore for storage.
- One of the key features of CloudNote is real-time synchronization. This means that when a user updates a note, the changes are instantly reflected.
- When users want to see their notes, the application retrieves them from Firestore. It requests all notes associated with the user's account and formats the data for easy reading and navigation. This ensures that users can quickly find and view their notes without any delay.
- The CloudNote application uses various computing processes to provide a seamless note-taking experience. It supports user authentication, allows for easy note creation and editing, ensures real-time updates, retrieves data quickly.

## **Milestones Achieved**

### **October 28 - November 3**

- Researched and defined project scope for CloudNote.
- Designed the project architecture by drawing Client-Service interaction and Data flow diagrams.
- Set up the development environment with Flask for the backend and started planning the database structure on Firestore.

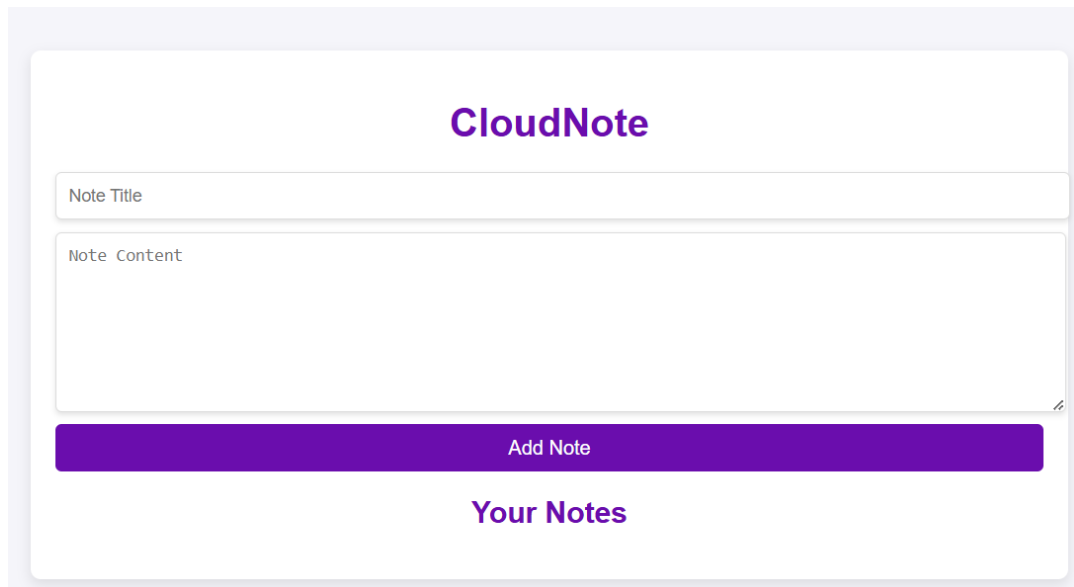
### **November 18 - November 24**

- Developed and tested API endpoints using Python Flask
- Integrated Firebase Firestore as the database for storing notes and Firebase Authentication for user management.

### **November 25 - December 1**

- Built frontend using HTML, CSS, and JavaScript.
- Implemented CRUD features for notes.
- Integrated frontend with backend APIs.
- Tested full application functionality.

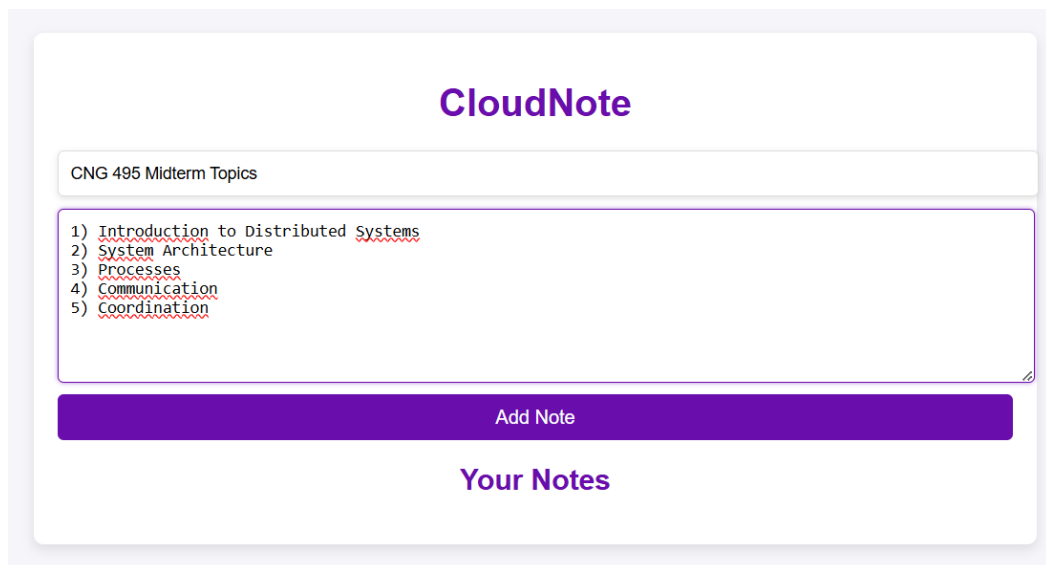
## User Interface



The image shows the CloudNote homepage. At the top, the title "CloudNote" is displayed in a bold, purple font. Below the title, there is a form with two input fields: "Note Title" and "Note Content". The "Note Content" field is a larger text area. Below these fields is a purple button labeled "Add Note". At the bottom of the form, the text "Your Notes" is displayed in a bold, purple font.

Figure 1: CloudNote Homepage

Figure 1 shows where users can quickly add new notes by typing a title and content. All saved notes are shown on the page, making it simple to view and manage them. The design is clean and user-friendly.



The image shows the CloudNote homepage with a note being added. The title "CloudNote" is at the top. The "Note Title" field contains the text "CNG 495 Midterm Topics". The "Note Content" field contains a list of topics: "1) Introduction to Distributed Systems", "2) System Architecture", "3) Processes", "4) Communication", and "5) Coordination". Below the content field is a purple button labeled "Add Note". At the bottom of the form, the text "Your Notes" is displayed in a bold, purple font.

Figure 2: CloudNote Adding a Note

Figure 2 shows the section of the CloudNote homepage where users can input the title and content of their notes. This simple form allows users to create new notes by filling in the title and the content fields, which are then saved and displayed on the page.



# CloudNote

Add Note

## Your Notes

### CNG 495 Midterm Topics

1) Introduction to Distributed Systems 2) System Architecture 3) Processes 4) Communication 5) Coordination

EditDelete

Figure 3:  
Saving Note

Figure 3 shows the process of adding a new note in CloudNote. After entering the note title and content, the user clicks the 'Add Note' button to save the note. Once saved, the new note appears in the list of existing notes below the form.

Home | 2024-2025F... METU

Bu sayfanın mesajı

Enter new title:

Cloud Computing Services

Tamamİptal

Add Note

## Your Notes

### CNG 495 Midterm Topics

1) Introduction to Distributed Systems 2) System Architecture 3) Processes 4) Communication 5) Coordination

EditDelete

### CNG 495

IaaS, PaaS, SaaS

EditDelete

Figure 4: Editing Note

Figure 4 shows the process of editing a note in CloudNote. By clicking the "Edit" button, users can modify either the title or the content of the note.

The screenshot displays the CloudNote application interface. At the top, the title "CloudNote" is centered in a purple font. Below the title, there is a form for adding a new note, consisting of a "Note Title" input field and a larger "Note Content" text area. A purple "Add Note" button is positioned below the form. Underneath the button, the section "Your Notes" is titled. It contains two note cards. The first card is titled "CNG 495 Midterm Topics" and contains the text "1) Introduction to Distributed Systems 2) System Architecture 3) Processes 4) Communication 5) Coordination". It has "Edit" and "Delete" buttons. The second card is titled "Cloud Computing Services" and contains the text "IaaS, PaaS, SaaS". It also has "Edit" and "Delete" buttons.

Figure 5: Edited Version

Figure 5 shows the edited version of selected the note.

This screenshot shows the CloudNote application interface, similar to Figure 5. The "Add Note" form and "Your Notes" section are visible. The "Your Notes" section contains two note cards. The first card, titled "CNG 495 Midterm Topics", is highlighted with a light purple background, indicating it is the selected note. The second card, titled "Cloud Computing Services", is not highlighted. Both cards have "Edit" and "Delete" buttons.

Figure 6: Deleting Notes

Figure 6 shows the process of deleting a note in CloudNote. The note “Cloud Computing Services” is deleted as it can be seen from the figure above.

## Cloud Technologies Used in CloudNote

### 1. Firebase Overview

Firebase is a cloud platform used for real-time databases and user authentication. It simplifies backend services and allows us to store and sync data across devices easily.

### 2. Firebase Firestore (Database)

Firestore is a NoSQL database that stores and syncs data in real-time.

#### Usage in CloudNote:

- Store notes in the Firestore database.
- Real-time syncing: Changes made to notes are reflected instantly across devices. (For the next progress)

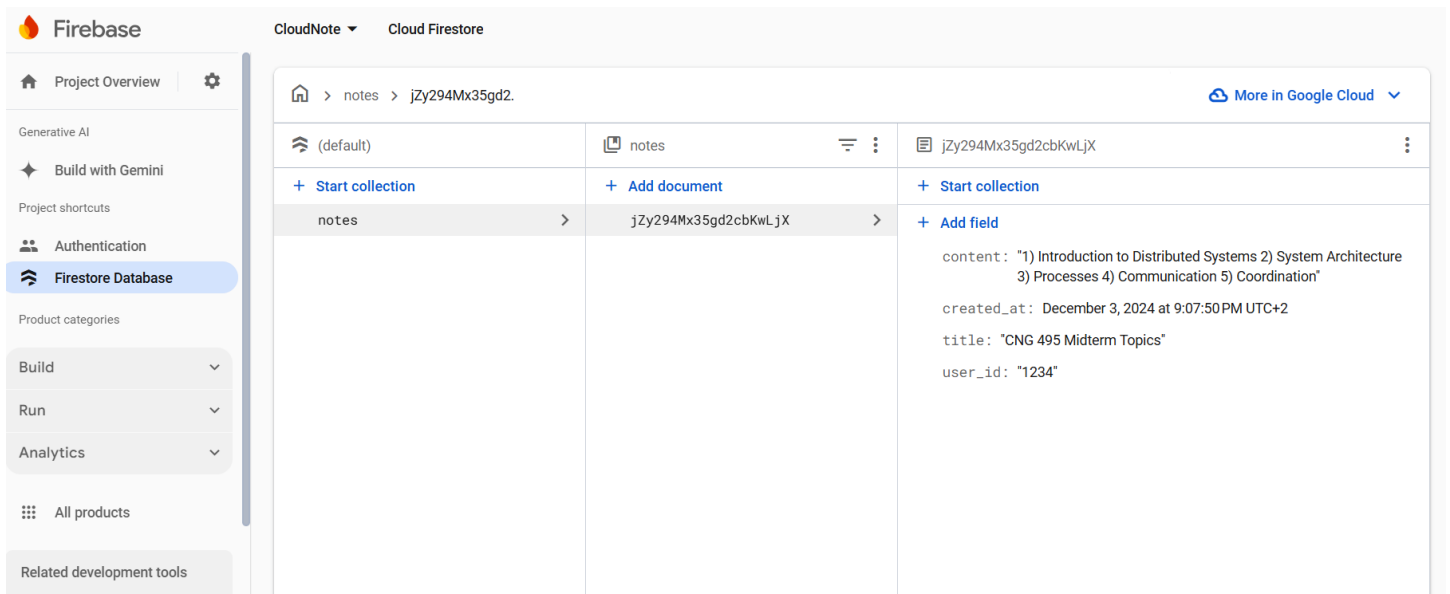


Figure 7: FireStore Database

Figure 7 shows the firestore database which store the note contents.

## Tutorials & Explanations

During the development of CloudNote, by utilizing Firebase and Flask to build the backend and frontend. The project involved setting up Firebase Firestore to store notes.

**Backend with Flask:** Flask server is set up to handle CRUD operations (Create, Read, Update, Delete) for notes. The backend communicates with Firestore to store and retrieve notes. I also handled errors to ensure smooth operation.

**Firestore Integration:** Firestore was used to store notes. Its real-time syncing feature allows updates to be reflected instantly across devices. Setting up Firestore was straightforward with the Firebase Admin SDK.

**Frontend with JavaScript:** The frontend was built using HTML, CSS, and JavaScript. It allows users

to create, view, and edit notes. The frontend communicates with the Flask backend using Fetch API to send and retrieve data.

## **Difficulties**

**CORS Issues:** While testing the frontend and backend, I encountered CORS (Cross-Origin Resource Sharing) issues when making requests from the frontend to the backend. This was resolved by enabling CORS in Flask using the flask-cors library.

## **Milestones Remaining**

### **December 9 - December 15**

- Deploy CloudNote app to Firebase
- Configure and test the production environment to ensure the app is accessible and fully functional.

### **December 16 - December 22**

- Enhance the user interface by improving the layout and design.
- Add user-friendly features like sharing notes.
- Integrate Firebase Authentication to allow users to securely sign in and manage their accounts.

### **December 23 - December 27**

- Conduct final testing to ensure all features work as expected.
- Fix any bugs and optimize the app for final deployment.

## **Remaining Deliverables**

- Frontend Code: All HTML, CSS, and JavaScript files for the user interface.
- Backend Code: Flask server code with all API endpoints and Firebase integration.
- Documentation: A detailed README file with setup instructions and usage guidelines.

Github Link: <https://github.com/iyagmurk/CloudNote>

## References

Firebase Documentation from <https://firebase.google.com/docs?hl=tr>

Firebase Firestore Documentation: <https://firebase.google.com/docs/firestore>