

Quick Recap Deployment Manual

Team - Bros of Balayya

Index

1. Introduction	3
2. System Requirements	4
3. Installation Process	5
4. Backend Setup	6
5. Frontend Setup	8
6. Database Configuration	11
7. Starting the Application	13
8. Deployment to Cloud	16
9. Troubleshooting	17

1. Introduction

Quick Recap is a web-based tool designed to streamline the process of consuming long audio and video content by generating concise, personalized summaries. Quick Recap utilizes advanced technologies such as automatic speech recognition (ASR) with tools like Google Speech-to-Text API, alongside natural language processing (NLP) to create context-aware summaries tailored to the user's specific role and preferences. Users can upload various recordings, and the tool transcribes and summarizes the content in a way that is relevant. It is designed to enhance productivity, improve accessibility, and transform the way users engage with recorded content, making it more efficient and effective for professionals, students, and content consumers alike.

This installation manual is intended for developers, system administrators, and technical teams who want to set up and deploy the Quick Recap application in the development environment. Whether you are setting up the project for local testing or deploying it to the cloud, this guide provides step-by-step instructions to ensure a smooth and successful setup process.

Important Features of the application

1. Upload audio/video files
2. Automatic transcription via speech-to-text
3. Basic user profiles (role, interests)
4. Personalized summaries based on user profiles
5. Core summarization algorithm
6. User-friendly web interface.

This document will guide you through:

- Setting up the development environment.
- Installing backend and frontend dependencies of the application.
- Setting up Firebase for storage and user authentication.
- Deploying the application.
- Troubleshooting common issues that may arise during setup or deployment.

2. System Requirements

For a successful setup and deployment of Quick Recap application your system must meet the below hardware and software requirements.

Hardware requirements:

- Memory (RAM): 8GB
- Processor: Core i5 2.0 GHz
- Storage: 10 GB free Space
- Network: Stable internet connection

Software requirements:

- Python
- Node.js
- NPM
- Git
- Operating System (Windows)
- Firebase CLI
- Web browser

Development Environment will require the following tools:

1. IDE or Code Editor
 - Visual Studio Code
 - Python IDE
2. Frontend Tools
 - Node.js
 - NPM

3. Installation Process

Installation Process gives step by step instructions to set up Quick Recap application for development environment

3.1. Development Environment

The development environment is used for feature development and testing the application locally

1. Clone the git Repository

In the first step start by cloning the git Quick Recap project Repository

```
git clone https://github.com/htmw/2024F-Biased.git  
cd 2024F-Biased
```

2. Frontend Setup

- Navigate to the frontend directory
- Install dependencies using NPM
- Start the deployment server
- The frontend will be available at

3. Backend Setup

4. Firebase Setup

- In firebase console create a new project
- Get the firebase configuration settings
- Update the firebase configuration in the config.js file with your credentials

5. Test that both the backend and frontend are up and running

6. Test the application by uploading an audio or video

4. Backend Setup

The backend of Quick Recap is built using Node.js with Express for handling API requests and interacting with Firebase services. Firebase Authentication is used for user login and registration, and Firestore is used for database management.

4.1. Installing Node.js and Dependencies

1. Verify that Node.js (version 18 or higher) and npm are installed on your machine.

Check the versions using the following commands:

```
node -v  
npm -v
```

If Node.js is not installed, download and install it from the official website.

2. Navigate to the Backend Directory:

Open a terminal and navigate to the backend directory of the project

```
bash  
  
cd backend
```

3. Install Backend Dependencies:

Use npm to install the backend dependencies from package.json

```
bash  
  
npm install
```

Verify Dependency Installation:

After installation, verify that all dependencies are installed correctly

```
bash  
  
npm list
```

This will display all installed packages and their versions.

4.2. Firebase Configuration for Backend

1. Create a Firebase Project:

Go to the Firebase Console -> Create a new project.

Enable Firebase Authentication, Firestore, and Firebase Storage from the Firebase Console.

2. Get Firebase Configuration:

In the Firebase Console, go to "Project Settings" and obtain the Firebase configuration details for your web app.

3. Update Firebase Configuration in Backend:

Replace the Firebase configuration in `backend/firebaseConfig.js` with your project's credentials

4.3. Running the Backend Server

- Run the following command to start the backend server:

```
bash  
  
npm start
```

5. Frontend Setup

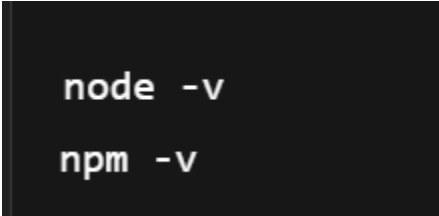
The frontend of Quick Recap is built using React.js. It interacts with Firebase for user authentication and data storage (Firestore and Firebase Storage).

5.1. Installing Node.js and Dependencies

1. Verify Node.js Installation:

Ensure that Node.js (version 18 or higher) and npm are installed on your machine.

Check the versions using the following commands:



```
node -v  
npm -v
```

2. Navigate to the Frontend Directory:

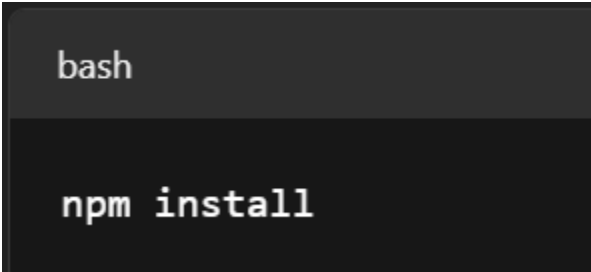
Open a terminal and navigate to the frontend directory:



```
bash  
cd frontend
```

3. Install Frontend Dependencies:

Install the required npm dependencies using the package.json file:



```
bash  
npm install
```


4. Verify Dependency Installation:

After installation, verify that all dependencies are installed:

```
bash  
  
npm list
```

5.2. Configuring Firebase for the Frontend

1. Create a Firebase Project:

Go to the Firebase Console and create a new project if you haven't done so already.

Enable Firebase Authentication, Firestore, and Firebase Storage.

2. Update Firebase Configuration in Frontend:

Replace the placeholder Firebase configuration in frontend/src/firebase.js with your Firebase project details

```
const firebaseConfig = {  
  apiKey: "your-api-key",  
  authDomain: "your-auth-domain",  
  projectId: "your-project-id",  
  storageBucket: "your-storage-bucket",  
  messagingSenderId: "your-messaging-sender-id",  
  appId: "your-app-id"  
};
```

5.3. Running the React Development Server

1. Start the React Development Server:

Run the following command to start the frontend development server:

```
bash  
  
npm start
```

2. Access the Frontend:

Open your browser and navigate to <http://localhost:3000>.

You should see the Quick Recap homepage.

5.4. Testing the Frontend

1. **Login:** Ensure that the login functionality works properly using Firebase Authentication.
2. **Audio/Video Upload:** Verify that the file upload functionality works and that files are stored in Firebase Storage.
3. **Summary Generation:** Test if summaries are displayed after uploading an audio/video file.

6. Database Configuration

Quick Recap uses Firebase Firestore for database management and Firebase Storage for storing media files.

6.1. Setting Up Firebase Firestore

1. Create a Firebase Project:

Log in to the Firebase Console and create a new project if you haven't already.

2. Enable Firestore from the Firestore Database section.

3. Configure Firestore Security Rules:

Set appropriate Firestore security rules.

```
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

6.2. Configuring Firebase Storage

1. Enable Firebase Storage:

In the Firebase Console, navigate to Storage and click "Get Started."

2. Set up a default storage bucket for your project.

3. Configure Storage Security Rules:

Set appropriate rules for Firebase Storage.

```
js

service firebase.storage {
  match /b/{bucket}/o {
    match /{allPaths=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

6.3. Integrating Firebase Configuration in Code

1. Obtain Firebase Configuration:

Go to the Firebase Console > Project Settings > Your Apps to find your Firebase configuration details.

2. Update Configuration in Code:

Update the Firebase configuration in frontend/src/firebase.js.

7. Starting the Application

This section explains how to start the Quick Recap application by running both the backend and front end.

7.1. Starting the Backend API

1. Navigate to the Backend Directory:

```
bash
cd backend
```

2. Run the Backend Server:

```
bash
npm start
```

7.2. Starting the Frontend UI

1. Navigate to the Frontend Directory:

```
bash
cd frontend
```

2. Install Frontend Dependencies:

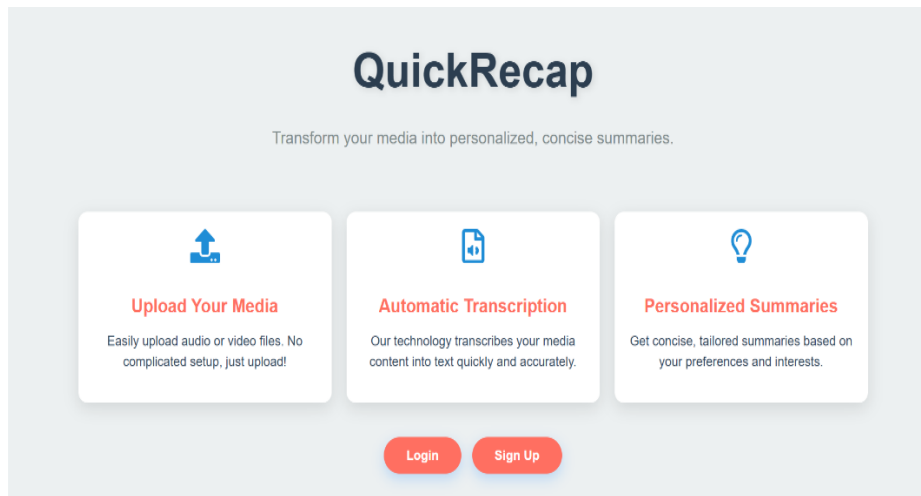
```
bash
npm install
```

3. Start the React Development Server:

```
bash  
npm start
```

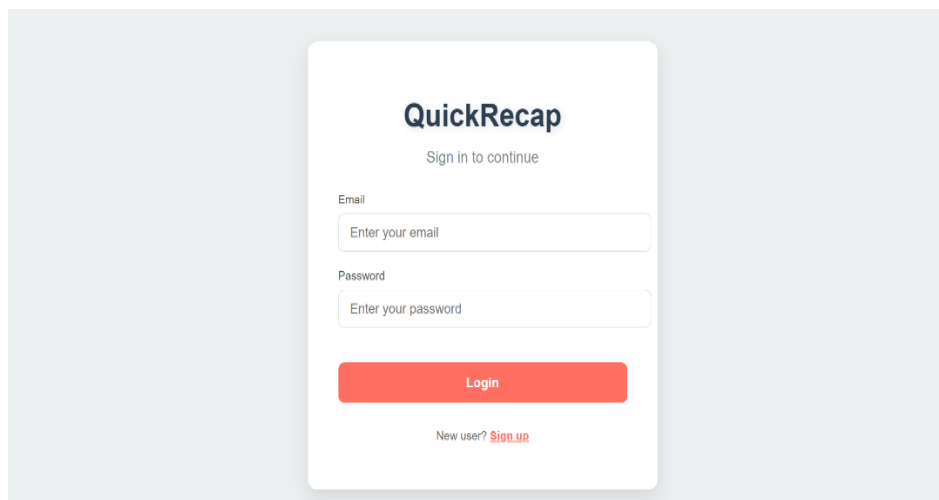
4. Verify Frontend:

Access the frontend at <http://localhost:3000>. Ensure that the frontend loads without errors.

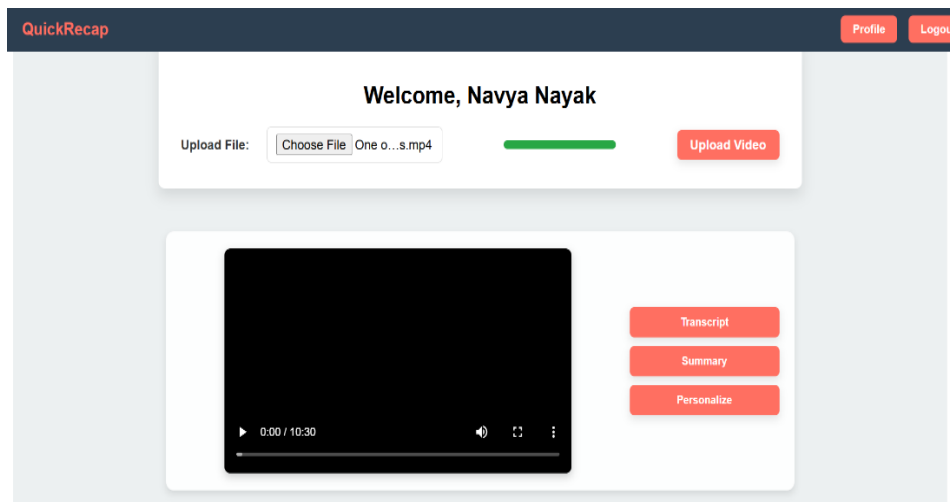


7.3. Testing the Integration

1. Login as a User.



2. Test user authentication using Firebase Authentication.
3. Upload a File.
4. Upload an audio/video file and verify it is stored in Firebase Storage.



5. Verify Summary Generation.
6. Ensure the summary appears after the file is processed.

8. Deployment to Cloud

8.1. Deploy Frontend:

Use Firebase Hosting to deploy the frontend:

```
bash  
  
firebase deploy --only hosting
```

8.2. Hosting the Backend

Deploy Using Google App Engine:

Create an app.yaml for your backend deployment.

8.3. Docker Deployment:

Build and push your Docker image to a repository:

```
bash  
  
docker build -t your-image .  
docker push your-image
```


9. Troubleshooting

1. Firebase Authentication not working

Cause: Misconfigured Firebase credentials

Solution: Double-check your Firebase config in both `frontend/src/firebase.js` and `backend/firebaseConfig.js`. Make sure Firebase Authentication is enabled in the Firebase Console.

2. File Upload Fails

Cause: Firebase Storage not configured or incorrect permissions

Solution: Ensure Firebase Storage is enabled. Go to Firebase Console > Storage > Rules, and set proper permissions (for dev use, you can start with `allow read, write: if true;`)

3. Frontend fails to start

Cause: Missing or outdated Node packages

Solution: Run `npm install` inside the frontend folder. Make sure you're using Node v18 or higher.

4. Backend API doesn't respond

Cause: Backend server not running or incorrect port

Solution: Run the backend using `node index.js` or `npm start` (as applicable). Ensure it's running on the expected port (default: 5000).

5. Summary is not generated after uploading

Cause: Functionality broken in backend or API limit reached

Solution: Check backend console logs for errors. If using external APIs like OpenAI, verify the API key and usage limits.