FreshLens: Deployment Guide

Team: Sierra

Mobile Frontend (React Native with Expo)

1. Target Users:

- **Mobile App Developers:** Developers who are involved in building and testing the mobile application.
- **System Administrators:** Individuals who manage the system settings and maintain the health of the application's backend services.
- **Full Stack Developers:** Developers who handle both frontend and backend parts of the application, ensuring seamless integration and functionality.

2. Platform Requirements:

- **iOS:**
    - **Minimum iOS version:** 12.0 - Ensures compatibility with modern iOS features and optimizations.
    - **Minimum device:** iPhone 7 - Guarantees that the app performs well on devices with adequate processing capabilities.
    - **Required free disk space:** 250 MB - Necessary to accommodate the application size and temporary files during updates.
    - **Supported browser:** Safari - The default browser for testing web content within the app.
- **Android:**
    - **Minimum Android version:** 9.0 - Supports newer Android APIs and user interface features.
    - **Required free disk space:** 250 MB - Similar to iOS, this space is to manage app installation and operational overhead.
    - **Supported browser:** Chrome - Preferred for its widespread use and support for latest web standards.

3. Installation Requirements:

- **Install Node.js:** Node.js is essential for running the JavaScript runtime environment.
- **Install npm:** npm is the package manager for JavaScript, used to install libraries and tools needed.

- **Install Expo CLI:** `npm install -g expo-cli` - Expo CLI is a command line app that is the main interface between a developer and Expo tools.

4. Project Configuration:

- Navigate to the project root directory.

- Create a `.env` file and populate it: This file will store environment-sensitive keys securely, aiding in the configuration of the Firebase and FastAPI services.

  REACT_NATIVE_FIREBASE_API_KEY=AIzaSyDm6AzKIkSgyFqpUVXcf-DcRY0iqBBV8SE

  REACT_NATIVE_FASTAPI_URL=https://localhost:8000

5. Project Setup and Execution:

- Enter the project directory: `cd frontend`
- Install dependencies: `npm install` to install necessary libraries and frameworks.
- Launch the project: `expo start` to start the Expo development server, which helps in live reloading and other development features.

6. Debugging and Validation:

- Utilize Expo's integrated debugging tools and the React Native Debugger for a comprehensive debugging experience, including real-time logs and state inspection.
- Employ `console.log` for internal debugging statements to trace values and application flow.

7. Build and Distribution:

- Execute `expo build:ios` and `expo build:android` to generate platform-specific builds.
- Follow Expo's guidance to deploy the app on the App Store or Google Play, ensuring adherence to submission guidelines and standards.

Backend API (FastAPI)

1. Target Users:

- **Backend Developers:** Responsible for API development and server management.
- **System Administrators:** Oversee the deployment and uptime of the server.

- **Data Scientists:** Utilize APIs for handling data processing tasks.

2. Hardware and Software Requirements:

- **RAM:** Minimum 6GB to ensure smooth operation of the server under load.
- **Disk Space:** Minimum 15GB free to store logs, data, and other necessary files.
- **CPU:** Quad-Core, 2.0 GHz or faster to manage multiple requests efficiently.

3. Core Installations:

- **Python 3.8 or higher:** Modern Python features and improved performance.
- **FastAPI:** A modern, fast (high-performance) web framework for building APIs.
- **Uvicorn:** A lightning-fast ASGI server implementation, running asynchronous Python web code.

4. Server Setup:

- Set up the Python environment within the backend directory for dependency management and isolation:

  python -m venv venv

  source venv/bin/activate -> On Unix/macOS

  venv\\\\Scripts\\\\activate -> On Windows

- Install the necessary packages: `pip install -r requirements.txt` to manage all required Python packages.

5. Launching the Server:

- Ensure the virtual environment is active.
- Launch FastAPI using Uvicorn: `uvicorn main:app --host 0.0.0.0 --port 8000` to start the server, making it accessible on the local network.

Firebase Integration (Auth, Firestore, Cloud Functions, Storage)

1. Firebase Setup:

- Initialize a new project in the Firebase Console to manage application data, authentication, and other backend services seamlessly.

- Enable and configure Firestore for database management, Authentication for user management, and Storage for file storage.

2. Configuration Details:

- Integrate Firebase Admin SDK with your backend to enhance backend capabilities like server-side authentication and data manipulation.
- Define environment variables for Firebase in the `.env` file to secure and manage sensitive data efficiently.

```
const firebaseConfig = {
  apiKey: "AIzaSyDm6AzKIkSgyFqpUVXcf-DcRY0iqBBV8SE",
  authDomain: "freshlens-e2cc4.firebaseapp.com",
  projectId: "freshlens-e2cc4",
  storageBucket: "freshlens-e2cc4.appspot.com",
  messagingSenderId: "298310083659",
  appId: "1:298310083659:web:c5e55777e650b04e4399ce"
};
```

3. Security Measures:

- Implement Firestore security rules to manage database access securely.
- Secure API endpoints using Firebase Authentication to verify and manage user access effectively.

Final Deployment Steps:

- Perform comprehensive testing of each system component individually and in integration to ensure all parts work seamlessly together.
- Observe and optimize the application post-launch to address any issues and enhance performance.
- Maintain and update dependencies to safeguard and enhance the application's performance, ensuring it stays up-to-date with the latest security patches and features.