WEB MANAGEMENT SYSTEM TO INTERACT EFFICIENTLY WITH APIs (FIREBASE)

1. Project Overview

1.1 **PURPOSE**

The primary purpose of this web page is to provide healthcare professionals with an intuitive and efficient tool for storing, retrieving, and visualizing patient medical data from a database. This web interface is a critical component of the VitalCare system, designed to facilitate real-time monitoring and logging of vital medical parameters, such as temperature, pulse rate, weight, and height. By integrating with Firebase, the web page ensures that data is managed seamlessly, offering reliable and secure access to patient records. This functionality aims to enhance patient care by enabling quick data entry, immediate visualization of patient status, and timely alerts for any abnormal readings.

**1.2 SCOPE**

This documentation focuses on the setup, configuration, and usage of the web page's integration + Firebase API.

The scope of this documentation includes:

1.2.1 Development Environment Setup: Instructions for setting up the necessary development tools and dependencies.

1.2.2 Firebase Configuration: Detailed steps to configure Firebase for authentication, real-time database, storage, and hosting.

1.2.3 User Interface Implementation: Guidelines for creating the user interface using HTML, CSS, and JavaScript, including forms for patient registration and medical data entry.

1.2.4 Data Handling: Explanation of how to store, retrieve, and display data using Firebase.

1.2.5 User Authentication: Procedures for implementing secure login and registration using Firebase Authentication.

1.2.6 Data Visualization: Techniques for displaying medical data in interactive table to facilitate easy monitoring.

1.2.7 Alert System: Implementation of a notification system to alert healthcare professionals of any abnormal test results, ensuring prompt medical attention.

1.2.8 Mobile Compatibility: Ensuring the web interface is mobile-friendly to support use on various devices by healthcare professionals.

**1.3 DEVELOPMENT ENVIRONMENT SETUP**

To begin developing the web interface, the following tools and dependencies are required:

1.3.1 Firebase Account: Create a Firebase account and set up a new project to obtain the necessary configuration details.

1.3.2 Firebase Configuration

Configure Firebase to support the necessary backend services:

1.3.2.1 Create a Firebase Project: Go to the [Firebase console](ole.firebase.google.com) and create a new project.

1.3.2.2 Add Configuration Details:

Obtain the Firebase configuration details (apiKey, authDomain, databaseURL, projectId, storageBucket, messagingSenderId, appId) and add them to a configuration file in the project:

|  |
| --- |
| *//firebaseConfig.js*  *const firebaseConfig = {*  *apiKey: "your-api-key",*  *authDomain: "your-auth-domain",*  *databaseURL: "your-database-url",*  *projectId: "your-project-id",*  *storageBucket: "your-storage-bucket",*  *messagingSenderId: "your-messaging-sender-id",*  *appId: "your-app-id"*  *};*  *// Initialize Firebase*  *firebase.initializeApp(firebaseConfig);* |

*Firebase setup - Fig.1*

*You can get the above setup in your firebase console account >> Project Settings*

*Setup Instructions on firebase DOCs*

**1.4 USER INTERFACE IMPLEMENTATION**

An intuitive user interface using HTML, CSS, and JavaScript:

1.4.1 HTML Structure: We develop the basic structure of the web page, including sections for patient registration (LOGIN/SIGNUP), data entry, data display, etc using HTML5 semantics and </tags>.

1.4.2 CSS Styles: Style the web page to make it visually appealing and user-friendly.

1.4.3 JavaScript Functions: Implement functions to handle form submissions, interact with Firebase, and update the UI dynamically.

­Writing a hundred or even a thousand lines of workable code is beautiful right? But what is more beautiful if other programmers can easily read and analyse your code, and also you should be able to view your web design in a browser without have to save and reload the browser every time, which might be a time consuming.

That is why we have to introduce two more tools(plugins) into our project.

* Prettier – for code formatting
* Live Server – for local host

To get these setups really easy in one goal, a general-purpose integrated development environment (IDE) is needed. It is just like a tool belt with a whole lot of tools to make your development process faster and easier.

In the context, I strongly recommend Visual Studio Code (VS code editor) – DOWNLOAD HERE

````````````````````````````````````````````````````````````

<!—WEB PAGES IMAGES HERE-->

````````````````````````````````````````````````````````````

**1.5 USER AUTHENTICATION**

Implement Firebase Authentication to manage user access:

1.5.1 Registration and Login: Develop forms for user registration and login.

1.5.2 Authentication Handling: Use Firebase Authentication methods to handle user sign-in, sign-out, and authentication state changes.

[*Click me*](https://youtu.be/WM178YopjfI?si=cBOPw-N0fS19rrBS) *to see a quick sample demonstration*

**Challenges / Common Issues:**

- Firebase Connection Errors: Ensure the Firebase configuration details in `firebaseConfig.js` are correct.

- Ensure all details in *Fig.1* are tallies with your firebase console account.

- Data Not Storing: Check Firebase rules to ensure write access is granted or downgrade to a lower version to bypass security errors.