**Currency Converter Web App**

**Project Documentation**

**Course Assignment Submission**

Course Name: Full Stack Web Development  
Instructor: Prashanth  
Submitted By: Sainam Akshay Sita Ram

**1. Introduction**

The Currency Converter Web App is a full-stack web application designed to provide real-time currency conversion. Users can sign up, log in, convert currencies, and save their conversion history. This project demonstrates the implementation of user authentication, API integration, and database management using modern web technologies.

**2. Objectives**

* Implement a secure user authentication system
* Fetch real-time exchange rates using an external API
* Store user conversion history in a database
* Provide an interactive and user-friendly UI

**3. Technologies Used**

Frontend:

* HTML, CSS (Bootstrap), JavaScript

Backend:

* Node.js, Express.js

Database:

* Firebase Firestore

Authentication:

* Firebase Auth (Email & Password)

API:

Exchange Rate API (via axios)

**4. System Requirements**

* Software: Node.js (LTS version), Firebase account
* Hardware: Any system with at least 4GB RAM
* Libraries: Express, Firebase Admin SDK, Axios, Bcrypt

**5. Installation & Setup**

1. Clone the Repository

git clone https://github.com/Akshay0920/currency-converter-app

cd currency-converter-app

2. Install Dependencies

npm install

3. Configure Firebase

* Create a Firebase Project in the Firebase Console.
* Enable Firestore Database.
* Replace the Firebase credentials in firebase.js.

4. Start the Server

node server.js

**6. Features Implemented**

* User Authentication
* Secure signup & login system
* Password hashing using bcrypt
* Email duplication prevention
* Real-Time Currency Conversion
* Fetches live exchange rates
* Uses Exchange Rate API
* User Conversion History
* Stores conversion records in Firestore
* Allows users to retrieve past conversions
* Interactive UI
* Glassmorphism design
* Responsive and user-friendly

**7. API Endpoints**

1. User Authentication

Signup

* Endpoint: POST /api/signup
* Body: {email, password, username}
* Response: User created successfully or Email already exists

Login

* Endpoint: POST /api/login
* Body: {email, password}
* Response: Login successful or Invalid credentials

**2. Currency Exchange**

Get Exchange Rates

* Endpoint: GET /api/exchange-rates
* Response: {conversion\_rates}

3. User Conversion History

Save Conversion

* Endpoint: POST /api/save-conversion
* Body: {username, fromCurrency, toCurrency, amount, result}

Fetch Recent Conversions

* Endpoint: GET /api/recent-conversions?username=xyz
* Response: {conversions: []}

**8. Testing & Debugging**

* Tested API endpoints using Postman
* Verified authentication and database operations
* Checked UI responsiveness across devices

**9. Challenges Faced & Solutions**

| Challenge | Solution |
| --- | --- |
| Handling API failures | Implemented proper error handling |
| Secure password storage | Used bcrypt for password hashing |
| UI responsiveness | Used Bootstrap for better layout |

**10. Future Enhancements**

* Implement a user dashboard with charts
* Add OTP-based authentication
* Improve UI/UX with animations & transitions

**11. Conclusion**

The Currency Converter Web App successfully demonstrates the integration of backend and frontend technologies to build a real-time web application. The project follows best coding practices and ensures data security, API efficiency, and user-friendly design.

**12. References**

1. Firebase Documentation: <https://firebase.google.com/docs/>
2. Node.js Express Guide: <https://expressjs.com/>
3. Exchange Rate API: <https://www.exchangerate-api.com/>