**Design Document for Currency Exchange System**

**1. Introduction**

**Purpose**

This document provides a detailed architectural overview of the Currency Exchange System. It outlines the frontend and backend components, key design choices, system architecture, and technologies used. The system supports exchange rate retrieval for general users, while authenticated users (such as admins) can register, log in, and modify exchange rates for historical analysis.

**Scope**

**The system allows users to:**

* Fetch live exchange rates for USD, AUD, CAD, and GBP against LKR
* Store and retrieve exchange rate history for the last 90 days (manual)
* Perform manual entry of exchange rates (authenticated User / Admin)
* Register and log in using a password less authentication system (OTP based authentication using email)
* interact with the backend API
* View audit trails to track user activity

**2. System Overview**

**Components**

* **Frontend**: Developed using Vue.js, handling user interface and interactions.
* **Backend**: Built with Laravel 12, managing API requests, authentication, data processing, and database interactions.
* **Database**: MySQL is used to store user information, exchange rate records, and audit logs.

**3. Architectural Strategy**

**Design Principles**

* **Modularity:** The frontend and backend are developed as separate modules to ensure scalability.
* **Security:** Secure data handling, validated input, authentication, and proper authorization.

**Technology Stack**

**Frontend**

* **Framework**: Vue 3 with Vite
* **Routing**: Vue Router
* **State Management**: vuex
* **HTTP Client**: Axios
* **Styling**: CSS

**Backend**

* **Framework:** Laravel 12
* **Environment Management: .**env files
* **Authentication:** Laravel Sanctum (password less login )
* **Database:** MySQL
* **ORM:** Eloquent

**Tools**

* **Version Control:** Git (Bitbucket)
* **Package Managers:** Composer (PHP), npm (JavaScript)
* **CI/CD**: Bitbucket Pipelines (frontend /netlify)

**4. Database Design**

**Tables**

**users**

* id (Primary Key)
* email (Unique, Required)
* created\_at, updated\_at

**exchange\_rates**

* id (Primary Key)
* currency\_code (ENUM: 'USD', 'AUD', 'CAD', 'GBP')
* rate (Decimal 10,4)
* date (Date)
* created\_at, updated\_at

**audit\_logs (on process)**

* id (Primary Key)
* user\_id (Foreign Key -> users.id)
* action (Text)
* timestamp (Datetime)

**5. API Design**

**Endpoints**

**Authentication (OTP password less)**

* **POST** /api/auth/verify-otp - OTP verification to login

**Exchange Rates (Target LKR only)**

* **GET** /api/exchange-rate/current/{baseCurrency?}?date={date} - Fetches exchange rates with currency and date
* **GET** /api/exchange-rate/current - Fetches today’s exchange rates (default base on USD )
* **GET** / api/exchange-rate/{currency}/last-seven-days - Fetches last 7 days' rates (base on currency) and weekly average
* **POST** /api/exchange-rate/manual - Manually enter an exchange rate (Register user or admin only)

**User Management (progress on magic link)**

* **POST** /api/auth/register - Registers a new user
* **POST** /api/auth/login – user login with send the OTP to email
* **POST** /api/logout - Logs out the user
* **GET** /api/user - Get the authenticated user

**Audit Logs**

* **GET** /api/audit-logs - Fetch user activity logs (admin/registered user only)

**6. Frontend Design**

**Components**

* **Login.vue -** Handles login (on process for magic link auth)
* **ExchangeRates.vue -** Displays current and historical exchange rates
* **ManualEntry.vue** - Allows manual input of exchange rates (register user only but on progress some auth part)
* **AuditLogs.vue -** Displays user activity logs(partial)

**Features**

* Users can change the selected date to fetch historical exchange rates.
* Admins can manually enter and modify exchange rates for the last 90.
* Users can view exchange rate trends with a simple table representation (with changes).

**7. Security Considerations**

* **CORS Configuration:** Configured to accept requests from the Vue frontend (cors.php).
* **Authentication:** Laravel Sanctum for token-based authentication.
* **Data Validation:** Input validation to prevent SQL injection and XSS attacks.
* **Password less :** OTP send to email and verify the login

**8. Challenges and Solutions**

* **Issue:** Managing real-time exchange rates efficiently.
  + **Solution:** Implement caching to store frequently accessed exchange.
* **Issue: Password less login security concerns. (gmail configuration / )**
  + **Solution:** Set expiration time for magic links and restrict reuse (on progress).

**9. Conclusion**

This document outlines the structure and implementation details of the Currency Exchange System. The system is designed with scalability, security, and usability in mind, providing a seamless experience for users while ensuring accurate currency exchange rate management.

**10. References**

* **Laravel Documentation:** <https://laravel.com/docs/12.x>
* **Vue.js Guide:** <https://vuejs.org/guide/introduction.html>
* **Currency API:** <https://www.exchangeratesapi.io/>
* **Chatgpt –** optimize /QA