**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.

**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
* Perform manual entry of exchange rates
* Register and log in using a passwordless authentication system (minimal level for poc on progress with magic link)
* 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.
* **Performance Optimization:** Implementing caching mechanisms and optimized database queries.

**Technology Stack**

**Frontend**

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

**Backend**

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

**Tools**

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

**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 (on process)**

* **POST** /api/auth/magic-link - Sends a magic link for login
* **POST** /api/auth/verify-link - Verifies the magic link

**Exchange Rates (Target LKR only)**

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

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

* **POST** /api/register - Registers a new user
* **POST** /api/login – normal a new user (normal user login)
* **POST** /api/logout - Logs out the user

**Audit Logs**

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

**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(on progress API)

**Features**

* Users can change the selected date to fetch historical exchange rates (implemented).
* Admins can manually enter and modify exchange rates for the last 90 days (auth on process).
* 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 (on process for better).

**8. Challenges and Solutions**

* **Issue:** Managing real-time exchange rates efficiently.
  + **Solution:** Implement caching to store frequently accessed exchange rates (not implement yet).
* **Issue: Passwordless login security concerns. (on process with magic link)**
  + **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