Software Requirements Specification (SRS) Document

**1. Project Information**

* **Project Name:** Spendwise - Your Personal Expense Tracker
* **Project Owner:** Haseeb Ahmed, Fatima Naeem, Fatima Shahid
* **Version:** 1.0
* **Date:** 9/12/2024

**2. Introduction**

**Purpose:**  
The purpose of this document is to outline the functional and non-functional requirements of Spendwise. The document serves as a guide for the development team and stakeholders, ensuring clarity on objectives, scope, and features.

**Scope:**  
Spendwise is designed to empower users to manage their finances effectively. Key features include expense tracking, budgeting, and financial reporting with an intuitive, secure, and user-friendly interface. Stakeholders include individuals, small businesses, and freelancers.

**3. Functional Requirements**

1. **Expense Tracking:**
   * Users can log daily, weekly, and monthly expenses.
   * Expenses can be categorized with notes for clarity.
2. **Budgeting and Alerts:**
   * Users can set spending limits and receive notifications for overspending.
3. **Report Generation:**
   * Users can view financial summaries with dynamic graphical representations like pie charts and bar graphs.
4. **User Profiles and Synchronization:**
   * Secure user profiles with cross-device synchronization.
5. **Expense Categorization**
   * Users can create custom categories (e.g., "Groceries," "Entertainment") to organize their expenses.
   * Ability to edit or delete categories as needed.
6. **Multi-Currency Support:**
   * Users can log expenses in different currencies, with automatic conversion to the base currency for consolidated reporting.
7. **Search and Filter:**
   * Advanced search functionality to locate specific transactions by date, amount, or category.
8. **Third-Party Integration::**
   * Support for integration with bank accounts to automatically fetch transaction data.
   * Compatibility with financial apps like PayPal or Stripe for seamless data syncing

**4. Non-Functional Requirements**

1. **Performance:**
   * Seamless performance with minimal latency.
2. **Security:**
   * AES encryption for sensitive data.
   * Anomaly detection for unauthorized access attempts.
3. **User Interface:**
   * Simple, accessible, and multi-language support.
4. **Scalability:**
   * Designed to accommodate a growing user base and future features.
5. **Availability:**
   * The system must have an uptime of 99.9% or higher.
   * Automatic failover mechanisms to ensure continuous service during server downtime.
6. **Compatibility:**
   * Compatible with major browsers (e.g., Chrome, Safari, Firefox, Edge).
   * Supports mobile and desktop operating systems, including Android, iOS, Windows, and macOS.
7. **Data Backup and Recovery:**
   * Daily backups of all user data.
   * Ability to restore data within 24 hours in case of a critical failure.
8. **Energy Efficiency:**
   * Optimized for low energy consumption, especially for mobile devices, to prolong battery life.

**5. User Interface Design**

* **Design Principles:**
  + **Simplicity:** Clear navigation and uncluttered interfaces.
  + **Consistency:** Uniform styles and branding.
  + **Accessibility:** High contrast and screen reader support.
* **Example:**  
  Dashboard showcasing metrics like budget status and recent transactions.

**6. System Architecture**

* **Layered Architecture:**
  + **Frontend (UI):** Manages interactions and displays.
  + **Backend (Logic):** Handles business logic and database interactions.
  + **Database:** Stores user data securely.

**7. Data Requirements**

* Store transaction details, user profiles, and budgeting information.
* Ensure redundancy and scalability for a growing database.

**8. Security Requirements**

1. **Encryption:** AES algorithms for secure storage.
2. **Breach Protocols:** Alerts and anomaly detection for unauthorized access.
3. **Transparency:** Clear privacy policies displayed to users.

**9. Performance Requirements**

1. **Load Handling:** Optimized for concurrent user interactions.
2. **Speed:** Response time under 5 seconds for major operations.

**10. Testing Requirements**

* **Unit Testing:** Validates individual components like expense entry.
* **Integration Testing:** Ensures smooth interaction between modules.
* **System Testing:** Assesses overall functionality in real-world scenarios.
* **Acceptance Testing:** Confirms alignment with user requirements.

**11. Project Timeline**

| **Task** | **Start Date** | **End Date** |
| --- | --- | --- |
| **Requirement Gathering** | **1/10/2024** | **5/10/2024** |
| **Design and Prototyping** | **6/10/2024** | **15/10/2024** |
| **Development** | **16/11/2024** | **20/11/2024** |
| **Testing** | **21/11/2024** | **30/11/2024** |
| **Deployment** | **1/12/2024** | **9/12/2024** |