**Smart Personal Budget Planner: Requirements Document**

**1. Introduction**

**1.1 Project Overview:** The Smart Personal Budget Planner is a mobile application designed to assist individuals in effectively managing their personal finances. It aims to empower users to track expenses, create and adhere to budgets, and achieve their savings goals by providing a user-friendly and intuitive platform.

**1.2 Project Goals:**

* Enable users to easily track their daily expenses.
* Assist users in creating and managing budgets effectively.
* Help users achieve their savings goals through personalized insights and recommendations.
* Provide users with a clear understanding of their spending habits.

**2. Requirements Analysis**

**2.1 User Stories/Scenarios:**

* As a user, I want to easily track my daily expenses so I can understand where my money is going.
* As a user, I want to create a budget that aligns with my income and financial goals.
* As a user, I want to receive alerts when I am approaching or exceeding my budget limits.
* As a user, I want to generate reports that visualize my spending patterns.
* As a user, I want to share expenses with friends and family members.
* As a user, I want to receive personalized financial advice and recommendations.

**3. Functional Requirements**

**3.1 User Management**

* **FR1:** Users shall be able to register and log in using a valid email address and password.
* **FR2:** Password reset functionality shall be available for forgotten credentials.

**3.2 Budget Management**

* **FR3:** Users shall be able to set monthly budgets based on their income and savings goals.
* **FR4:** The system shall track remaining funds in real-time by comparing expenses against the budget.
* **FR5:** The system shall send alerts to users when they are approaching or exceeding budget thresholds.

**3.3 Expense Tracking**

* **FR6:** Users shall be able to manually add expenses with date, amount, and category.
* **FR7:** The system shall provide predefined expense categories (e.g., Food, Rent, Transport) and allow users to create custom categories.
* **FR8:** The system shall calculate daily spending limits dynamically based on the remaining budget and savings goals.

**3.4 Group Expense Sharing**

* **FR9:** Users shall be able to create groups and invite other users to join.
* **FR10:** Users shall be able to record shared expenses and assign contributions to group members.
* **FR11:** The system shall generate summaries indicating balances owed between group members.

**3.5 Visual Reports**

* **FR12:** The system shall generate pie charts to visualize the distribution of expenses across different categories.
* **FR13:** The system shall generate bar charts to compare expenses over time (e.g., monthly, weekly).
* **FR14:** The system shall provide monthly financial summaries in graphical and textual formats.

**3.6 Recommendations and Advice**

* **FR15:** The system shall display personalized advisory messages based on spending habits (e.g., "Reduce dining out to save more").
* **FR16:** The system shall suggest daily spending limits to meet monthly savings goals.

**4. Non-Functional Requirements**

* **NFR1:** **Usability:** The application shall have a user-friendly interface with intuitive navigation and clear instructions.
* **NFR2:** **Scalability:** The system shall be able to handle a growing number of users and data without significant performance degradation.
* **NFR3:** **Security:** User data shall be stored securely using encryption and appropriate security measures.
* **NFR4:** **Performance:** The system shall respond quickly to user interactions and generate reports with minimal latency.
* **NFR5:** **Accessibility:** The application shall be accessible to users with disabilities (e.g., screen reader compatibility).
* **NFR6:** **Cross-Platform Compatibility:** The application shall be compatible with both mobile and desktop platforms.

**4. Use Cases**

**4.1 Use Case ID and Name:** UC-01 - Login

**4.1.1 Created By:** [Your Name/Team]

**4.1.2 Date Created:** [Date]

**4.1.3 Primary Actor:** Registered User

**4.1.4 Secondary Actors:** None

**4.1.5 Description:** This use case describes the process of a registered user logging into the Smart Personal Budget Planner application.

**4.1.6 Trigger:** User initiates the login process.

**4.1.7 Preconditions:**

* The user has an existing account in the system.
* The user has access to the internet and a device capable of accessing the application.
* The application is available and functional.

**4.1.8 Postconditions:**

* The user is successfully logged in to the application.
* The application displays the user's dashboard or home screen.

**4.1.9 Normal Flow:**

1. The User navigates to the login screen.
2. The User enters their registered email address.
3. The User enters their password.
4. The User clicks the "Login" button.
5. The system verifies the provided credentials.
6. The system authenticates the user and grants access to the application.
7. The system redirects the user to the application's main screen or dashboard.

**4.1.10 Alternative Flows:**

* **4.1.10.1 Invalid Credentials:**
  1. The User enters incorrect email address or password.
  2. The system displays an error message (e.g., "Invalid username or password").
  3. The User is prompted to re-enter their credentials.
* **4.1.10.2 Multiple Login Attempts:**
  1. The User enters incorrect credentials multiple times.
  2. The system may temporarily lock the account or display a security message.
* **4.1.10.3 System Error:**
  1. The system encounters an unexpected error during the login process.
  2. The system displays an error message to the user.
  3. The system logs the error for troubleshooting.

**4.1.11 Exceptions:** None

**4.1.12 Priority:** High

**4.1.13 Frequency of Use:** Very High (expected to occur each time a user wants to access the application)

**4.1.14 Business Rules:** None

**4.1.15 Other Information:** None

**4.1.16 Assumptions:**

* The user has a stable internet connection.
* The user has the necessary software and hardware to access the application.

**4.1 Use Case ID and Name:** UC-01 - Register New User

**4.1.1 Created By:** [Your Name/Team]

**4.1.2 Date Created:** [Date]

**4.1.3 Primary Actor:** New User

**4.1.4 Secondary Actors:** None

**4.1.5 Description:** This use case describes the process of a new user creating an account within the Smart Personal Budget Planner application.

**4.1.6 Trigger:** New User initiates the account creation process.

**4.1.7 Preconditions:**

* The user has access to the internet and a device capable of accessing the application.
* The application is available and functional.

**4.1.8 Postconditions:**

* A new user account is created in the system database.
* The user receives a confirmation message or email.
* The user is able to log in to the application using their new credentials.

**4.1.9 Normal Flow:**

1. The New User navigates to the "Create Account" screen.
2. The New User enters their desired email address.
3. The New User enters and confirms their password.
4. The New User clicks the "Create Account" button.
5. The system validates the entered information (e.g., checks for valid email format, password strength).
6. The system creates a new user account in the database.
7. The system sends a confirmation message/email to the user's provided email address.
8. The system displays a success message and redirects the user to the login page.

**4.1.10 Alternative Flows:**

* **4.1.10.1 Invalid Input:**
  1. The New User enters invalid information (e.g., empty fields, invalid email format, weak password).
  2. The system displays an error message indicating the invalid input.
  3. The New User corrects the information and resubmits the form.
* **4.1.10.2 Username Already Exists:**
  1. The New User enters an email address that is already in use.
  2. The system displays an error message indicating that the email address is already registered.
  3. The New User enters a different email address and resubmits the form.
* **4.1.10.3 System Error:**
  1. The system encounters an unexpected error during account creation.
  2. The system displays an error message to the user.
  3. The system logs the error for troubleshooting.

**4.1.11 Exceptions:** None

**4.1.12 Priority:** High

**4.1.13 Frequency of Use:** High (expected to occur frequently as new users join the system)

**4.1.14 Business Rules:**

* Password must meet minimum complexity requirements (e.g., length, character types).

**4.1.15 Other Information:** None

**4.1.16 Assumptions:**

* The user has a stable internet connection.
* The user has the necessary software and hardware to access the application.

**4.2 Use Case ID and Name:** UC-02 - Set a Monthly Budget

**4.2.1 Created By:** [Your Name/Team]

**4.2.2 Date Created:** [Date]

**4.2.3 Primary Actor:** User

**4.2.4 Secondary Actors:** None

**4.2.5 Description:** This use case describes the process of a user setting a monthly budget within the Smart Personal Budget Planner application.

**4.2.6 Trigger:** User initiates the budget setting process.

**4.2.7 Preconditions:**

* The User is logged in to the application.

**4.2.8 Postconditions:**

* A new monthly budget is created or updated for the user.
* The system calculates and displays the available budget for expenses.

**4.2.9 Normal Flow:**

1. The User navigates to the "Budget" section of the application.
2. The User enters their monthly income.
3. The User (optionally) sets a savings goal.
4. The system calculates the available budget for expenses.
5. The system displays the calculated budget and suggested daily spending limits.
6. The User reviews and confirms the budget.
7. The system saves the budget settings for the user.

**4.2.10 Alternative Flows:**

* **4.2.10.1 Missing Input:**
  1. The User fails to enter their monthly income.
  2. The system displays an error message prompting the user to enter their income.
  3. The User enters their income and proceeds to step 4.
* **4.2.10.2 Invalid Input:**
  1. The User enters invalid input (e.g., negative income).
  2. The system displays an error message indicating invalid input.
  3. The User corrects the input and proceeds to step 4.

**4.2.11 Exceptions:** None

**4.2.12 Priority:** High

**4.2.13 Frequency of Use:** High (expected to occur frequently, potentially at the beginning of each month)

**4.2.14 Business Rules:** None

**4.2.15 Other Information:** None

**4.2.16 Assumptions:** None

**4.3 Use Case ID and Name:** UC-03 - Add an Expense

**4.3.1 Created By:** [Your Name/Team]

**4.3.2 Date Created:** [Date]

**4.3.3 Primary Actor:** User

**4.3.4 Secondary Actors:** None

**4.3.5 Description:** This use case describes the process of a user recording an expense within the Smart Personal Budget Planner application.

**4.3.6 Trigger:** User initiates the expense recording process.

**4.3.7 Preconditions:**

* The User is logged in to the application.
* A budget has been set for the current month.

**4.3.8 Postconditions:**

* A new expense record is successfully added to the user's expense history.
* The user's available budget is updated to reflect the new expense.

**4.3.9 Normal Flow:**

1. The User navigates to the "Add Expense" screen.
2. The User selects the expense date.
3. The User enters the expense amount.
4. The User selects or enters the expense category.
5. The User (optionally) adds a description for the expense.
6. The User clicks the "Save" button.
7. The system validates the entered data.
8. The system saves the expense record to the database.
9. The system updates the user's available budget.
10. The system displays a success message and updates the expense list.

**4.3.10 Alternative Flows:**

* **4.3.10.1 Invalid Input:**
  1. The User enters invalid data (e.g., negative amount, invalid date).
  2. The system displays an error message indicating invalid input.
  3. The User corrects the input and resubmits the expense.
* **4.3.10.2 No Internet Connection:**
  1. The User attempts to add an expense while offline.
  2. The system stores the expense locally.
  3. Once an internet connection is restored, the application synchronizes the local data with the server.

**4.3.11 Exceptions:** None

**4.3.12 Priority:** High

**4.3.13 Frequency of Use:** Very High (expected to occur multiple times per day for most users)

**4.3.14 Business Rules:** None

**4.3.15 Other Information:** None

**4.3.16 Assumptions:** None

**Scenario 1: Successful Registration**

1. **User:** Navigates to the "Create Account" page.
2. **User:** Enters a valid email address (e.g., "example@email.com").
3. **User:** Enters a strong password (e.g., "Password123!").
4. **User:** Confirms the password.
5. **User:** Clicks the "Create Account" button.
6. **System:** Verifies the email format and password strength.
7. **System:** Creates a new user account in the database.
8. **System:** Sends a confirmation email to the provided address.
9. **System:** Displays a success message ("Account created successfully!") and redirects the user to the login page.

**Scenario 2: Invalid Email Format**

1. **User:** Navigates to the "Create Account" page.
2. **User:** Enters an invalid email address (e.g., "example").
3. **User:** Enters a valid password.
4. **User:** Confirms the password.
5. **User:** Clicks the "Create Account" button.
6. **System:** Displays an error message ("Invalid email address format.").
7. **User:** Corrects the email address (e.g., "example@email.com").
8. **User:** Continues with the registration process.

**Scenario 3: Weak Password**

1. **User:** Navigates to the "Create Account" page.
2. **User:** Enters a valid email address.
3. **User:** Enters a weak password (e.g., "password").
4. **User:** Confirms the password.
5. **User:** Clicks the "Create Account" button.
6. **System:** Displays an error message ("Password is too weak. Please use a combination of uppercase and lowercase letters, numbers, and symbols.").
7. **User:** Enters a stronger password.
8. **User:** Continues with the registration process.

**Scenario 4: Username Already Exists**

1. **User:** Navigates to the "Create Account" page.
2. **User:** Enters an email address that is already registered.
3. **User:** Enters a valid password.
4. **User:** Confirms the password.
5. **User:** Clicks the "Create Account" button.
6. **System:** Displays an error message ("Email address is already registered.").
7. **User:** Enters a different email address and continues with the registration process.

ss