**Features:**

1. **Account Management**: Allow the user to create and manage multiple accounts (like checking, savings, credit card, etc.).
2. **Transaction Logging**: Provide a way to log income and expenses. Each transaction should include the date, amount, category (like groceries, utilities, entertainment, etc.), and a note field for additional details.
3. **Budgeting**: Allow the user to set a budget for each category and track spending against the budget.
4. **Reports and Analysis**: Provide reports that show spending by category and over time. This could include pie charts, line graphs, etc.
5. **Recurring Transactions**: Allow the user to set up recurring transactions for bills that are the same amount and due on the same day every month.
6. **Alerts and Reminders**: Send alerts when the user is close to exceeding their budget, and reminders for upcoming bills.
7. **Data Import/Export**: Provide a way to import transactions from bank statements (like CSV files) and export data for backup or use in other applications.
8. **Security**: Since financial data is sensitive, make sure to implement strong security measures. This could include encryption for data storage and transmission, a strong password policy, and possibly two-factor authentication.

**DB Schema:**

1. Define your database schema. You'll need tables for users, accounts, transactions, categories, and possibly others depending on your features.
2. Create your Laravel models, migrations, and controllers for each of these tables.
3. Implement the account management features. This includes creating, viewing, updating, and deleting accounts.
4. Implement the transaction logging features. This includes creating, viewing, updating, and deleting transactions.
5. Implement the budgeting features. This includes setting a budget for each category and tracking spending against it.
6. Implement the reports and analysis features. This could involve creating a separate reports controller, or it could be part of the transactions or categories controller.
7. Implement the recurring transactions, alerts and reminders, and data import/export features.
8. Implement the security features. This includes encrypting sensitive data, enforcing a strong password policy, and possibly implementing two-factor authentication.
9. Test your application thoroughly to make sure everything works as expected.
10. Deploy your application to a web server.

**Pages:**

**1. Dashboard:**

* Overview of financial status.
* Summary of account balances.
* Graphical representation of spending by categories against allocated budgets.
* Alerts for nearing budget limits or upcoming bills.

**2. Accounts Page:**

* List of user accounts.
* Ability to add, delete, and view individual accounts.
* Transaction history for each account.

**3. Transactions Page:**

* Ability to log income and expenses.
* Form to add new transactions (date, amount, category, note).
* Filter and search options for transactions.

**4. Budgets Page:**

* List of budget categories with allocated amounts.
* Ability to set/edit budgets for each category.
* Visual representation (graphs/charts) comparing actual spending against budgeted amounts.

**5. Reports & Analysis Page:**

* Detailed breakdown of spending by category and time period.
* Graphs/charts showing spending trends.
* Export options for reports (CSV, PDF).

**6. Recurring Transactions Page:**

* List of recurring bills/expenses.
* Ability to add, edit, and delete recurring transactions.
* Option to link recurring transactions to specific accounts.

**7. Settings Page:**

* Account settings (password change, two-factor authentication).
* Preferences (currency, date format).
* Data import/export options.

**8. Authentication Pages:**

* Login page.
* Registration page.
* Forgot password/reset password page.

**9. Alerts and Notifications:**

* Notification center for alerts about approaching budget limits, upcoming bills, etc.
* Ability to customize notification preferences.

**Design Considerations:**

* **Navigation**: Implement a clear and consistent navigation menu to access different sections.
* **Responsive Design**: Ensure the interface is mobile-friendly for easy access on various devices.
* **Visualizations**: Use graphs, charts, and infographics for better data representation.
* **Accessibility**: Make sure the UI is accessible and easy to use for all users.
* **Security**: Incorporate security measures, especially for sensitive areas like authentication and data handling.