Design Document
Detective Dollar

12/05/2023

Project Owner:

Thomas Toy

Developers:

Hersh Rudrawal

Rohan Venkatapuram

Ben Styles

Cecile Noguera

1. Overview:

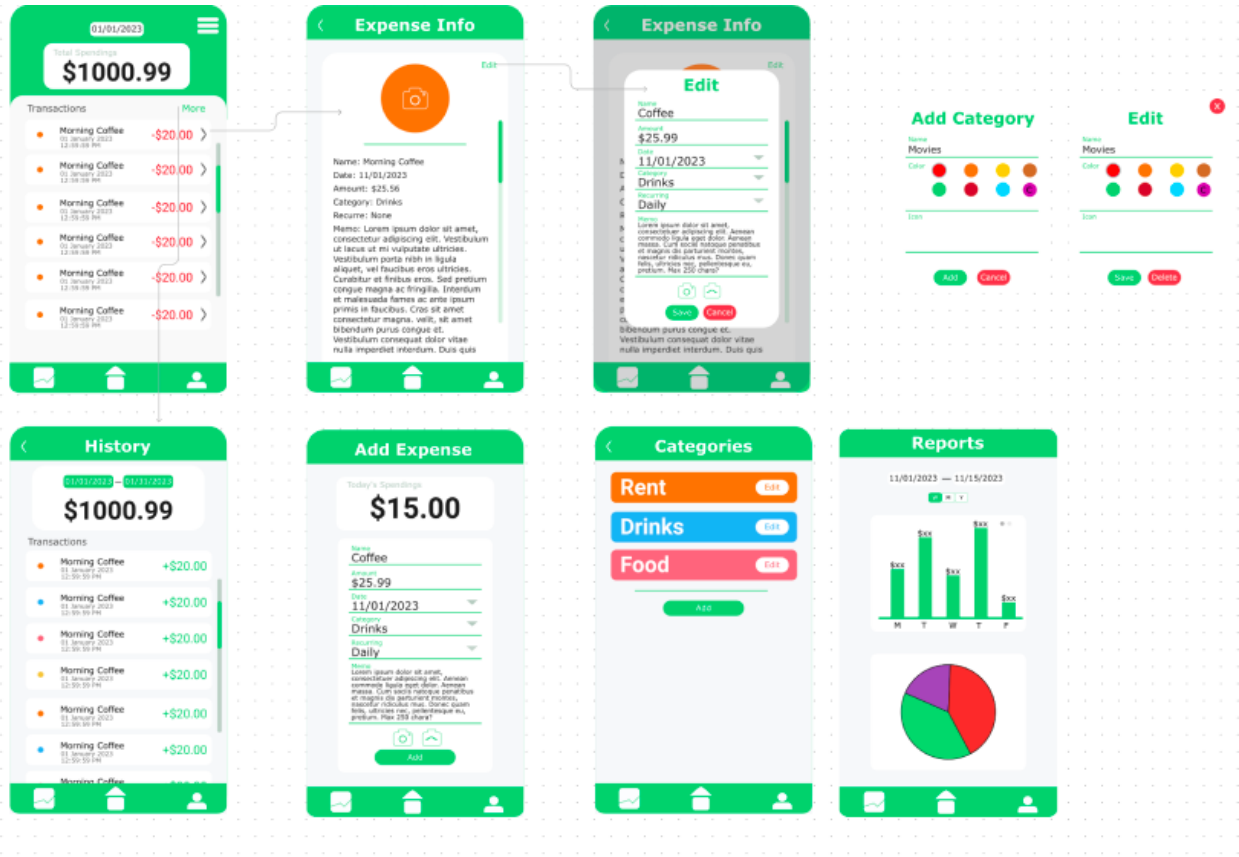
- *Briefly describe the purpose and goals of the spending app.*

- *Provide an overview of the architecture and major components.*

The expense tracker app is designed to help users efficiently track and manage their expenses. The primary goals include providing a user-friendly interface for expense entry, offering insightful visualizations of spending patterns, and encouraging financial awareness. The app aims to empower users to make informed financial decisions and improve their overall financial well-being.

The app leverages Expo Go's built-in SQLite for efficient data storage on the client side, allowing seamless cross-platform compatibility with React Native for a unified user experience across iOS and Android. React Native Gifted Charts is used to build our graphs, and React Native Community DateTimePicker was used for our date selections. SQLite manages local data storage. Major components include UI, local database management, and third-party libraries for charts and date selection.

2. User Interface (UI) Design:



The UI is designed with simplicity and intuitiveness in mind, enhanced by React Native Gifted Charts for interactive graphs and React Native Community DateTimePicker for smooth date selection. Touch gestures and dynamic UI elements contribute to a positive user experience.

3. System Architecture:

The app follows a client-side architecture. Expo Go serves as the frontend, handling UI and user interactions, with React Native Gifted Charts and DateTimePicker enhancing graph and date selection functionalities. Data flows from user generated inputs and this goes to SQLite

which manages this with local data storage. React Native Gifted Charts dynamically generates graphs based on the stored queried data.

4. Data Model:

Entities include expenses, categories, and user data. Relationships link expenses to categories. Data is stored locally using SQLite and is organized by timestamp and category, which allows for efficient retrieval.

5. Functionality and Features:

Major Features:

1. Expense Tracking
    - Via intuitive input forms
2. Categorized Spending
    - Customizable categories
3. Interactive Graphs with React Native Gifted Charts
4. Date Selection with React Native Community DateTimePicker
5. Local Data Storage
    - Expo Go's SQLite

6. Security Considerations:

Potential security risks include people being able to get insights on a person's spending data. One way we have mitigated this was through local storage.

7. External Services and APIs:

Integration with React Native Gifted Charts and DateTimePicker for enhanced graph and date selection features.

8. Performance Considerations:

Optimization Strategies:

- Optimize database queries for faster data retrieval.
- Implement caching mechanisms for frequently accessed data.
- Asynchronous loading of UI components, including charts.

Bottlenecks:

- Potential bottlenecks may include large datasets affecting graph rendering and history page retrieval.

9. Testing Strategy:

- Unit testing for individual components.
- Integration testing for UI, database interactions, and third-party libraries.
- User acceptance testing for overall functionality.

10. Deployment Plan:

Check out here on how to deploy the app.

11. Maintenance and Upgrades:

Considerations:

- Scalability adjustments based on user growth.
- Compatibility checks for Expo Go, SQLite, and third-party library updates.