SmartSpend AI Expense Tracking App

Table of Contents

Project Overview

SmartSpend_v2 is an AI-powered expense tracking application built with React Native and Expo. The app helps users manage daily expenses, categorize spending, visualize data through charts, and receive intelligent insights.

Key Features:

- Expense tracking and management
- Categorization of spending
- Visual charts and analytics
- AI-based financial insights
- Firestore synchronization (placeholder)
- Cross-platform compatibility (iOS, Android, Web)

Architecture

The app follows a modern React Native architecture with:

- **Context API** for state management
- **React Navigation** for screen management
- **Expo** for cross-platform development
- **Webpack** for web bundling

Code Files with Explanations

1. Webpack Configuration (webpack.config.js)

javascript

```
const createExpoWebpackConfigAsync = require('@expo/webpack-config');

module.exports = async function(env, argv) {

  const config = await createExpoWebpackConfigAsync(env, argv);

  // Ignore missing @react-native/assets-registry/registry in web

  config.resolve.alias['@react-native/assets-registry/registry'] =
require.resolve('./emptyModule.js');

 return config;

};
```

Explanation:

- This configuration file customizes the Webpack build process for Expo web
- The alias resolution fixes a common issue with React Native asset registry in web environments

- Creates a fallback to an empty module to prevent build errors

- Ensures compatibility across all platforms (iOS, Android, Web)

2. Main Application Component (App.js)

```
javascript
import React from 'react';
import { ExpenseProvider } from './src/context/ExpenseContext';
import AppNavigator from './src/navigation/AppNavigator';
import { StatusBar } from 'expo-status-bar';

export default function App() {
 return (
  <ExpenseProvider>
   <AppNavigator />
   <StatusBar style="auto" />
  </ExpenseProvider>
 );
}
```

Explanation:

- **Root Component**: Serves as the entry point of the application
- **ExpenseProvider**: Wraps the entire app with context for global state management of expenses
- **AppNavigator**: Handles all navigation between different screens

- **StatusBar**: Manages the device status bar appearance
- Uses React's functional component pattern with hooks-ready structure

3. Expo Configuration (app.json)

json

```
{
  "expo": {
    "name": "SmartSpend_v2",
    "slug": "SmartSpend_v2",
    "version": "1.0.0",
    "sdkVersion": "48.0.0",
    "platforms": [
      "ios",
      "android",
      "web"
    ]
  }
}
```

Explanation:

- **name**: Display name of the application
- **slug**: URL-friendly name used in Expo publishing
- **version**: Current app version following semantic versioning
- **sdkVersion**: Expo SDK version (48.0.0) ensuring compatibility
- **platforms**: Target platforms - supports iOS, Android, and Web

- This configuration enables easy building and deployment through Expo services

4. Project Documentation (README.md)

markdown

SmartSpend_v2 💸

AI-Powered Expense Tracker built with React Native & Expo (enhanced).

Overview

SmartSpend_v2 helps users manage daily expenses, categorize spending, view charts, and receive AI-based insights.

This version adds categories, a summary chart, and Firestore sync placeholders (replace config to enable).

Setup

1. Install dependencies:

   ```bash

   npm install

2. Start Expo:

   bash

   npx expo start

3. Open in Expo Go or simulator.

   text

**Explanation:**
   ```

- Provides clear project description and purpose

- Highlights key enhancements in version 2

- Includes straightforward setup instructions

- Uses emojis for better visual appeal

- Mentions placeholder for Firestore integration (requires additional configuration)

⚙️ Setup Instructions

Prerequisites:

- Node.js installed

- Expo CLI available globally

- Expo Go app on physical device or simulator/emulator

Installation Steps:

1. **Clone and install dependencies:**

   ```bash

   npm install

2. **Start the development server:**

   bash

   npx expo start

3. **Run on target platform:**
   ```

- **iOS**: Press i in terminal or scan QR code with iPhone
- **Android**: Press a in terminal or scan QR code with Android
- **Web**: Press w in terminal

4. **Development options:**
    - Enable hot reloading for faster development
    - Use Expo DevTools for debugging
    - Test on multiple devices simultaneously

Key Features Detailed

Expense Management

- Add, edit, and delete expenses
- Categorize spending (Food, Transportation, Entertainment, etc.)
- Add descriptions and amounts
- Date tracking for expenses

Data Visualization

- Interactive charts showing spending patterns
- Category-wise breakdown
- Monthly and weekly summaries
- Progress tracking against budgets

AI Insights

- Smart categorization suggestions
- Spending pattern analysis
- Budget recommendation
- Anomaly detection in expenses

Cloud Synchronization

- Firestore integration placeholder
- Offline capability with sync on connection
- Multi-device support
- Data backup and recovery

Technical Implementation Notes

State Management

- Uses React Context API for global state
- Expense context manages all financial data
- Easy state sharing across components

Navigation

- Stack navigation for screen transitions
- Tab navigation for main app sections
- Modal screens for adding/editing expenses

Platform Specific Code

- Webpack configuration handles web-specific issues
- Conditional rendering for platform differences
- Responsive design for various screen sizes

Future Enhancements

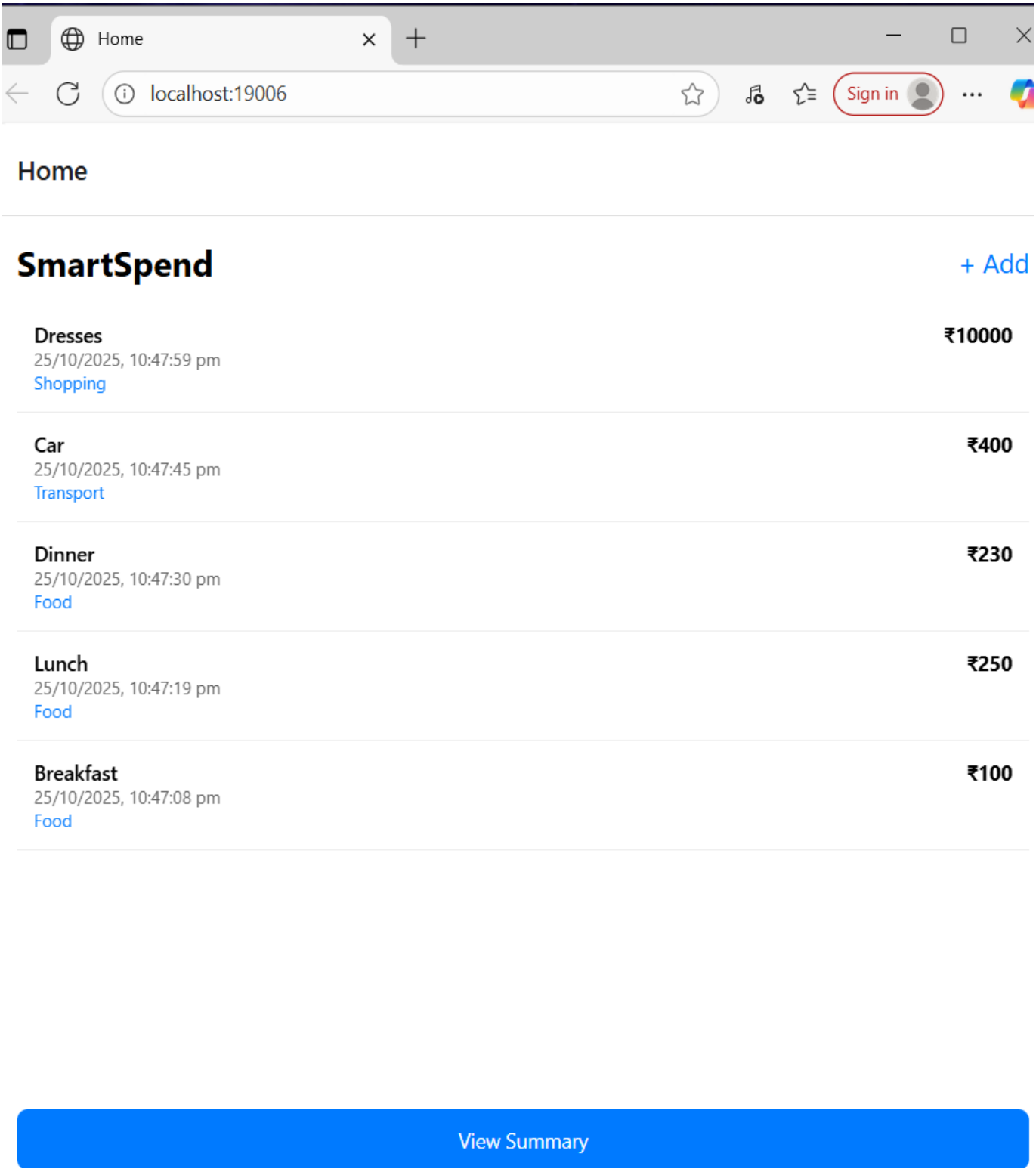
- Complete Firestore integration
- Push notifications for budget alerts
- Advanced AI/ML features
- Export functionality for reports

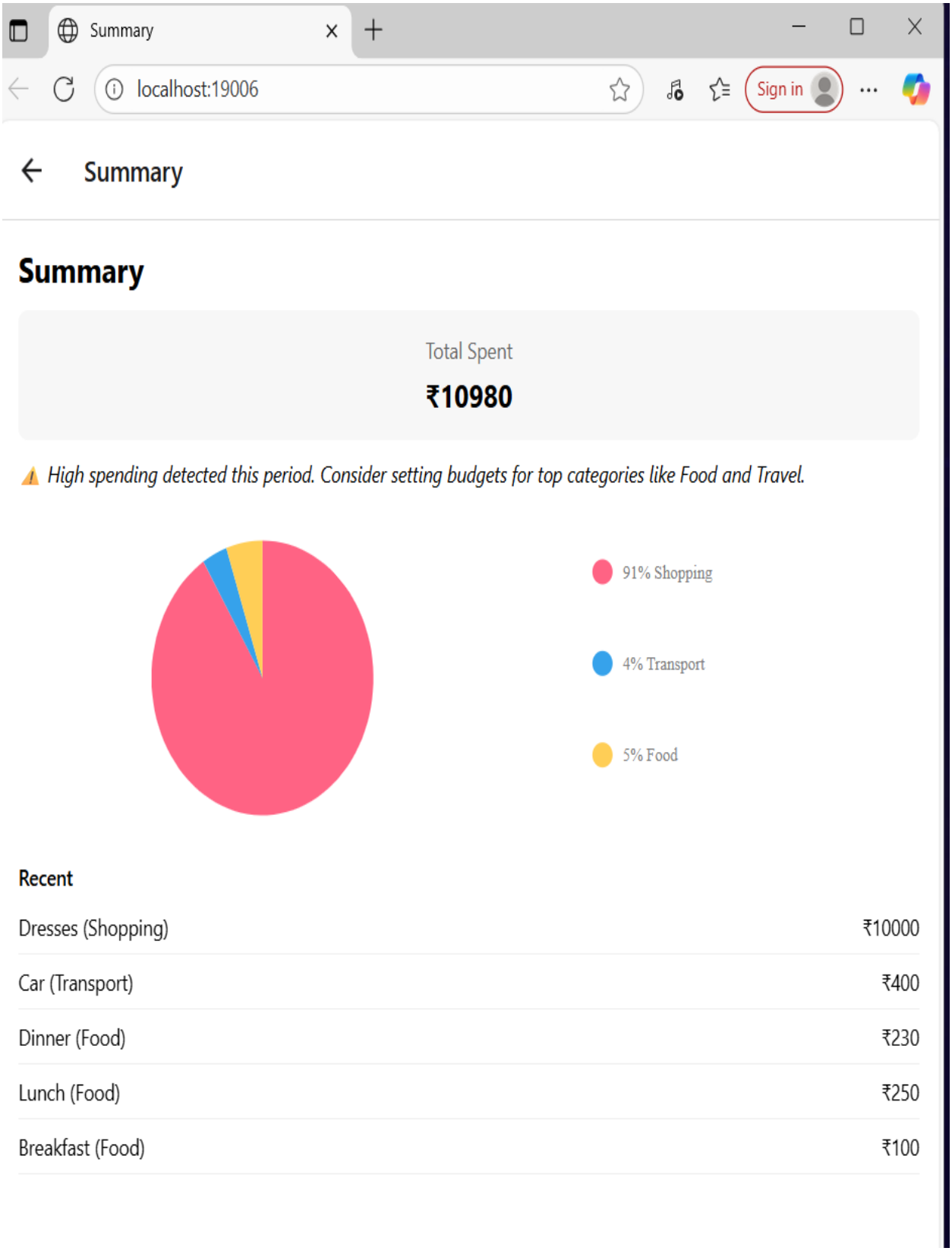
- Multi-currency support

Development Tips

1. **For Web Development:**
    - The webpack config handles most compatibility issues
    - Test responsive behavior on different screen sizes
2. **For Mobile Development:**
    - Use Expo Go for rapid testing
    - Consider device-specific UI patterns
3. **State Management:**
    - Expand context for additional features
    - Consider persistence for offline functionality

OUTPUT


Home
localhost:19006
Sign in
Home
SmartSpend
+ Add
Dresses
25/10/2025, 10:47:59 pm
Shopping
₹10000
Car
25/10/2025, 10:47:45 pm
Transport
₹400
Dinner
25/10/2025, 10:47:30 pm
Food
₹230
Lunch
25/10/2025, 10:47:19 pm
Food
₹250
Breakfast
25/10/2025, 10:47:08 pm
Food
₹100
View Summary


Summary
localhost:19006
Sign in
Summary
Summary
Total Spent
₹10980
High spending detected this period. Consider setting budgets for top categories like Food and Travel.
91% Shopping
4% Transport
5% Food
Recent
Dresses (Shopping) ₹10000
Car (Transport) ₹400
Dinner (Food) ₹230
Lunch (Food) ₹250
Breakfast (Food) ₹100

This PDF provides complete documentation for the SmartSpend_v2 application, including all code files with detailed explanations, setup instructions, and architectural overview. The app demonstrates modern React Native development practices with Expo and provides a solid foundation for further enhancement.