CALCULATING FAMILY EXPENSES USING SERVICENOW

PROJECT DOCUMENTATION FORMAT

1. **INTRODUCTION :**
    - **Project Title :** Calculating Family Expenses Using Service Now
    - Team ID : LTVIP2025TMID31080
    - Team Size : 4
    - Team Leader : N Teja
        Team Member: Lakum Sai Thanusha
        Team Member: Bharath Talasila
        Team Member: Kandula Uday Kiran
2. **Project Overview :**

Purpose:

- The purpose of this project is to create a digital system using ServiceNow to efficiently track, manage, and analyze family expenses. It aims to help users maintain a clear and organized record of daily spending, categorize expenses, and ensure better financial planning and control.
- Goals:
- Automate the tracking of family expenses.
- Provide categorized expense summaries for better insights.
- Enable easy data entry and access via forms and dashboards.
- Improve financial decision-making with reports and visual analytics.
- Ensure data security, reliability, and user-friendly experience.

Features :

1. User Registration & Authentication
    - Secure sign-up/login through email, Gmail, or LinkedIn ○
      OTP and email verification for user validation
2. Expense Entry & Management
    - Add, edit, or delete daily/monthly expenses
    - Categorize expenses (e.g., food, bills,education, travel)
3. Dashboard & Visualization
    - Interactive dashboard displaying expense trends

Graphs and charts for category-wise analysis

4. Reports & Summaries
   - Monthly/weekly expense summaries

     Exportable reports (PDF/Excel)
5. Notifications & Alerts
   - Alerts for budget limits or unusual spending
   - Reminder for upcoming bills or dues
   -

     Role-Based Access
   - Different access levels for individual or family members
   - Admin control for expense approvals and updates
6. Data Security & Integrity
   - Encrypted storage of financial data
   - Backup and recovery mechanisms
7. Mobile Accessibility

   ○ Responsive design or integration with mobile platforms for

   easy access
8. Integration Capabilities

   ○ Potential integration with payment systems or bank feeds

   ○ API support for third-party tools

These features work together to provide a complete, reliable, and user-friendly family expense management system within the ServiceNow platform.

3. ARCHITECTURE :

- **FRONTEND :**

The frontend of the "Calculating Family Expenses Using ServiceNow" project is built using React with a component-based architecture. It features reusable UI components, responsive layouts, and secure routing. State is managed using Context API or Redux, and the interface

communicates with the ServiceNow backend via REST APIs. The design is enhanced with data visualizations using chart libraries, and user authentication is handled with token-based security for protected routes.

- **BACKEND :**

The backend is powered by **ServiceNow**, utilizing its platform capabilities for data storage, workflows, and automation. It includes custom tables for storing expense data, business rules for validation, and scripted REST APIs for frontend integration. Role-based access controls ensure data security, while scheduled jobs and notifications handle reminders and alerts. The architecture supports scalable data operations with minimal custom coding through ServiceNow's low-code/no-code environment.

- **DATABASE :**

The database is designed using **ServiceNow's built-in relational data model**, where custom tables are created to store and manage family expense data. Key tables include:

- **User Table** – Stores user profiles and authentication details.
- **Expense Table** – Records individual expense entries with fields like amount, category, date, and description.
- **Category Table** – Defines expense categories (e.g., Food, Rent, Utilities).
- **Budget Table** – Tracks monthly or yearly budget limits.

Relationships are established using reference fields, enabling data linkage (e.g., each expense is linked to a user and category). The structure ensures data integrity, scalability, and efficient querying within the ServiceNow platform.

4. SET UP INSTRUCTIONS :

PREREQUISITES :

Software Dependencies (Pre-requisites):

1. **ServiceNow PDI** – Backend development and data storage
2. **React.js** – Frontend development
3. **Node.js & npm** – React setup and package management
4. **Axios** – API communication
5. **Chart.js / Recharts** – Data visualization
6. **Tailwind CSS / Bootstrap** – UI styling
7. **Postman** – API testing

8. **VS Code** – Code editing
9. **Web Browser** – App testing

INSTALLATION :

- Clone the Repository

git clone <repo-url>

cd <project-folder>

- Install Dependencies npm
- install Set Up Environment Variables

Create a `.env` file in the root folder

Add:

```
REACT_APP_API_URL=https://your-servicenow-instance/api
```

- Run the App npm start
- Access Locally

Open http://localhost:3000 in your browser

5. FOLDER STRUCTURE :

CILENT:

- /components – Contains reusable UI elements like expense forms, charts, and navigation bars.
- /pages – Includes main screens such as Login, Register, Dashboard, and Reports.
- /services – Manages API requests to ServiceNow (e.g., add/get expenses).
- /context – Handles global state (e.g., user authentication, expense data).
- /assets – Stores images, icons, and custom styles.
- App.js – Defines the main layout and routes.
- index.js – Entry point that initializes and renders the app.

This structure supports a clean, scalable, and efficient interface for tracking family expenses.

SERVER :

The server is handled by ServiceNow, which acts as the backend platform. It includes:

- Custom Tables – For storing users, expenses, and categories
- Scripted REST APIs – To handle CRUD operations from the React frontend
- Business Rules & Workflows – For automation (e.g., budget alerts, data validation) • Access Control Rules – To ensure secure, role-based access to data

6. RUNNING THE APPLICATION :

FRONTEND :

To run the frontend of the *Calculating Family Expenses* project, first navigate to the project folder and install all necessary dependencies using npm install. Then, start the development server with npm start. This will compile the React app and launch it on http://localhost:3000 in your browser. The frontend will interact with the ServiceNow backend via REST APIs to fetch and display family expense data in real time.

BACKEND :

Log in to your ServiceNow Personal Developer Instance, create custom tables (e.g., Users, Expenses), and configure business rules and scripted REST APIs. The backend runs on the cloud and is always active—no manual server start is needed. It handles data storage, logic, and API communication with the frontend.

7. API DOCUMENT :

The backend exposes a set of **Scripted REST API endpoints** in ServiceNow that allow the React frontend to interact with the expense data. These endpoints handle all CRUD operations for users, expenses, and categories. Each endpoint supports specific HTTP methods and accepts JSON input for creating, updating, or deleting records. Responses are returned in JSON format with success messages or data objects. This setup enables secure, real-time communication between the frontend and the ServiceNow backend.

8. AUTHENTICATION :

Authentication is handled using JWT tokens. Users log in with their credentials via the /api/users/login endpoint. On success, a token is returned and stored in the frontend (e.g., local storage). This token is then attached to all API requests for secure access. The backend validates the token to ensure only authenticated users can view or modify expense data.

9. USER INTERFACE :

The user interface is built with **React** and designed for simplicity and ease of use. It includes pages for user login/registration, a dashboard to view expenses, a form to add/edit entries, and charts for visualizing spending. The UI is responsive, styled with Tailwind CSS or Bootstrap, and provides smooth navigation with interactive elements for a better user experience.

10 . **TESTING :**

The project uses both **functional** and **performance** testing strategies:

- **Functional Testing**:
  Ensures each feature (login, expense entry, dashboard) works as expected.
  Tool: Jest & React Testing Library (for frontend)
  Tool: ServiceNow ATF (Automated Test Framework) for backend workflows
- **Performance Testing**:
  Checks speed, responsiveness, and stability under load.
  Tool: Postman (API testing), JMeter or BlazeMeter (for load testing APIs)

This approach ensures the system is reliable, accurate, and scalable.

10. SCREENSHOT or DEMO:

https://drive.google.com/file/d/1xRlq0EBRBG1e4NMYSZ8k9zBvExHDmIBB/view?usp=drive_link

11. KNOWN ISSUES:

1. Session Timeout – JWT token may expire without user notification.
2. Limited Offline Support – App requires internet to fetch/update data from ServiceNow.
3. API Rate Limits – ServiceNow instance may throttle high-frequency API calls.
4. UI Rendering Delay – Charts may briefly lag on slower networks.
5. Validation Gaps – Some form fields may lack strict input validation.

12. FUTURE ENHANCEMENTS

1. Mobile App Integration – Build a mobile-friendly version for on-the-go access.
2. Voice Input for Expenses – Add voice-based entry using speech recognition.
3. AI-based Budget Suggestions – Use AI to analyze spending and suggest budgets.
4. Multi-Currency Support – Enable tracking expenses in different currencies.
5. Export to PDF/Excel – Allow users to download reports for records or tax use.

6. Bank Integration – Auto-fetch expenses from bank statements via APIs.