Personal Budget Tracker

1. Introduce

Team Name: IP_9

Member Name: Kim Sang Huynh and Kelly Kla-Diihbah

Our team brainstormed a project that addresses a common real-world need—managing personal finances. We chose a Personal Budget Tracker because it's a practical tool that helps users monitor income, expenses, and savings, and it aligns with the course's focus on building functional, user-friendly web applications.

2. Purpose of the Proposal

The Personal Budget Tracker aims to provide users with an intuitive web application to track their income and expenses, set budget goals, and visualize their financial progress.

Many individuals struggle to manage their finances effectively due to a lack of accessible, user-friendly tools tailored to their needs. This project addresses this by offering a simple yet powerful solution.

Financial literacy and budgeting are critical life skills. Our app will empower users to make informed financial decisions, making it a valuable and relevant project.

3.Goals & Objectives

Goals:

- Build a fully functional, responsive web application for personal budget management.
- Gain hands-on experience with full-stack development using React, Node.js, and a database.
- Deliver an intuitive and visually appealing user interface.

Objectives:

- Complete initial planning and wireframes.
- Implement basic UI with React Router for navigation.
- Set up backend with CRUD operations for budget data.
- Add advanced features (e.g., charts) and polish UI/UX.

- Final Deliverables: Submit software, architecture document, and demo video by May 11, 2025.

---

4 Project Description

Overview:
The Personal Budget Tracker will allow users to:

- Register and log in to a personal account.
- Add, view, edit, and delete income and expense entries.
- Set monthly budget goals and track progress.
- View financial summaries and visualizations (e.g., pie charts).

Technologies:

- **Frontend:** React (JSX) with React Router for navigation.
- **Backend:** Node.js with Express for API routes.
- **Database:** MongoDB to store user data, income, and expenses.

Pages (Minimum 6):

1. **Login Page:** User authentication.
2. **Signup Page:** Account creation.
3. **Home/Dashboard:** Overview of budget, recent transactions, and progress.
4. **Transaction Management:** Add/edit/delete income and expense entries (Admin-like features for the user).
5. **Budget Goal Setup/Confirmation:** Set and confirm budget goals.
6. **About/Team Info:** Course details and team information.

5.Project Path Selection

We opted for a new project to explore a fresh idea that excites us and allows us to apply the full range of skills learned in the course. A Personal Budget Tracker offers ample opportunity to implement CRUD operations, responsive design, and API integration from scratch.

Implementation Plan:

Design UI wireframes and plan data structure.

Build frontend with React and backend with Node.js/Express.

Integrate MongoDB for persistent storage and test functionality weekly.

6.Feature Ownership & Responsibility

Feature Breakdown:

1. Login & Signup Pages
    - Description: User authentication and account creation with secure password handling.
    - Assigned Developer: Kim Sang Huynh
    - Tech Involvement: React (frontend) + Node.js/Express (backend API for authentication).
2. Transaction Management
    - Description: CRUD operations for income/expense entries.
    - Assigned Developer: Kelly Kla-Diihbah
    - Tech Involvement: React (frontend forms) + Node.js/Express (backend CRUD routes).
3. Home/Dashboard
    - Description: Displays budget overview and recent transactions.
    - Assigned Developer: Kim Sang Huynh
    - Tech Involvement: React (frontend with hooks) + Node.js (backend data retrieval).
4. Budget Goal Setup
    - Description: Set and track monthly budget goals.

- Assigned Developer: Kelly Kla-Diihbah
- Tech Involvement: React (frontend input) + Node.js/Express (backend storage).

7. Resources and Tools

- **Technologies:** React, Node.js, Express, MongoDB, GitLab, REST API.
- **Tools:** Figma (wireframes), Axios (API calls), Chart.js (visualizations).
- **Time Commitment:** 5-7 hours/week per member.
- **Responsibility Division:** Equal split between frontend and backend tasks, as outlined above.

8.File Structure and Project Organization

- **frontend/**
  - src/
    - assets/ (e.g., icons, images)
    - components/ (e.g., LoginForm.jsx, TransactionList.jsx)
    - App.jsx, main.jsx
- **backend/**
  - Server logic, API routes (e.g., /api/transactions), MongoDB connection.
- **Documents/**
  - Wireframes, architecture document, demo video.

Communication: Frontend and backend will interact via RESTful APIs (e.g., POST /api/transactions to add an expense).

9.Data Sources and Management

Data Sources:

- User input for transactions (amount, category, date).
- Local seed data for testing (e.g., sample expenses).

CRUD Operations:

- **Create:** Add new transactions via POST requests.
- **Read:** Fetch transactions and budget data with GET requests.
- **Update:** Edit transactions via PUT requests.
- **Delete:** Remove transactions with DELETE requests.

10.User Experience Views

Wireframes (to be created in Figma/Excalidraw):

1. **Login Page:** Username/password fields, submit button.
2. **Dashboard:** Summary stats, recent transactions list.
3. **Transaction Management:** Form to add/edit entries, table of transactions.
4. **Budget Goal Setup:** Input field for goal amount, confirmation button.

(Describe each screen's purpose and user actions—attach sketches in your PDF.)

11.Final Comments

We aim to build a practical tool while mastering full-stack development. We're excited to learn React Hooks, API integration, and database management.
Primary Contact: Kim Sang Huynh and Kelly Kla-Diihbah