Budget Planning Web Application

July 6, 2025

Overview

The **Budget Planning Web Application** is a web-based tool to help users:

- Track income and expenses
- Define and monitor budget goals
- Analyze spending using visual charts
- Export transaction data

This application includes user authentication and a clean, responsive design with support for dark mode.

Features

- **User Authentication**
  - Registration, login, logout
  - JWT-based sessions
  - Password reset by username
- **Transaction Management**
  - Add, edit, and delete transactions
  - Mark transactions as recurring (daily, weekly, monthly)
- **Budget Goals**
  - Define spending limits per category
  - Track completed and uncompleted goals
- **Analytics**
  - Expense breakdown chart using Chart.js
  - Monthly budget progress bar
  - Export transactions as CSV
- **Dark Mode**
  - Toggle between light and dark themes

Tech Stack

- **Frontend:** HTML, CSS, JavaScript
- **Backend:** Node.js (`http` module without Express)
- **Database:** MongoDB (Mongoose)
- **Authentication:** JSON Web Tokens (JWT)
- **Email (optional):** Nodemailer

Installation

1. Clone the repository

```
git clone https://your-repo-url.git
cd budget-planning-app
```

2. Install dependencies

```
npm install
```

3. Configure environment variables

Create a `.env` file in the project root:

```
MONGO_URI=mongodb://localhost:27017/budgetapp
JWT_SECRET=your_secret_key
EMAIL_USER=your_email@example.com
EMAIL_PASS=your_email_password
BASE_URL=http://localhost:3000
```

Note: If you don't plan to use email features, you can leave `EMAIL_USER` and `EMAIL_PASS` as placeholders.

4. Start MongoDB

Ensure MongoDB is running locally:

```
mongod
```

5. Start the server

```
node server.js
```

Visit: http://localhost:3000

Usage

- Register and login to create your account.
- Add transactions for income and expenses.
- Set budget goals by category.
- Export your transactions as a CSV file.
- Toggle dark mode for the interface.

Folder Structure

```
/public
  index.html          (Login page)
  register.html       (Registration page)
  forgotpass.html     (Password reset request)
  reset.html          (New password setup)
  dashboard.html      (User dashboard)
  style.css           (Styling and dark mode)
  script.js           (Client-side logic)
server.js             (Main server code)
```

Important Notes

- This project does not use Express; routing is handled by the native Node.js `http` module.
- Password reset is implemented via username.
- For production deployment, consider:
  - Adding CSRF protection
  - JWT expiration
  - Rate limiting
  - Email sending for password resets