**BUDGET TRACKER**

This app will help to monitor your budget everyday/weekly. App will allow users to set a budget, track expenses, and get alerts when they're close to exceeding their budget. Users can also create savings goals and track their progress towards those goals.

**Features:**

1. Expense tracking: Allow users to track their expenses, including the name of the expense, the amount spent, and the date.
2. Income tracking: Allow users to track their income, including the source of the income, the amount earned, and the date.
3. Budget tracking: Allow users to set a budget for a particular time period (e.g. a week, a month) and track their progress towards their budget.
4. Expense categories: Allow users to categorize their expenses (e.g. groceries, entertainment, transportation) so that they can see where they're spending the most money.
5. Expense analysis: Provide users with charts or graphs that show them how they're spending their money and which categories they're spending the most money on.
6. Goal tracking: Allow users to set financial goals (e.g. saving for a vacation, paying off debt) and track their progress towards those goals.
7. Reminders: Provide users with reminders to pay bills or make deposits.

**User stories:**

1. As a user, I want to add expenses to the budget tracker so that I can keep track of my spending.

* The application should have a form where users can enter the amount and description of their expenses.
* After entering the expense details, users should be able to save it to their budget tracker.

1. As a user, I want to categorize my expenses (e.g. groceries, rent, entertainment) so I can see where I'm spending the most money.

* The application should have a category field where users can select or create a category for their expenses.
* The application should allow users to view their expenses by category and display the total amount spent in each category.

1. As a user, I want to set a budget for a particular time period (e.g. every day, a week) so that I can track my progress towards my financial goals.

* The application should allow users to set a budget limit for each time period they want to track.

1. Users should be able to select the time period they want to track (e.g. daily, weekly, monthly).

* The application should display the remaining amount of money left to spend before the user reaches their budget limit.

1. As a user, I want to see how much money I have left to spend before I reach my budget limit.

* The application should display the remaining amount of money left to spend before the user reaches their budget limit.
* The application should alert users when they have exceeded their budget limit in a particular time period.

1. As a user, I want to be able to see charts or graphs that show me how I'm spending my money and which categories I'm spending the most money on.
2. As a user, I want to be able to set financial goals (e.g. saving for a vacation, paying off debt) and track my progress towards those goals.
3. As a user, I want to be able to receive reminders to pay bills or make deposits.

**MVP scope:**

1. Allow users to add expenses to the budget tracker.
2. Allow users to set a budget for a particular time period (e.g. every day, a week).
3. Allow users to see how much money they have left to spend before they reach their budget limit.
4. Allow users to categorize their expenses (e.g. groceries, rent, entertainment).
5. Provide users with a dashboard that shows their spending and budget progress in a simple and easy-to-understand format.
6. Allow users to receive reminders to pay bills or make deposits.
7. Use a responsive design that works well on desktop and mobile devices.

// import React, { useState, useReducer } from 'react';

// import { BrowserRouter as Router, Route, Link, Routes } from 'react-router-dom';

// const initialState = {

//   transactions: []

// };

// function reducer(state, action) {

//   switch (action.type) {

//     case 'ADD\_TRANSACTION':

//       return {

//         ...state,

//         transactions: [...state.transactions, action.payload]

//       };

//     case 'DELETE\_TRANSACTION':

//       return {

//         ...state,

//         transactions: state.transactions.filter(

//           transaction => transaction.id !== action.payload

//         )

//       };

//     default:

//       return state;

//   }

// }

// function BudgetTracker() {

//   const [state, dispatch] = useReducer(reducer, initialState);

//   const [name, setName] = useState('');

//   const [amount, setAmount] = useState('');

//   function handleAddTransaction(e) {

//     e.preventDefault();

//     const newTransaction = {

//       id: Math.random(),

//       name: name,

//       amount: +amount

//     };

//     dispatch({ type: 'ADD\_TRANSACTION', payload: newTransaction });

//     setName('');

//     setAmount('');

//   }

//   function handleDeleteTransaction(id) {

//     dispatch({ type: 'DELETE\_TRANSACTION', payload: id });

//   }

//   return (

//     <div>

//       <h1>Budget Tracker</h1>

//       <TransactionList transactions={state.transactions} onDeleteTransaction={handleDeleteTransaction} />

//       <TransactionForm onAddTransaction={handleAddTransaction} name={name} setName={setName} amount={amount} setAmount={setAmount} />

//     </div>

//   );

// }

// function TransactionList({ transactions, onDeleteTransaction }) {

//   return (

//     <ul>

//       {transactions.map(transaction => (

//         <TransactionItem key={transaction.id} transaction={transaction} onDeleteTransaction={onDeleteTransaction} />

//       ))}

//     </ul>

//   );

// }

// function TransactionItem({ transaction, onDeleteTransaction }) {

//   return (

//     <li>

//       {transaction.name}: ${transaction.amount}{' '}

//       <button onClick={() => onDeleteTransaction(transaction.id)}>

//         Delete

//       </button>

//     </li>

//   );

// }

// function TransactionForm({ onAddTransaction, name, setName, amount, setAmount }) {

//   return (

//     <form onSubmit={onAddTransaction}>

//       <label>

//         Name:

//         <input

//           type="text"

//           value={name}

//           onChange={e => setName(e.target.value)}

//         />

//       </label>

//       <label>

//         Amount:

//         <input

//           type="number"

//           value={amount}

//           onChange={e => setAmount(e.target.value)}

//         />

//       </label>

//       <button type="submit">Add Transaction</button>

//     </form>

//   );

// }

// function About() {

//   return (

//     <div>

//       <h2>About</h2>

//       <p>This is a budget tracker app built with React.</p>

//     </div>

//   );

// }

// function App() {

//   return (

//     <Router>

//       <div>

//         <nav>

//           <ul>

//             <li>

//               <Link to="/">Budget Tracker</Link>

//             </li>

//             <li>

//               <Link to="/about">About</Link>

//             </li>

//           </ul>

//         </nav>

//         <Routes>

//           <Route path="/" exact element={<BudgetTracker />} />

//           <Route path="/about" element={<About />} />

//         </Routes>

//       </div>

//     </Router>

//   );

// }

//export default App;

import React, { useState } from "react";

function BudgetTracker() {

  const [expenses, setExpenses] = useState([]);

  const [categories, setCategories] = useState([

    "groceries",

    "rent",

    "entertainment",

  ]);

  const [budget, setBudget] = useState(0);

  const [newBudget, setNewBudget] = useState(0);

  const [newExpense, setNewExpense] = useState("");

  const [newCategory, setNewCategory] = useState("");

  const [newAmount, setNewAmount] = useState(0);

  // set budget by weekly

  const [budgetPeriod] = useState("week");

  // add a new expense to the list of expenses

  const addExpense = () => {

    const newExpenseItem = {

      expense: newExpense,

      category: newCategory,

      amount: newAmount,

    };

    setExpenses([...expenses, newExpenseItem]);

    setNewExpense("");

    setNewCategory("");

    setNewAmount(0);

  };

  // calculate the total amount spent in a category

  const getTotalForCategory = (category) => {

    const filteredExpenses = expenses.filter(

      (expense) => expense.category === category

    );

    return filteredExpenses.reduce((acc, expense) => acc + expense.amount, 0);

  };

  // calculate the total amount spent for all categories

  const getTotalExpenses = () => {

    return expenses.reduce((acc, expense) => acc + expense.amount, 0);

  };

  // calculate the amount left to spend before reaching budget limit

  const getAmountLeftToSpend = () => {

    return budget - getTotalExpenses();

  };

  // calculate summary per category

  const renderCategorySummary = () => {

    return categories.map((category, index) => (

      <div key={index}>

        <span>{category}: </span>

        <span>{getTotalForCategory(category)}</span>

      </div>

    ));

  };

  // check if user has no money left

  const checkBudget = () => {

    if (getAmountLeftToSpend() < 0) {

      return (

        <div>

          <p>You have no money left!</p>

        </div>

      );

    }

  };

  return (

    <div>

      <h1>Budget Tracker</h1>

      <div>

        <label>Budget:</label>

        <input

          type="number"

          value={newBudget}

          onChange={(e) => setNewBudget(parseInt(e.target.value))}

        />

        <button onClick={() => setBudget(newBudget)}>Set Budget</button>

      </div>

      <div>

  <label>Expense:</label>

  <input

    type="text"

    value={newExpense}

    onChange={(e) => setNewExpense(e.target.value)}

  />

  <label>Category:</label>

  <select

    value={newCategory}

    onChange={(e) => setNewCategory(e.target.value)}

  >

    {categories.map((category) => (

      <option key={category} value={category}>

        {category}

      </option>

    ))}

  </select>

  <label>Amount:</label>

  <input

    type="number"

    value={newAmount}

    onChange={(e) => setNewAmount(parseInt(e.target.value))}

  />

  <button onClick={addExpense}>Add Expense</button>

</div>

  <div>

    <h2>Expenses:</h2>

    <table>

      <thead>

        <tr>

          <th>Category</th>

          <th>Expense</th>

          <th>Amount</th>

        </tr>

      </thead>

      <tbody>

        {expenses.map((expense) => (

          <tr key={expense.id}>

            <td>{expense.category}</td>

            <td>{expense.expense}</td>

            <td>{expense.amount}</td>

          </tr>

        ))}

      </tbody>

    </table>

    <div>

      <h2>Summary:</h2>

      <p>Budget: {budget}</p>

      <p>Total Expenses: {getTotalExpenses()}</p>

      <p>Remaining Budget: {budget - getTotalExpenses()}</p>

    </div>

  </div>

</div>

);

}

export default BudgetTracker;