## **Project Document: Finance Tracker App**

### **1. Project Overview**

The Finance Tracker App is a web application designed to help users manage their finances by tracking income, expenses, and overall balance. The app will offer users the ability to log in, record financial transactions, view their current balance, and analyse their spending habits through visual reports. The application will be developed using the MERN (MongoDB, Express.js, React.js, Node.js) stack, with the frontend and back end deployed separately.

### **2. Features**

* **User Authentication**:
  1. Users will be able to create an account and log in securely.
  2. User sessions will be managed to ensure data privacy.
* **Income and Expense Tracking**:
  1. Users can add income and expenses with a description, date, and amount.
  2. Transactions will be stored and categorised for easy retrieval.
* **Balance Overview**:
  1. The app will display the current balance based on the logged income and expenses.
  2. Users can see a breakdown of their financial status at a glance.
* **Financial Reports**:
  1. Pie charts to display the proportion of different categories of expenses.
  2. Line graphs to show income and expenses trends over time.
* **Pages**:
  1. **Registration Page**: User creation interface.
  2. **Login Page**: User authentication interface.
  3. **Dashboard**: Overview of current balance and quick add transaction feature.
  4. **Transactions Page**: Detailed list of all transactions with options to add, edit or delete.

### **3. Technology Stack**

* **Frontend**: React.js
  + State Management: Context API
  + React-Bootstrap Library: React-Bootstrap
  + Charts: Chart.js or Recharts
* **Backend**: Node.js with Express.js
  + Database: MongoDB (hosted on MongoDB Atlas)
  + Authentication: JWT (JSON Web Tokens)
* **Deployment**:
  + **Frontend**: Deployed on Netlify or Vercel
  + **Backend**: Deployed on Render or Cyclic

### **4. Database Design**

* **Users Collection**:
  + \_id: ObjectId
  + username: String
  + email: String (unique)
  + password: String (hashed)
  + createdAt: Date
* **Transactions Collection**:
  + \_id: ObjectId
  + userId: ObjectId (reference to Users collection)
  + type: String (Income/Expense)
  + amount: Number
  + date: Date
  + description: String
  + createdAt: Date

### **5. Security Considerations**

* Passwords will be hashed before storage using bcrypt.
* JWT will be used for authentication with secure cookie storage.
* HTTPS will be enforced on both frontend and backend deployments.

## **Project Plan**

### **Phase 1: Planning & Design (Day 1)**

* **Requirement Gathering**: Finalize app features and scope.
* **Wireframing & UI Design**: Create wireframes for each page.
* **Technology Setup**: Set up Git repository and CI/CD pipeline.

### **Phase 2: Backend Development (Day 2-3)**

* **User Authentication**: Develop the user login and registration API.
* **Transaction Management**: Create APIs for adding, editing, and deleting transactions.
* **Balance Calculation**: Implement logic for balance computation based on transactions.
* **Data Security**: Implement JWT authentication and secure storage practices.

### **Phase 3: Frontend Development (Day 3-5)**

* **Sign Up and Login Page**: Develop the user sign up and login interface.
* **Dashboard Page**: Implement the dashboard with balance overview and quick transaction add.
* **Transactions Page**: Develop the transaction list with edit and delete functionality.

### **Phase 4: Manual Testing (Day 5-6)**

* **E2E Testing**: Perform end-to-end testing to ensure app functionality.

### **Phase 5: Deployment (Day 7)**

* **Backend Deployment**: Deploy the backend to Heroku or AWS.
* **Frontend Deployment**: Deploy the frontend to Netlify or Vercel.
* **Final Testing**: Perform a full system test post-deployment.

### **Phase 6: Post-Launch (Day 8)**

* **Monitoring**: Set up monitoring and error tracking
* **Feedback & Iteration**: Gather user feedback and iterate on the product.
* **Documentation**: Finalize project documentation and API references.