**Full Stack Development with MERN**

**Database Design and Development Report**

|  |  |  |  |
| --- | --- | --- | --- |
| Date | 19 July 2024 |  | |
| Team ID | SWTID1721040922 |  | |
| Project Name | WalletWatch – The Expense Tracker app | |  |
| Maximum Marks | 5 |  | |

**Project Title**: WalletWatch – The Expense Tracker app

**Date**: 19th July 2024

**Prepared by**: M.Madhuri Nagalakshmi

**Objective**

The objective of this report is to outline the database design and implementation details for the WalletWatch – The Expense Tracker app project, including schema design and database management system (DBMS) integration.

**Technologies Used**

* **Database Management System (DBMS):** MongoDB
* **Object-Document Mapper (ODM):** Mongoose

**Design the Database Schema**

The database schema is designed to accommodate the following entities and relationships:

**The database schema for the expense tracker app is designed to accommodate the following entities and relationships, ensuring data integrity and efficient query performance. The primary entities include Users, Expenses, Categories, Budgets, and Reports. Here’s a detailed breakdown:**

**Entities and Relationships**

1. **Users**
   * **Fields: id, username, email, password\_hash, created\_at, updated\_at**
   * **Relationships: One-to-Many with Expenses, Categories, and Budgets**
2. **Expenses**
   * **Fields: id, user\_id, category\_id, amount, description, date, created\_at, updated\_at**
   * **Relationships: Many-to-One with Users, Many-to-One with Categories**
3. **Categories**
   * **Fields: id, user\_id, name, created\_at, updated\_at**
   * **Relationships: One-to-Many with Expenses, Many-to-One with Users**
4. **Budgets**
   * **Fields: id, user\_id, category\_id, amount, start\_date, end\_date, created\_at, updated\_at**
   * **Relationships: Many-to-One with Users, Many-to-One with Categories**
5. **Reports**
   * **Fields: id, user\_id, type, generated\_at, data**
   * **Relationships: Many-to-One with Users**

**Explanation of Key Entities and Relationships**

1. **Users**
   * **Purpose**: Stores user information for authentication and authorization.
   * **Key Fields**:
     + id: Primary key for uniquely identifying a user.
     + username, email: Unique identifiers for user login.
     + password\_hash: Securely stores user passwords.
   * **Relationships**:
     + One user can have multiple expenses, categories, and budgets.
2. **Expenses**
   * **Purpose**: Records individual expense transactions.
   * **Key Fields**:
     + id: Primary key for uniquely identifying an expense.
     + user\_id: Foreign key linking to the Users table.
     + category\_id: Foreign key linking to the Categories table.
     + amount: The amount spent.
     + date: The date of the expense.
   * **Relationships**:
     + Each expense belongs to one user and one category.
3. **Categories**
   * **Purpose**: Defines different categories for classifying expenses.
   * **Key Fields**:
     + id: Primary key for uniquely identifying a category.
     + user\_id: Foreign key linking to the Users table.
     + name: The name of the category.
   * **Relationships**:
     + One user can have multiple categories.
     + One category can have multiple expenses.
4. **Budgets**
   * **Purpose**: Tracks budget allocations for different categories over specific periods.
   * **Key Fields**:
     + id: Primary key for uniquely identifying a budget.
     + user\_id: Foreign key linking to the Users table.
     + category\_id: Foreign key linking to the Categories table.
     + amount: The budgeted amount.
     + start\_date, end\_date: The period of the budget.
   * **Relationships**:
     + One user can have multiple budgets.
     + Each budget is associated with one category.
5. **Reports**
   * **Purpose**: Stores generated reports summarizing expense data.
   * **Key Fields**:
     + id: Primary key for uniquely identifying a report.
     + user\_id: Foreign key linking to the Users table.
     + type: The type of report (e.g., summary, category-specific).
     + data: The report data stored in JSONB format.
   * **Relationships**:
     + One user can have multiple reports.

**Implement the Database using MongoDB**

The MongoDB database is implemented with the following collections and structures:

**Collections and Documents Structure**

1. **Users Collection**
2. **Expenses Collection**
3. **Categories Collection**
4. **Budgets Collection**
5. **Reports Collection**

**MongoDB Schema Design**

**1. Users Collection**

* **Purpose**: Stores user information for authentication and authorization.
* **Fields**:
  + \_id: ObjectId, primary key for uniquely identifying a user.
  + username: String, unique identifier for the user.
  + email: String, unique identifier for user login.
  + passwordHash: String, securely stores user passwords.
  + createdAt: Date, timestamp of user creation.
  + updatedAt: Date, timestamp of last user update.

**. Expenses Collection**

* **Purpose**: Records individual expense transactions.
* **Fields**:
  + \_id: ObjectId, primary key for uniquely identifying an expense.
  + userId: ObjectId, references the Users collection.
  + categoryId: ObjectId, references the Categories collection.
  + amount: Number, the amount spent.
  + description: String, details about the expense.
  + date: Date, the date of the expense.
  + createdAt: Date, timestamp of expense creation.
  + updatedAt: Date, timestamp of last expense update.

3. **. Categories Collection**

* **Purpose**: Defines different categories for classifying expenses.
* **Fields**:
  + \_id: ObjectId, primary key for uniquely identifying a category.
  + userId: ObjectId, references the Users collection.
  + name: String, the name of the category.
  + createdAt: Date, timestamp of category creation.
  + updatedAt: Date, timestamp of last category update.

**Budgets Collection**

* **Purpose**: Tracks budget allocations for different categories over specific periods.
* **Fields**:
  + \_id: ObjectId, primary key for uniquely identifying a budget.
  + userId: ObjectId, references the Users collection.
  + categoryId: ObjectId, references the Categories collection.
  + amount: Number, the budgeted amount.
  + startDate: Date, the start date of the budget.
  + endDate: Date, the end date of the budget.
  + createdAt: Date, timestamp of budget creation.
  + updatedAt: Date, timestamp of last budget update

**5. Reports Collection**

* **Purpose**: Stores generated reports summarizing expense data.
* **Fields**:
  + \_id: ObjectId, primary key for uniquely identifying a report.
  + userId: ObjectId, references the Users collection.
  + type: String, the type of report (e.g., summary, category-specific).
  + generatedAt: Date, timestamp of report generation.
  + data: Object, the report data stored in JSON format.

**Integration with Backend**

* Database connection: Give Screenshot of Database connection done using Mongoose

