**Full Stack Development with MERN**

**Database Design and Development Report**

| Date | 10 JUL 2024 |
| --- | --- |
| Team ID | SWTID1720004076 |
| Project Name | Wise Wallet: Your Budgeting Partner |
| Maximum Marks |  |

### 

### **Objective**

The objective of this document is to outline the database design and implementation details for the Wise Wallet: Your Budgeting Partner project, including schema design and integration with MongoDB using Mongoose.

### 

### 

### **Technologies Used**

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

### 

### 

### **Design the Database Schema**

The database schema for Wise Wallet is designed to handle users, categories, and transactions efficiently.

1. **Users**
   * **Attributes**:
     + username: User’s username
     + email: User’s email address
     + password: Encrypted password
2. **Categories**
   * **Attributes**:
     + user: Reference to the user (ObjectId from users)
     + name: Name of the category
     + type: description of the category

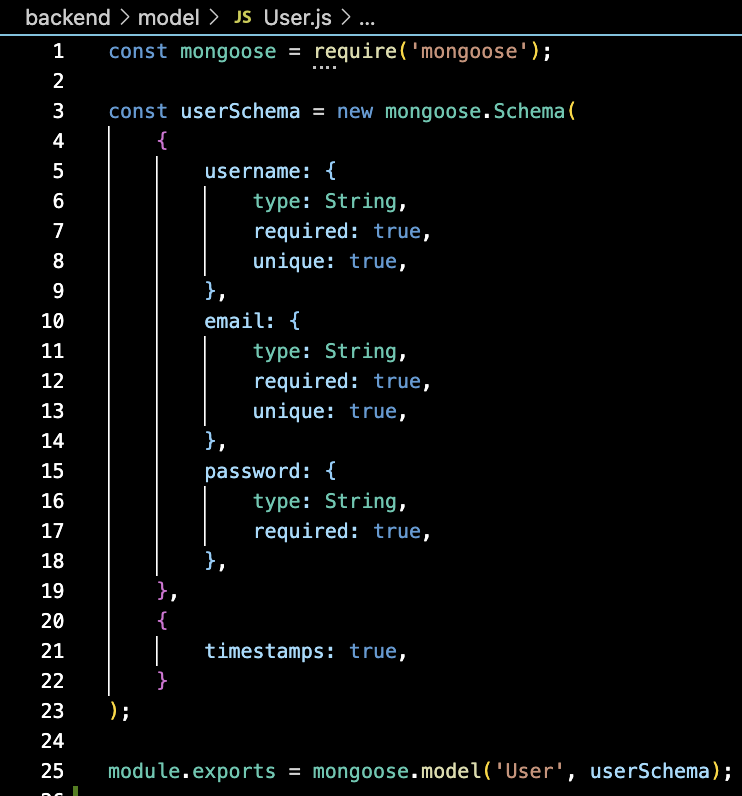
**3. Transactions**

* + **Attributes**:
    - user: Reference to the user (ObjectId from users)
    - type: type of the transaction (income,expense)
    - category: Reference to the category (ObjectId from categories)
    - amount: Amount of the transaction
    - date: Date of the transaction
    - description: Description of the transaction

### 

### **Implement the Database using MongoDB**

The MongoDB database for Wise Wallet is implemented with the following collections and schema structures:

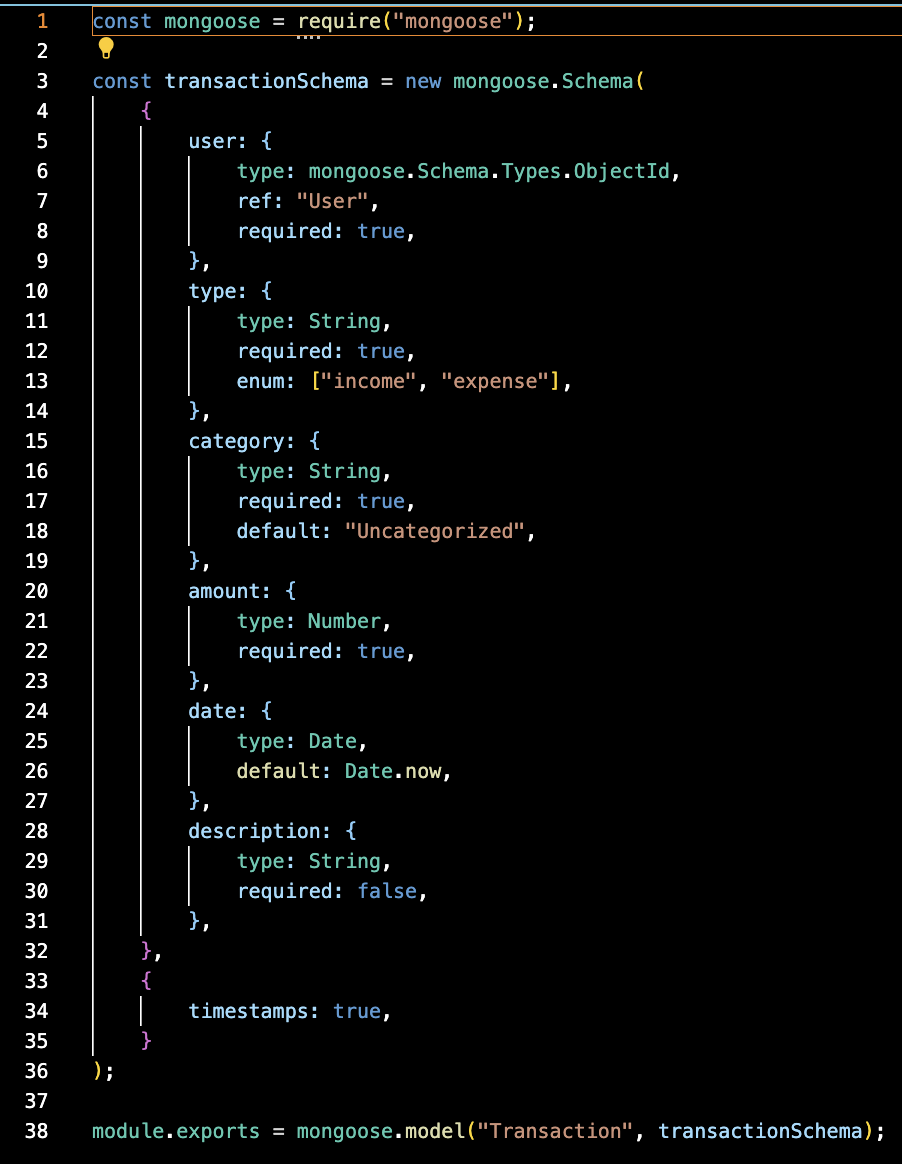
1. **Collection: users**

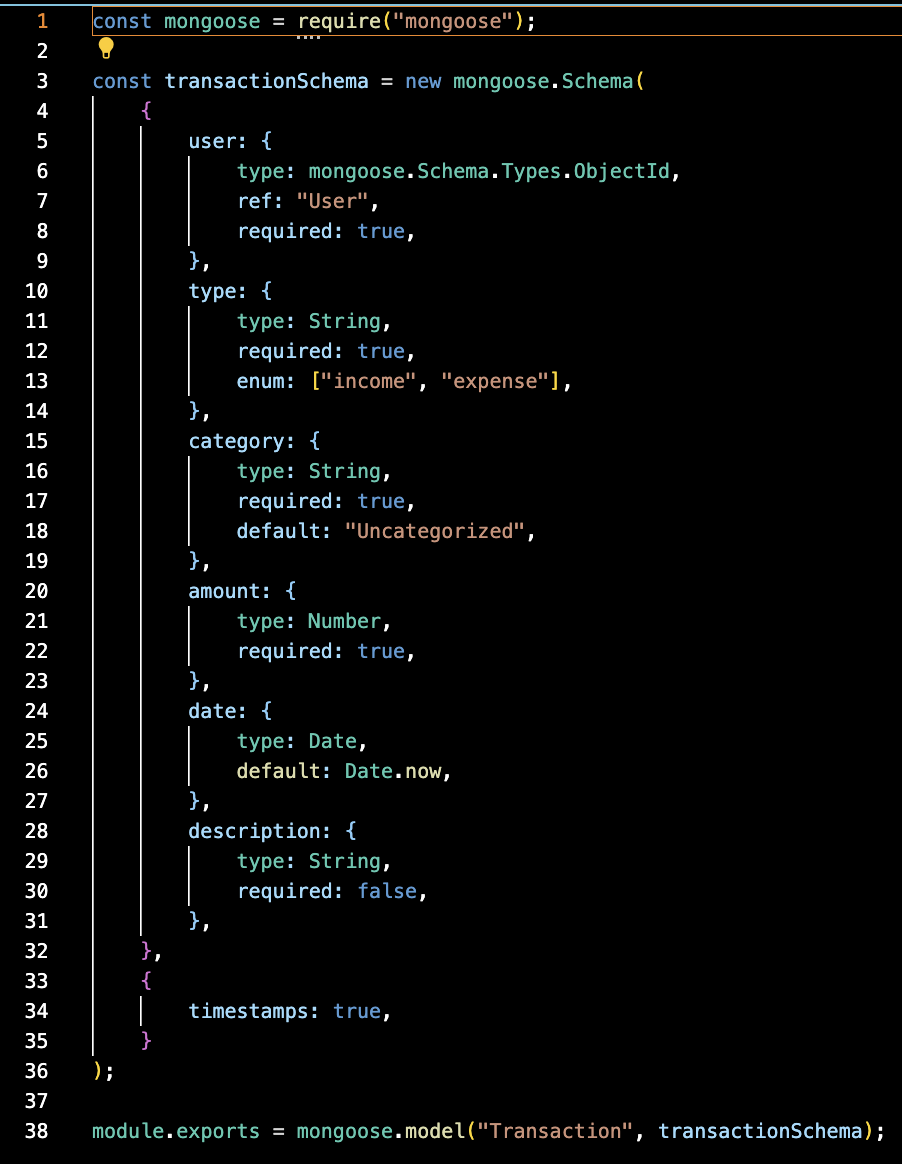
**Schema**:

**2. Collection: categories**

**Schema**:

**3. Collection: transactions**

**Schema:**

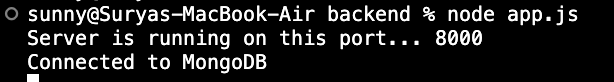


### **Integration with Backend**

**Database Connection:** The backend connects to MongoDB using Mongoose.

Here’s how to establish the connection, which can be found in the backend/app.js file:





**Key Interactions**:

* **User Management**: Perform CRUD operations on the users collection to manage user data.
* **Category Management**: Handle CRUD operations on the categories collection for managing categories used in transactions.
* **Transaction Management**: Execute CRUD operations on the transactions collection to manage financial records, linking them to users and categories.