Full Stack Development with MERN

Database Design and Development Report

Date	17-07-2024
Team ID	SWTID1720199985
Project Name	Grocery Webapp
Maximum Marks	5 Marks

Project Title: Grocery Webapp

Date: 17-07-2024

Prepared by: Lohith Pattubala & Patchigolla Moulika

Objective

The objective of this report is to outline the database design and implementation details for the Grocery Webapp 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:

1. Users

- Attributes: {

"_id": { "$oid": },

"name": " ",

"email": "",

"password": ",

"isAdmin":,

"createdAt": { "$date": { "$numberLong": "" }

}, "updatedAt": { "$date": { "$numberLong": "" }

},

"__v": { "$numberInt": "" }

}

2. Products

- Attributes: {

"_id": { "$oid": "" },

"name": "",

"slug": "",

"image": "",

"images": "",

"brand": "",

"category": “",

"description": " ",

"price": "",

"countInStock": "",

"rating": "",

"numReviews": ""

}

3. Order

- Attributes: {

"_id": { "$oid": "" },

"orderItems": [

{

"slug": "",

"name": "",

"quantity": { "$numberInt": "" },

```
    "image": "",
    "price": { "$numberInt": "" },
    "product": { "$oid": ""  },
    "_id": { "$oid": "”    }
]
 "shippingAddress": {
  "fullName": "",
  "address": "",
  "city": "",
  "postalCode": "",
  "country": ""
 },
 "paymentMethod": "",
 "itemsPrice": { "$numberInt": "" },
 "shippingPrice": { "$numberInt": "" },
 "taxPrice": { "$numberInt": "" },
 "totalPrice": { "$numberInt": "" },
 "user": { "$oid": "" },
 "isPaid": ,
 "isDelivered": ,
 "createdAt": {
  "$date": { "$numberLong": "" }
 },
 "updatedAt": {
  "$date": { "$numberLong": "" }
 },
 "__v": { "$numberInt": "" }
}
```


Implement the Database using MongoDB

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

Database Name: [Schooldb]

1. **Collection: users**

- Schema:

```
{
  name: { type: String, required: true },
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
  resetToken: { type: String },
  isAdmin: { type: Boolean, default: false, required: true },
 },
 {
  timestamps: true,
 }
```

2. **Products**

- Schema:

```
name: { type: String, required: true, unique: true },
  slug: { type: String, required: true, unique: true },
  image: { type: String, required: true },
  images: [String],
  brand: { type: String, required: true },
  category: { type: String, required: true },
  description: { type: String, required: true },
  price: { type: Number, required: true },
  countInStock: { type: Number, required: true },
  rating: { type: Number, required: true },
  numReviews: { type: Number, required: true },
  reviews: [reviewSchema],
```

```
  },
  {
    timestamps: true,
  }
```

3. **Order**

- Schema:

```
{
  orderItems: [
    {
      slug: { type: String, required: true },
      name: { type: String, required: true },
      quantity: { type: Number, required: true },
      image: { type: String, required: true },
      price: { type: Number, required: true },
      product: {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'Product',
        required: true,
      },
    },
  ],
  shippingAddress: {
    fullName: { type: String, required: true },
    address: { type: String, required: true },
    city: { type: String, required: true },
    postalCode: { type: String, required: true },
    country: { type: String, required: true },
    location: {
      lat: Number,
      lng: Number,
```

```
    address: String,
    name: String,
    vicinity: String,
    googleAddressId: String,
   },
  },
  paymentMethod: { type: String, required: true },
  paymentResult: {
   id: String,
   status: String,
   update_time: String,
   email_address: String,
  },
  itemsPrice: { type: Number, required: true },
  shippingPrice: { type: Number, required: true },
  taxPrice: { type: Number, required: true },
  totalPrice: { type: Number, required: true },
  user: { type: mongoose.Schema.Types.ObjectId, ref: 'User', required: true },
  isPaid: { type: Boolean, default: false },
  paidAt: { type: Date },
  isDelivered: { type: Boolean, default: false },
  deliveredAt: { type: Date },
},
{
  timestamps: true,
}
```

Integration with Backend

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

```
dotenv.config();

mongoose
  .connect(process.env.MONGODB_URI)
  .then(() => {
    console.log("connected to db");
  })
  .catch((err) => {
    console.log(err.message);
  });

const app = express();
```

```
MONGODB_URI=mongodb+srv://root:root@schooldb.foeyhgj.mongodb.net/?retryWrites=true&w=majority&appName=schooldb
JWT_SECRET=somethingsecret
```

- The backend APIs interact with MongoDB using Mongoose ODM Key interactions include:
    - User Management: CRUD operations for admin.
    - Product Management: CRUD operations for products, with admin authentication.
    - review Management: CR operations for reviews associated with products.