**Full Stack Development with MERN**

**Database Design and Development Report**

|  |  |
| --- | --- |
| Date | 10/07/2024 |
| Team ID | SWTID1720194751 |
| Project Name | Project – Flight Booking App(Fly High) |
| Maximum Marks | 4 |

**Project Title**: Flight Booking App(Fly High)

**Date**: 10/07/2024

**Prepared by**: SWTID1720194751

**Objective**

The objective of this report is to outline the database design and implementation details for the Fly HIgh 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: ObjectId
     + email: String
     + Full\_name: String
     + Gender: String
     + Age: Number
     + Passport\_Number: String
     + Adhaar\_Number: String
     + Contact\_Number: String
     + Password: String
     + month: String
     + day: String
     + year: String
     + createdAt: Date
     + updatedAt: Date
2. **Flights**
   * **Attributes**:
     + \_id: ObjectId
     + id: String
     + company: String
     + type: String
     + rows: Number
     + cols: Number
     + gap\_one: Number
     + gap\_two: Number
     + seats\_avl: Number
     + start: String
     + end: String
     + day: String
     + month: String
     + year: String
     + time: String
     + price: String
     + seats: Array of Objects
     + createdAt: Date
     + updatedAt: Date

**Implement the Database using MongoDB**

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

**Database Name**: TRAVEL\_HUB

1. Collection: users

- Schema:

```

{

"\_id": "ObjectId",

"email": "String",

"Full\_name": "String",

"Gender": "String",

"Age": "Number",

"Passport\_Number": "String",

"Adhaar\_Number": "String",

"Contact\_Number": "String",

"Password": "String",

"month": "String",

"day": "String",

"year": "String",

"createdAt": "Date",

"updatedAt": "Date"

}

```

2. Collection: flights

- Schema:

```

{

"\_id": "ObjectId",

"id": "String",

"company": "String",

"type": "String",

"rows": "Number",

"cols": "Number",

"gap\_one": "Number",

"gap\_two": "Number",

"seats\_avl": "Number",

"start": "String",

"end": "String",

"day": "String",

"month": "String",

"year": "String",

"time": "String",

"price": "String",

"seats": [

{

"seat": "String"

}

],

"createdAt": "Date",

"updatedAt": "Date"

}

```

**Integration with Backend**

* **Database connection**: Mongoose is used to connect to the MongoDB database.  
  Code for connection:-  
  -Schema  
  “””

const mongoose = require('mongoose');

mongoose.connect('mongodb+srv://Cluster15452:QkJGS25gWXBc@cluster15452.gkfkzll.mongodb.net/TRAVEL\_HUB?retryWrites=true&w=majority', {

useNewUrlParser: true,

useUnifiedTopology: true

}).then(() => {

console.log('Connected to MongoDB');

}).catch((error) => {

console.error('Connection error', error);

});  
“””

* The backend APIs interact with MongoDB using Mongoose ODM Key interactions include:
  + User Management: CRUD operations for users.

Example: Add a new user if the email doesn't exist.

app.post('/get\_user\_details', async function(req, res) {

try {

await mongoose.connect('mongodb+srv://Cluster15452:QkJGS25gWXBc@cluster15452.gkfkzll.mongodb.net/TRAVEL\_HUB?retryWrites=true&w=majority');

const user = await User.findOne({ email: req.body.email });

if (!user) {

const newUser = new User(req.body);

await newUser.save();

res.status(201).json({ success: 'User created successfully' });

} else {

res.status(409).json({ success: 'Email already exists' });

}

} catch (e) {

res.status(500).json({ success: 'Unsuccessful', error: e.message });

}

});

* Flight Management: CRUD operations for posts, with user authentication.

Example: Add a new flight with seat details

app.post('/add\_flight', async function(req, res) {

try {

await mongoose.connect('mongodb+srv://Cluster15452:QkJGS25gWXBc@cluster15452.gkfkzll.mongodb.net/TRAVEL\_HUB?retryWrites=true&w=majority');

const newFlight = new Flight(req.body);

await newFlight.save();

res.status(201).json({ success: 'Flight added successfully' });

} catch (e) {

res.status(500).json({ success: 'Unsuccessful', error: e.message });

}

});