**Full Stack Development with MERN**

**Database Design and Development Report**

| Date | 08/07/2024 |
| --- | --- |
| Team ID | SWTID1720278137 |
| Project Name | Online Learning Platform |
| Maximum Marks |  |

**Project Title:** **Online Learning Platform (EduVerse)**  
**Date: 08/07/2024**  
**Prepared by:** SWTID1720278137

**Objective**

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

* fullname: {

type: String,

required: true,

}

* email: {

type: String,

required: true,

unique: true,

}

* password: {

type: String,

required: true,

}

**2. Courses**

- Attributes:

* id: Object\_Id
* name: String
* price: Number
* category: String
* image: String
* title: String

**Implement the Database using MongoDB**

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

Database Name: EduVerse

1. Collection: users

- Schema:

```

{

\_id: ObjectId,

fullname: String,

email: String,

password: String,

createdAt: Date,

updatedAt: Date

}

```

2. Collection: courses

- Schema:

```

{

\_id: ObjectId,

name: String,

title: String,

price: Number,

category: String,

image: String,

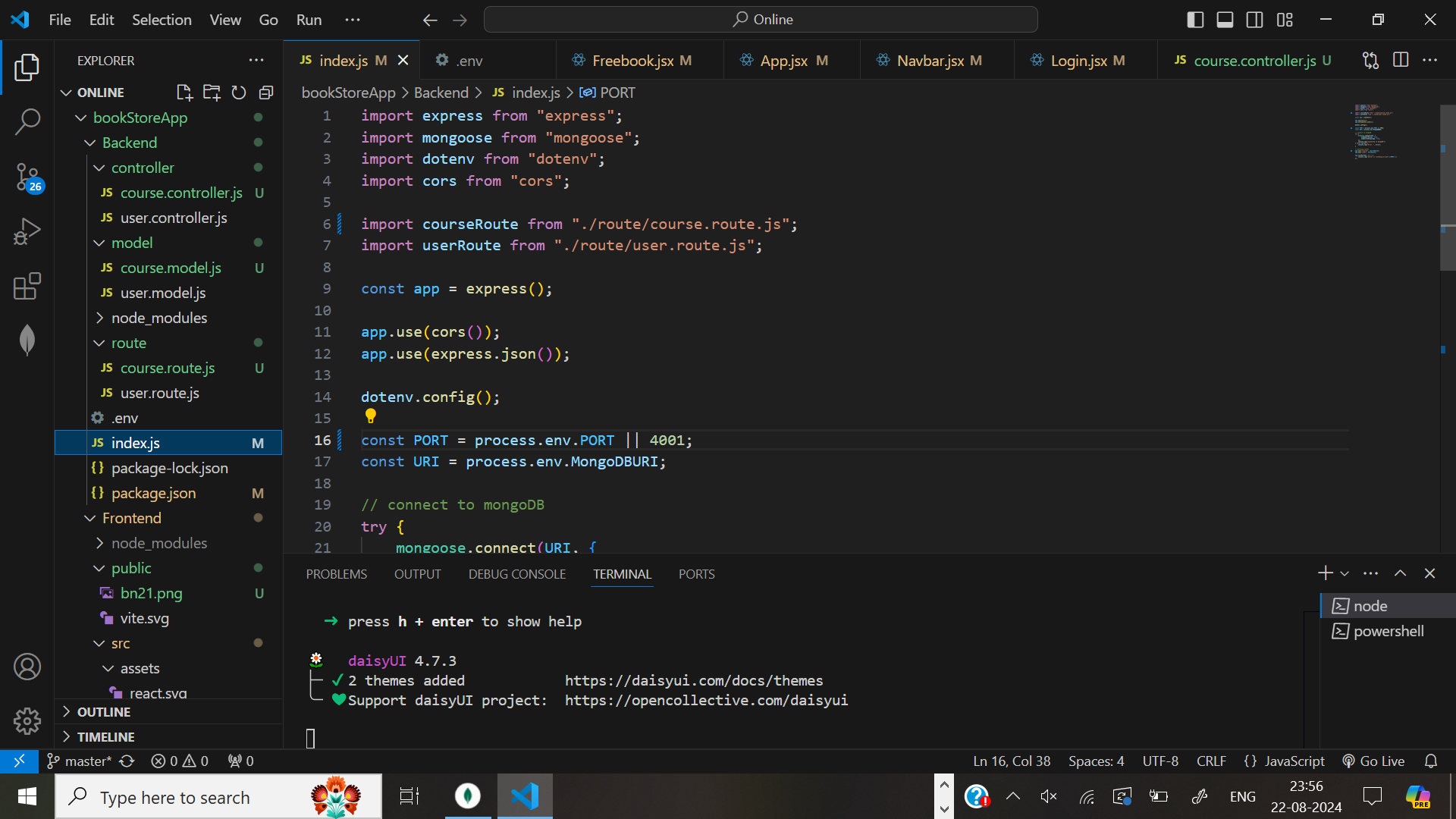
createdAt: Date,

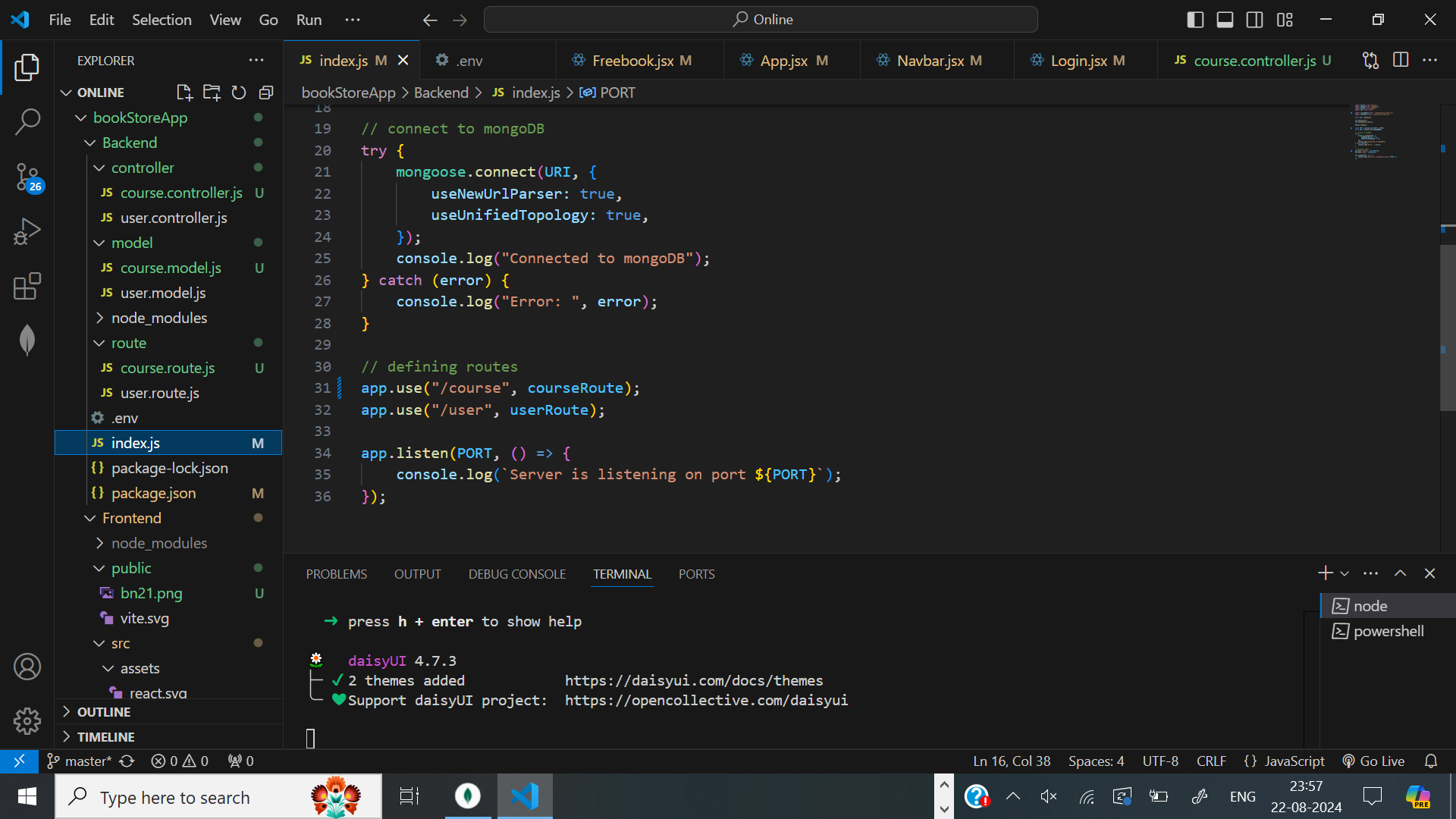
updatedAt: Date

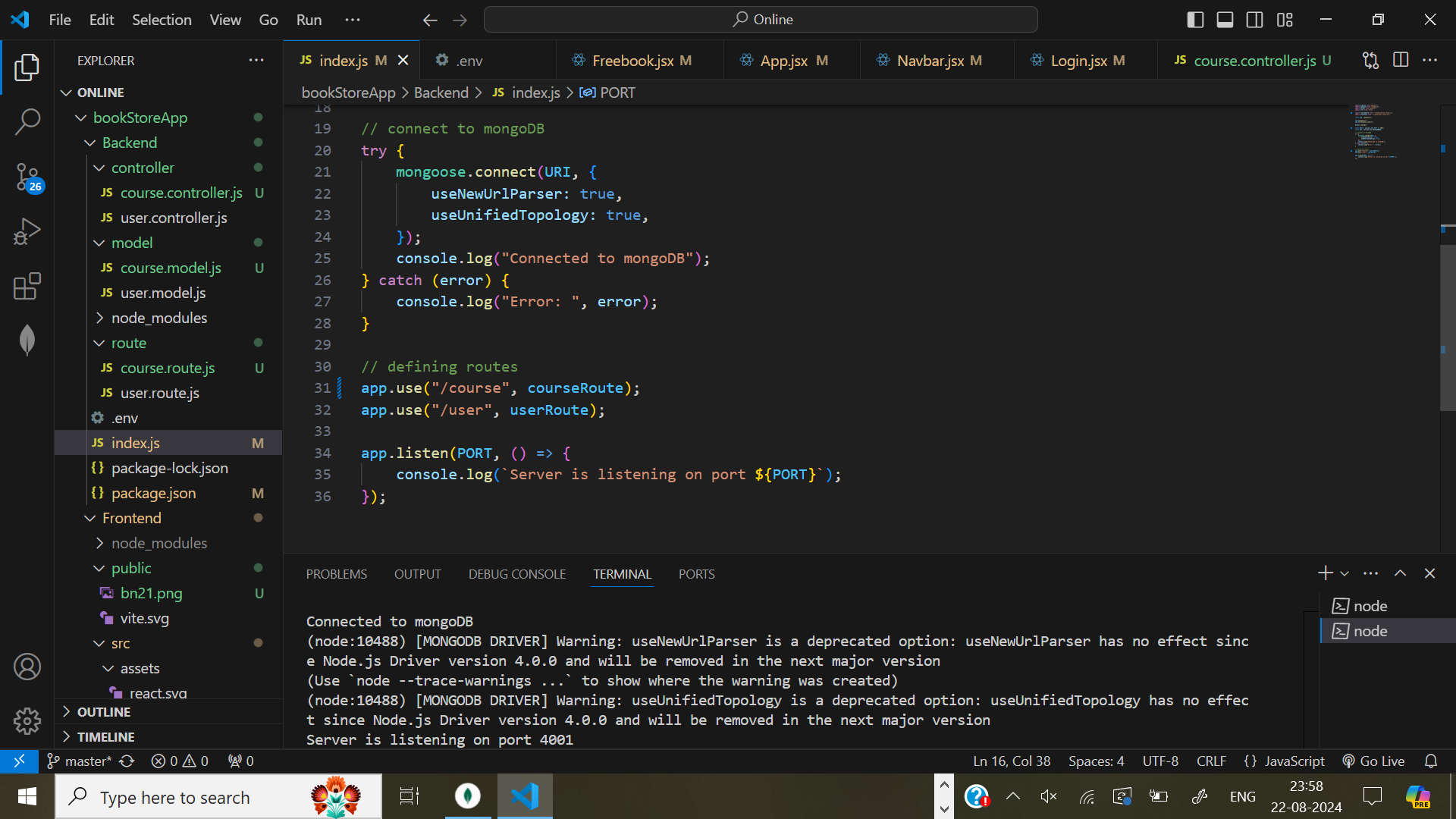
}

```

**Integration with Backend**







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

**Ex**: Adding new user if email doesn’t exist

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 });

}

});

* + Course Management: CRUD operations for courses

**Ex**: Adding a new course

const newCourse = new Course(req.body);

await newCourse.save();

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

} catch (e) {

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

}

});