**FULL STACK DEVELOPMENT**

**LAB TASK – 7**

**THEJA N**

**21BCS213**

**Step 1: Set Up MongoDB Atlas**

1. **Create a MongoDB Atlas account:**
   * If you don't already have an account, visit [MongoDB Atlas](https://www.mongodb.com/cloud/atlas/register) and sign up.
2. **Create a Cluster:**
   * Once logged in, create a new cluster. MongoDB Atlas will provide instructions for setting up your cluster.
   * Choose the free-tier (M0 Sandbox) cluster for learning purposes.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. **Create a Database User:**
   * In the "Database Access" section of the Atlas dashboard, create a new database user with a username and password. Make sure to save the credentials securely.

A screenshot of a computer

Description automatically generated

1. **Allow IP Access:**
   * Go to "Network Access" in your Atlas dashboard and click "Add IP Address."
   * Choose "Allow access from anywhere" (0.0.0.0/0) if you're working locally, or add your specific IP address.

A screenshot of a computer

Description automatically generated

1. **Get the Connection String:**
   * In your cluster dashboard, click "Connect" and choose "Connect your application."
   * Copy the provided connection string, which will look something like this:

A screenshot of a computer

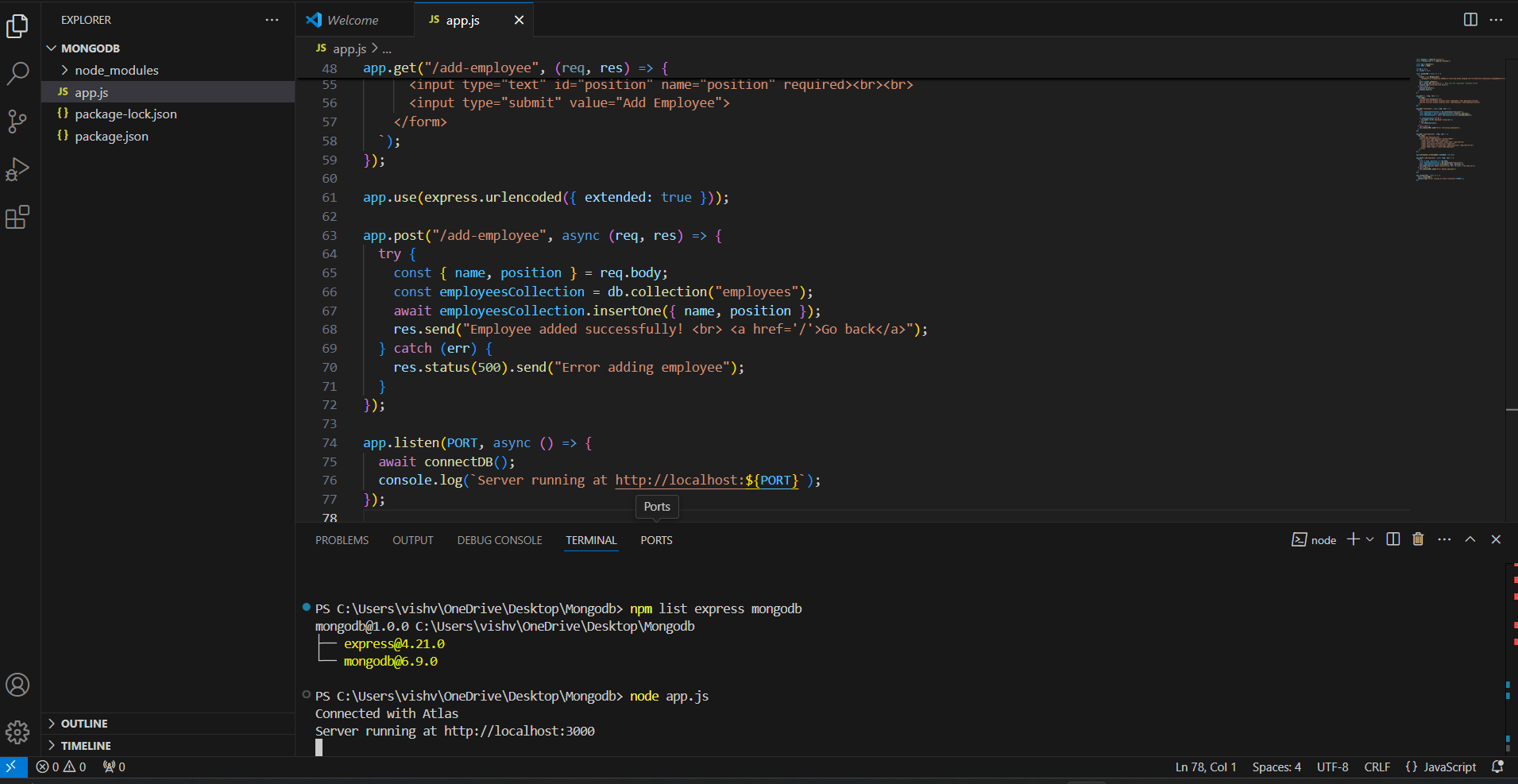
Description automatically generated

Replace <username>, <password>, and myFirstDatabase with your actual database name and credentials.

**Step 2: Node.js Application**

Now, update your Node.js application to connect to MongoDB Atlas.

* + 1. Install the required MongoDB package



* + 1. Code

1. const express = require("express");
2. const { MongoClient } = require("mongodb");
3. const app = express();
4. const PORT = 3000;
5. let db = null;
6. let client = null;
7. const connectDB = async () => {
8. try {
9. client = new MongoClient(
10. "mongodb+srv://cayyanraj:2305@full-stack-lab.ptod1.mongodb.net/?retryWrites=true&w=majority&appName=full-stack-lab"
11. );
12. await client.connect();
13. db = client.db("employees"); // Make sure the 'employees' database exists
14. console.log("Connected with Atlas");
15. } catch (err) {
16. console.error(err);
17. process.exit(1);
18. }
19. };
20. app.get("/", (req, res) => {
21. res.send(`
22. <h1>Employee Management</h1>
23. <button onclick="window.location.href='/employees'">Get Employees</button>
24. <button onclick="window.location.href='/add-employee'">Add Employee</button>
25. `);
26. });
27. app.get("/employees", async (req, res) => {
28. try {
29. const employeesCollection = db.collection("employees");
30. const employees = await employeesCollection.find({}).toArray();
31. const employeesCount = await employeesCollection.countDocuments();
33. if (employeesCount === 0) {
34. res.send(`<h2>No employees found</h2>`);
35. } else {
36. res.send(employees);
37. }
38. } catch (err) {
39. res.status(500).send("Error retrieving employees");
40. }
41. });
42. app.get("/add-employee", (req, res) => {
43. res.send(`
44. <h2>Add New Employee</h2>
45. <form action="/add-employee" method="POST">
46. <label for="name">Name:</label><br>
47. <input type="text" id="name" name="name" required><br>
48. <label for="position">Position:</label><br>
49. <input type="text" id="position" name="position" required><br><br>
50. <input type="submit" value="Add Employee">
51. </form>
52. `);
53. });
54. app.use(express.urlencoded({ extended: true }));
55. app.post("/add-employee", async (req, res) => {
56. try {
57. const { name, position } = req.body;
58. const employeesCollection = db.collection("employees");
59. await employeesCollection.insertOne({ name, position });
60. res.send("Employee added successfully! <br> <a href='/'>Go back</a>");
61. } catch (err) {
62. res.status(500).send("Error adding employee");
63. }
64. });
65. app.listen(PORT, async () => {
66. await connectDB();
67. console.log(`Server running at http://localhost:${PORT}`);
68. });

**Step 3: Run Your Application**

1. Make sure MongoDB Atlas is set up correctly, and your IP address is whitelisted in MongoDB Atlas.
2. Start your Node.js server by running:

Your app will now connect to MongoDB Atlas instead of a local MongoDB instance. You can visit http://localhost:3000 to test it, and use Postman or curl to interact with the /add-employee and /employees endpoints.

**Output:**

A white background with black text

Description automatically generated

**Add Employee Example:**

A white background with black text

Description automatically generated

A white background with black dots

Description automatically generated

**Get Employee:**

A screenshot of a computer

Description automatically generated