**Portfolio (Week-8)**

This week, we focused on learning how to perform CRUD (Create, Read, Update, Delete) operations using MongoDB in a Node.js application. The tutorial guided us through integrating MongoDB commands into our application using Visual Studio Code. Below is a detailed account of what I learned and implemented during this week’s lab:

**Setting Up the Environment**

1. **Project Initialization**
   * Created a new folder named "MongoWeek" to store project files.
   * Opened the terminal, navigated to the "MongoWeek" directory, and initialized a new Node.js project:

A black rectangular object with white text

Description automatically generated

This command generated a package.json file.

1. **Installing Dependencies**

* Installed Mongoose, a popular library for MongoDB interactions in Node.js:

A screenshot of a computer

Description automatically generated

A screenshot of a computer program

Description automatically generated

1. **Creating the Main File**

* Created a new file named index.js for all the coding tasks.

A screenshot of a computer

Description automatically generated

To connect to MongoDB, I needed to install Mongoose. I ran the following commands in the terminal:

A screenshot of a computer

Description automatically generated

In index.js, I added code to connect to MongoDB:

A screenshot of a computer

Description automatically generated

This code connects to a MongoDB database named "Week8". It also handles connection errors and logs when the connection is successful.

**Implementing CRUD Operations**

**1. Creating a Schema**

I created a schema for storing person data:

A screenshot of a computer program

Description automatically generated

Then, I added code to create and save a single document:

The doc1.save() method saves the document to the database, and .then is used to handle successful insertion.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer program

Description automatically generated

To add multiple documents at once, I used this code:

A screenshot of a computer program

Description automatically generated

The insertMany method is used here to insert multiple documents into the collection.

To retrieve all documents from the collection without filtering, I added this code:

A screenshot of a computer program

Description automatically generated

For filtering documents based on criteria (e.g., gender), I wrote:

A screenshot of a computer program

Description automatically generated

To delete records where age is greater than 25, I used this code:

A screenshot of a computer program

Description automatically generated

Finally, to update all female records' salaries to 5555, I wrote:

A screenshot of a computer program

Description automatically generated

This week's lab provided me with hands-on experience in using MongoDB with Node.js for CRUD operations. I learned how to connect to a database using Mongoose, create schemas and models, insert single and multiple documents, fetch data with filtering options, delete records based on criteria, and update existing documents.The tutorial emphasized understanding how each function works within Mongoose and how they interact with MongoDB. This knowledge will be valuable for developing applications that require database interactions in real-world scenarios.I will include all relevant code snippets along with explanations in my portfolio for future reference.

**Reflections and Learning Outcomes**

This week’s lab provided hands-on experience in using MongoDB with Node.js for CRUD operations. Key takeaways include:

1. **Connecting to MongoDB**: I learned how to use Mongoose to establish a connection and handle errors.
2. **Schemas and Models**: Understanding how to define schemas and create models helped in structuring data effectively.
3. **CRUD Operations**: I gained practical knowledge of creating, reading, updating, and deleting data using Mongoose methods.
4. **Error Handling**: The lab emphasized proper error handling to ensure robust application behavior.

**Challenges and Solutions**

1. **Connection Errors**:
   * **Issue:** Encountered an error while connecting to MongoDB.
   * **Solution:** Verified the database URL and ensured MongoDB service was running locally.
2. **Schema Validation**:
   * **Issue:** Faced issues when inserting documents with missing fields.
   * **Solution:** Updated the schema to handle optional fields appropriately.
3. **Bulk Insert Errors**:
   * **Issue:** Encountered errors during bulk inserts.
   * **Solution:** Checked for duplicate keys and ensured data structure consistency.

By addressing these challenges, I enhanced my understanding of database interactions and strengthened my problem-solving skills.

**Conclusion**

This week’s experience of working with MongoDB and Node.js has equipped me with the foundational skills needed for database-driven application development. The knowledge of CRUD operations and Mongoose will be invaluable in future projects that require robust and scalable database solutions.