Week 1 Lab: Environment Setup, Git Workflows & Hello MongoDB

Objective: Set up core development tools, learn basic Git workflows, and create a simple NodeJS script that connects to MongoDB.

Lab Overview

Tools to Install

1. **VSCode**: Code editor.
2. **NodeJS & npm**: JavaScript runtime and package manager.
3. **MongoDB**: Database (local or cloud instance).
4. **Git**: Version control system.
5. **MongoDB Compass**: GUI for MongoDB (optional).

Deliverables

1. Documented installation steps for each tool.
2. A NodeJS script that connects to MongoDB and inserts/reads a document.
3. A GitHub repository with branches (`main` and `feature/setup`).

Lab Procedures

Step 1: Install Development Tools

1. **Install VSCode**
   - Download from code.visualstudio.com.
   - Install recommended extensions: **MongoDB for VSCode**.
2. **Install NodeJS & npm**
   - Download the LTS version from nodejs.org.
     Verify installation:

     ```
     node -v
     npm -v
     ```

3. **Install MongoDB**
   - Follow MongoDB Community Server installation guide.

- Start MongoDB service

4. **Install Git**
   - Download from [git-scm.com](git-scm.com).
     Configure Git username/email:

```
git config --global user.name "Your Name"
git config --global user.email "your@email.com"
```

5. **Install MongoDB Compass (Optional)**
   - Download from MongoDB Compass.

Step 2: Git Basics & Repository Setup

1. **Create a GitHub Account**
   - https://education.github.com/pack

   Initialize a Git Repository

```
mkdir my-first-project
cd my-first-project
git init
```

4. **Create a `README.md` File**
   - Document your installation steps.

5. **Commit and Push to GitHub**

```
git add .
git commit -m "Initial commit: Setup documentation"
git branch -M main
git remote add origin https://github.com/your-username/my-first-project.git
git push -u origin main
```

Step 3: Create a "Hello MongoDB" NodeJS Script

1. **Initialize a NodeJS Project**

```
npm init -y
```

2. **Install MongoDB Driver**

```
npm install mongodb
```

3. Create **index.js**

```
const { MongoClient } = require('mongodb');

async function main() {
  // Replace <connection-string> with your MongoDB URI
  const uri = "mongodb://localhost:27017"
  const client = new MongoClient(uri);

  try {
    await client.connect();
    console.log("Connected to MongoDB!");

    const db = client.db("testDB");
    const collection = db.collection("users");

    // Insert a document
    await collection.insertOne({ name: "Alice", age: 25 });
    console.log("Document inserted!");

    // Query the document
    const result = await collection.findOne({ name: "Alice" });
    console.log("Query result:", result);
  } catch (err) {
    console.error("Error:", err);
  } finally {
    await client.close();
  }
}

main();
```

4. Run the Script

```
node index.js
```

5. Verify in MongoDB Compass
    - Connect to your MongoDB instance and check the testDB database.

Step 4: Push Code to GitHub

1. Commit Changes

```
git add .
git commit -m "Add NodeJS script and MongoDB connection"
```

2. Push to GitHub

```
git push origin feature/setup
```

Lab Questions

Answer these by completing the lab steps and observing results.

1. **Code Execution & Output**
    - After running your `index.js` script:
        - What **exact text** does the console display when the document is inserted?
        - What **_id value** is automatically assigned to the document?

2. **Modify and Observe**
    - Change the `name` field in `index.js` to your own name and the `age` to your birth year. Run the script again.
        - What **new _id** is generated for this document?
        - What error occurs if you forget to call **`await client.connect()`**?

3. **MongoDB Connection Failure**
    - **Intentionally break** the MongoDB connection string (e.g., change the port to `27018`).
        - What **error message** does NodeJS throw?
        - What is the **exact text** of the error code (e.g., `ECONNREFUSED`)?

4. **MongoDB Shell Query**
    - Use the **MongoDB shell** (not Compass) to:
        - List all documents in the `testDB.users` collection.
        - What command did you use? Paste the **full output**.
        -

5. **File System & Dependencies**
    - What is the **absolute path** to your project's `package-lock.json` file?
    - What **exact version** of the `mongodb` driver is installed?

6. **Troubleshooting Practice**
    - **Stop the MongoDB service** and run the script.
        - What error occurs?
        - What **command** restarts the service?

7. **GitHub Repository Structure**
    - On GitHub, navigate to your repository's.
        - What **timestamp** is listed for your last commit?

        - How many **files** are present in this branch?

8. **Performance Observation**
    - Time how long it takes for the script to print "Connected to MongoDB!".
        - What is the **duration** (in milliseconds)?
        - Does this time change if you run the script again? Why?

Submission Requirements

1. A **GitHub repository** with:
    - README.md documenting installation steps.
    - index.js script.

2. **Screenshots** of:
    - Successful MongoDB connection in NodeJS.
    - The document in MongoDB Compass/Atlas.