

Vince Jemuel A. Pelaez.

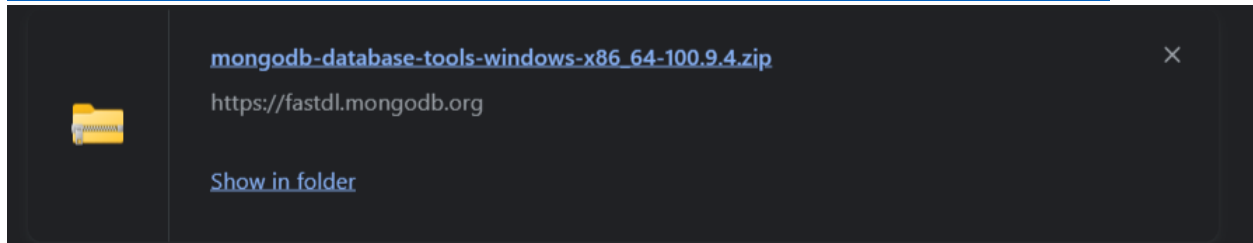
BSI/T – 3E

Individual Performance Test #1

Test Procedure:

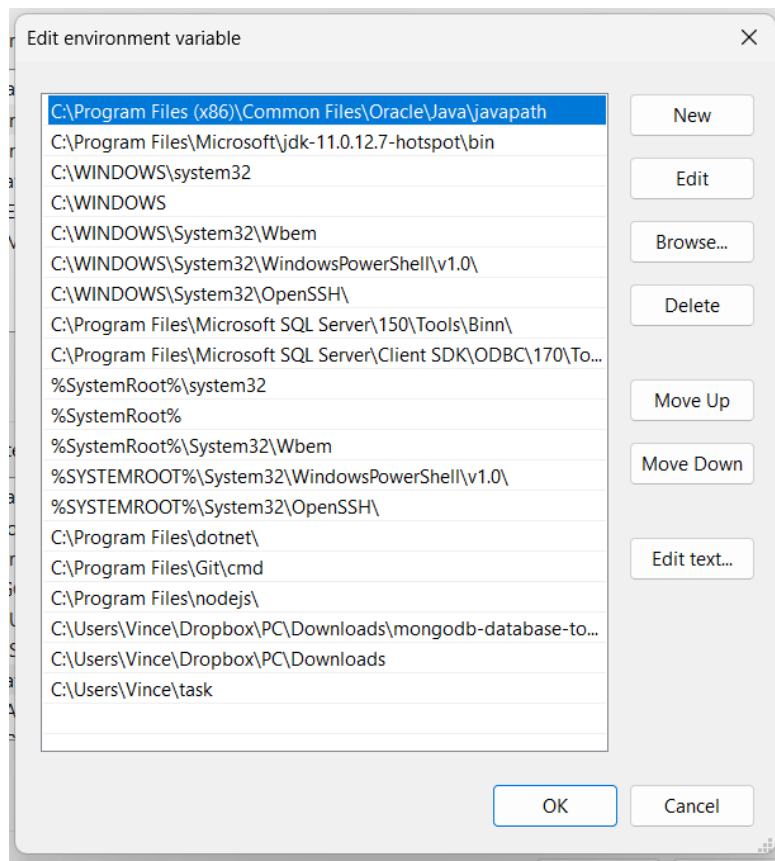
Step 1: Download and install the MongoDB Database tool:

https://fastdl.mongodb.org/tools/db/mongodb-database-tools-windows-x86_64-100.9.4.zip



Step 2: Follow the installation instructions for the MongoDB Database tool:

<https://www.mongodb.com/docs/database-tools/installation/installation-windows/>

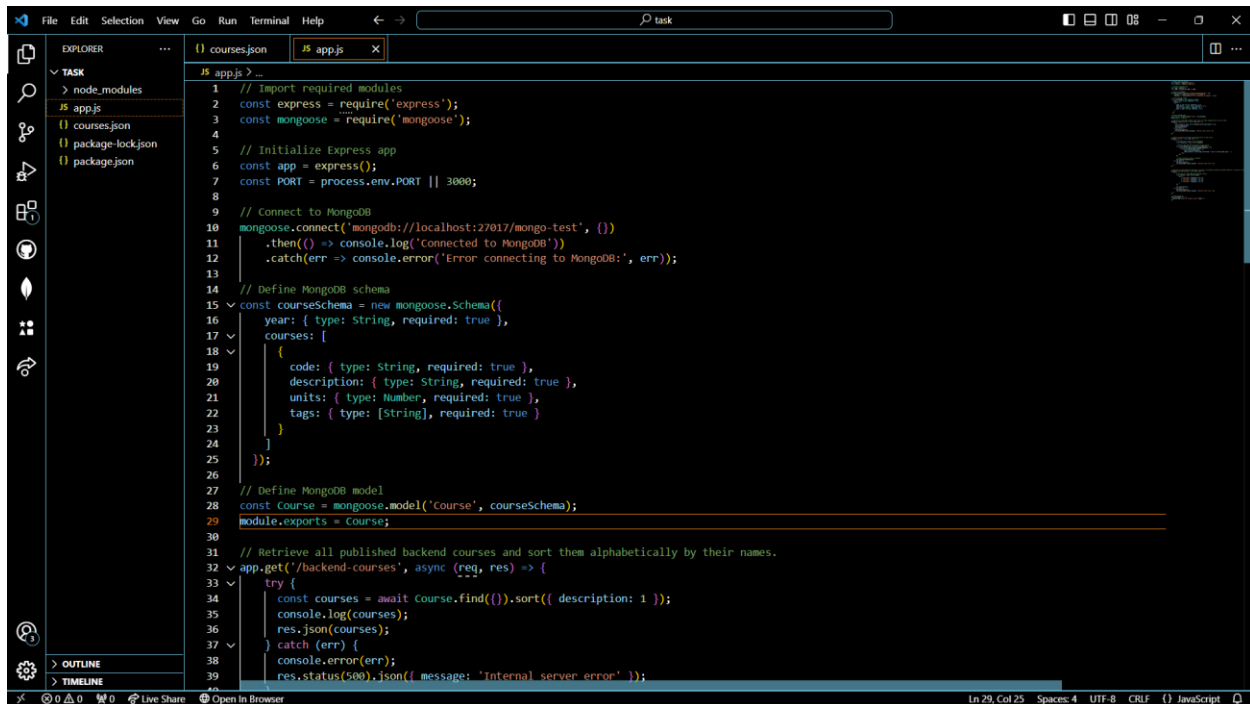


Step 3: Used the 'mongoimport' command to import the provided course data into MONGODB:

```
mongoimport --db mongo-test --collection courses --file courses.json --jsonArray
```

```
C:\Users\Vince\task>mongoimport --db mongo-test --collection courses --file courses.json --jsonArray
2024-02-23T23:24:41.357+0800    connected to: mongodb://localhost/
2024-02-23T23:24:41.420+0800    2 document(s) imported successfully. 0 document(s) failed to import.
```

Step 4: Created a Node.js application using Express to implement backend API endpoints.



```
1 // Import required modules
2 const express = require('express');
3 const mongoose = require('mongoose');
4
5 // Initialize Express app
6 const app = express();
7 const PORT = process.env.PORT || 3000;
8
9 // Connect to MongoDB
10 mongoose.connect('mongodb://localhost:27017/mongo-test', {})
11   .then(() => console.log('Connected to MongoDB'))
12   .catch(err => console.error('Error connecting to MongoDB:', err));
13
14 // Define MongoDB schema
15 const courseSchema = new mongoose.Schema({
16   year: { type: String, required: true },
17   courses: [
18     {
19       code: { type: String, required: true },
20       description: { type: String, required: true },
21       units: { type: Number, required: true },
22       tags: { type: [String], required: true }
23     }
24   ]
25 });
26
27 // Define MongoDB model
28 const Course = mongoose.model('Course', courseSchema);
29 module.exports = Course;
30
31 // Retrieve all published backend courses and sort them alphabetically by their names.
32 app.get('/backend-courses', async (req, res) => {
33   try {
34     const courses = await Course.find().sort({ description: 1 });
35     console.log(courses);
36     res.json(courses);
37   } catch (err) {
38     console.error(err);
39     res.status(500).json({ message: 'Internal server error' });
40   }
41 });
```

Step 5: API Endpoint Testing

1. Retrieve all published backend courses and sort them alphabetically by their names.
 - Using postman, sent a GET request to 'http://localhost:3000/backend-courses'.
2. Select and extract the name and specialization of each course.
 - Using postman, sent a GET request to 'http://localhost:3000/courses'.
3. Retrieve all published BSIS (Bachelor of Science in Information Systems) and BSIT (Bachelor of Science in Information Technology) courses from the curriculum.
 - Using postman, sent a GET request to 'http://localhost:3000/bsis-bsit-courses'.

Step 6: Ensured data validation at each step:

- Validated the structure of the course data during import.
- Implemented Mongoose schema validation to ensure correctness.

Step 7: Document the whole process

Challenges faced and solutions implemented.

1. Connection Error

Solution: Ensure that your MongoDB connection string is correct and that your MongoDB server is running.

2. Empty output in retrieving backend courses

Solution: Make sure that you stored first the data in mongodb using the 'mongoimport'

3. Data not sorting correctly

Solution: Double-check the field you're using for sorting. Ensure that it exists in the documents you're querying and that its values are correct.