Name: Ronit Santwani

Class : D15A

Roll no. : 48

Experiment – 7: MongoDB

1) **Aim:** To study CRUD operations in MongoDB 2)
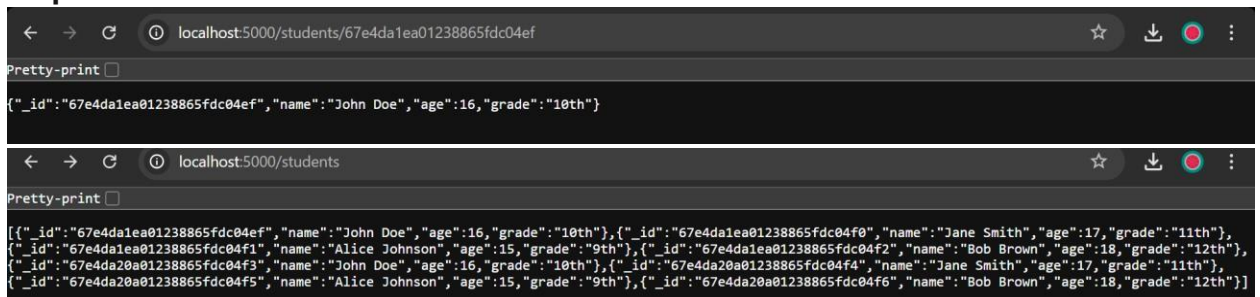**Problem Statement:**

A) Create a new database to storage student details of IT dept(Name, Roll no, class name) and perform the following on the database
   a) Insert one student details
   b) Insert at once multiple student details
   c) Display student for a particular class
   d) Display students of specific roll no in a class
   e) Change the roll no of a student
   f) Delete entries of particular student

B) Create a set of RESTful endpoints using Node.js, Express, and Mongoose for handling student data operations.

The endpoints should support:
- Retrieve a list of all students.
- Retrieve details of an individual student by ID.
- Add a new student to the database.
- Update details of an existing student by ID.
- Delete a student from the database by ID.

Connect the server to MongoDB using Mongoose, and store student data with attributes: name, age, and grade.

3) **Output:**

Insert Document

To collection sturts

VIEW {} ≡

```
1 ▾ [
2 ▾     {
3          "name": "John Doe",
4          "age": 16,
5          "grade": "10th"
6      },
7 ▾     {
8          "name": "Jane Smith",
9          "age": 17,
10         "grade": "11th"
11     },
12 ▾    {
13         "name": "Alice Johnson",
14         "age": 15
```

Cancel   Insert

+ Create Database

Q Search Namespaces

▸ inventory

▸ itdept

▾ **test**

    students

    sturts

    stus

test.sturts

STORAGE SIZE: 4KB   LOGICAL DATA SIZE: 0B   TOTAL DOCUMENTS: 0   INDEXES TOTAL SIZE: 4KB

Find   Indexes   Schema Anti-Patterns ⓪   Aggregation   Search Indexes

Generate queries from natural language in Compass☐   INSERT DOCUMENT
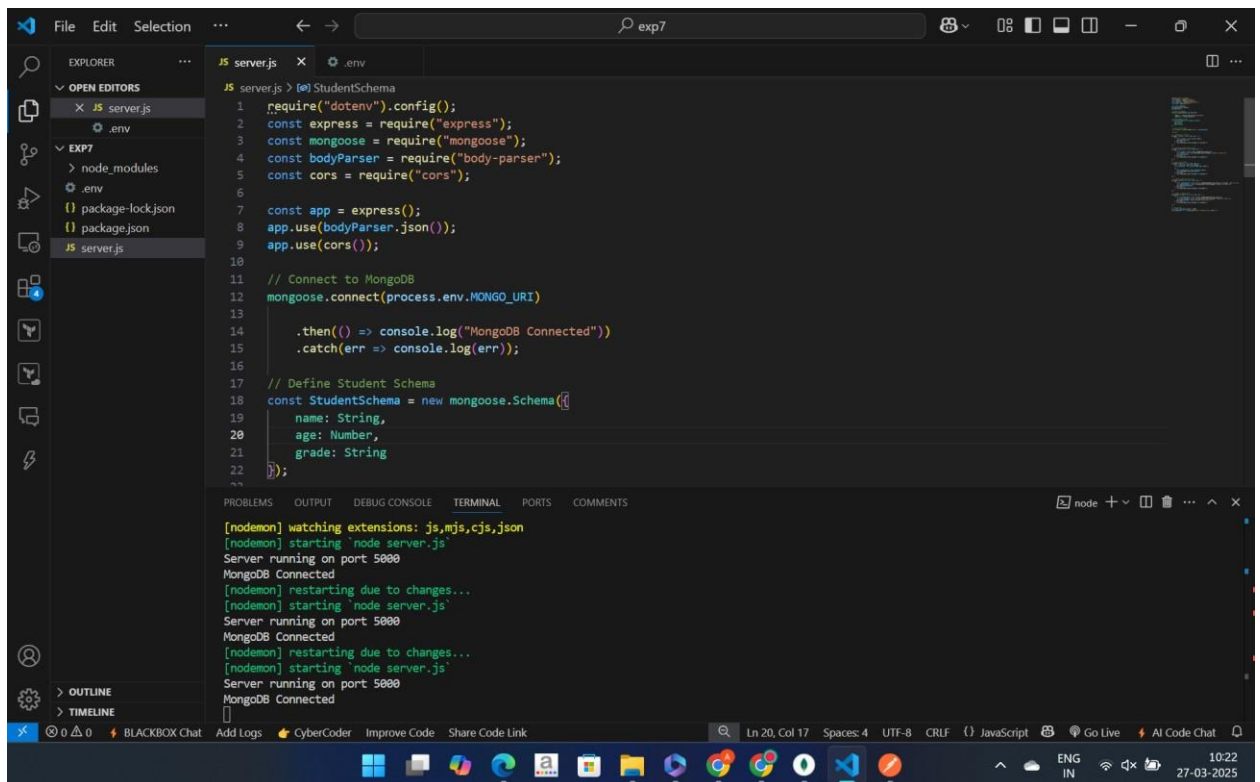
Filter☐   Type a query: { field: 'valu   Reset   Apply   Options ▸

QUERY RESULTS: 0

```js
require("dotenv").config();
const express = require("express");
const mongoose = require("mongoose");
const bodyParser = require("body-parser");
const cors = require("cors");

const app = express();
app.use(bodyParser.json());
app.use(cors());

// Connect to MongoDB
mongoose.connect(process.env.MONGO_URI)

    .then(() => console.log("MongoDB Connected"))
    .catch(err => console.log(err));

// Define Student Schema
const StudentSchema = new mongoose.Schema({
    name: String,
    age: Number,
    grade: String
});
```

```
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node server.js`
Server running on port 5000
MongoDB Connected
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server running on port 5000
MongoDB Connected
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server running on port 5000
MongoDB Connected
```

```
> db.students.updateOne({ RollNo: "IT101" }, { $set: { RollNo: "IT110" } })
< {
    acknowledged: true,
    insertedId: null,
    matchedCount: 1,
    modifiedCount: 1,
    upsertedCount: 0
  }
> db.students.deleteOne({ Name: "Charlie Brown" })
< {
    acknowledged: true,
    deletedCount: 1
  }
Atlas atlas-jsa0wb-shard-0 [primary] itdept>
```

```
>_MONGOSH

> db.students.find({ ClassName: "IT-1" }).pretty()
< {
    _id: ObjectId('67e4d54b85c12e85bbe0e052'),
    Name: 'John Doe',
    RollNo: 'IT101',
    ClassName: 'IT-1'
  }
  {
    _id: ObjectId('67e4d55585c12e85bbe0e053'),
    Name: 'Alice Smith',
    RollNo: 'IT102',
    ClassName: 'IT-1'
  }
  {
    _id: ObjectId('67e4d55585c12e85bbe0e055'),
    Name: 'Charlie Brown',
    RollNo: 'IT104',
    ClassName: 'IT-1'
  }
> db.students.find({ RollNo: "IT103", ClassName: "IT-2" }).pretty()
< {
    _id: ObjectId('67e4d55585c12e85bbe0e054'),
    Name: 'Bob Johnson',
    RollNo: 'IT103',
    ClassName: 'IT-2'
  }
```

```
>_MONGOSH

> use itdept
< switched to db itdept
>
> db.students.insertOne({
      Name: "John Doe",
      RollNo: "IT101",
      ClassName: "IT-1"
  })
< {
    acknowledged: true,
    insertedId: ObjectId('67e4d54b85c12e85bbe0e052')
  }
> db.students.insertMany([
      { Name: "Alice Smith", RollNo: "IT102", ClassName: "IT-1" },
      { Name: "Bob Johnson", RollNo: "IT103", ClassName: "IT-2" },
      { Name: "Charlie Brown", RollNo: "IT104", ClassName: "IT-1" },
      { Name: "David White", RollNo: "IT105", ClassName: "IT-3" }
  ])
< {
    acknowledged: true,
    insertedIds: {
      '0': ObjectId('67e4d55585c12e85bbe0e053'),
      '1': ObjectId('67e4d55585c12e85bbe0e054'),
      '2': ObjectId('67e4d55585c12e85bbe0e055'),
      '3': ObjectId('67e4d55585c12e85bbe0e056')
```

ARVA'S

C

REGION
AWS Mun

Overv

y Insights

BASES:

ISUALIZE YOUR DA

Create Database ✕

Database name ❓

```
itdept
```

Collection name ❓

```
student
```

Additional Preferences

```
Select                                        ▼
```

Searc

UMENTS: INDEX

36KB

nvent

Cancel   Create

prod

Aggregation

IN