**Name : Chirag A Choudhary**

**Class : D15A**

**Roll no. : 10**

**Experiment – 7: MongoDB**

1. **Aim:** To study CRUD operations in MongoDB
2. **Problem Statement:**
3. Create a new database to storage student details of IT dept( Name, Roll no, class name) and perform the following on the database
   1. Insert one student details
   2. Insert at once multiple student details
   3. Display student for a particular class
   4. Display students of specific roll no in a class
   5. Change the roll no of a student
   6. Delete entries of particular student

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

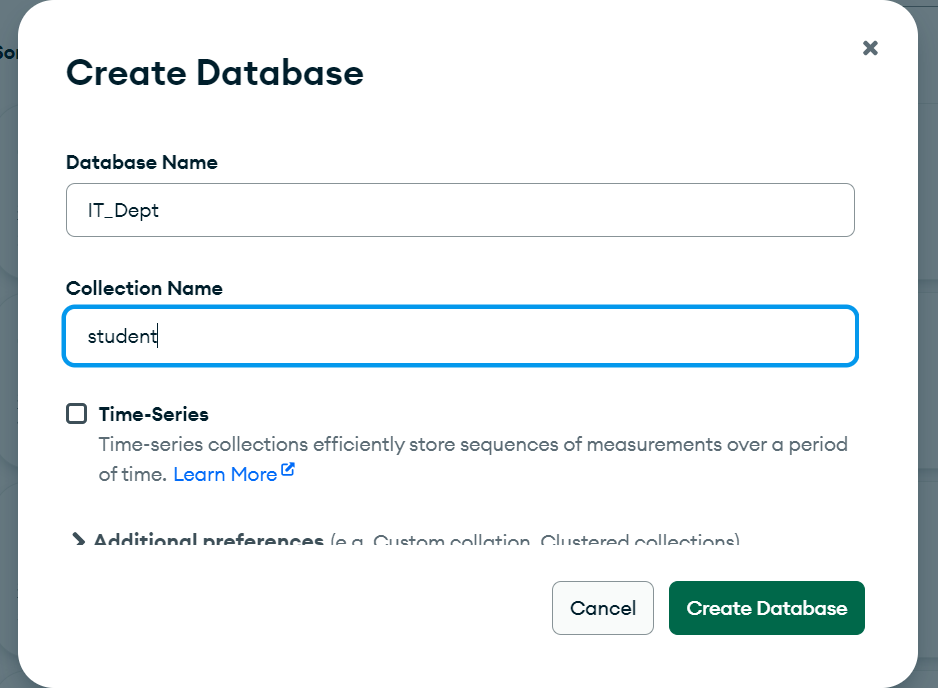
The endpoints should support:

* 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.

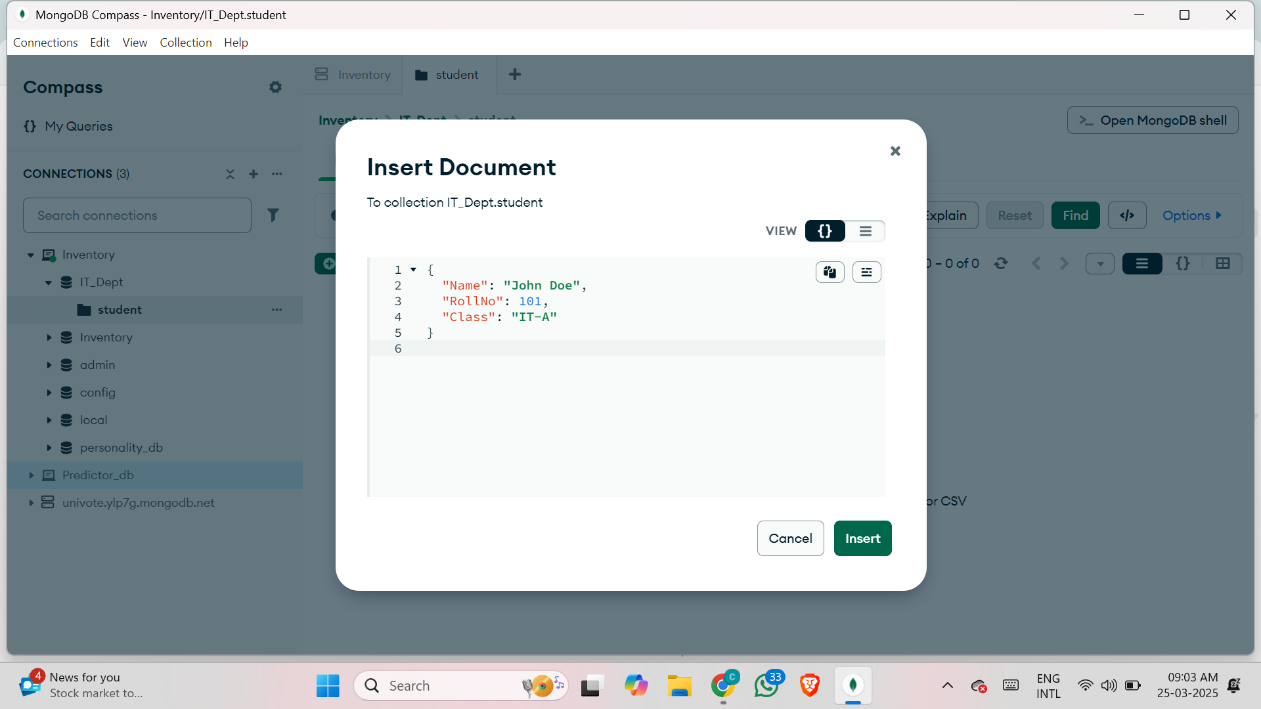
Connect the server to MongoDB using Mongoose, and store student data with attributes: name, age, and grade.

1. **Output:**

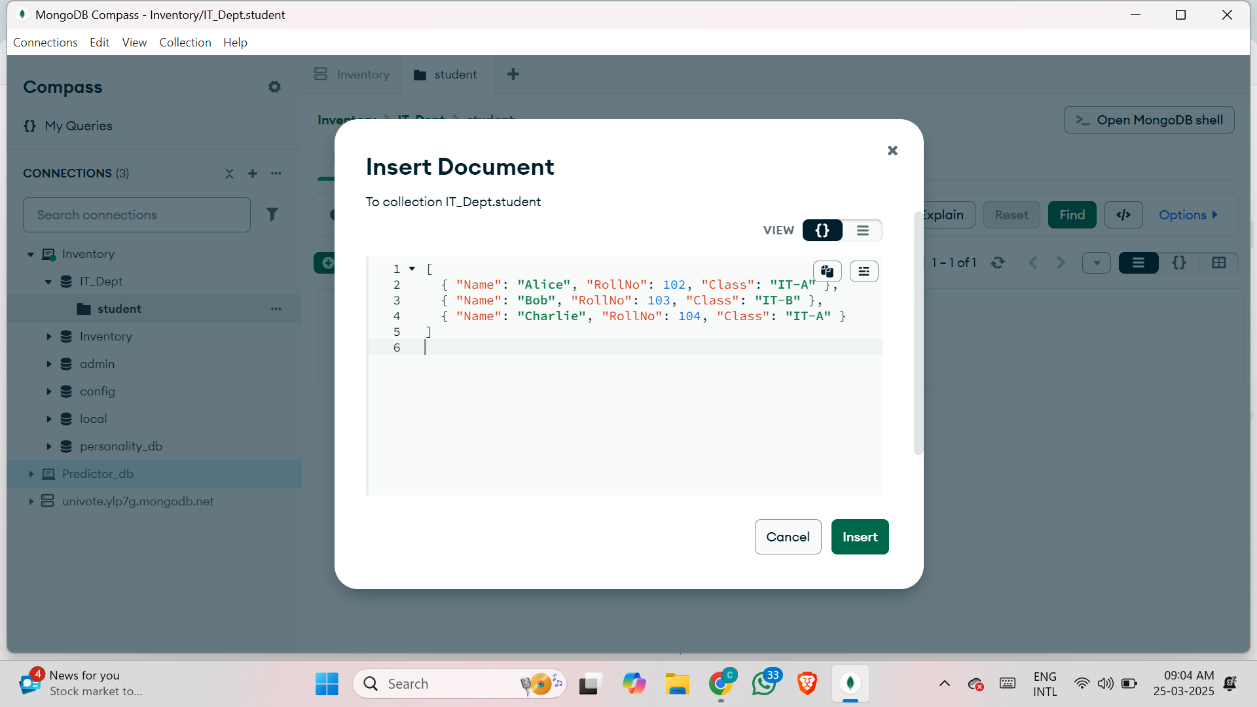
**Part A:** MongoDB CRUD Operations in Compass. 1) Create Database and Collection



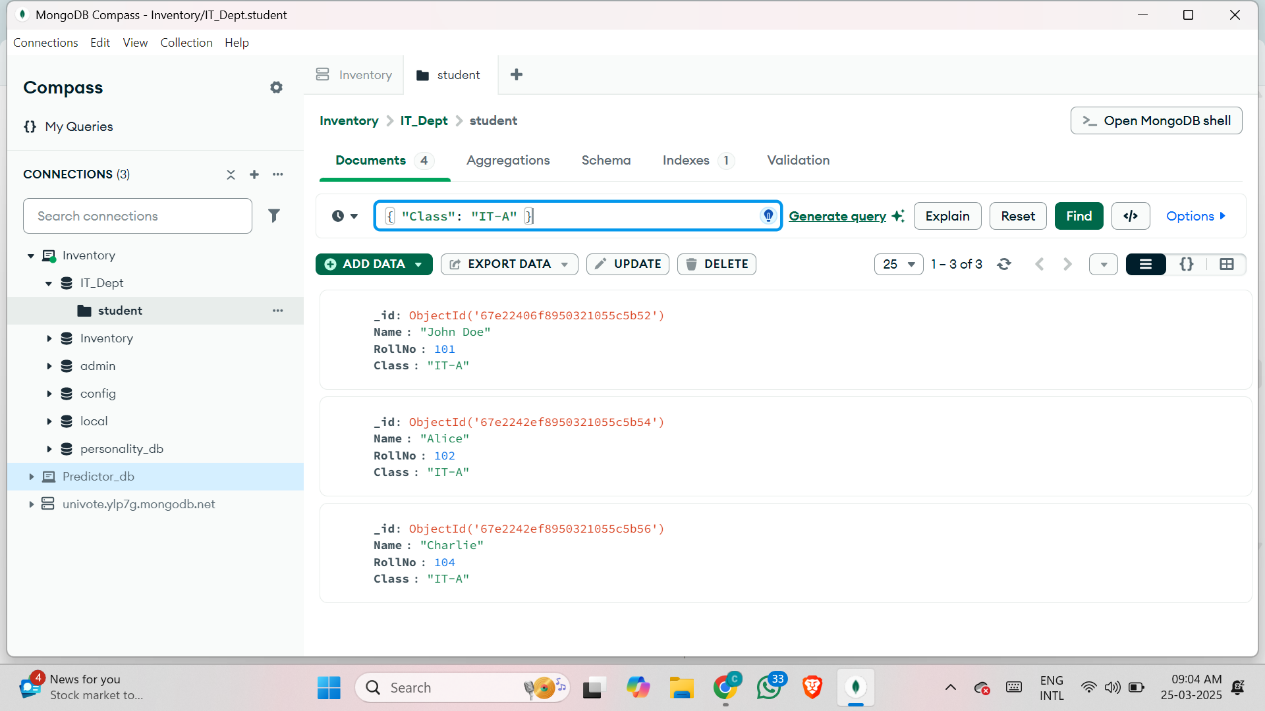
2) Insert One Student Record



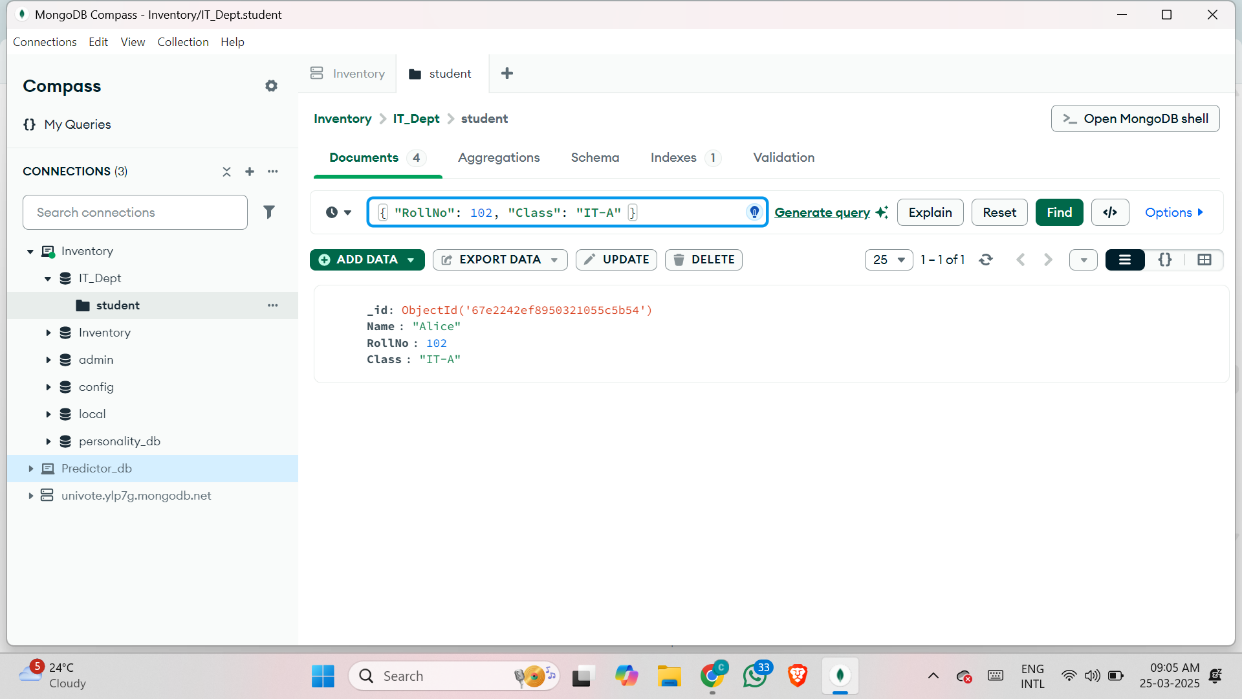
3) Insert Multiple Students at Once



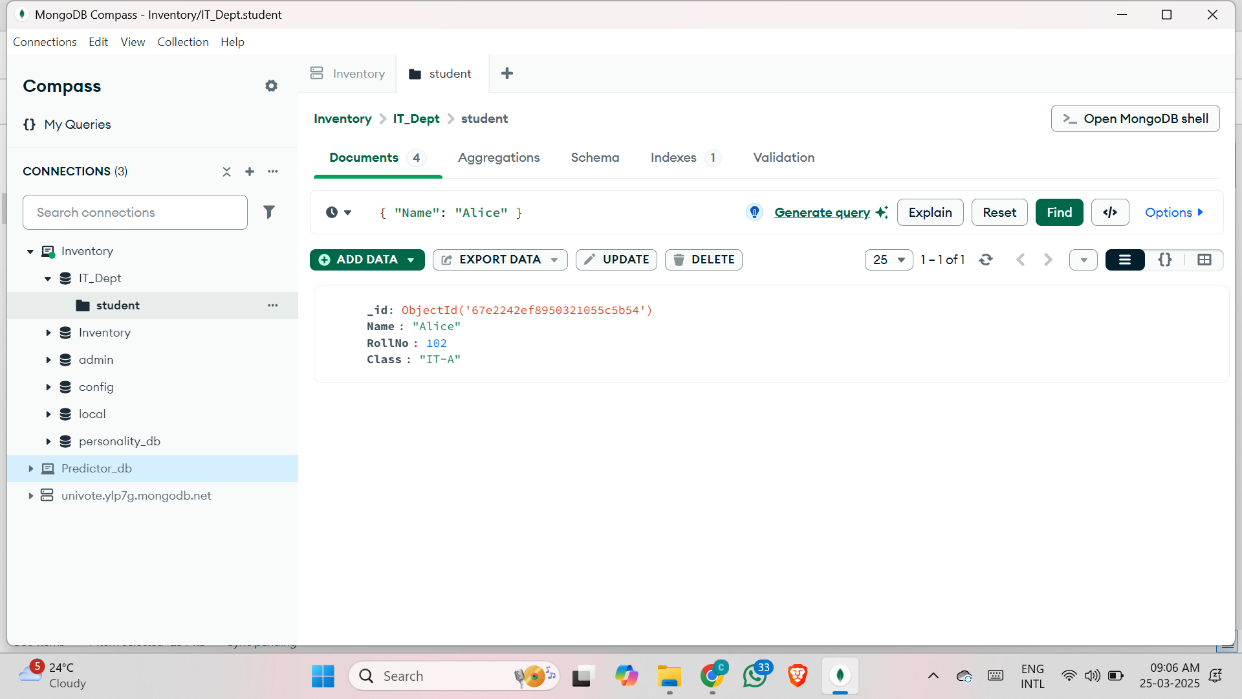
4) Display Students for a Particular Class

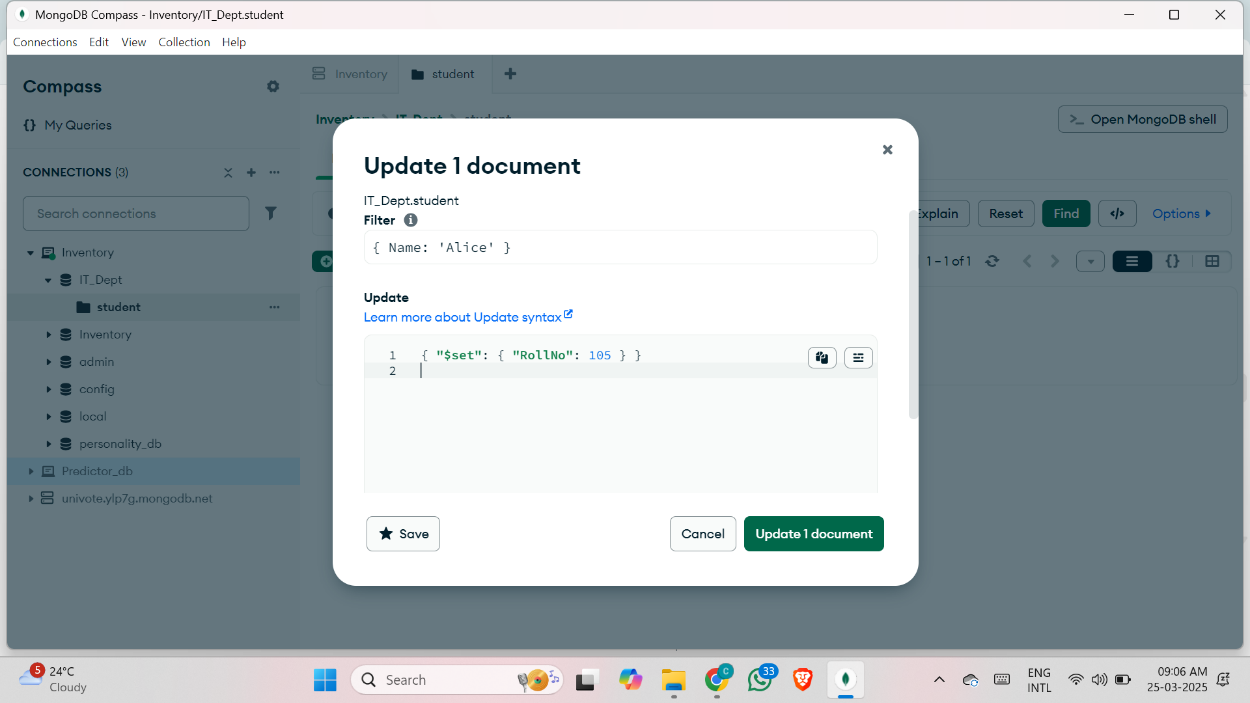


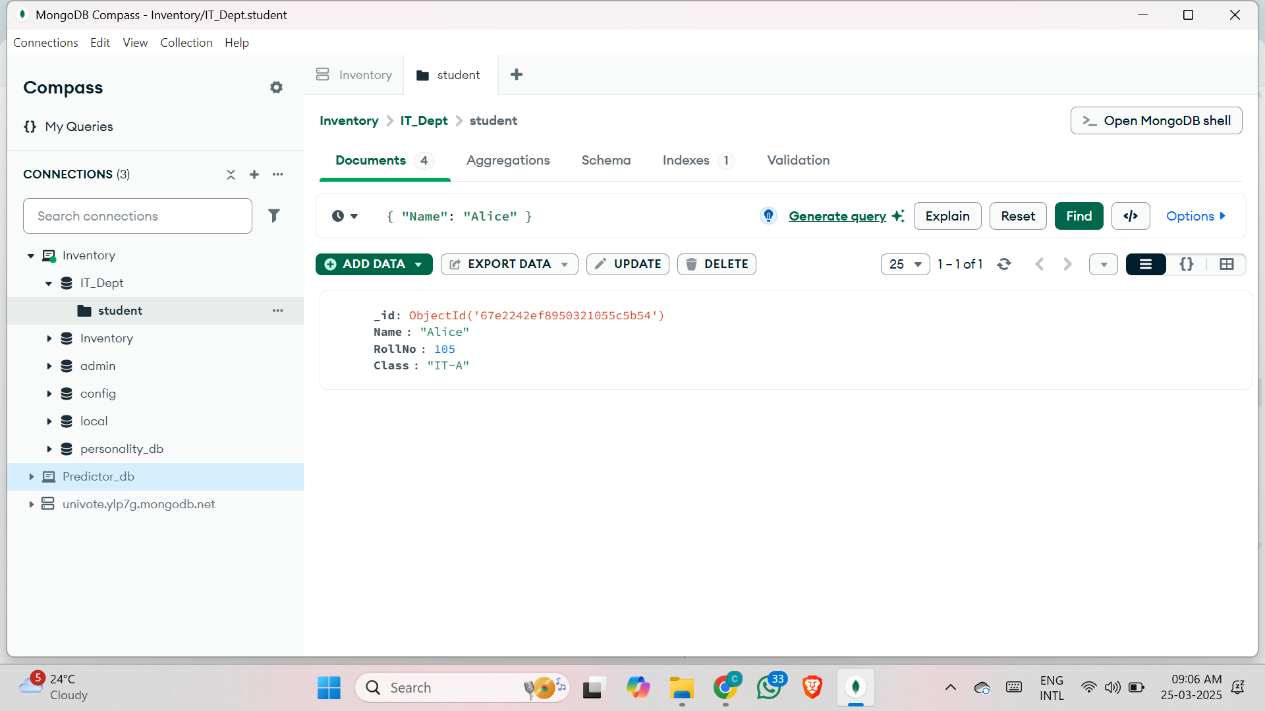
5) Display a Student With a Specific Roll No in a Class

****

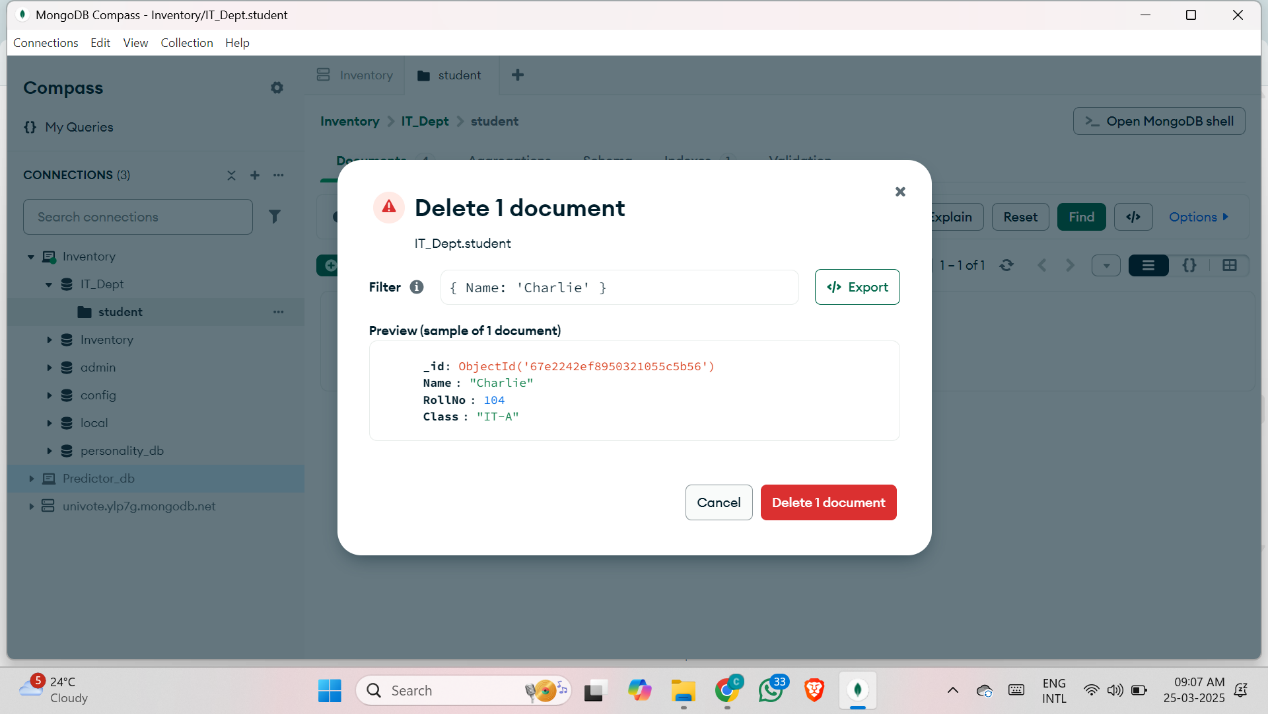
6) Change the Roll No of a Student

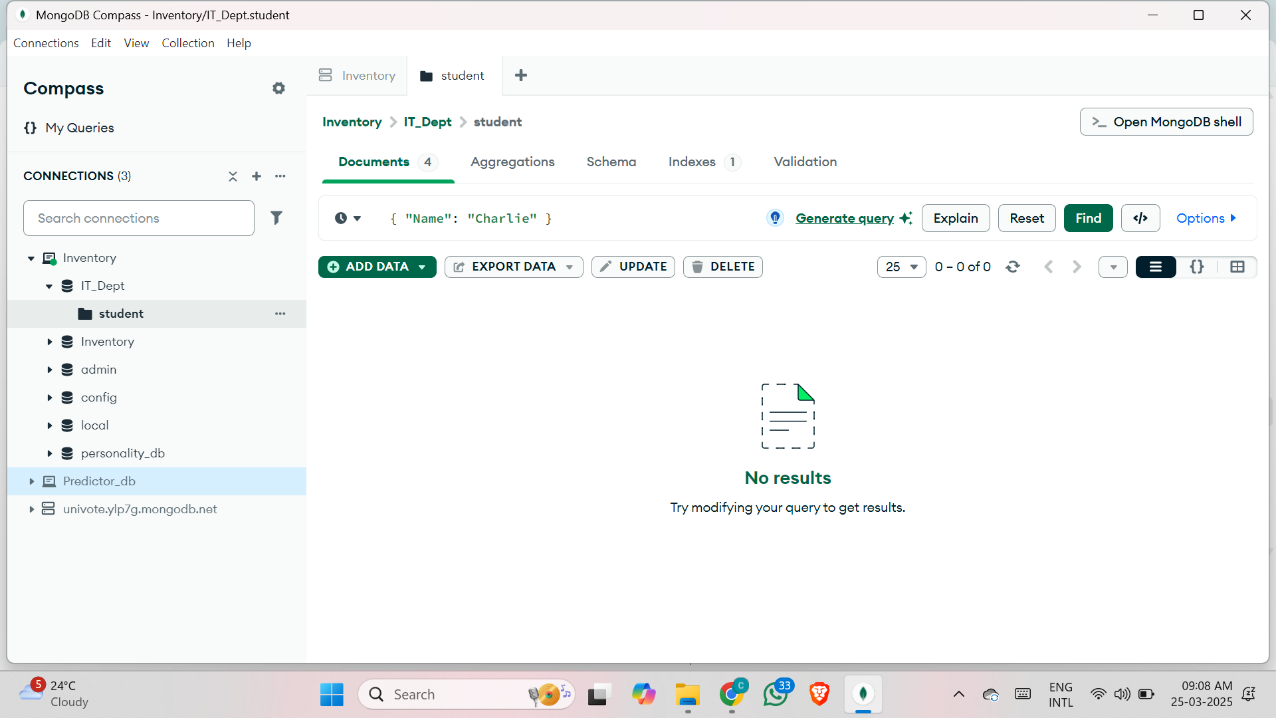
****

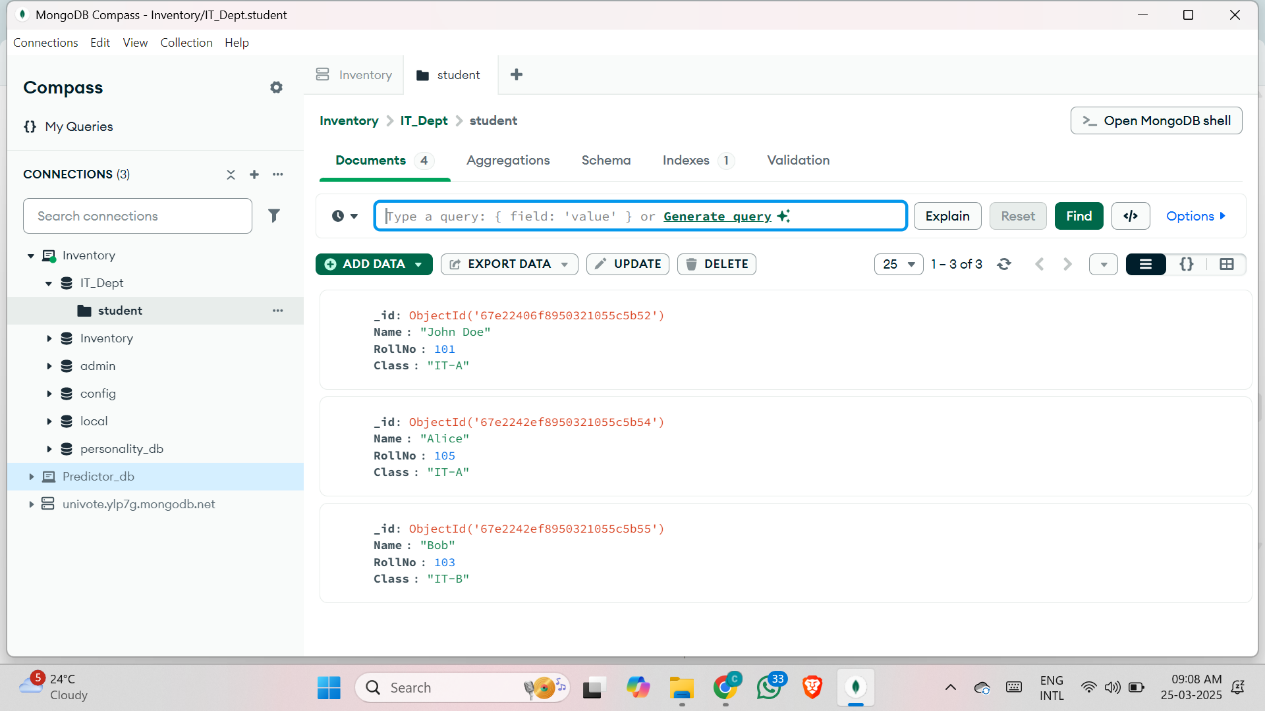
****

****

7) Delete a Particular Student Entry

****

****

****

**Part B:** Creating RESTful API with Node.js, Express, and Mongoose

**server.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());

// MongoDB Connection

mongoose.connect(process.env.MONGO\_URI, { useNewUrlParser: true, useUnifiedTopology: true })

  .then(() => console.log('MongoDB Connected'))

  .catch(err => console.error('MongoDB Connection Failed:', err));

// Define Student Schema

const studentSchema = new mongoose.Schema({

  name: String,

  age: Number,

  grade: String

});

const Student = mongoose.model('Student', studentSchema);

// Routes

app.get('/students', async (req, res) => {

  const students = await Student.find();

  res.json(students);

});

app.get('/students/:id', async (req, res) => {

  try {

    const student = await Student.findById(req.params.id);

    res.json(student);

  } catch (err) {

    res.status(404).json({ error: 'Student not found' });

  }

});

app.post('/students', async (req, res) => {

  const newStudent = new Student(req.body);

  await newStudent.save();

  res.json(newStudent);

});

app.put('/students/:id', async (req, res) => {

  try {

    const updatedStudent = await Student.findByIdAndUpdate(req.params.id, req.body, { new: true });

    res.json(updatedStudent);

  } catch (err) {

    res.status(404).json({ error: 'Student not found' });

  }

});

app.delete('/students/:id', async (req, res) => {

  try {

    await Student.findByIdAndDelete(req.params.id);

    res.json({ message: 'Student deleted' });

  } catch (err) {

    res.status(404).json({ error: 'Student not found' });

  }

});

// Server Start

const PORT = process.env.PORT || 5000;

app.listen(PORT, () => console.log(`Server running on port ${PORT}`));

**.env:**

MONGO\_URI=mongodb://localhost:27017/IT\_Dept

PORT=5000

