|  |  |
| --- | --- |
| NAME | Bhumi |
| UID | 23BCS13453 |
| CLASS | 622-B |

* NodeJs PRACTISE 5.2
* CODE

Server.js (Main server):

// server.js

const express = require('express');

const mongoose = require('mongoose');

const studentRoutes = require('./routes/studentRoutes');

const app = express();

app.use(express.json());

// MongoDB Connection

mongoose.connect('mongodb+srv://alpha:alpha2025@cluster0.gttfsb5.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0')

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

.catch(err => console.error('MongoDB connection error:', err.message));

// Routes

app.use('/students', studentRoutes);

// Start Server

const PORT = 3000;

app.listen(PORT, () => {

  console.log(`Server running on http://localhost:${PORT}`);

});

/models/Student.js :

// models/Student.js

const mongoose = require('mongoose');

const StudentSchema = new mongoose.Schema({

  name: {

    type: String,

    required: true,

    trim: true

  },

  age: {

    type: Number,

    required: true,

    min: 1

  },

  course: {

    type: String,

    required: true,

    trim: true

  },

  createdAt: {

    type: Date,

    default: Date.now

  }

});

module.exports = mongoose.model('Student', StudentSchema);

/controller/studentController.js :

// controllers/studentController.js

const Student = require('../models/Student');

// Create new student

exports.createStudent = async (req, res) => {

  try {

    const { name, age, course } = req.body;

    const student = new Student({ name, age, course });

    const savedStudent = await student.save();

    res.status(201).json(savedStudent);

  } catch (error) {

    res.status(400).json({ message: error.message });

  }

};

// Get all students

exports.getAllStudents = async (req, res) => {

  try {

    const students = await Student.find();

    res.json(students);

  } catch (error) {

    res.status(500).json({ message: error.message });

  }

};

// Get single student by ID

exports.getStudentById = async (req, res) => {

  try {

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

    if (!student) return res.status(404).json({ message: 'Student not found' });

    res.json(student);

  } catch (error) {

    res.status(500).json({ message: error.message });

  }

};

// Update student by ID

exports.updateStudent = async (req, res) => {

  try {

    const { name, age, course } = req.body;

    const updatedStudent = await Student.findByIdAndUpdate(

      req.params.id,

      { name, age, course },

      { new: true }

    );

    if (!updatedStudent) return res.status(404).json({ message: 'Student not found' });

    res.json(updatedStudent);

  } catch (error) {

    res.status(400).json({ message: error.message });

  }

};

// Delete student by ID

exports.deleteStudent = async (req, res) => {

  try {

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

    if (!student) {

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

    }

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

    res.json({

      message: 'Student deleted successfully',

      deletedStudent: {

        id: student.\_id,

        name: student.name,

        age: student.age,

        course: student.course

      }

    });

  } catch (error) {

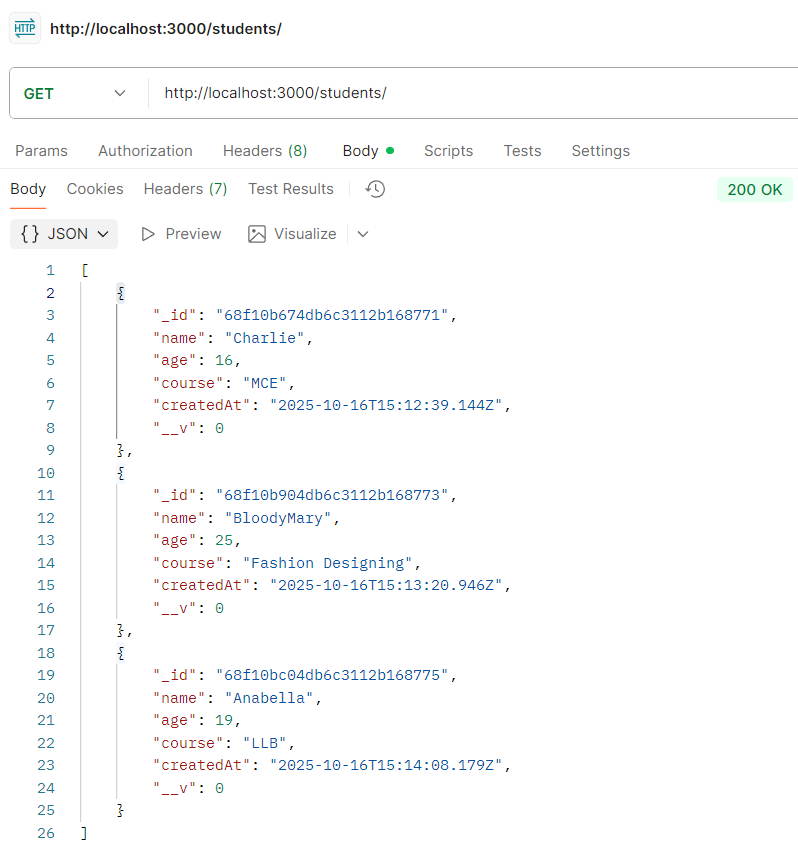
    res.status(500).json({ message: error.message });

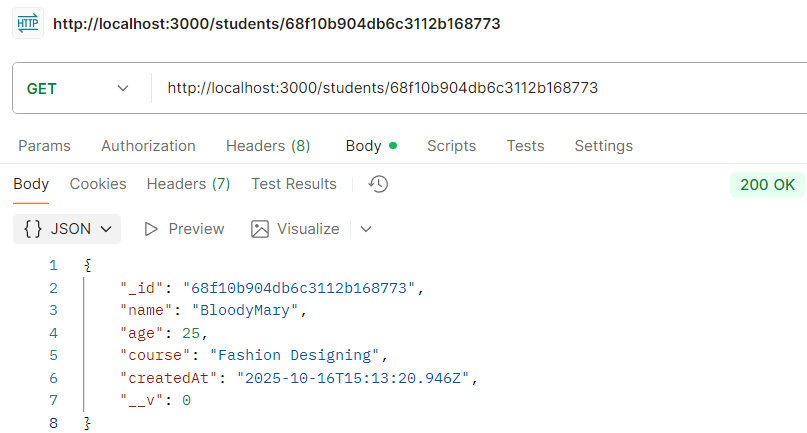
  }

};

/routes/studentRoutes.js :

* // routes/studentRoutes.js
* const express = require('express');
* const router = express.Router();
* const studentController = require('../controllers/studentController');
* // CRUD Routes
* router.post('/add', studentController.createStudent);
* router.get('/', studentController.getAllStudents);
* router.get('/:id', studentController.getStudentById);
* router.put('/:id', studentController.updateStudent);
* router.delete('/:id', studentController.deleteStudent);
* module.exports = router;
* OUTPUT





A screenshot of a computer

AI-generated content may be incorrect.

Output Link (Postman) : <http://localhost:3000/students/>

API’s: GET <http://localhost:3000/students/>

GET<http://localhost:3000/students/(id)>

DELETE <http://localhost:3000/students/(id)>

POST <http://localhost:3000/students/add>