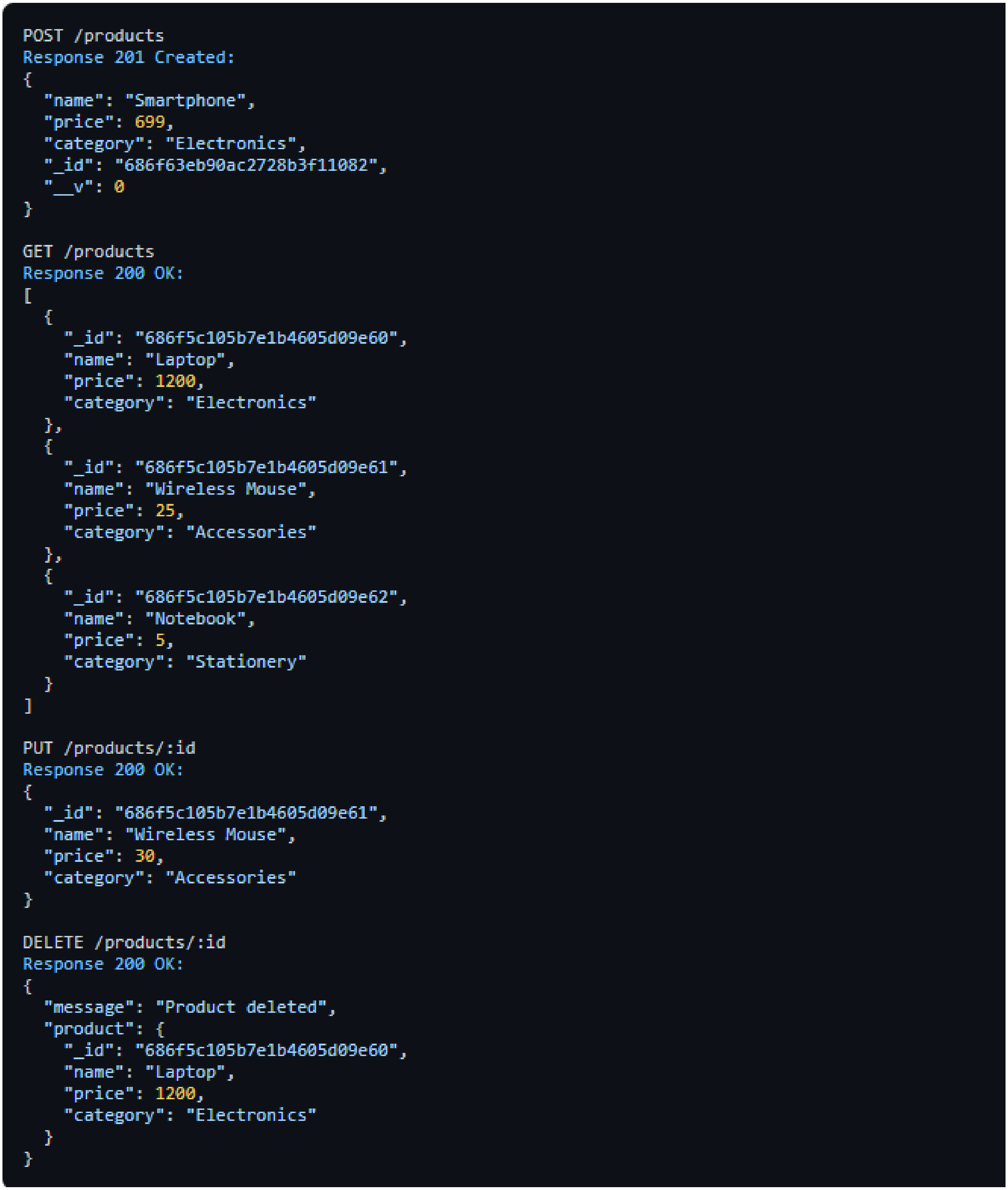
Name: KONIKA KHURANA	**UID**:23BCS12925
Branch: CSE	**Section**: 23 BCS_FS 622-A
Semester: 5	**Date of Performance**: 09/10/2025
Subject: FULL STACK-LAB	**Subject Code**: 23CSP-339

Mongo db 1:

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');

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

mongoose.connect('mongodb://localhost:27017/productsDB', {
  useNewUrlParser: true,
  useUnifiedTopology: true
})
.then(() => console.log('MongoDB connected'))
.catch(err => console.log(err));

const productSchema = new mongoose.Schema({
  name: { type: String, required: true },
  price: { type: Number, required: true },
  category: { type: String, required: true }
});

const Product = mongoose.model('Product', productSchema);

app.post('/products', async (req, res) => {
  try {
    const product = new Product(req.body);
    const savedProduct = await product.save();
    res.status(201).json(savedProduct);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});
```

```
app.get('/products', async (req, res) => {
  try {
    const products = await Product.find();
    res.status(200).json(products);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

app.put('/products/:id', async (req, res) => {
  try {
    const updatedProduct = await Product.findByIdAndUpdate(
      req.params.id,
      req.body,
      { new: true }
    );
    if (!updatedProduct)
      return res.status(404).json({ message: 'Product not found' });
    res.status(200).json(updatedProduct);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
});

app.delete('/products/:id', async (req, res) => {
  try {
    const deletedProduct = await
Product.findByIdAndDelete(req.params.id);
    if (!deletedProduct)
      return res.status(404).json({ message: 'Product not found' });
    res.status(200).json({
      message: 'Product deleted',
      product: deletedProduct
    });
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
});

const PORT = 3000;
app.listen(PORT, () => console.log(`Server running on port
${PORT}`));
```

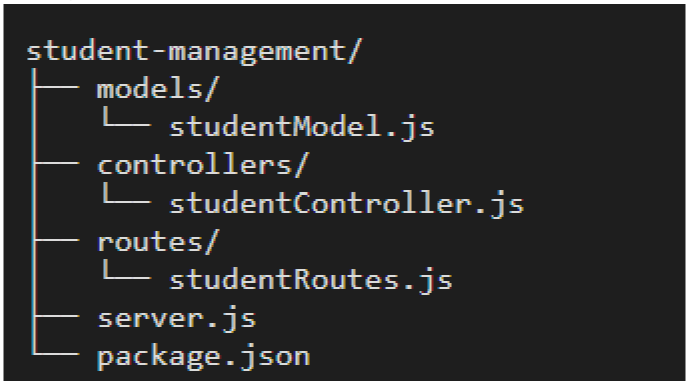
Expected output:



```
POST /products
Response 201 Created:
{
  "name": "Smartphone",
  "price": 699,
  "category": "Electronics",
  "_id": "686f63eb90ac2728b3f11082",
  "__v": 0
}

GET /products
Response 200 OK:
[
  {
    "_id": "686f5c105b7e1b4605d09e60",
    "name": "Laptop",
    "price": 1200,
    "category": "Electronics"
  },
  {
    "_id": "686f5c105b7e1b4605d09e61",
    "name": "Wireless Mouse",
    "price": 25,
    "category": "Accessories"
  },
  {
    "_id": "686f5c105b7e1b4605d09e62",
    "name": "Notebook",
    "price": 5,
    "category": "Stationery"
  }
]

PUT /products/:id
Response 200 OK:
{
  "_id": "686f5c105b7e1b4605d09e61",
  "name": "Wireless Mouse",
  "price": 30,
  "category": "Accessories"
}

DELETE /products/:id
Response 200 OK:
{
  "message": "Product deleted",
  "product": {
    "_id": "686f5c105b7e1b4605d09e60",
    "name": "Laptop",
    "price": 1200,
    "category": "Electronics"
  }
}
```

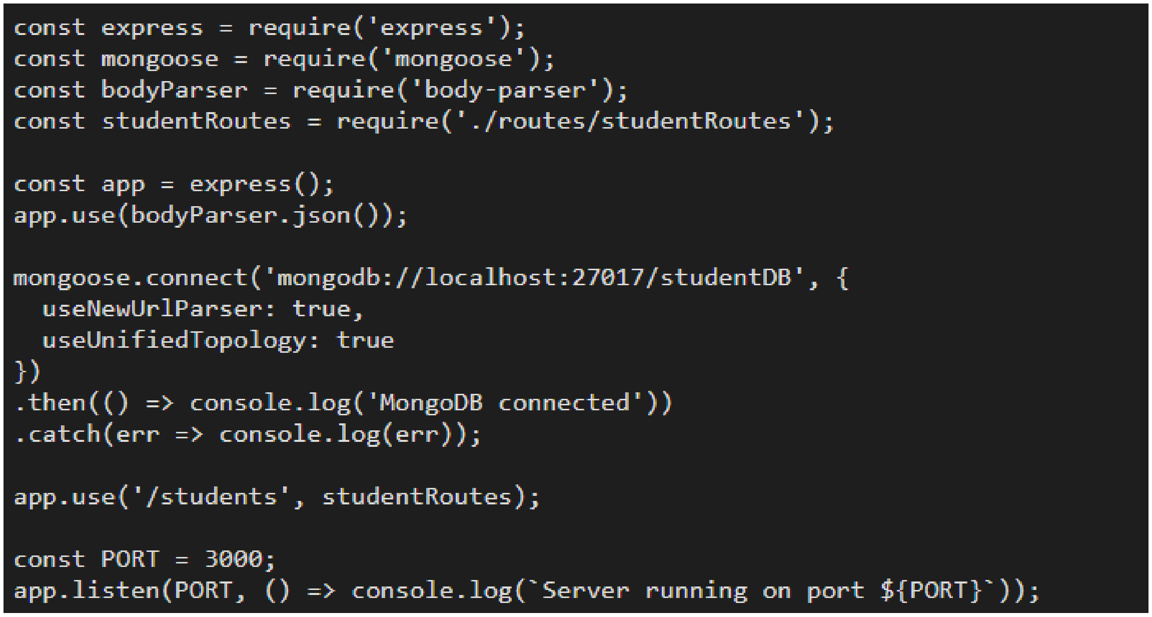
Mongo db 2:

Project Structure



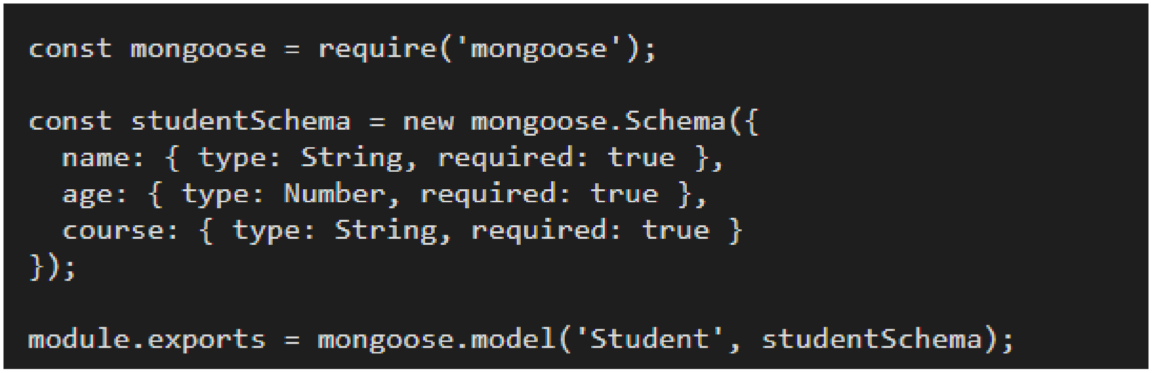
```
student-management/
├── models/
│   └── studentModel.js
├── controllers/
│   └── studentController.js
├── routes/
│   └── studentRoutes.js
├── server.js
└── package.json
```

server.js



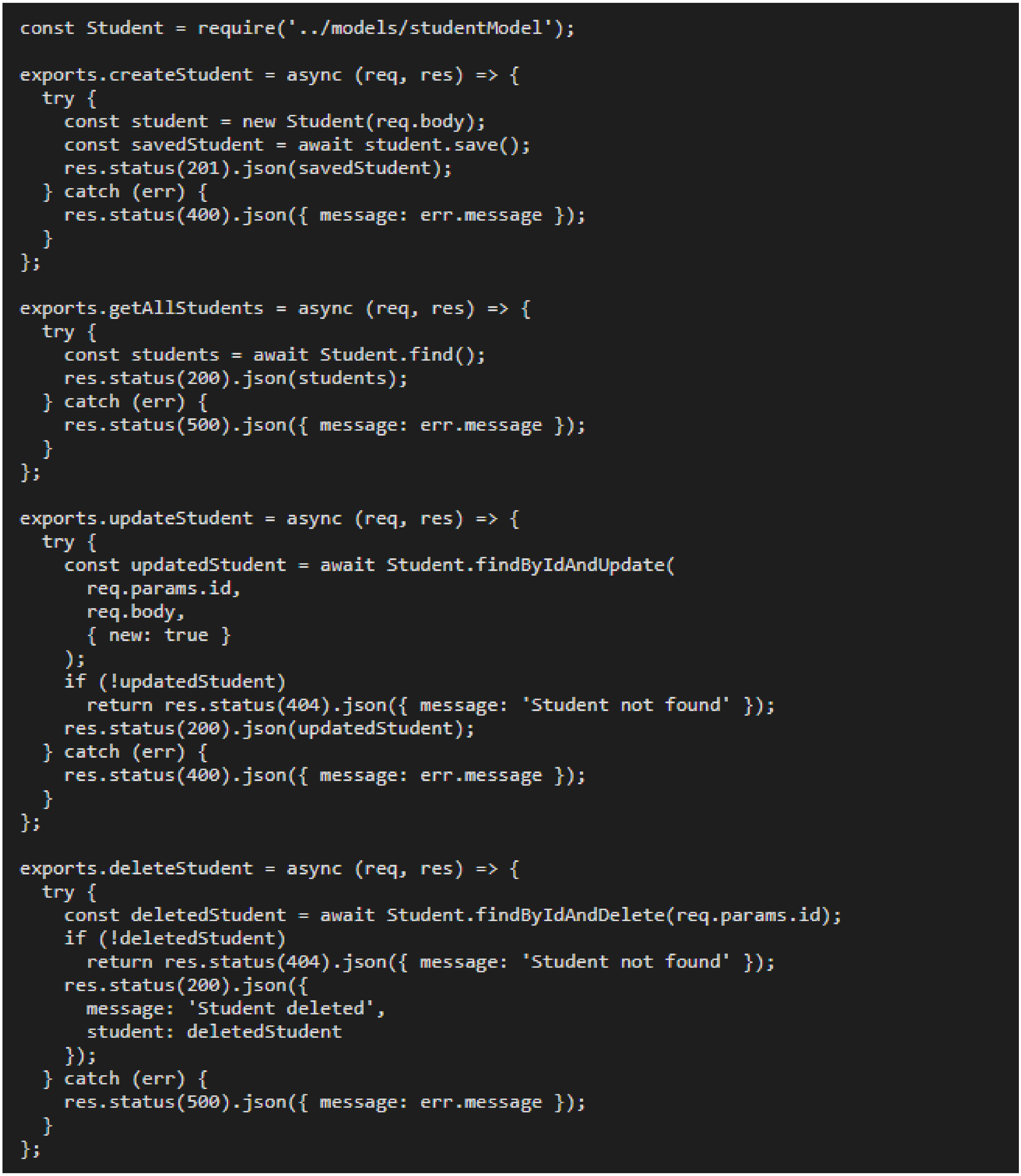
```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const studentRoutes = require('./routes/studentRoutes');

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

mongoose.connect('mongodb://localhost:27017/studentDB', {
  useNewUrlParser: true,
  useUnifiedTopology: true
})
.then(() => console.log('MongoDB connected'))
.catch(err => console.log(err));

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

const PORT = 3000;
app.listen(PORT, () => console.log(`Server running on port ${PORT}`));
```

Models/studentModel.js



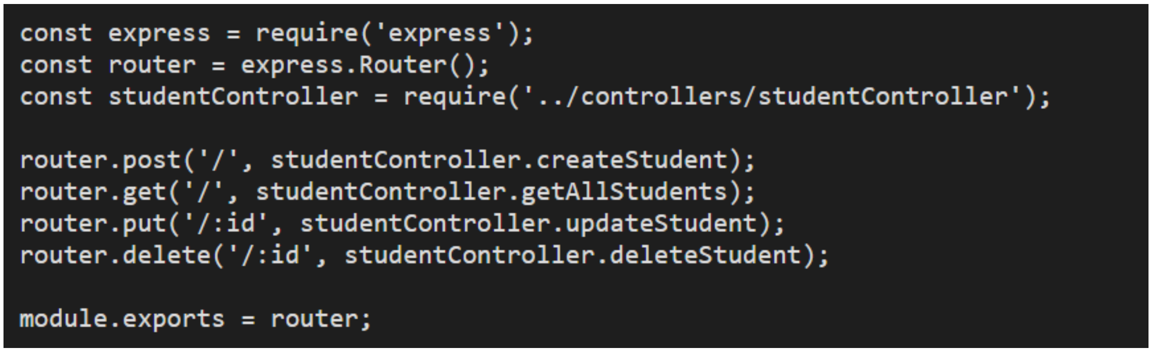
```
const mongoose = require('mongoose');

const studentSchema = new mongoose.Schema({
  name: { type: String, required: true },
  age: { type: Number, required: true },
  course: { type: String, required: true }
});

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

Controllers/studentController.js



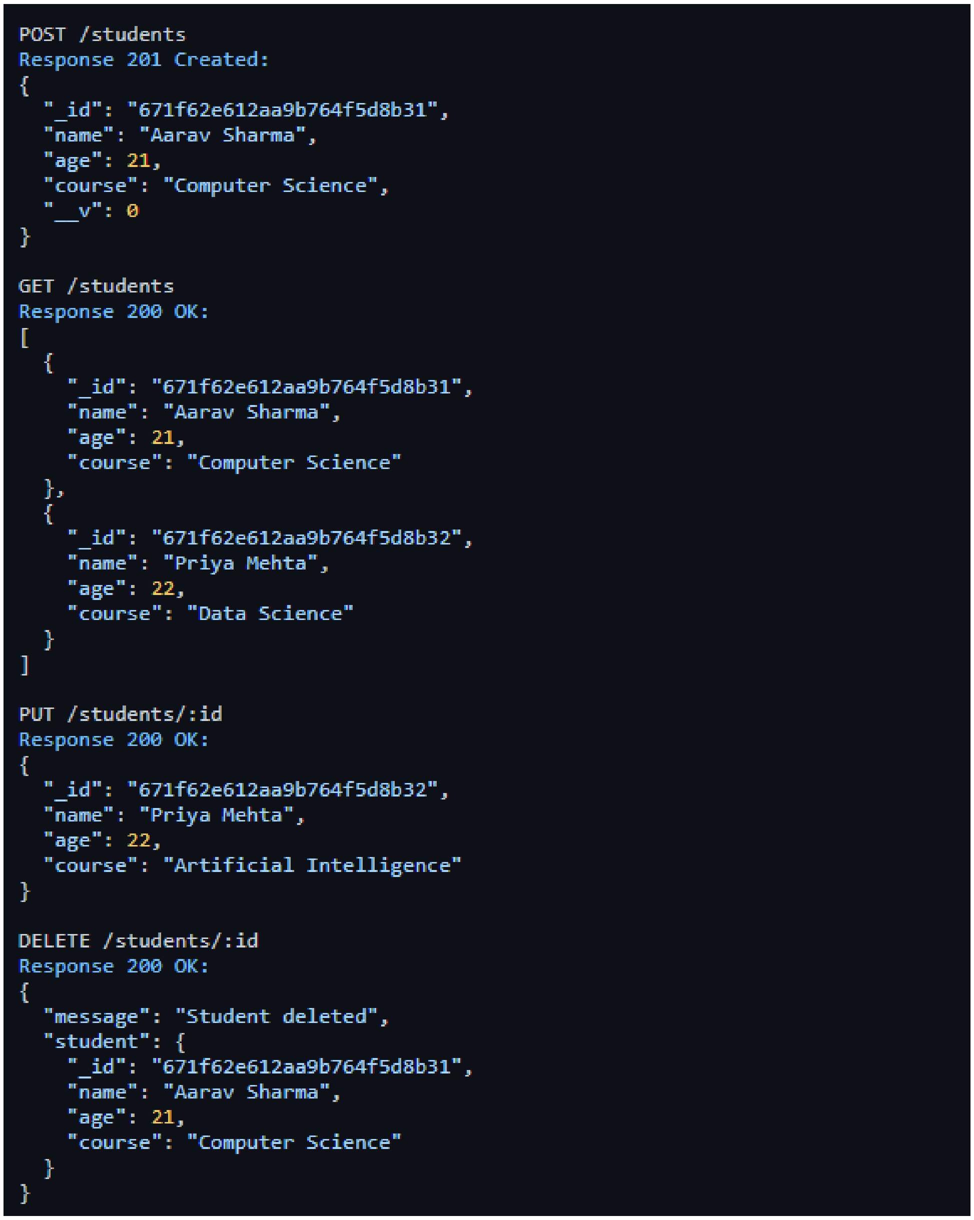
```
const Student = require('../models/studentModel');

exports.createStudent = async (req, res) => {
  try {
    const student = new Student(req.body);
    const savedStudent = await student.save();
    res.status(201).json(savedStudent);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
};

exports.getAllStudents = async (req, res) => {
  try {
    const students = await Student.find();
    res.status(200).json(students);
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
};

exports.updateStudent = async (req, res) => {
  try {
    const updatedStudent = await Student.findByIdAndUpdate(
      req.params.id,
      req.body,
      { new: true }
    );
    if (!updatedStudent)
      return res.status(404).json({ message: 'Student not found' });
    res.status(200).json(updatedStudent);
  } catch (err) {
    res.status(400).json({ message: err.message });
  }
};

exports.deleteStudent = async (req, res) => {
  try {
    const deletedStudent = await Student.findByIdAndDelete(req.params.id);
    if (!deletedStudent)
      return res.status(404).json({ message: 'Student not found' });
    res.status(200).json({
      message: 'Student deleted',
      student: deletedStudent
    });
  } catch (err) {
    res.status(500).json({ message: err.message });
  }
};
```

Routes/studentRoutes.js



```
const express = require('express');
const router = express.Router();
const studentController = require('../controllers/studentController');

router.post('/', studentController.createStudent);
router.get('/', studentController.getAllStudents);
router.put('/:id', studentController.updateStudent);
router.delete('/:id', studentController.deleteStudent);

module.exports = router;
```

Expected Output (Postman Style):



```
POST /students
Response 201 Created:
{
  "_id": "671f62e612aa9b764f5d8b31",
  "name": "Aarav Sharma",
  "age": 21,
  "course": "Computer Science",
  "__v": 0
}

GET /students
Response 200 OK:
[
  {
    "_id": "671f62e612aa9b764f5d8b31",
    "name": "Aarav Sharma",
    "age": 21,
    "course": "Computer Science"
  },
  {
    "_id": "671f62e612aa9b764f5d8b32",
    "name": "Priya Mehta",
    "age": 22,
    "course": "Data Science"
  }
]

PUT /students/:id
Response 200 OK:
{
  "_id": "671f62e612aa9b764f5d8b32",
  "name": "Priya Mehta",
  "age": 22,
  "course": "Artificial Intelligence"
}

DELETE /students/:id
Response 200 OK:
{
  "message": "Student deleted",
  "student": {
    "_id": "671f62e612aa9b764f5d8b31",
    "name": "Aarav Sharma",
    "age": 21,
    "course": "Computer Science"
  }
}
```

Mongo db 3:

```
use ecommerceDB;
db.createCollection("products");

db.products.insertMany([
  {
    name: "Smartphone",
    price: 699,
    category: "Electronics",
    variants: []
  },
  {
    name: "Winter Jacket",
    price: 200,
    category: "Apparel",
    variants: [
      { color: "Black", size: "S", stock: 8 },
      { color: "Gray", size: "M", stock: 12 }
    ]
  },
  {
    name: "Running Shoes",
    price: 120,
    category: "Footwear",
    variants: [
      { color: "Red", size: "M", stock: 10 },
      { color: "Blue", size: "L", stock: 5 }
    ]
  }
]);

db.products.find().pretty();

db.products.find({ category: "Electronics"
}).pretty();

db.products.find({ "variants.color": "Blue"
}).pretty();

db.products.find({}, { name: 1,
"variants.color": 1, _id: 0 }).pretty();

db.products.updateOne(
  { name: "Running Shoes",
"variants.color": "Blue" },
  { $set: { "variants.$.stock": 8 } }
);

db.products.deleteOne({ name:
"Smartphone" });
```