PRACTICAL 2

1. CRUD operation using NodeJS .

CODE :

Run Terminal Help index.js - D:\Node Js\COLLAGE PRACTICALS\PRACTICALS\PRACTICAL_2 - PRACTICAL_2 - Visual Studio Code

JS index.js U TC New Request JS studentmodel.js U

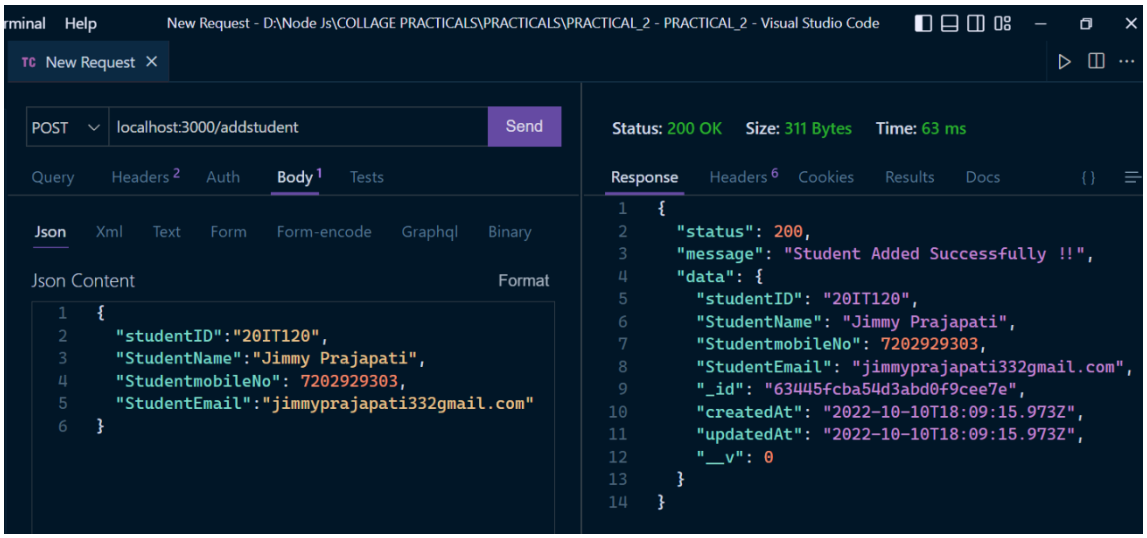
JS index.js > mongoose.connection.on("disconnected") callback

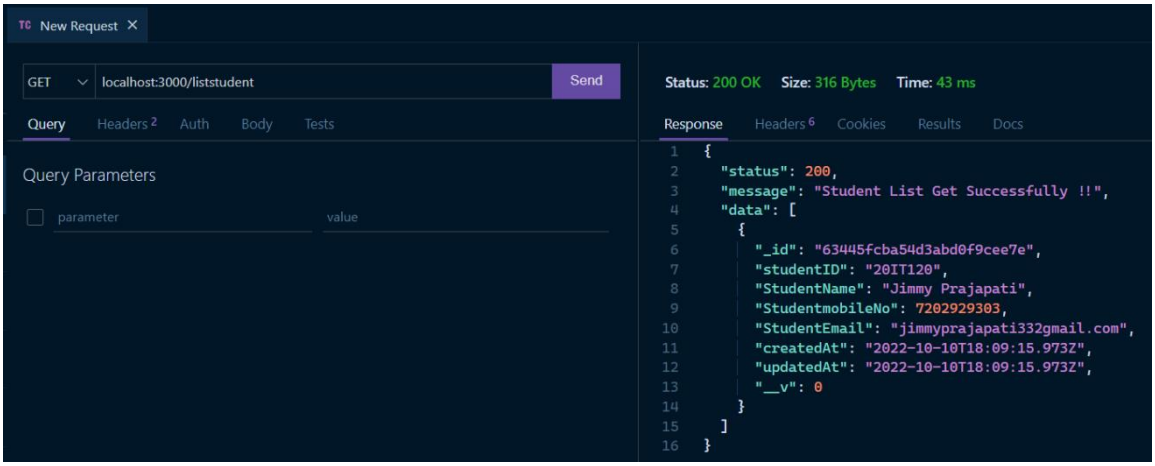
```
import express, { request } from "express";
import mongoose from "mongoose";
import bodyParser from "body-parser";
import Student from "./model/studentmodel.js";

const app = express();

app.use(express.json());

const connect = async () => {
  try {
    await mongoose.connect(
      "mongodb+srv://Jimmy0915:Jimmy$$0912@studentmangementsystem.ker0nuf
    );
    console.log("Connected to MongoDB !!!😍😍😍");
  } catch (error) {
    throw error;
  }
};

mongoose.connection.on("disconnected", () => {
  console.log("Disconnected from MongoDB !!!😒😡😡");
});

app.post("/addstudent", async (req, res) => {
  try {
    let data = new Student(req.body);
    await data.save();
    res.status(200).json({
      status: 200,
      message: "Student Added Successfully !!",
      data: data,
    });
  } catch (error) {
    res.send(error.message);
  }
});

app.listen(3000, () => {
  connect();
  console.log("Listening on port 3000!!!😍😍😍");
});


app.delete("/deletestudent/:id", async (req, res) => {
  try {
    let data = Student.findByIdAndDelete(req.params.id);
    res.status(200).json({
      status: 200,
      message: "Student De;ete Successfully !!",
      data: data,
    });
  } catch (error) {
    res.send(error.message);
  }
});

app.put("/editstudent/:id", async (req, res) => {
  try {
    let data = Student.findByIdAndUpdate(
      req.params.id,
      { $set: req.body },
      { new: true }
    );
    res.status(200).json({
      status: 200,
      message: "Student Edit Successfully !!",
      data: data,
    });
  } catch (error) {
    res.send(error.message);
  }
});

app.get("/liststudent", async (req, res) => {
  try {
    let data = Student.find();
    res.status(200).json({
      status: 200,
      message: "Student List Get Successfully !!",
      data: data,
    });
  } catch (error) {
    res.send(error.message);
  }
});
```

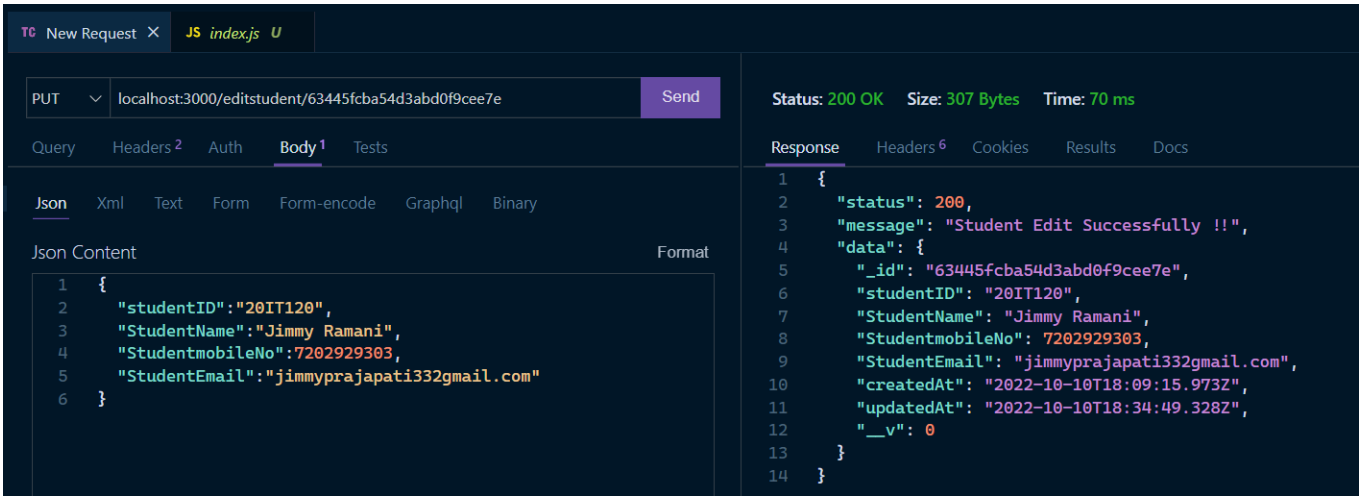
model > JS studentmodel.js > ...

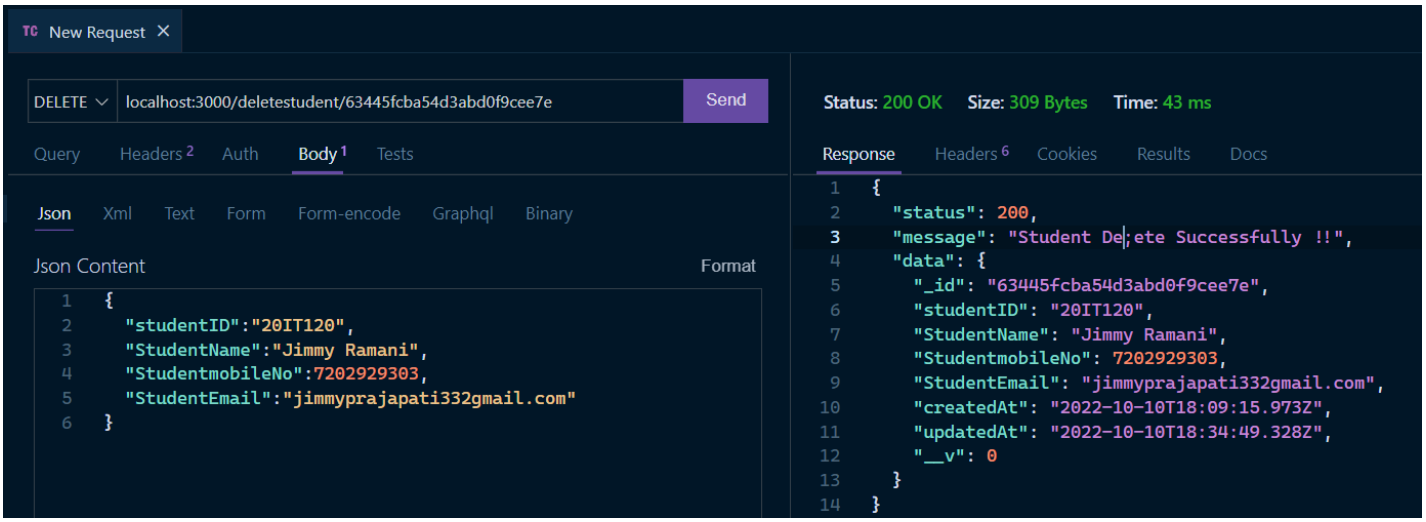
```
import mongoose from "mongoose";

const { Schema } = mongoose;

const StudentSchema = new mongoose.Schema(
  {
    studentID: {
      type: String,
      required: true,
    },
    StudentName: {
      type: String,
      required: true,
    },
    Studentmobileno: {
      type: Number,
      required: false,
      unique: true, // define a unique value
      trim: true,
    },
    StudentEmail: {
      type: String,
      required: true,
      unique: true,
      trim: true,
      lowercase: true,
    },
  },
  { timestamps: true } // show created and updated time in database
);

export default mongoose.model("Student", StudentSchema);
```

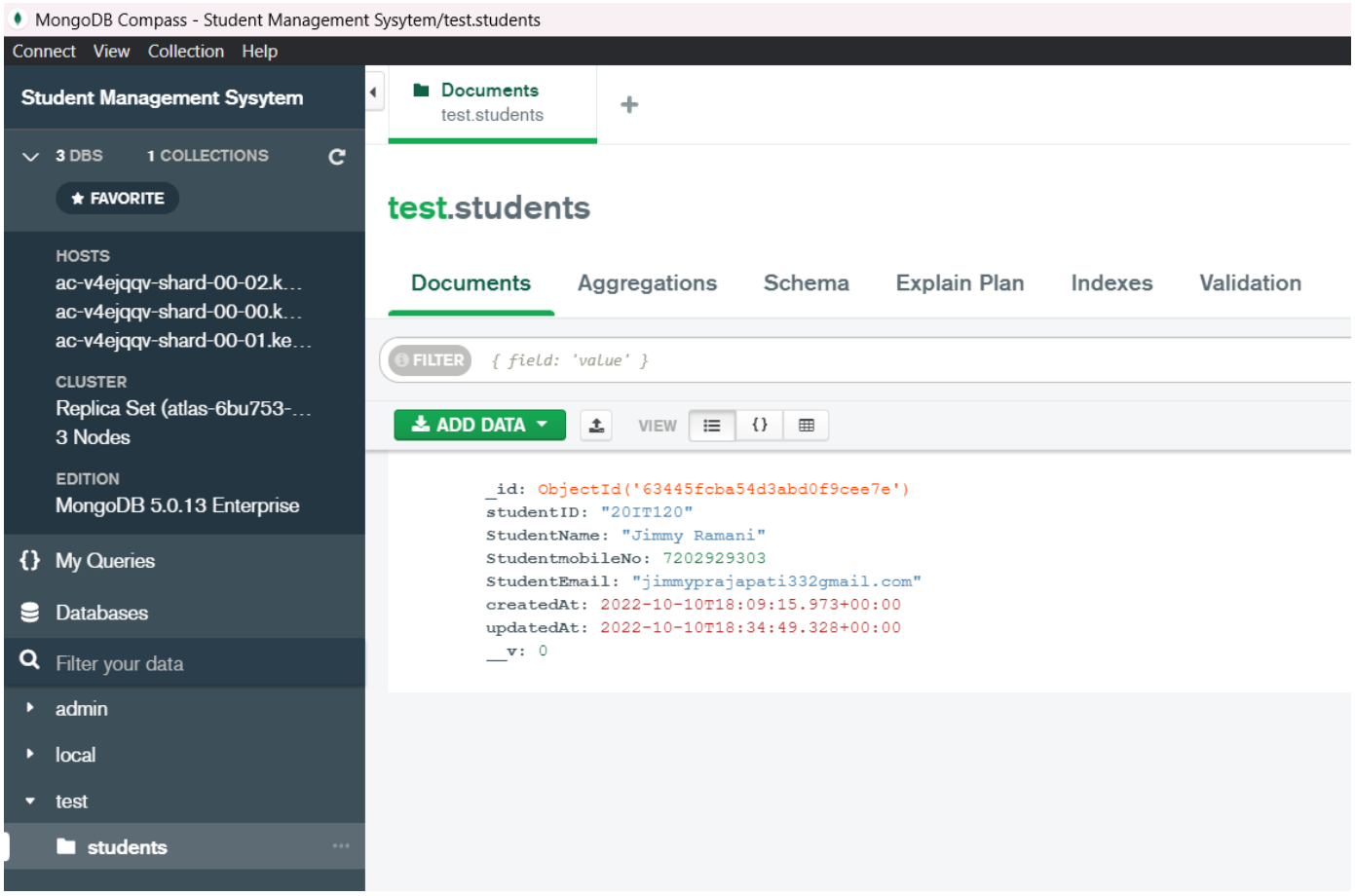
OUTPUT :


rminal Help New Request - D:\Node Js\COLLAGE PRACTICALS\PRACTICALS\PRACTICAL_2 - PRACTICAL_2 - Visual Studio Code
New Request
POST localhost:3000/addstudent Send
Query Headers 2 Auth Body 1 Tests
Json Xml Text Form Form-encode Graphql Binary
Json Content Format
{
"studentID":"20IT120",
"StudentName":"Jimmy Prajapati",
"StudentmobileNo": 7202929303,
"StudentEmail":"jimmyprajapati332gmail.com"
}
Status: 200 OK Size: 311 Bytes Time: 63 ms
Response Headers 6 Cookies Results Docs
{
"status": 200,
"message": "Student Added Successfully !!",
"data": {
"studentID": "20IT120",
"StudentName": "Jimmy Prajapati",
"StudentmobileNo": 7202929303,
"StudentEmail": "jimmyprajapati332gmail.com",
"_id": "63445fcba54d3abd0f9cee7e",
"createdAt": "2022-10-10T18:09:15.973Z",
"updatedAt": "2022-10-10T18:09:15.973Z",
"__v": 0
}
}


New Request
GET localhost:3000/liststudent Send
Query Headers 2 Auth Body Tests
Query Parameters
parameter value
Status: 200 OK Size: 316 Bytes Time: 43 ms
Response Headers 6 Cookies Results Docs
{
"status": 200,
"message": "Student List Get Successfully !!",
"data": [
{
"_id": "63445fcba54d3abd0f9cee7e",
"studentID": "20IT120",
"StudentName": "Jimmy Prajapati",
"StudentmobileNo": 7202929303,
"StudentEmail": "jimmyprajapati332gmail.com",
"createdAt": "2022-10-10T18:09:15.973Z",
"updatedAt": "2022-10-10T18:09:15.973Z",
"__v": 0
}
]
}


New Request
JS index.js U
PUT localhost:3000/editstudent/63445fcba54d3abd0f9cee7e Send
Query Headers 2 Auth Body 1 Tests
Json Xml Text Form Form-encode Graphql Binary
Json Content Format
{
"studentID":"20IT120",
"StudentName":"Jimmy Ramani",
"StudentmobileNo":7202929303,
"StudentEmail":"jimmyprajapati332gmail.com"
}
Status: 200 OK Size: 307 Bytes Time: 70 ms
Response Headers 6 Cookies Results Docs
{
"status": 200,
"message": "Student Edit Successfully !!",
"data": {
"_id": "63445fcba54d3abd0f9cee7e",
"studentID": "20IT120",
"StudentName": "Jimmy Ramani",
"StudentmobileNo": 7202929303,
"StudentEmail": "jimmyprajapati332gmail.com",
"createdAt": "2022-10-10T18:09:15.973Z",
"updatedAt": "2022-10-10T18:34:49.328Z",
"__v": 0
}
}


New Request
DELETE
localhost:3000/deletestudent/63445fcba54d3abd0f9cee7e
Send
Query
Headers 2
Auth
Body 1
Tests
Json
Xml
Text
Form
Form-encode
Graphql
Binary
Json Content
Format
{
"studentID":"20IT120",
"StudentName":"Jimmy Ramani",
"StudentmobileNo":7202929303,
"StudentEmail":"jimmyprajapati332gmail.com"
}
Status: 200 OK
Size: 309 Bytes
Time: 43 ms
Response
Headers 6
Cookies
Results
Docs
{
"status": 200,
"message": "Student Delete Successfully !!",
"data": {
"_id": "63445fcba54d3abd0f9cee7e",
"studentID": "20IT120",
"StudentName": "Jimmy Ramani",
"StudentmobileNo": 7202929303,
"StudentEmail": "jimmyprajapati332gmail.com",
"createdAt": "2022-10-10T18:09:15.973Z",
"updatedAt": "2022-10-10T18:34:49.328Z",
"__v": 0
}
}


MongoDB Compass - Student Management Sysytem/test.students
Connect
View
Collection
Help
Student Management Sysytem
3 DBS
1 COLLECTIONS
FAVORITE
HOSTS
ac-v4ejqqv-shard-00-02.k...
ac-v4ejqqv-shard-00-00.k...
ac-v4ejqqv-shard-00-01.ke...
CLUSTER
Replica Set (atlas-6bu753-...
3 Nodes
EDITION
MongoDB 5.0.13 Enterprise
My Queries
Databases
Filter your data
admin
local
test
students
Documents
test.students
test.students
Documents
Aggregations
Schema
Explain Plan
Indexes
Validation
FILTER
{ field: 'value' }
ADD DATA
VIEW
_id: ObjectId('63445fcba54d3abd0f9cee7e')
studentID: "20IT120"
StudentName: "Jimmy Ramani"
StudentmobileNo: 7202929303
StudentEmail: "jimmyprajapati332gmail.com"
createdAt: 2022-10-10T18:09:15.973+00:00
updatedAt: 2022-10-10T18:34:49.328+00:00
__v: 0