

Assignment -3 : RESTful API Creation

Name : Adigopula Nikhitha

Email : 208x1a4201@khitguntur.ac.in

Phone No : 9398751981

Roll No : 208X1A4201

College Name : Kallam Haranadhareddy Institute Of Technology

Source Code :

- **models→course.js**

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

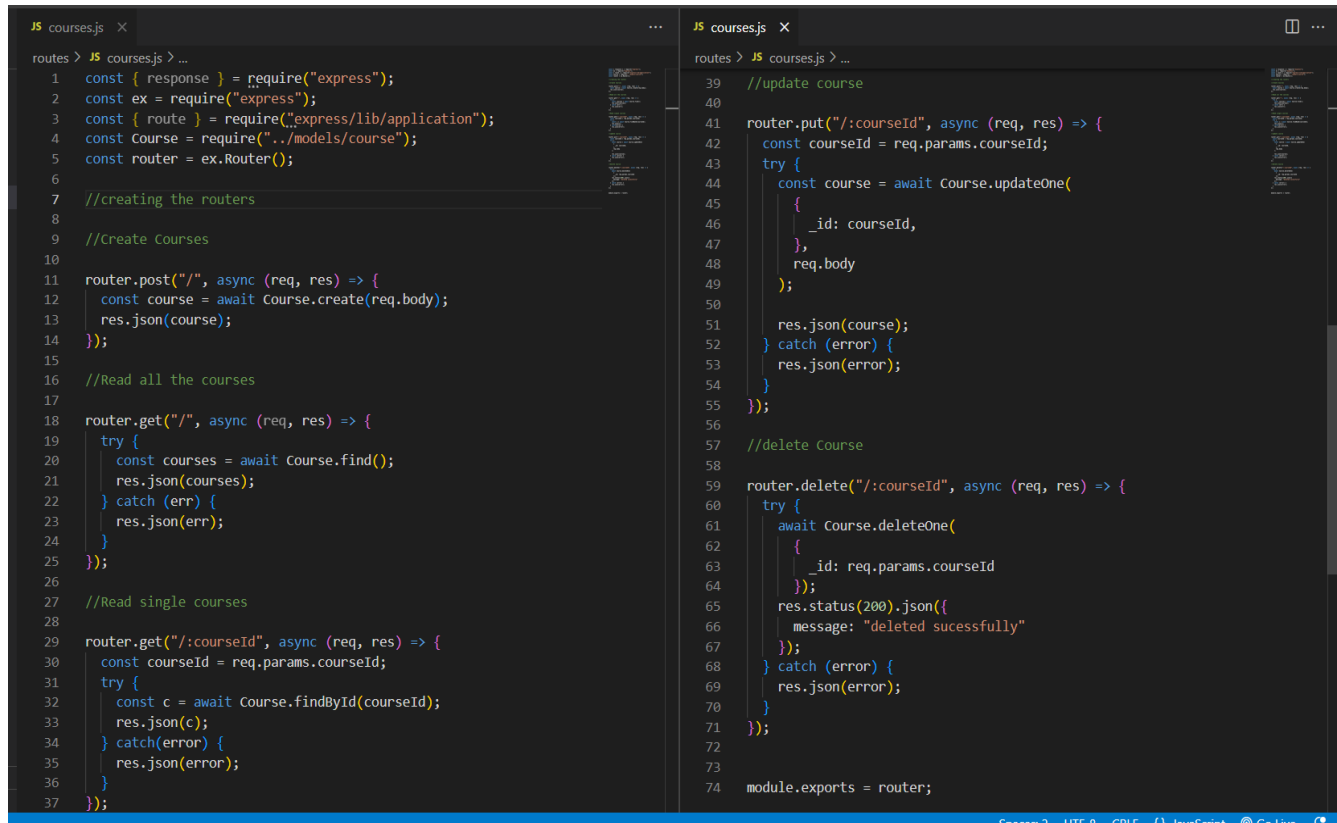
const Course = mongoose.Schema({
  title: {
    type: String,
    require: true,
  },
  content: {
    type: String,
    require: true,
  },
  videos: {
    type: Number,
    require: true,
  },
  active: Boolean,
});

module.exports = mongoose.model("courses", Course);
```

- **.env file :**

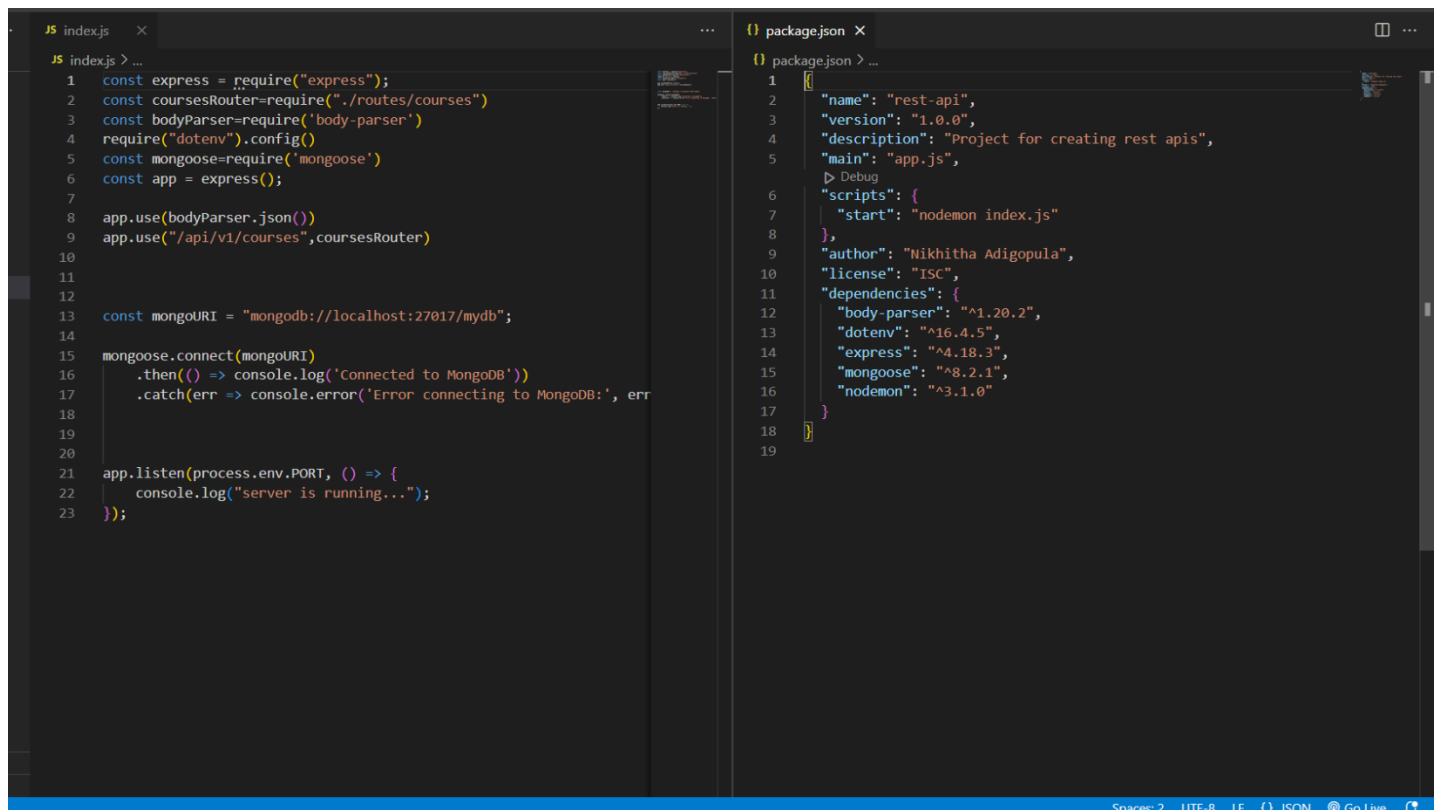
```
PORT=2000
DB_CONNECTION_URL=mongodb://localhost:27017/mydb
```

- routes → courses.js :



```
JS courses.js X
routes > JS courses.js > ...
1 const { response } = require("express");
2 const ex = require("express");
3 const { route } = require("express/lib/application");
4 const Course = require("../models/course");
5 const router = ex.Router();
6
7 //creating the routers
8
9 //Create Courses
10
11 router.post("/", async (req, res) => {
12   const course = await Course.create(req.body);
13   res.json(course);
14 });
15
16 //Read all the courses
17
18 router.get("/", async (req, res) => {
19   try {
20     const courses = await Course.find();
21     res.json(courses);
22   } catch (err) {
23     res.json(err);
24   }
25 });
26
27 //Read single courses
28
29 router.get("/:courseId", async (req, res) => {
30   const courseId = req.params.courseId;
31   try {
32     const c = await Course.findById(courseId);
33     res.json(c);
34   } catch (error) {
35     res.json(error);
36   }
37 });
38
39 //update course
40
41 router.put("/:courseId", async (req, res) => {
42   const courseId = req.params.courseId;
43   try {
44     const course = await Course.updateOne(
45       {
46         _id: courseId,
47       },
48       req.body
49     );
50
51     res.json(course);
52   } catch (error) {
53     res.json(error);
54   }
55 });
56
57 //delete Course
58
59 router.delete("/:courseId", async (req, res) => {
60   try {
61     await Course.deleteOne(
62       {
63         _id: req.params.courseId
64       }
65     );
66     res.status(200).json({
67       message: "deleted sucessfully"
68     });
69   } catch (error) {
70     res.json(error);
71   }
72 });
73
74 module.exports = router;
```

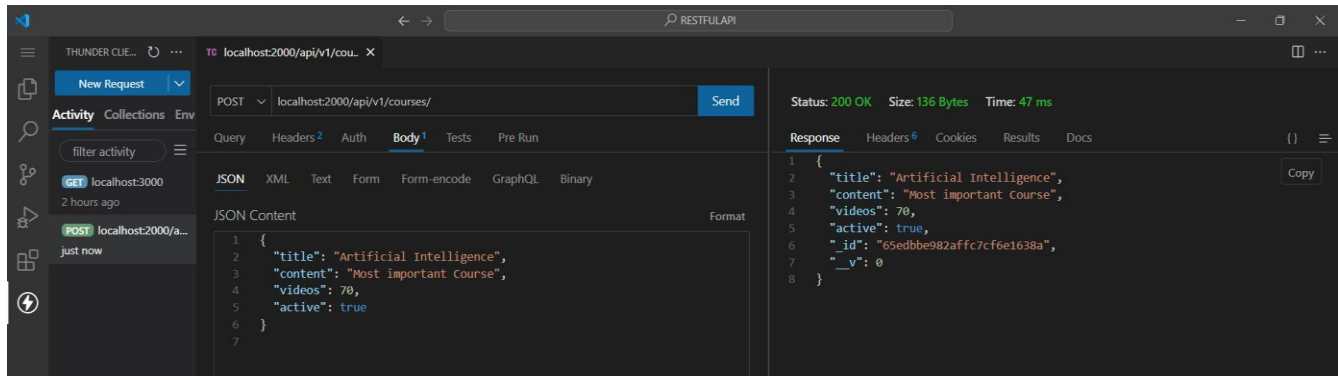
- index.js & package.json :



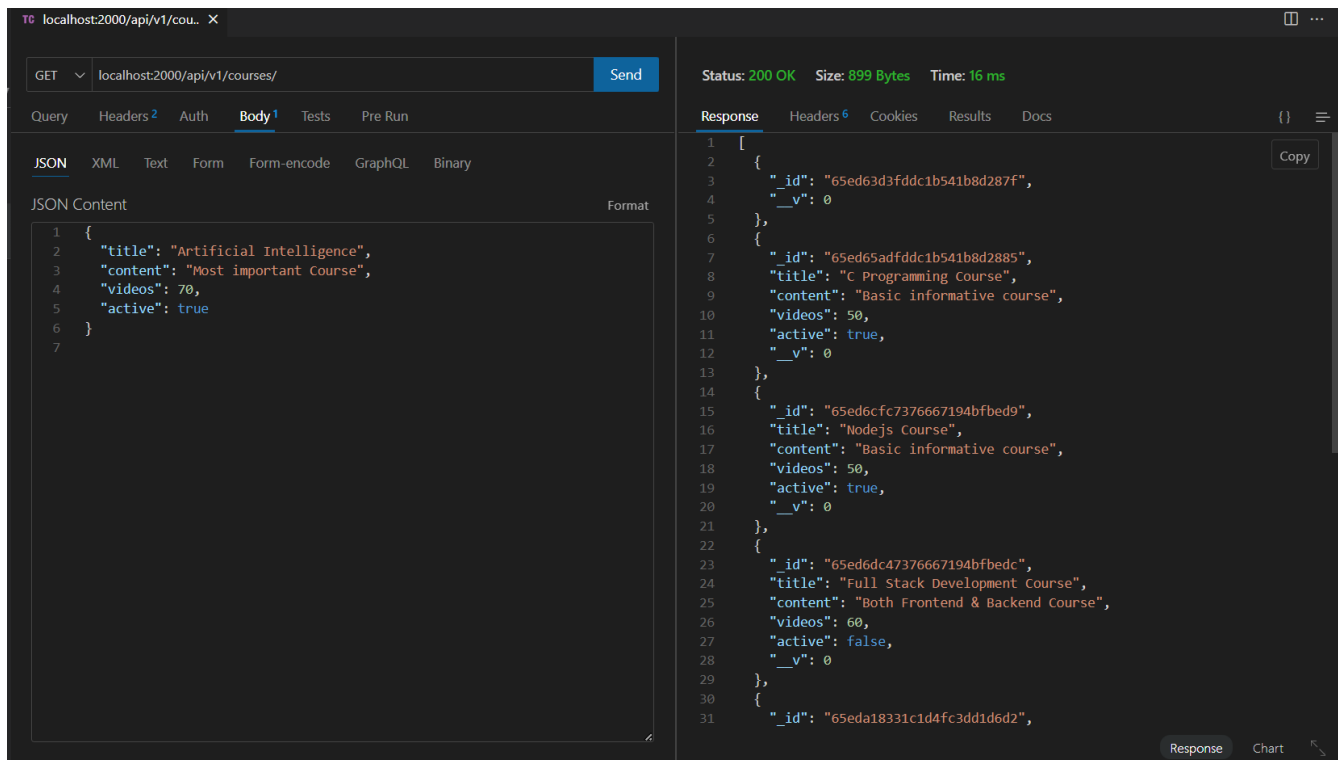
```
JS index.js X
JS index.js > ...
1 const express = require("express");
2 const coursesRouter=require("./routes/courses")
3 const bodyParser=require('body-parser')
4 require("dotenv").config()
5 const mongoose=require('mongoose')
6 const app = express();
7
8 app.use(bodyParser.json())
9 app.use("/api/v1/courses",coursesRouter)
10
11
12
13 const mongoURI = "mongodb://localhost:27017/mydb";
14
15 mongoose.connect(mongoURI)
16   .then(() => console.log('Connected to MongoDB'))
17   .catch(err => console.error('Error connecting to MongoDB:', err));
18
19
20
21 app.listen(process.env.PORT, () => {
22   console.log("server is running...");
23 });

() package.json X
() package.json > ...
1 {
2   "name": "rest-api",
3   "version": "1.0.0",
4   "description": "Project for creating rest apis",
5   "main": "app.js",
6   "scripts": {
7     "start": "nodemon index.js"
8   },
9   "author": "Nikhitha Adigopula",
10  "license": "ISC",
11  "dependencies": {
12    "body-parser": "^1.20.2",
13    "dotenv": "^16.4.5",
14    "express": "^4.18.3",
15    "mongoose": "^8.2.1",
16    "nodemon": "^3.1.0"
17  }
18 }
19
```

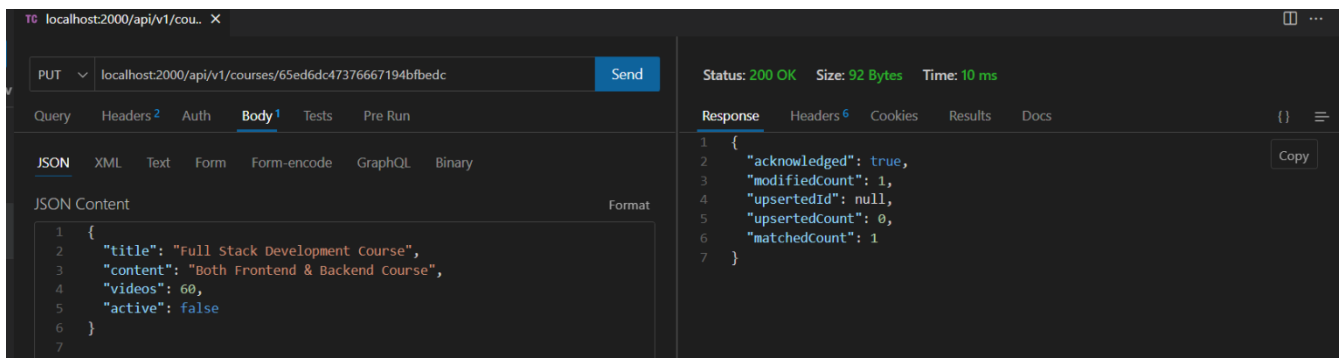
• Output : Create:



• Read:

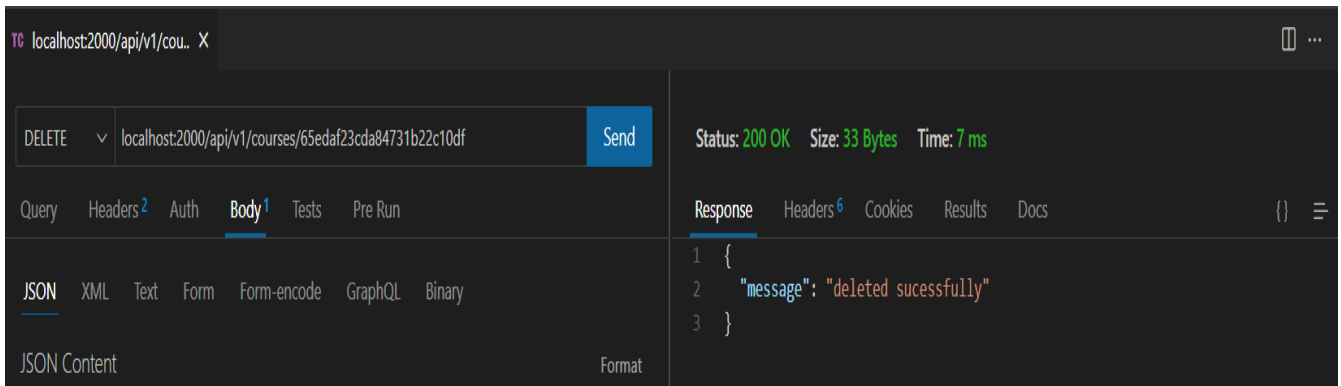


• Update:

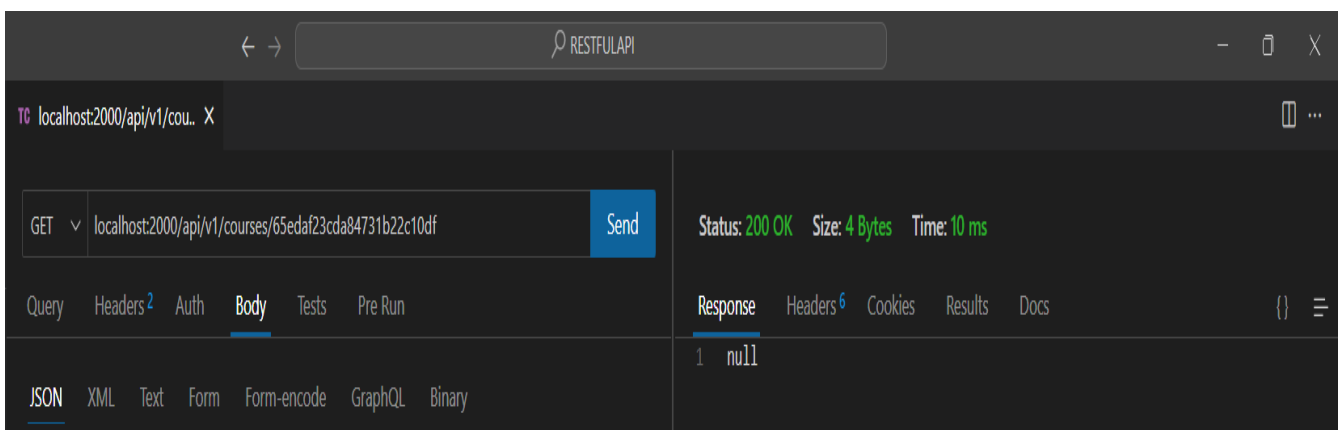


Delete:

- Before deleting:



- After deleting:



Brief Description :

- Create a new folder.
- Open that folder in any code editor (vs code).
- Open terminal (ctrl + ~) on code edit.
- In new folder create a .env file and create a PORT and variables and assign value
- now simply paste this code in .env file .

PORT=2000

DB_CONNECTION_URL=mongodb://localhost:27017/mydb

- After in terminal type this commands to install all dependencies and run server.

`npm i` and `npm start`

- Finally, if you get below message in terminal then your server will running successfully.

```
PS C:\Users\nikhi\OneDrive\Desktop\RESTFULAPI> npm start

> rest-api@1.0.0 start
> nodemon index.js

[nodemon] 3.1.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node index.js`
server is running...
Connected to MongoDB
```

- For this use any API using tools like Postman or Curl or Thunder Client.
- I used THUNDER CLIENT.

CREATE ROUTE :

- This route is used to create a new collection in database with a req fields.
- In thunder client click on new request and select this options
- method as post url as <http://localhost:2000/api/v1/courses/>
- pass this json data as a body as your required value.

```
{
  "title": "C Programming Course",
  "content": "Basic informative course",
  "videos": 50,
  "active": true
}
```

- Finally press send to insert data in mongodb data base and get a inserted data as a response.
- If user is already in db it will return course is already exist as response.

READ ONE : This is used to read specific course info by passing that course id as param.

method as get:

url as: [http://localhost:2000/api/v1/courses/ 65ed65adfddc1b541b8d2885](http://localhost:2000/api/v1/courses/65ed65adfddc1b541b8d2885)

- After clicking send you will get that specific course details as response.

READ ALL :

- Read all route is used to get all courses data existing in the mongodb data base .

method as get:

url as: <http://localhost:2000/api/v1/courses/>

- After clicking send you will get all courses details as response.

UPDATE :

- This route is used to update specific course by passing that course id as a param.

method as put:

url as: [http://localhost:2000/api/v1/update/ 65ed65adfddc1b541b8d2885](http://localhost:2000/api/v1/update/65ed65adfddc1b541b8d2885)

- After sending you will get updated course details as response.

DELETE :

- This route is used to delete specific course by passing that course id as a param.

method as delete:

url as: [http://localhost:2000/api/v1/delete/ 65ed65adfddc1b541b8d2885](http://localhost:2000/api/v1/delete/65ed65adfddc1b541b8d2885)

- After clicking send your data will be deleted successfully as response.

Source Code link : <https://github.com/Nikhitha4201/RESTful-API>