# Creating a RESTful API using express.js and creating a database and index in MongoDB.

**Name : Sadhanala.Anusha**

**email : anushasadhanala10@gmail.com**
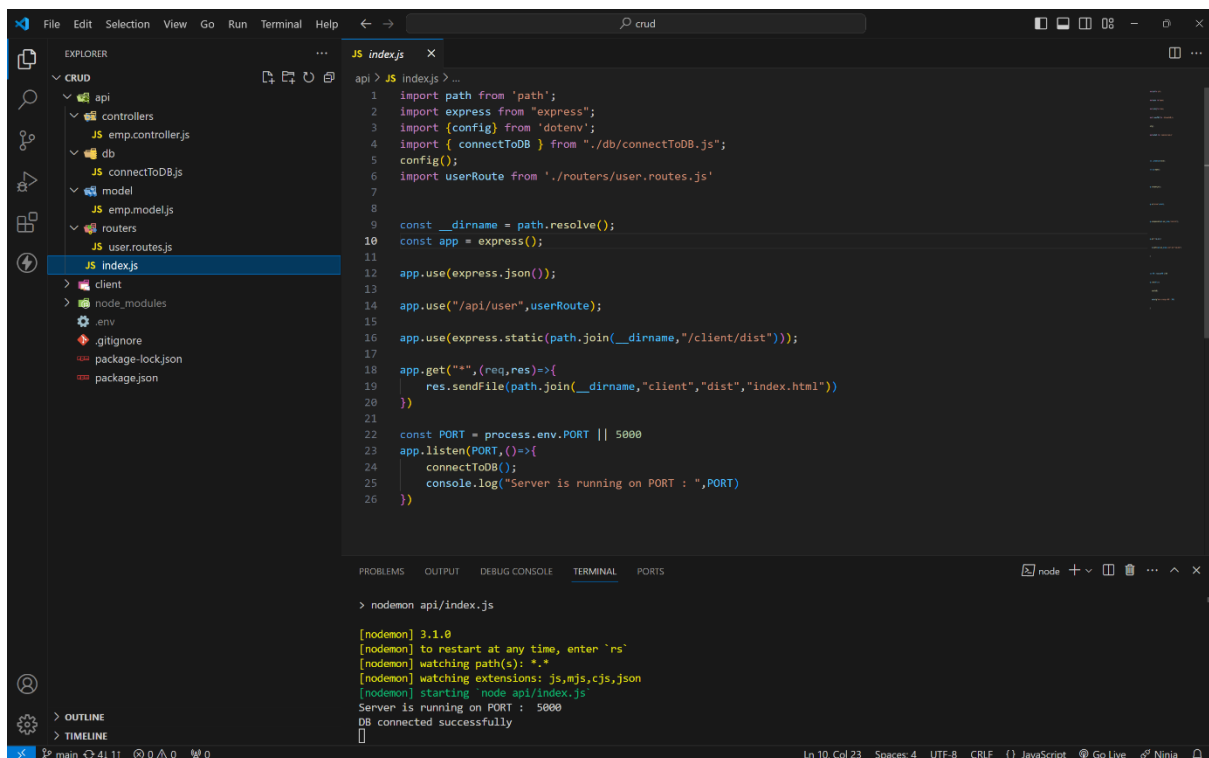
**Phone no : 9392494816**

**Roll no : 2221338**

**College name : P R Govt college (A)**

## source code :

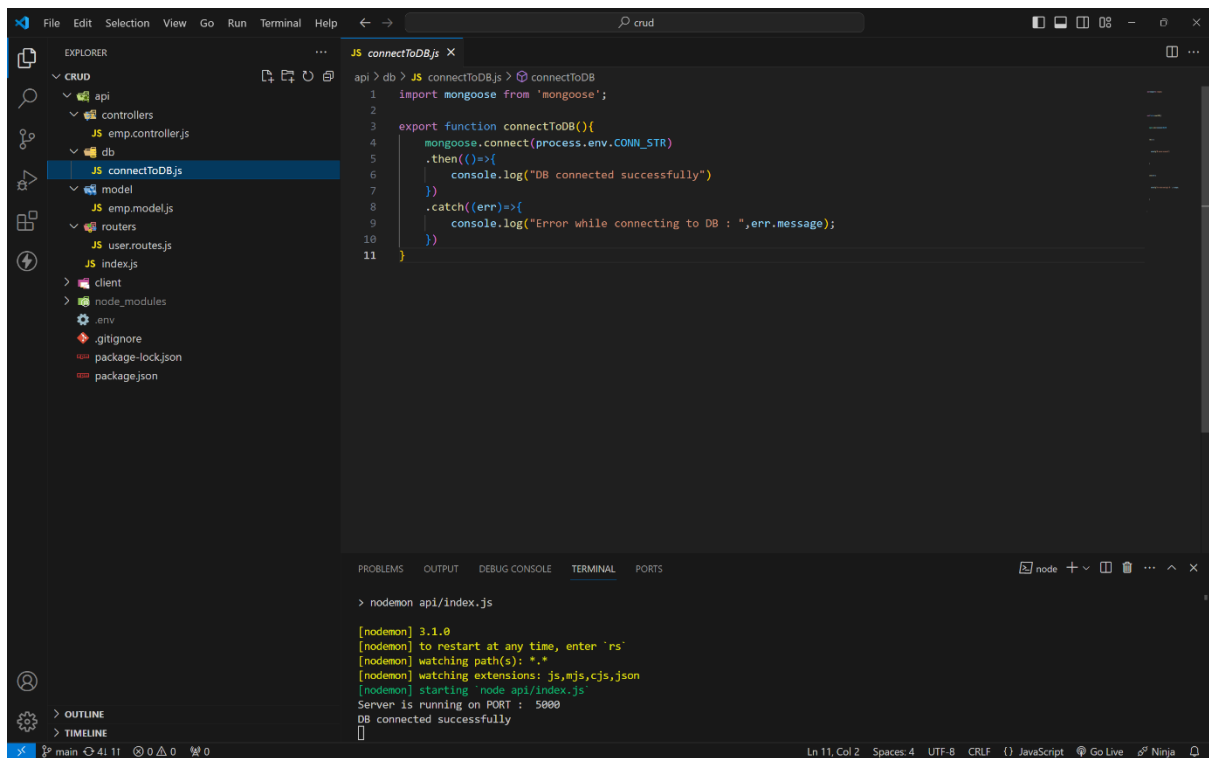### index.js file :

# MONGODB CONNECTION :



```javascript
import mongoose from 'mongoose';

export function connectToDB(){
    mongoose.connect(process.env.CONN_STR)
    .then(()=>{
        console.log("DB connected successfully")
    })
    .catch((err)=>{
        console.log("Error while connecting to DB : ",err.message);
    })
}
```

```
> nodemon api/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 api/index.js`
Server is running on PORT :  5000
DB connected successfully
```

# MODEL :



```javascript
import mongoose from 'mongoose';

const userSchema = new mongoose.Schema({
    username:{
        type:String,
        unique:true,
        required:true
    },
    empname:{
        type:String,
        required:true
    },
    email:{
        type:String,
        required:true
    },
    role:{
        type:String,
        required:true
    },
    salary:{
        type: Number,
        required: true,
    }
},{timestamps:true})

const Emp = mongoose.model("User",userSchema);

export default Emp;
```

```
> nodemon api/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 api/index.js`
Server is running on PORT :  5000
DB connected successfully
```

## ROUTES:



```javascript
import express from 'express'
import { create, readAll  , read,remove, update,  } from '../controllers/emp.controller.js';

const router = express.Router();

router.post('/create',create);
router.get("/readall",readAll);
router.get("/read/:id",read);
router.put('/update/:id',update);
router.delete('/remove/:id',remove);

export default router;
```
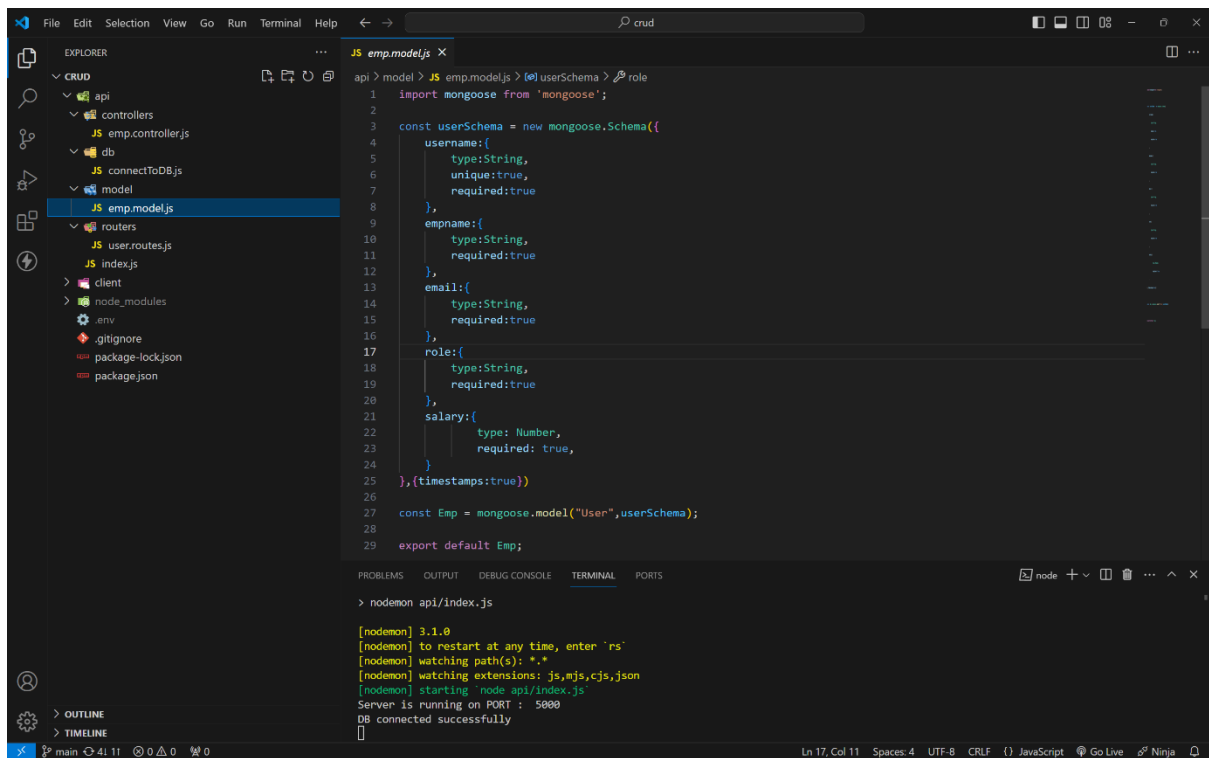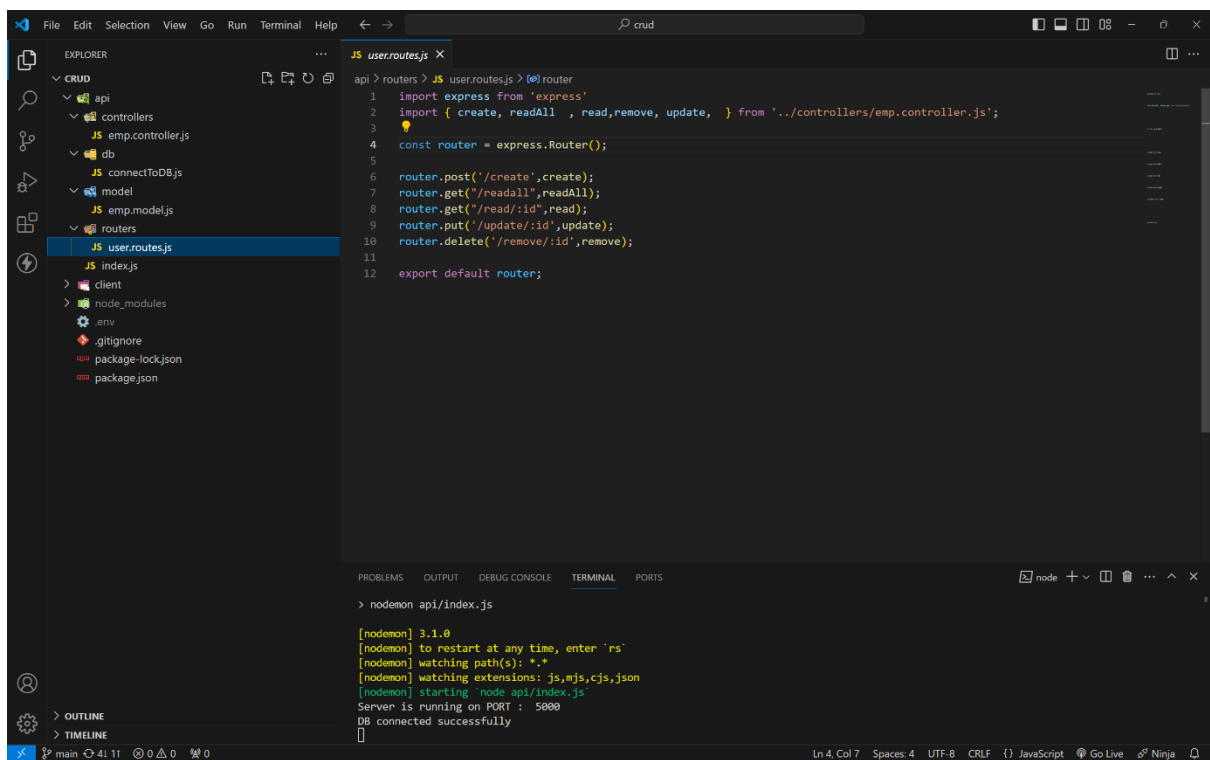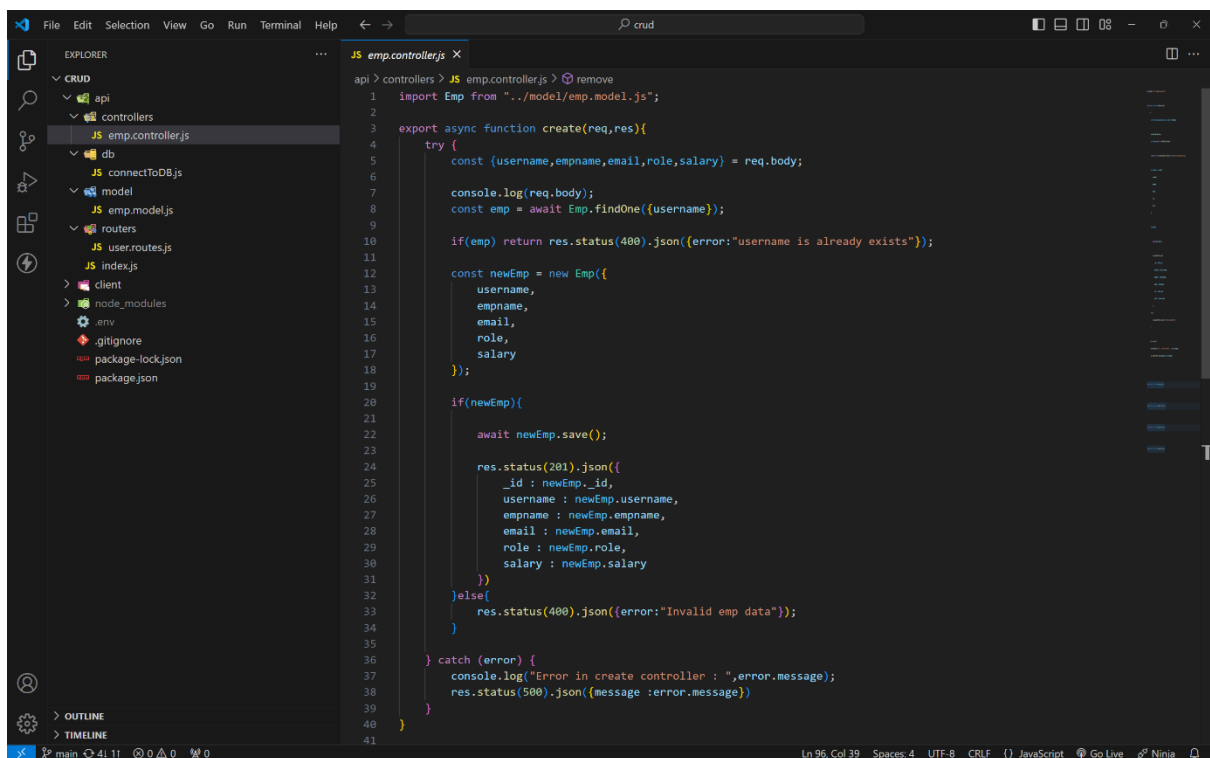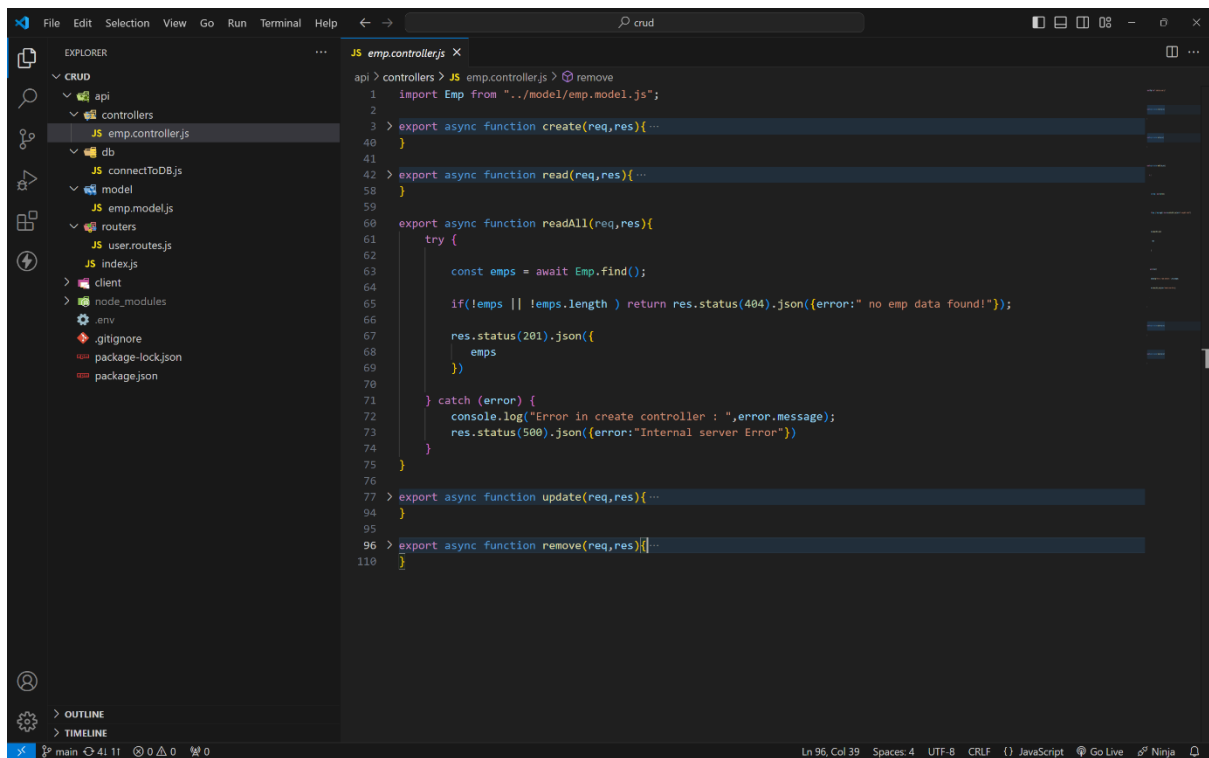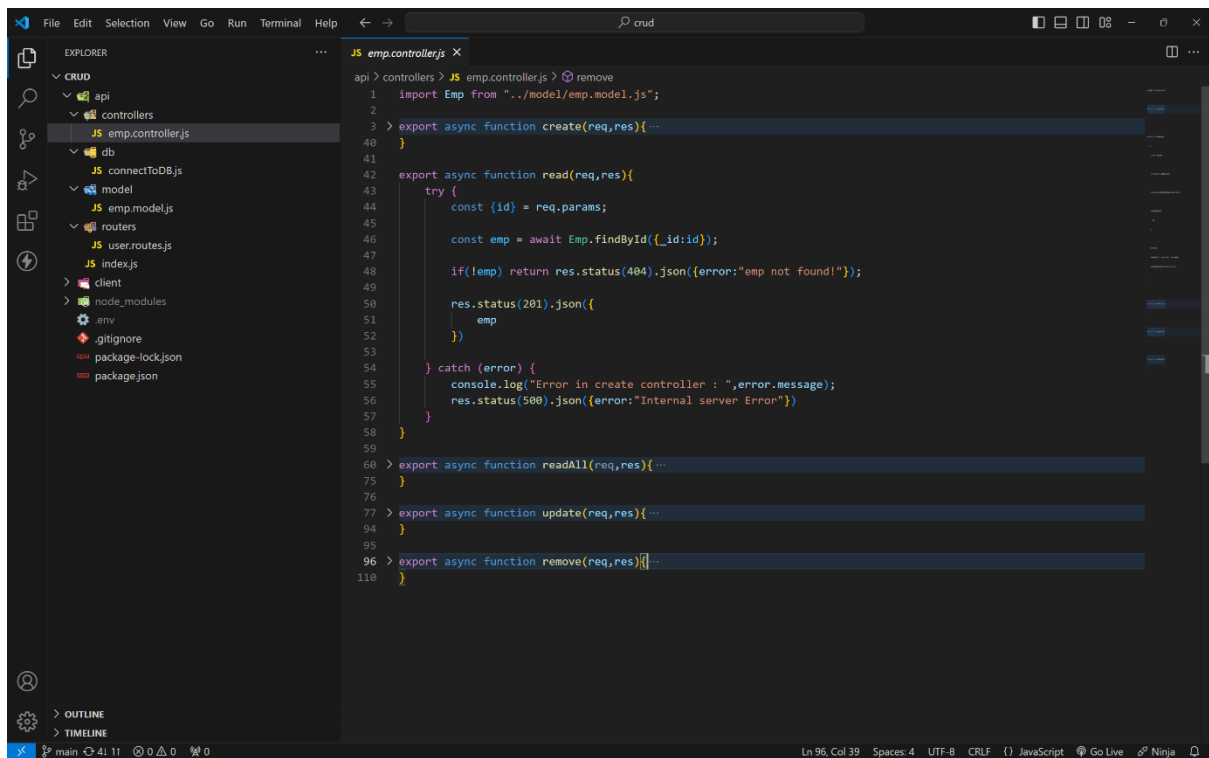
## CONTROLLERS :

## CREATE :



```javascript
import Emp from "../model/emp.model.js";

export async function create(req,res){
    try {
        const {username,empname,email,role,salary} = req.body;

        console.log(req.body);
        const emp = await Emp.findOne({username});

        if(emp) return res.status(400).json({error:"username is already exists"});

        const newEmp = new Emp({
            username,
            empname,
            email,
            role,
            salary
        });

        if(newEmp){

            await newEmp.save();

            res.status(201).json({
                _id : newEmp._id,
                username : newEmp.username,
                empname : newEmp.empname,
                email : newEmp.email,
                role : newEmp.role,
                salary : newEmp.salary
            })
        }else{
            res.status(400).json({error:"Invalid emp data"});
        }

    } catch (error) {
        console.log("Error in create controller : ",error.message);
        res.status(500).json({message :error.message})
    }
}
```

## READALL:

```
import Emp from "../model/emp.model.js";

export async function create(req,res){...
}

export async function read(req,res){...
}

export async function readAll(req,res){
    try {

        const emps = await Emp.find();

        if(!emps || !emps.length ) return res.status(404).json({error:" no emp data found!"});

        res.status(201).json({
            emps
        })

    } catch (error) {
        console.log("Error in create controller : ",error.message);
        res.status(500).json({error:"Internal server Error"})
    }
}

export async function update(req,res){...
}

export async function remove(req,res){...
}
```
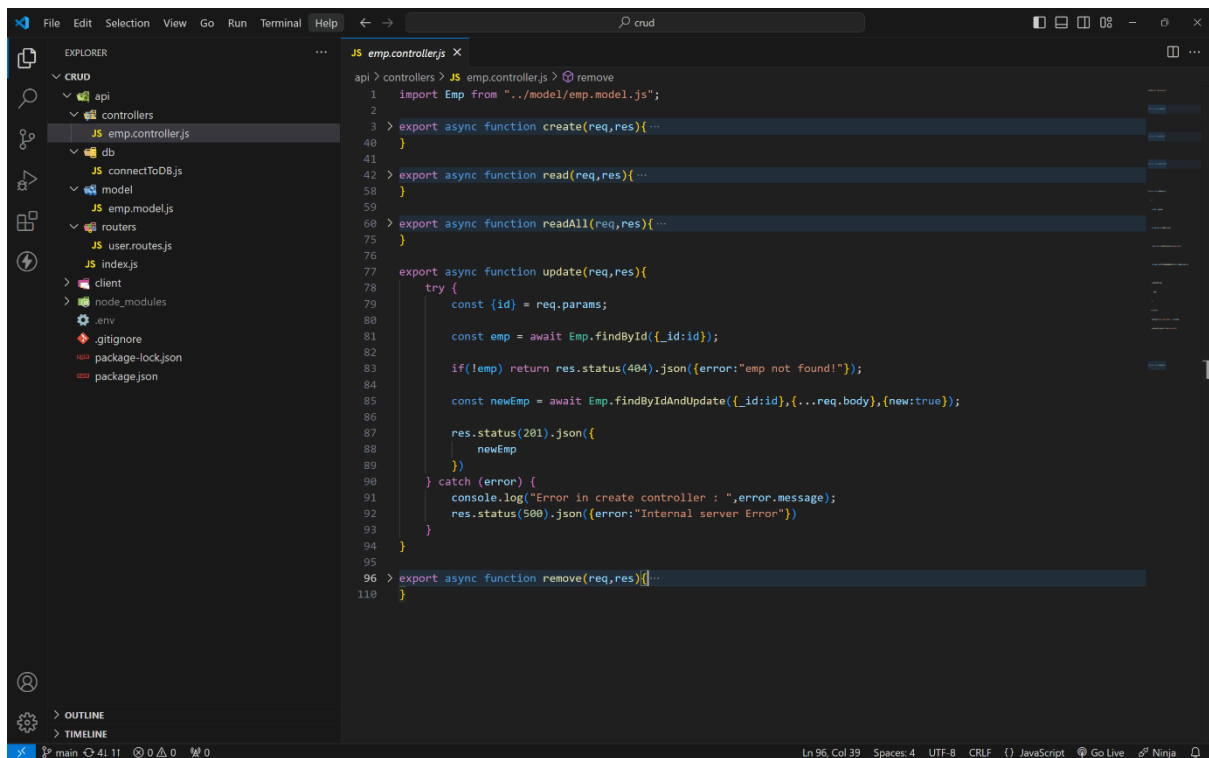
## READONE :

```
import Emp from "../model/emp.model.js";

export async function create(req,res){...
}

export async function read(req,res){
    try {

        const {id} = req.params;

        const emp = await Emp.findById({_id:id});

        if(!emp) return res.status(404).json({error:"emp not found!"});

        res.status(201).json({
            emp
        })

    } catch (error) {
        console.log("Error in create controller : ",error.message);
        res.status(500).json({error:"Internal server Error"})
    }
}

export async function readAll(req,res){...
}

export async function update(req,res){...
}

export async function remove(req,res){...
}
```

# UPDATE :

```javascript
import Emp from "../model/emp.model.js";

> export async function create(req,res){ ...
}

> export async function read(req,res){ ...
}

> export async function readAll(req,res){ ...
}

export async function update(req,res){
    try {
        const {id} = req.params;

        const emp = await Emp.findById({_id:id});

        if(!emp) return res.status(404).json({error:"emp not found!"});

        const newEmp = await Emp.findByIdAndUpdate({_id:id},{...req.body},{new:true});

        res.status(201).json({
            newEmp
        })
    } catch (error) {
        console.log("Error in create controller : ",error.message);
        res.status(500).json({error:"Internal server Error"})
    }
}

> export async function remove(req,res){ ...
}
```
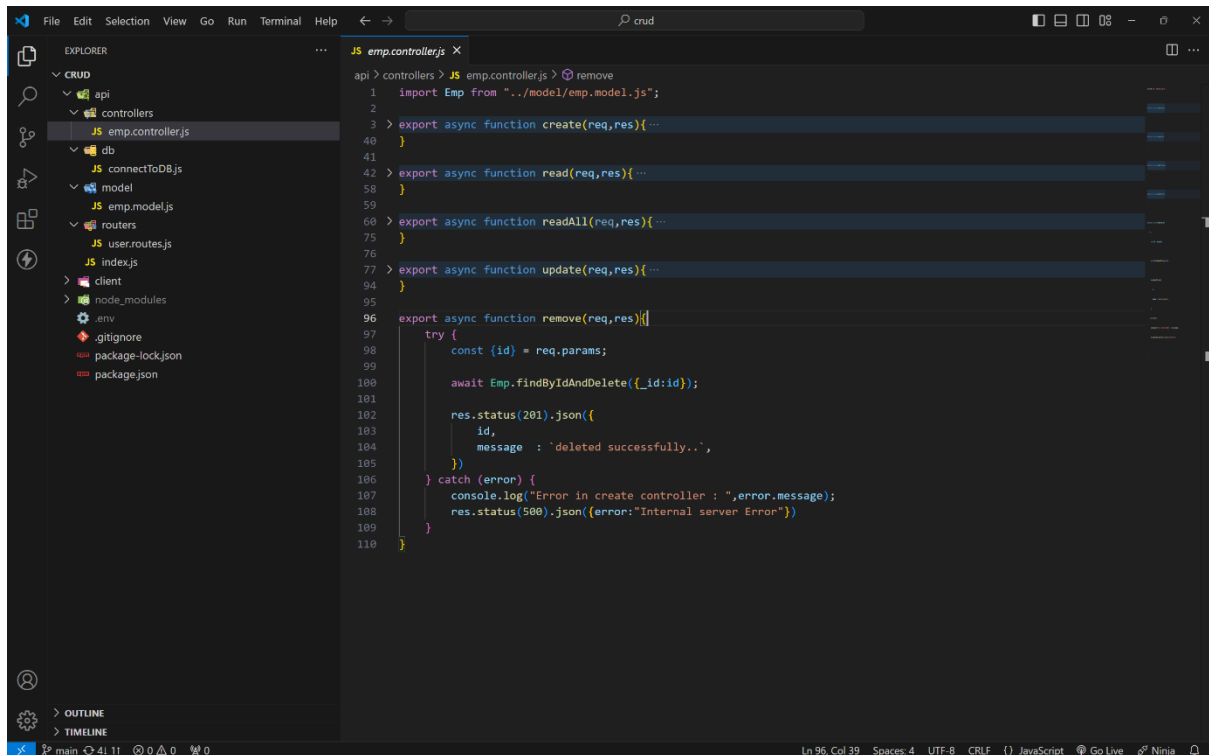
# DELETE :

```javascript
import Emp from "../model/emp.model.js";

> export async function create(req,res){ ...
}

> export async function read(req,res){ ...
}

> export async function readAll(req,res){ ...
}

> export async function update(req,res){ ...
}

export async function remove(req,res){
    try {
        const {id} = req.params;

        await Emp.findByIdAndDelete({_id:id});

        res.status(201).json({
            id,
            message  : `deleted successfully..`,
        })
    } catch (error) {
        console.log("Error in create controller : ",error.message);
        res.status(500).json({error:"Internal server Error"})
    }
}
```

# HOW TO RUN ON LOCALLY :

1 . **Create a folder as any name.**

2 . **Open that folder in any code editor (vs code).**

3 . **Open terminal ( ctrl + ~ ) on code editor.**

4 . **Type this code to get code locally.**

5 . **Now move to crud folder (cd crud in terminal)**

6 . **Ignore client folder.**

7 . **Here crud is root folder.**

8 . **In root folder create a .env file and create a PORT and**

    **CONN_STR variables and assign value.**

**ex : PORT = 3000     ( commonly any number between 3000 - 8080).**

    **CONN_STR = your mongodb_connection_string.**
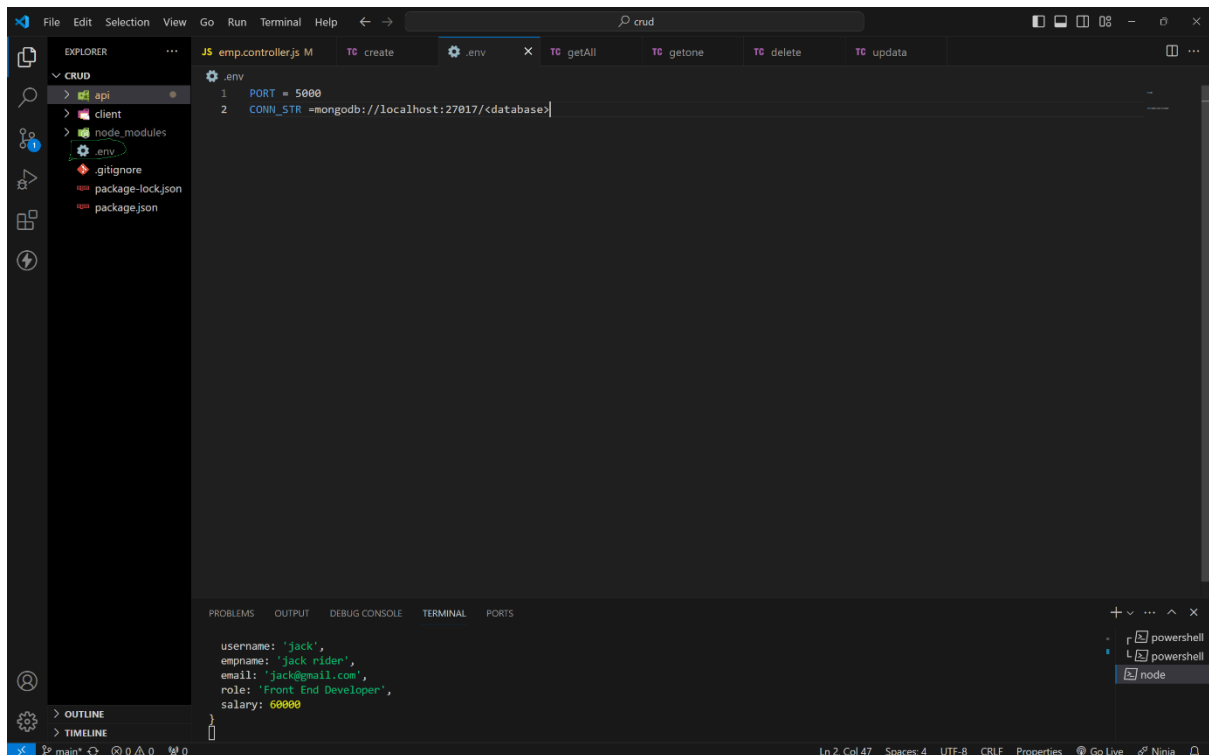
**--- trouble in above process ? :**

**simply paste this code in .env file .**

PORT = 5000

CONN_STR=mongodb+srv://bhavanidibbidi:rough@cluster0.wbclvtg.mongodb.net /?retryWrites=true&w=majority&appName=Cluster0

**9 . After in terminal (in crud folder as root folder) type this command to run server.**

**npm i (installing all dependencies)**

**npm run dev (to run server)**

**10 . if you get below message in terminal then your server will running successfully.**

```
PS C:\Users\4727y\OneDrive\Desktop\internshala\crud> npm run dev

> crud@1.0.0 dev
> nodemon api/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 api/index.js`
Server is running on PORT :  5000
DB connected successfully
```

# route and its functionality :

For this use any API using tools like Postman or Thunder Client.

i use THUNDER CLIENT.

## CREATE ROUTE :

1 . This route is used to create a new employee in database with a

below fields.

username, empname, email, role, salary

2 . in thunder client click on new request and select this options

method as post

url as http://localhost:5000/api/user/create

pass this json data as a body as your required value.

{

"username": "jack",

"empname": "jack rider",

"email": "jack@gmail.com",

"role": "Front End Developer",

"salary": 60000

}

3 . finally press send to insert data in mongodb data base and get a

inserted

data as a response.

4 . If user is already in db it will return User is already exist as

response.

for more details visit below output images...

**READONE :**

   **1 . This route is used to read specific user info by passing that user id**

     **as a param.**

      **method as get**

      **url as**
      http://localhost:5000/api/user/read/65ed7b3d76e1dcc9a51654ca

   **2 . After sending you will get that specific user details as response.**

**READALL :**

   **1 . Read all route is used to get all the user data existing in the mongodb**

     **data  base .**

      **method as get**

      **url as** http://localhost:5000/api/user/readall

   **2 . After sending you will get that all user details as response.**

**UPDATE :**

   **1 . This route is used to update specific user by passing that user id as**

     **a param.**

      **method as put**

      **url as**
      http://localhost:5000/api/user/update/65ed7b3d76e1dcc9a51654ca

   **2 . After sending you will get updated user details as response.**

**DELETE :**

**1 . This route is used to delete specific user by passing that user id as a param.**
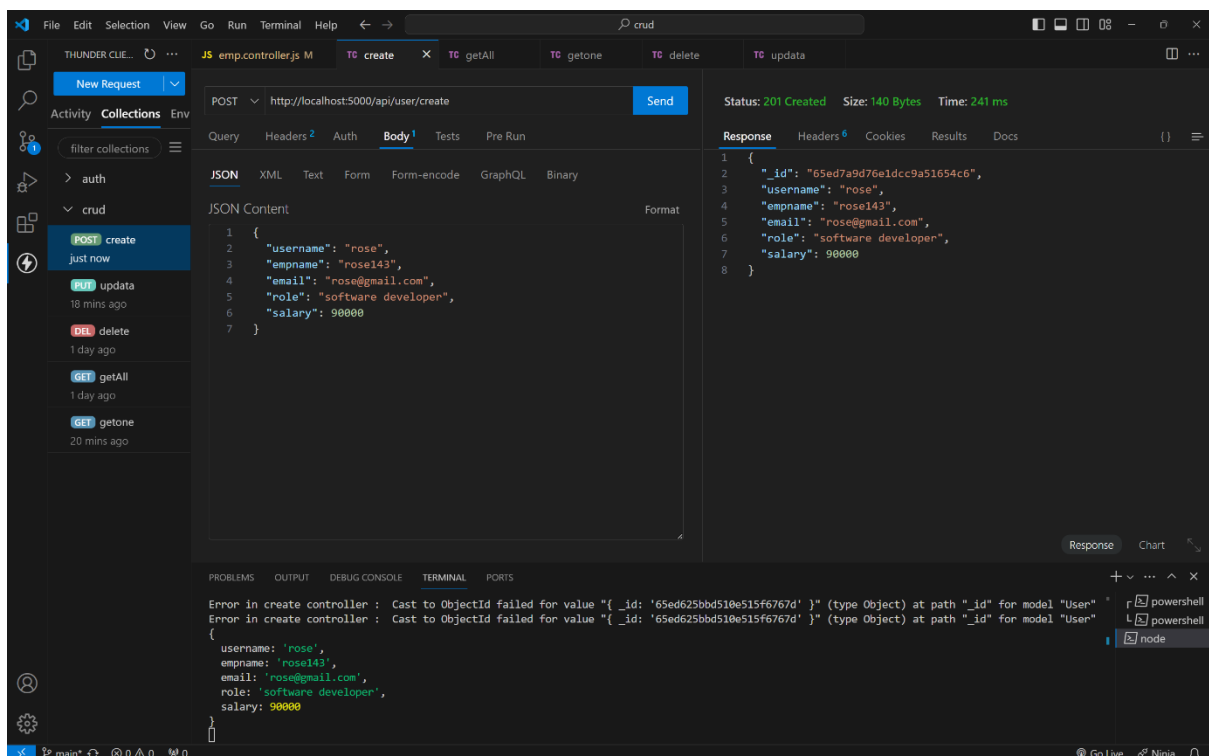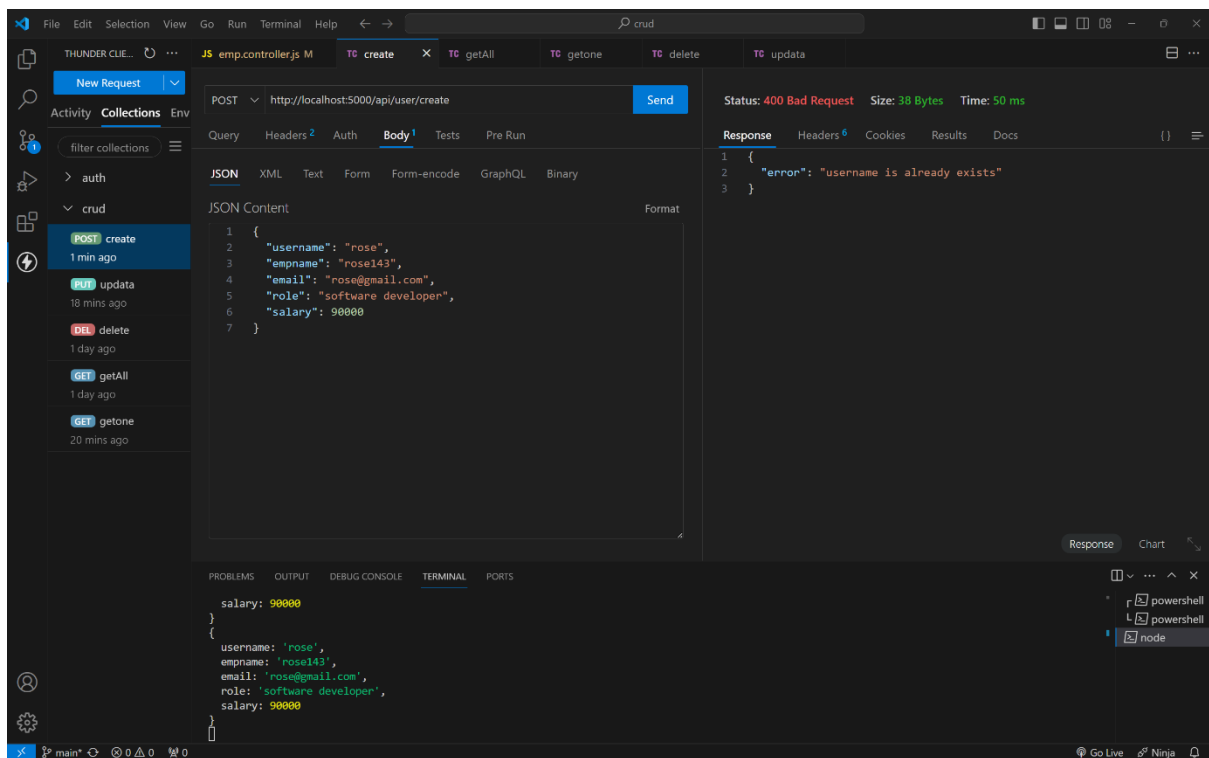
**method as delete**

**url as http://localhost:5000/api/user/delete/65ed7b3d76e1dcc9a51654ca**

**2 . After sending you will deleted successfully as response.**

**OUTPUT :**
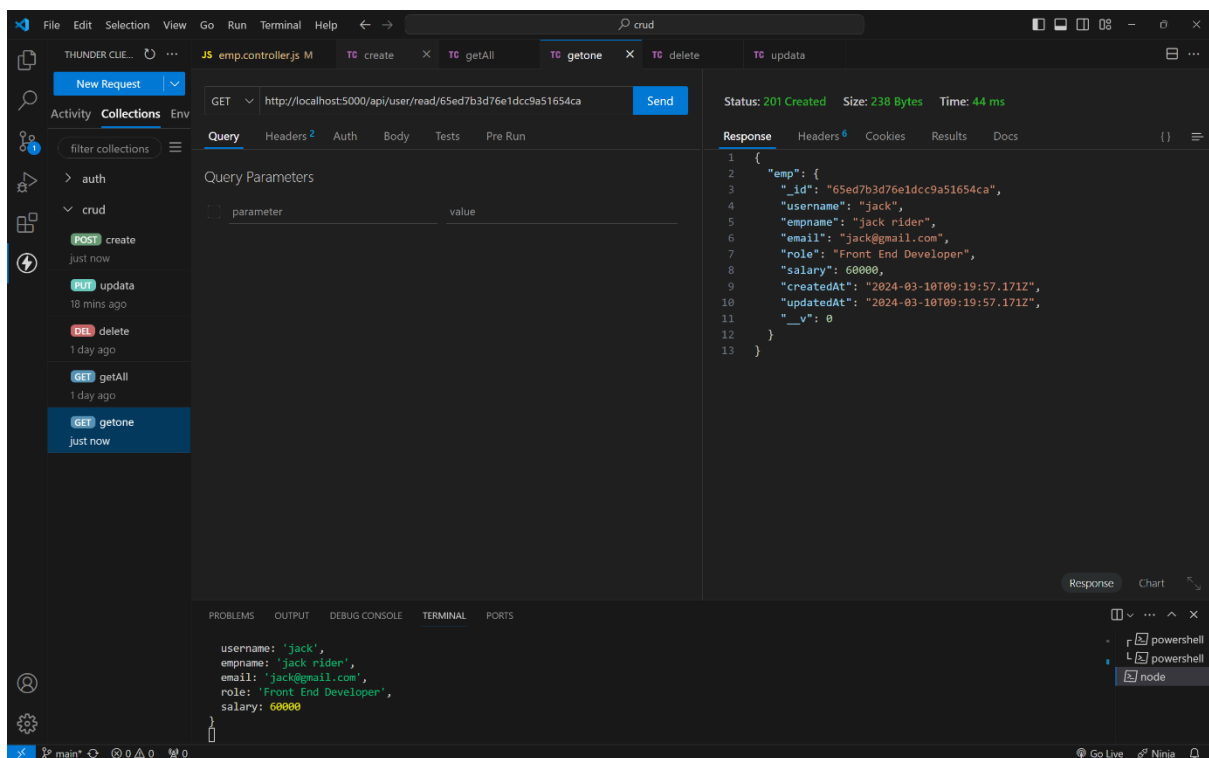
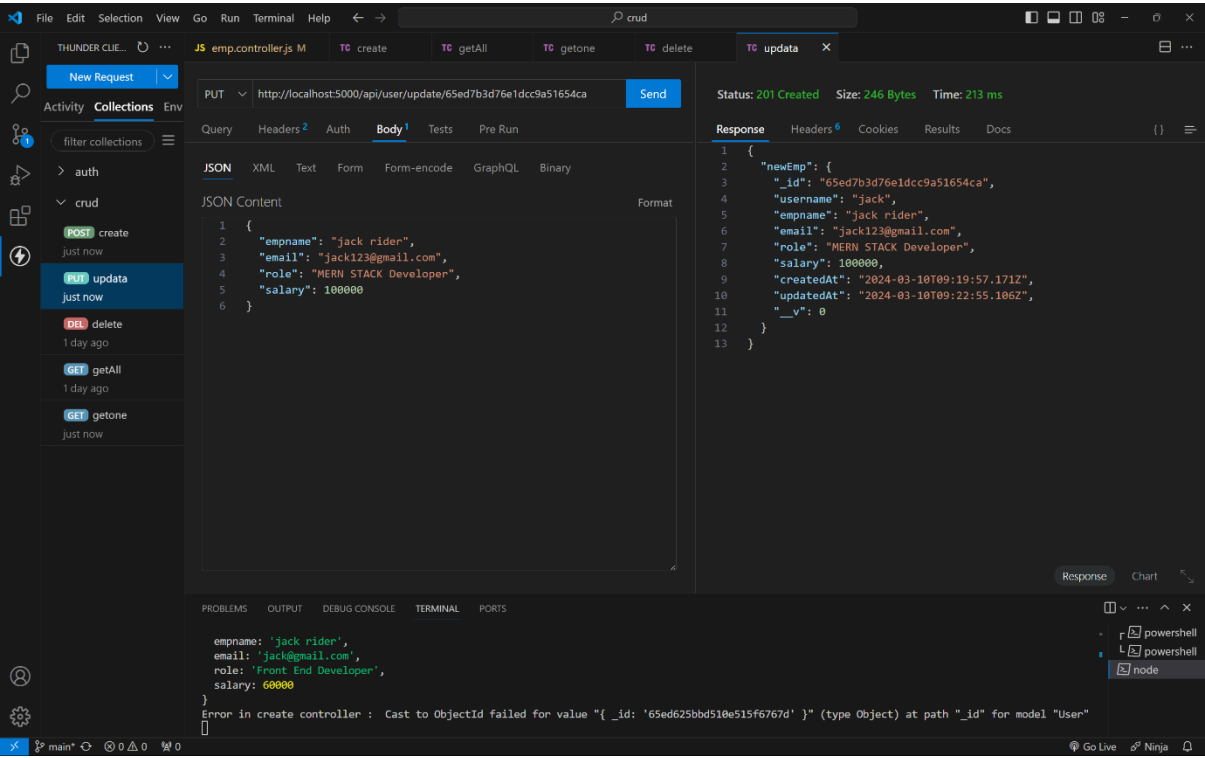**CREATE A NEW USER :**

# CREATING USER WITH EXISTING USERNAEM :



# READONE :

## READ ALL :



## UPDATE :

**DELETE :**