Sd.Shabana Azmi

IV CSE-C (20NN1A0509)

VNITSW, Guntur, AP.

ASSIGNMENT-3

Create a RESTFUL API using express.js and integrate it with MongoDB to perform CRUD operations on a database.

Code :

To set up a express.js :

>> npm init -y

>> npm i express

>> npm i mongodb

>> npm i mongoose

Create a file index.js:

```
const express=require("express")
const mongoose=require("mongoose")

const app=express()
const url='mongodb://localhost/Assignment3'
app.use(express.json())

mongoose.connect(url,{useNewUrlParser:true})

const con=mongoose.connection

con.on('open',()=>{
console.log('connected.....')
})

const router=require('./routes/route')
```

```
app.use('/route',router)


app.listen(3000, ()=>{
console.log("server started")
})
```

Created a routes folder for routing :

In the routes folder, create a route.js file.

```
const express=require('express')
const router=express.Router()
const Rout=require('../models/rout')
router.get('/',async(req,res)=>{
    try{

      const route=await Rout.find()
      res.json(route)

    }catch(err){
         res.send('Error'+err)
   }
})
router.get('/:id',async(req,res)=>{
    try{
       const route=await Rout.findById(req.params.id)
      res.json(route)
    }catch(err){
         res.send('Error'+err)
    }
})
router.post('/',async(req,res)=>{
    const r=new Rout({
        username: req.body.username,
        password: req.body.password
    })
    try{
        const a1=await r.save()
        res.json(a1)
    }catch(err){
        res.send('Error'+err)
    }
```

```
})
router.patch('/:id',async(req,res)=>{
  try{
    const r=await Rout.findById(req.params.id)
    r.password=req.body.password
    const a1=await r.save()
    res.json(a1)
  }
  catch(err){
    res.send('Error'+err)
  }
})
module.exports=router
```

Now for defining the database model we need to create the schema of the data.

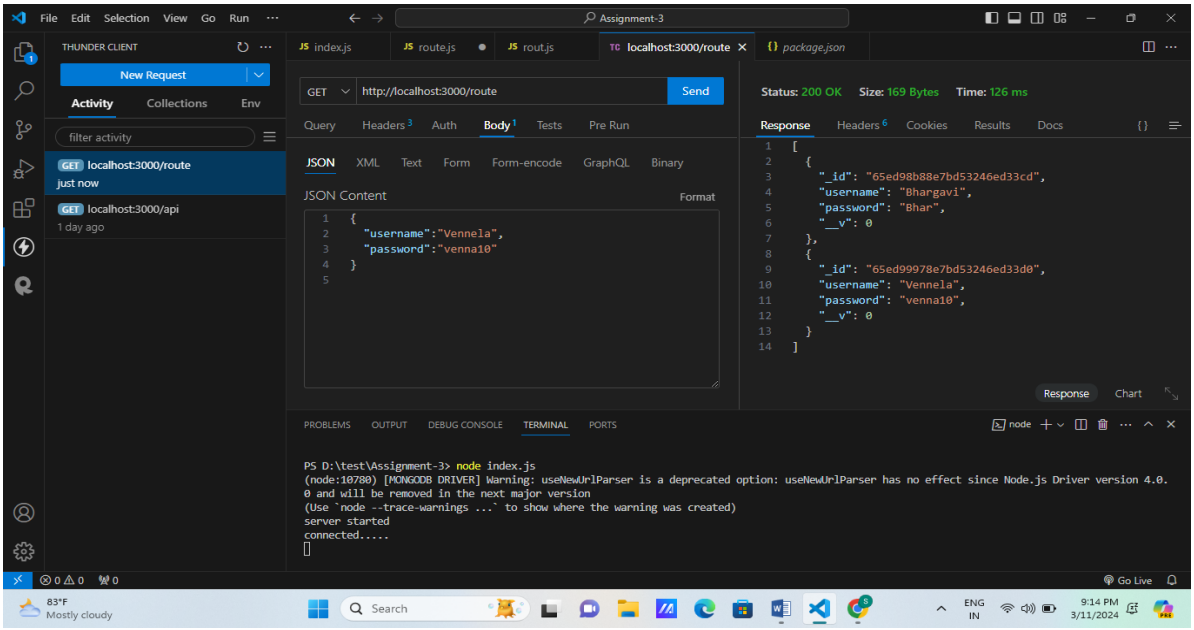
So, here rout.js is a file in the models folder in which the schema is defined.

```
const mongoose=require('mongoose')
const schema= new mongoose.Schema({
    username:{
        type:String,
        required:true
    },
    password:{
        type:String,
        required:true
    }
})
 module.exports=mongoose.model('rout',schema)
```

Here we can verify how the CRUD operations work in server side by using the Thunder Client extension in Vs code:

1) GET Operation:

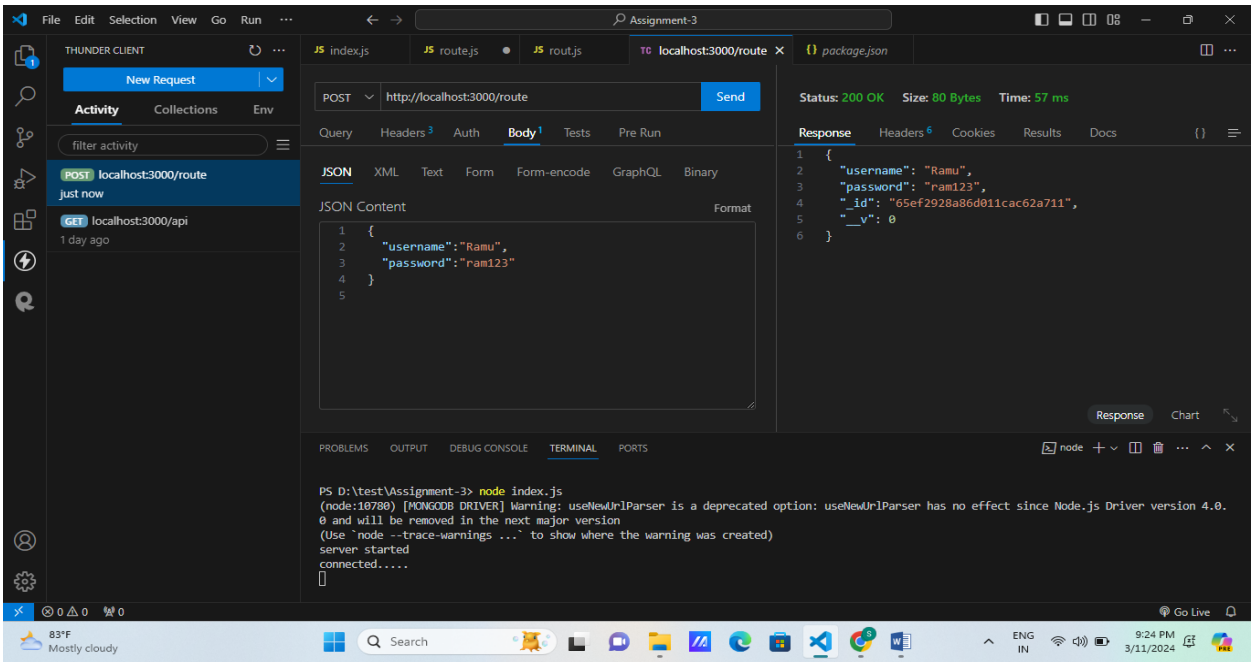
The get operation is used to get the data from the server.



In the above figure, the response section showing the data present in the server. We can see the two json responses.

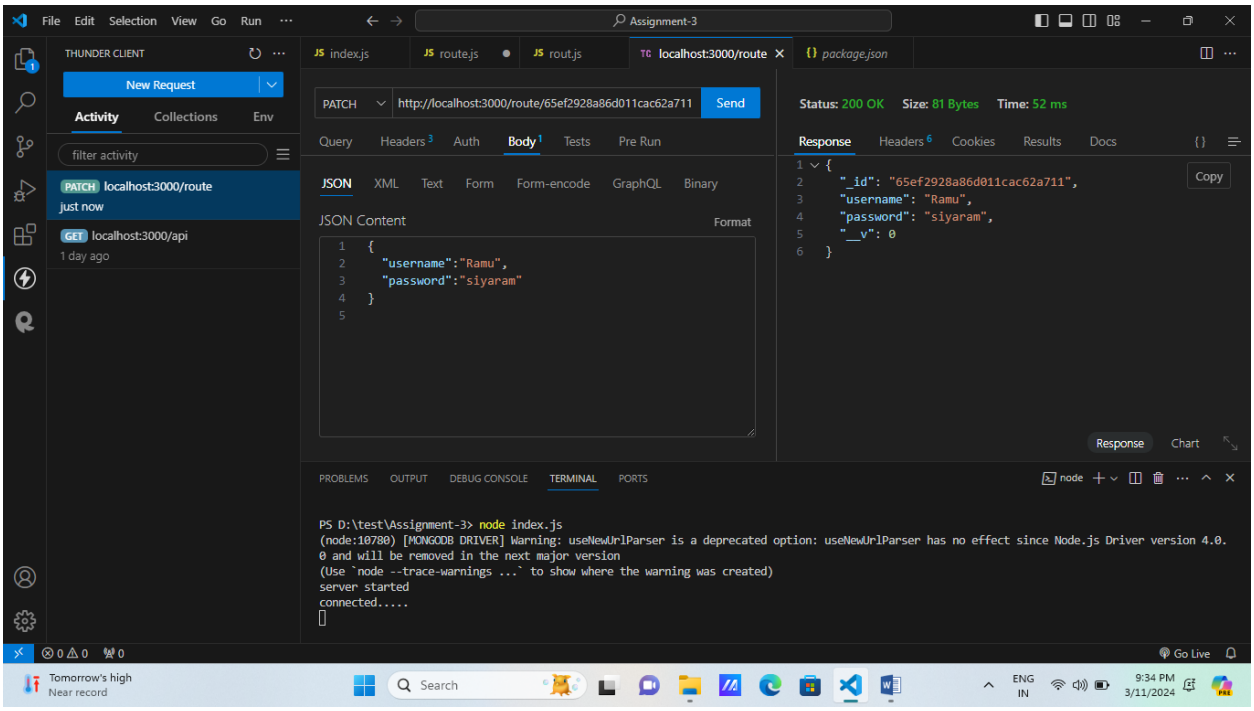
2)POST Operation:

In the post operation we can see in the body section using json format we can add/send the data to the server.



3)PATCH Operation:

In the patch operation we can modify the content of data.

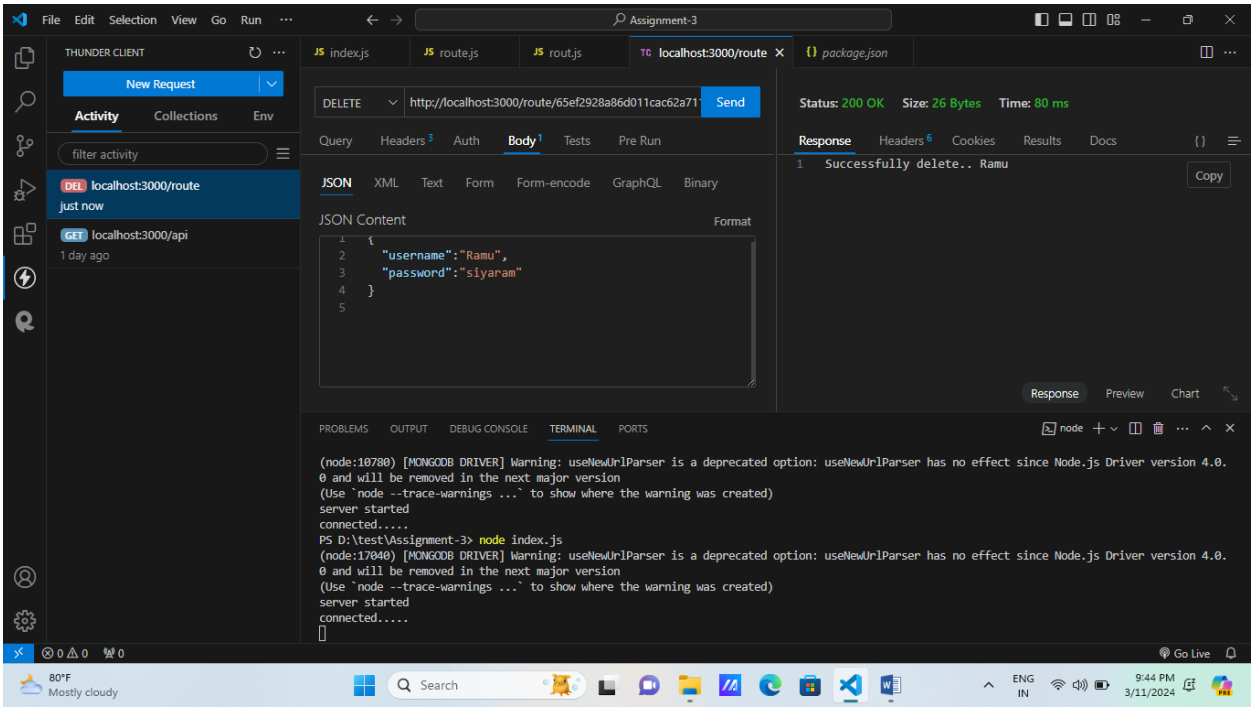


Here in the above code the password of a user is changed. We can change the data by passing the id of the user.

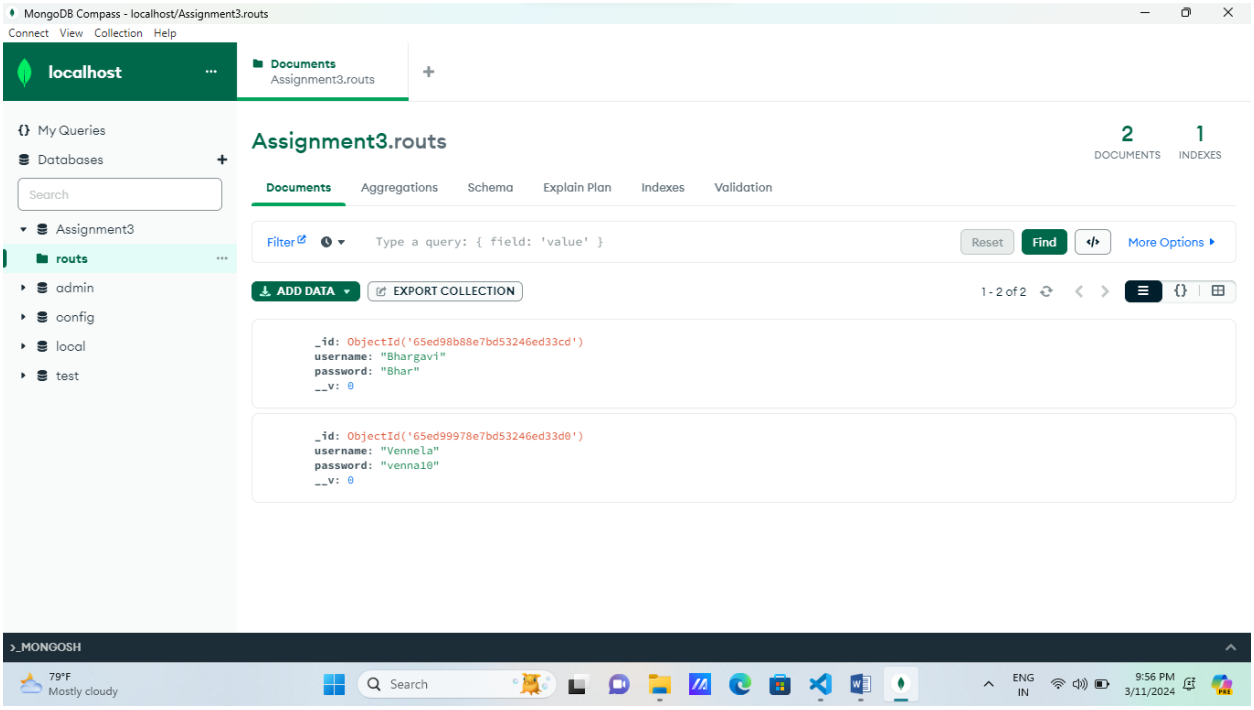
4) DELETE Operation:

We can delete one of the user by using id of the each data.

After deleting the user data we can check by using get operation, if the data is present or not.



We connect it to MongoDb By using the url to connect.



We can see the data collections in the Assignment3 database.