Ex. No: 8	Building a REST API with Express, Node, and MongoDB
05.10.2023	

Aim:

To create a REST API with express node and mongoDB.

Algorithm:

1. Ensure Node.js, npm, and MongoDB are installed on your system.
2. Create a project directory and set up its structure.
3. Use npm to install necessary packages, including Express and a MongoDB driver like Mongoose.
4. Create API routes and handlers for various HTTP methods to manage different data operations.
5. Establish a connection to your MongoDB database using the installed MongoDB driver.
6. Define data models and schemas to structure the data you'll work with in the MongoDB database.
7. Implement Create, Read, Update, and Delete (CRUD) operations in your API routes for database interaction.
8. Test API endpoints using tools like Postman. Debug, refine, and handle errors as needed.

Program:

1) Index.js (server):

• Connecting to mongo dB, mongoose and express

```
const express=require("express");
const mongoose=require("mongoose");
const url='mongodb://127.0.0.1:27017/studentDB';
const app=express();
mongoose.connect(url,{
useNewUrlParser:true
})
const con =mongoose.connection
con.on('open',function(){
console.log("connected to mongodb database")
})
app.use(express.json())
const studentRouter=require('./routes/students')
app.use('/students',studentRouter)
app.listen(3000,function(){
console.log("Server started")
})
```

2) students.js

• Creating routes (GET, PATCH, GET single object by ID, POST)

```
const express=require("express");
const router=express.Router()
const Student=require('../models/student')
router.get('/',async(req,res)=>{
try{
const stud=await Student.find()
res.json(stud)
}catch(err){
res.send("Error")
}
res.send("Get request made")
})
router.get('/:id',async(req,res)=>{
try{
const stud1=await Student.findById(req.params.id)
res.json(stud1)
}catch(err){
res.send("Error")
}
})
router.patch('/:id', async (req, res) => {
try {
const studPatch = await Student.findById(req.params.id);
studPatch.name = req.body.name;
const s = await studPatch.save();
res.json(studPatch);
} catch (err) {
res.status(500).send("Error"); // Sending an error response
}
});
router.post('/',async(req,res)=>{
const student=new Student({
name: req.body.name,
course: req.body.course
})
try{
const s=await student.save()
res.json(s)
}catch(err){
res.send("Error")
}
})
module.exports=router
```

3) student.js

• Creating mongoose schema for a single object (student here)

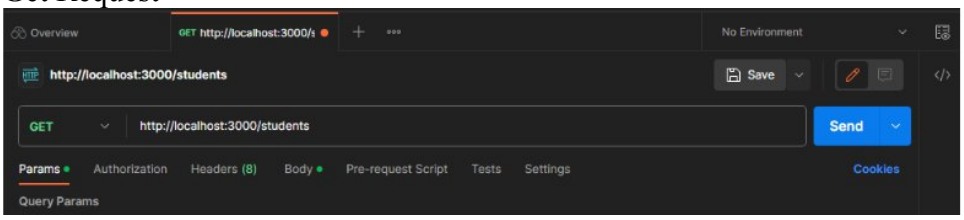
```
const mongoose=require("mongoose")
const studSchema=new mongoose.Schema({
name:{
type: String,
required:true
},
course:{
type: String,
required:true
}
})
module.exports=mongoose.model('Student',studSchema)
```

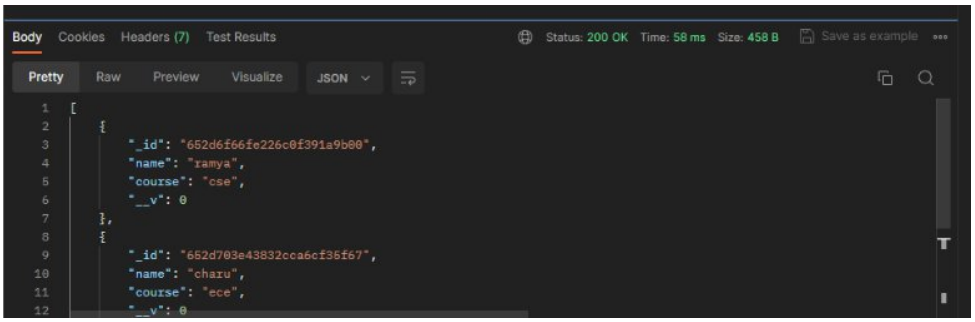
Output:

Github Link:

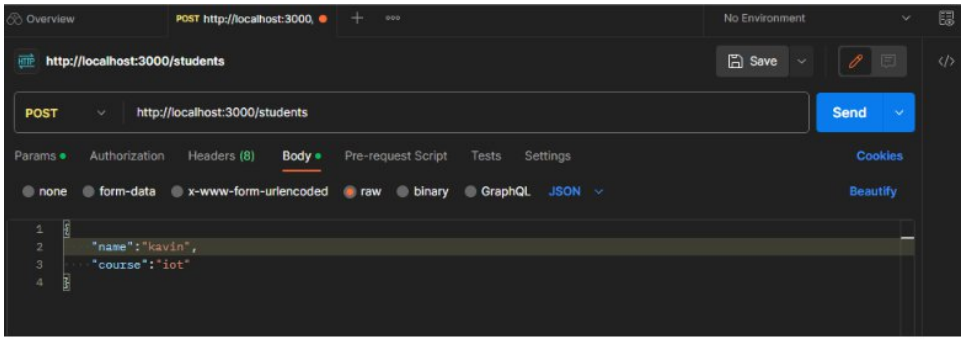
Request made by the postman API:

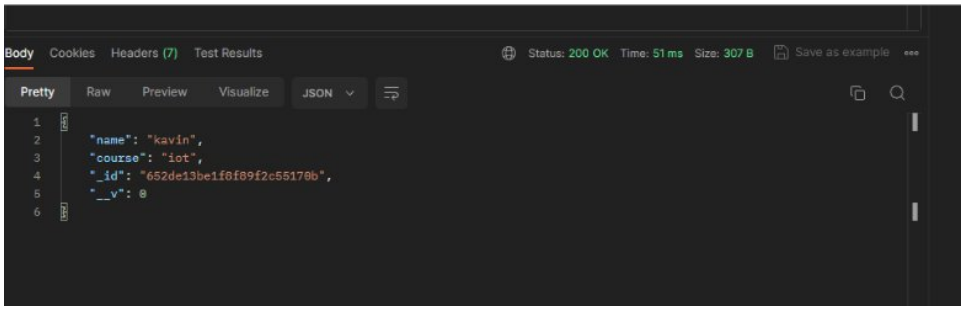
1. Get Request



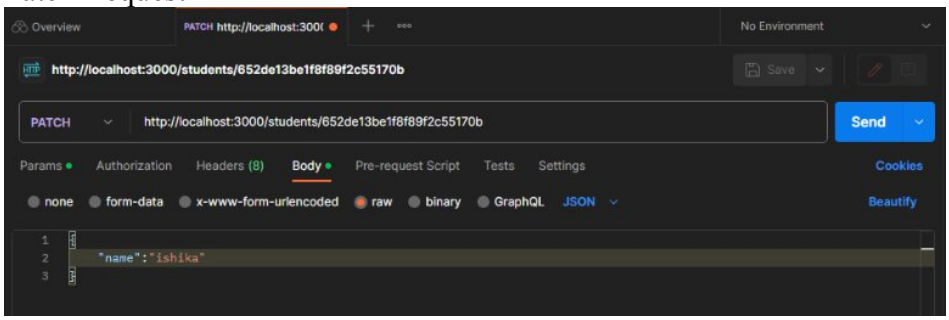


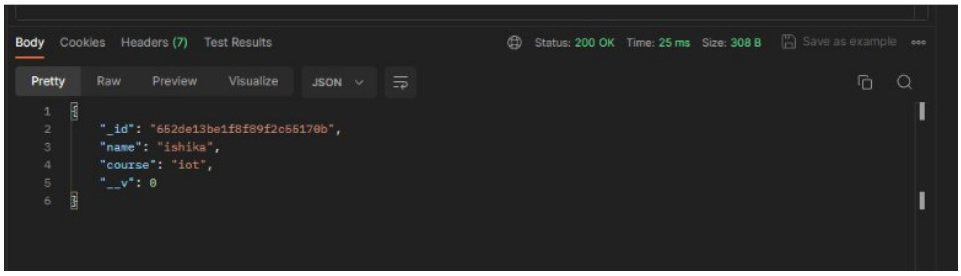
2. Post Request



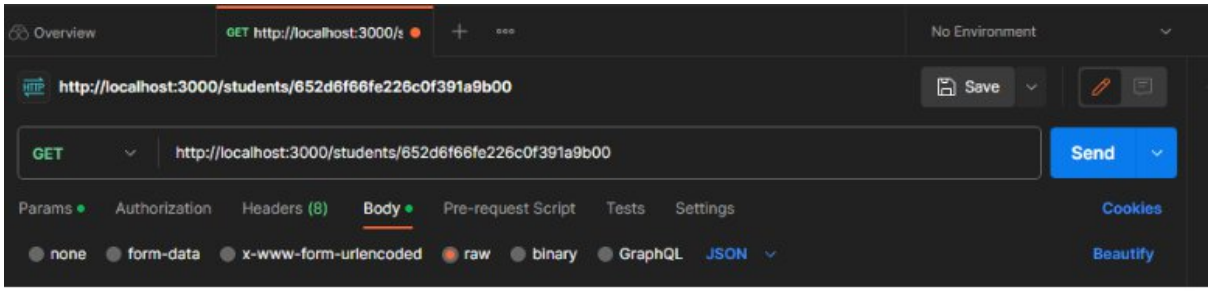


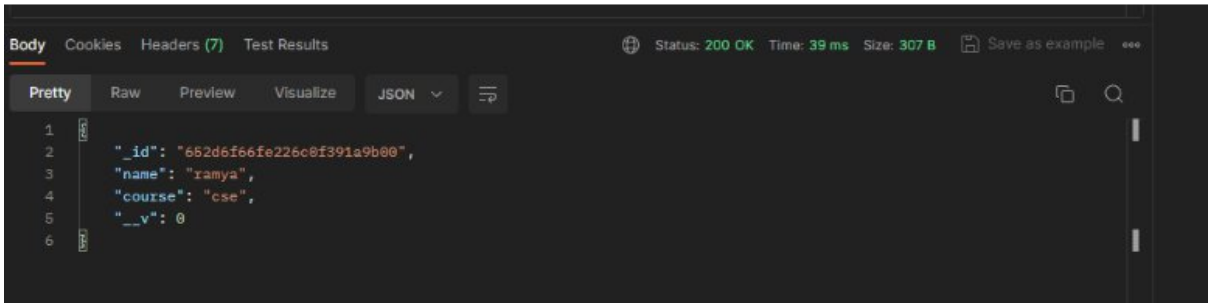
3. Patch Request





4. Get by ID





Result:
Therefore, we've successfully implemented the creation of a REST API with express node and mongoDB.