**ASSIGNMENT – 7**

REST API WITH NODEJS, EXPRESSJS, MONGODB

Q. Create a NodeJS application to provide endpoints for the collections **studentInfo** and **studentAcademicInfo** with a similar structure of objects created in Assignment 6. Install required packages like – express, mongoose, nodemon and perform the following operations. Handle asynchronous behaviour and exceptions properly.

1. **Define schema and model for studentInfo where every field must be required and rollno, mobile, and email must be unique.**

**//model>studentInfo.model.js**

//Swati Nath || 21BCSD21

const mongoose = require('mongoose')

const studentSchema = new mongoose.Schema({

name: {

type: String,

required: true

},

rollno: {

type: Number,

required: true,

unique: true

},

mobile: {

type: String,

required: true,

unique: true

},

email: {

type: String,

required: true,

unique: true

},

address: {

city: {

type: String,

required: true

},

state: {

type: String,

required: true

},

pin: {

type: String,

required: true

}

}

})

const studentInfo = mongoose.model("studentInfo", studentSchema)

module.exports = studentInfo

1. **Define schema and model for studentAcademicInfo where every field is required and rollno is unique again.**

**//model>studentAcademicInfo.model.js**

//Swati Nath || 21BCSD21

const mongoose = require('mongoose')

const studentAcademicSchema = new mongoose.Schema({

rollno: {

type: Number,

required: true,

unique: true

},

program: {

type: String,

required: true

},

branch: {

type: String,

required: true

},

cgpa: {

type: String,

required: true

}

})

const studentAcademic = mongoose.model("studentAcademic", studentAcademicSchema)

module.exports = studentAcademic

**3. Create a file to establish the connection with MongoDB and select the database.**

**//db>db.js**

// Swati Nath || 21BCSD21

const mongoose = require('mongoose')

async function dbConnect(){

DBURL="mongodb+srv://21bcsd21:21bcsd21@cluster0.vnejuvk.mongodb.net"

DBNAME="student"

try {

await mongoose.connect(DBURL+"/"+DBNAME)

console.log("Database Connected");

} catch (error) {

console.log("Connection Error "+error );

}

}

module.exports = dbConnect

**4. Create a controller file for studentInfo and add functionalities for –**

**a. Add a student’s info**

**b. Retrieve all students' records**

**c. Retrieve a single student based on roll number**

**d. Update a student based on roll number**

**e. Delete a student based on roll number**

**//controller>studentInfo.controller.js**

// Swati Nath || 21BCSD21

const mongoose = require('mongoose');

const studentInfo = require('../model/studentInfo.model');

async function addStudent(req,res){

try {

let newStudent = await studentInfo.create(req.body)

res.status(201).json(newStudent)

} catch (error) {

console.log(error);

res.status(400).json({"message": error.message})

}

}

async function allStudentsInfo(req, res){

try {

let students = await studentInfo.find()

res.status(200).json(students)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function getStudentById(req, res){

try {

let { roll } = req.params

console.log(req.params);

let student = await studentInfo.find({rollno: roll})

if(student.length >0){

res.status(200).json(student)

} else {

res.status(404).json({"message": "Data not found"})

}

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function updateStudent(req, res){

try {

let { roll } = req.params

let data = req.body

let student = await studentInfo.findOneAndUpdate({rollno: roll}, data, {new: true})

res.status(200).json(student)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function deleteStudent(req, res){

try {

let { roll } = req.params

let student = await studentInfo.findOneAndDelete({rollno: roll})

res.status(200).json(student)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

module.exports = {

addStudent,

allStudentsInfo,

getStudentById,

updateStudent,

deleteStudent,

}

**5. Create a controller file for studentAcademicInfo and add functionalities for –**

**a. Add a student’s academic record**

**b. Retrieve all students’ academic records**

**c. Retrieve a single student’s academic record based on roll number**

**d. Update a student’s academic record based on roll number**

**e. Delete a student’s academic record based on roll number**

**//controller>studentAcademicInfo.controller.js**

// Swati Nath || 21BCSD21

const mongoose = require('mongoose');

const studentAcademicInfo = require('../model/studentAcademic.model')

async function addStudent(req,res){

try {

let newStudent = await studentAcademicInfo.create(req.body)

res.status(201).json(newStudent)

} catch (error) {

console.log(error);

res.status(400).json({"message": error.message})

}

}

async function allStudentsInfo(req, res){

try {

let students = await studentAcademicInfo.find()

res.status(200).json(students)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function getStudentById(req, res){

try {

let { roll } = req.params

console.log(req.params);

let student = await studentAcademicInfo.find({rollno: roll})

if(student.length >0){

res.status(200).json(student)

} else {

res.status(404).json({"message": "Data not found"})

}

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function updateStudent(req, res){

try {

let { roll } = req.params

let data = req.body

let student = await studentAcademicInfo.findOneAndUpdate({rollno: roll}, data, {new: true})

res.status(200).json(student)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function deleteStudent(req, res){

try {

let { roll } = req.params

let student = await studentAcademicInfo.findOneAndDelete({rollno: roll})

res.status(200).json(student)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

module.exports = {

addStudent,

allStudentsInfo,

getStudentById,

updateStudent,

deleteStudent

}

**6. Now use the controllers and provide end-points for all the functionalities and test the endpoints with Postman**

**//route>studentInfo.route.js**

Swati Nath || 21BCSD21

const express = require('express')

const{

addStudent,

allStudentsInfo,

getStudentById,

updateStudent,

deleteStudent,

allDetails,

insertBoth

} = require('../controller/studentInfo.controller')

const studentInfoRouter = express.Router()

studentInfoRouter.get("/details", allDetails)

studentInfoRouter.get("/", allStudentsInfo)

studentInfoRouter.post("/", addStudent)

studentInfoRouter.get("/:roll", getStudentById)

studentInfoRouter.put("/:roll", updateStudent)

studentInfoRouter.delete("/:roll", deleteStudent)

studentInfoRouter.post("/insertBoth", insertBoth)

module.exports = studentInfoRouter

**//route.studentAcademic.router.js**

const express = require('express')

const {

addStudent,

allStudentsInfo,

getStudentById,

updateStudent,

deleteStudent

} = require('../controller/studentAcademic.controller')

const studentAcademicRouter = express.Router()

studentAcademicRouter.get("/", allStudentsInfo)

studentAcademicRouter.get("/:roll", getStudentById)

studentAcademicRouter.post("/", addStudent)

studentAcademicRouter.put("/:roll", updateStudent)

studentAcademicRouter.delete("/:roll", deleteStudent)

module.exports = studentAcademicRouter

**7. Add two more routes/end-points one of which will be capable of adding records to both the collections and another one will fetch the records from both the collections.**

async function allDetails(req, res){

try {

let data = await studentInfo.aggregate([

{

$lookup: {

from: "studentacademics",

localField: "rollno",

foreignField: "rollno",

as: "studentacademics"

}

}

])

res.status(200).json(data)

} catch (error) {

console.log(error);

res.status(500).json({"message": error.message})

}

}

async function insertBoth(req,res){

try {

const {

name,rollno,mobile,email,address,program,branch,cgpa

} = req.body

let newStudent = await studentInfo.create({

name: name,

rollno: rollno,

mobile: mobile,

email: email,

address: address

})

let newStudentAcademic = await studentAcademic.create({

rollno: rollno,

program: program,

branch: branch,

cgpa: cgpa

})

res.status(201).json(newStudent)

res.status(201).json(newStudentAcademic)

} catch (error) {

console.log(error);

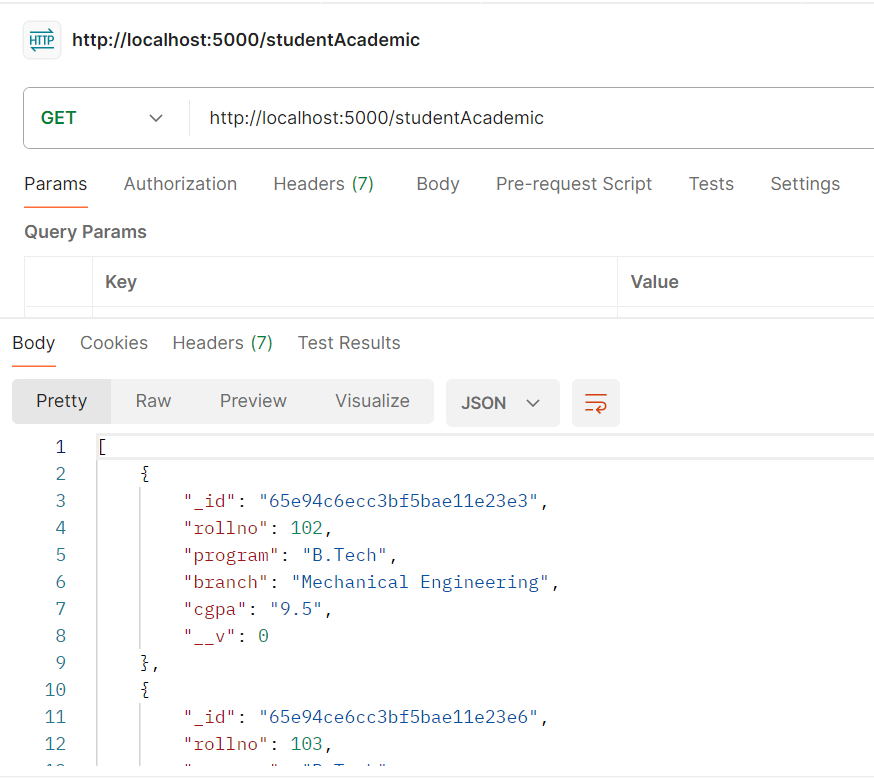
res.status(400).json({"message": error.message})

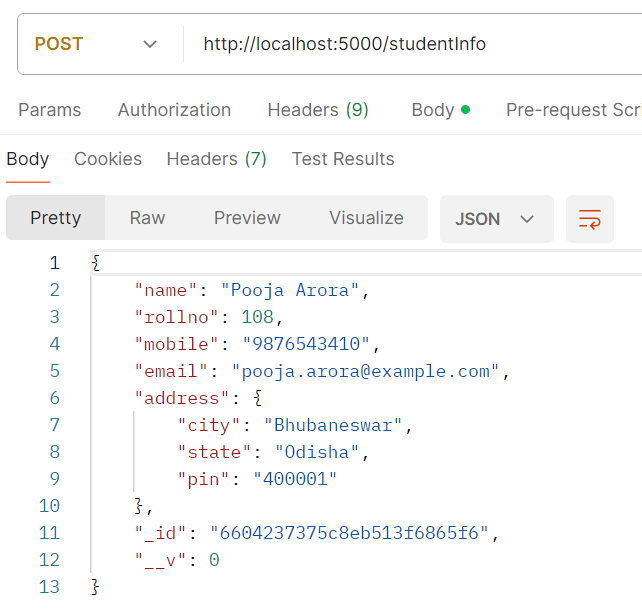
}

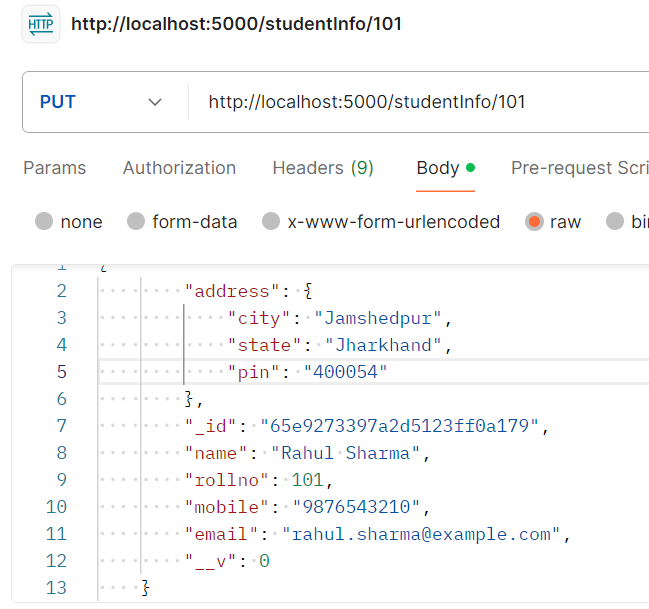
}

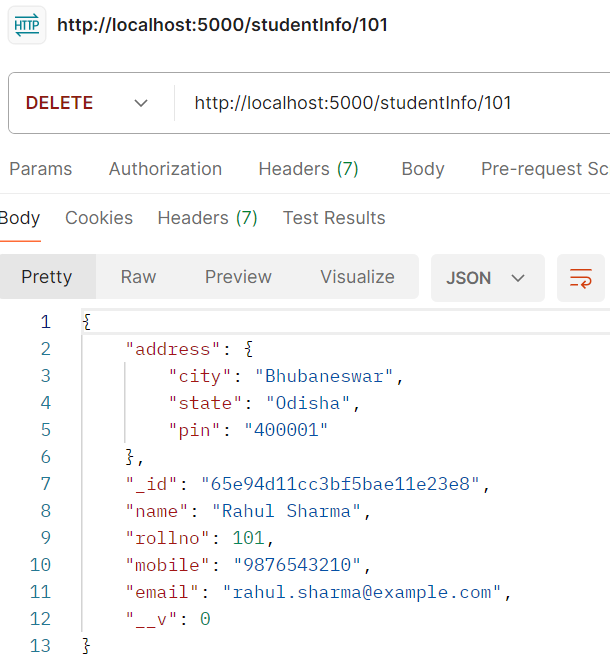
**OUTPUT**

**GET REQUEST:**



**POST** **REQUEST**:

**PUT** **REQUEST**:

**DELETE** **REQUEST**: