**EX NO: 10**

**DATE:**

**IMPLEMENT THE MODIFIED SYSTEM AND TEST IT FOR VARIOUS SCENARIOS.**

**AIM:**

To implement the modified system and test it for various scenarios.

**Modified Program:**

**//server.js**

const express = require('express');

const cors = require('cors');

const { MongoClient, ObjectId } = require('mongodb');

require('dotenv').config();

const app = express();

const PORT = process.env.PORT || 5000;

const uri = process.env.MONGODB\_URI;

let client; let db;

async function connectToDatabase() {

if (db) return db;

client = new MongoClient(uri);

await client.connect();

db = client.db("sis");

console.log("Connected to MongoDB");

return db;}

app.use(cors());

app.use(express.json());

connectToDatabase().catch(err => {

console.error("Database connection failed", err);

process.exit(1);});

app.post('/register', async (req, res) => {

try {

const db = await connectToDatabase();

const newUser = req.body;

const result = await db.collection("users").insertOne(newUser);

res.status(201).json({ message: 'User Registered Successfully', \_id: result.insertedId});

} catch (error) {

res.status(500).json({ message: 'Error registering user' }); }});

app.post('/students', async (req, res) => {

try {

const db = await connectToDatabase();

const result = await db.collection("students").insertOne(req.body);

res.status(201).json({ message: 'Student added successfully', \_id: result.insertedId});

} catch (error) {

res.status(500).json({ message: 'Error adding student' }); }});

app.delete('/students/:id', async (req, res) => {

try {

const db = await connectToDatabase();

const result = await db.collection("students").deleteOne({ \_id: new ObjectId(req.params.id) });

res.json({ message: 'Student removed successfully' });

} catch (error) {

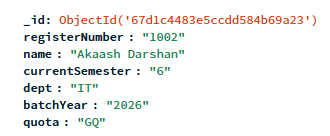
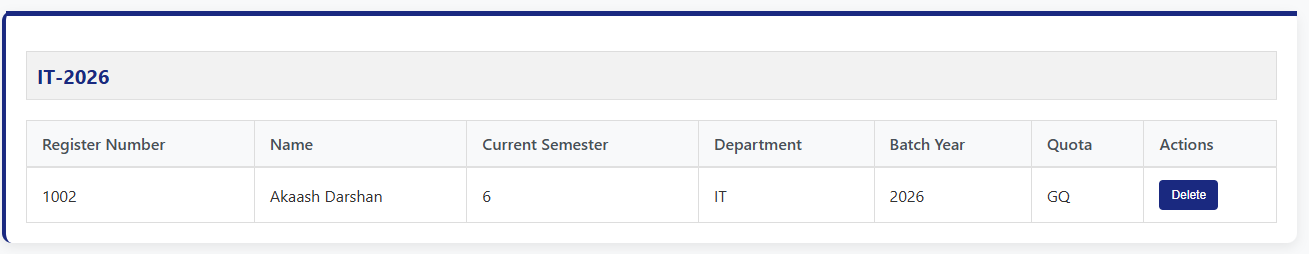
res.status(500).json({ message: 'Error deleting student' }); }});

app.listen(PORT, () => console.log(`Server running on port ${PORT}`));

**Test Cases:**

#### **Test Case 1: Admin Dashboard - Student Management**

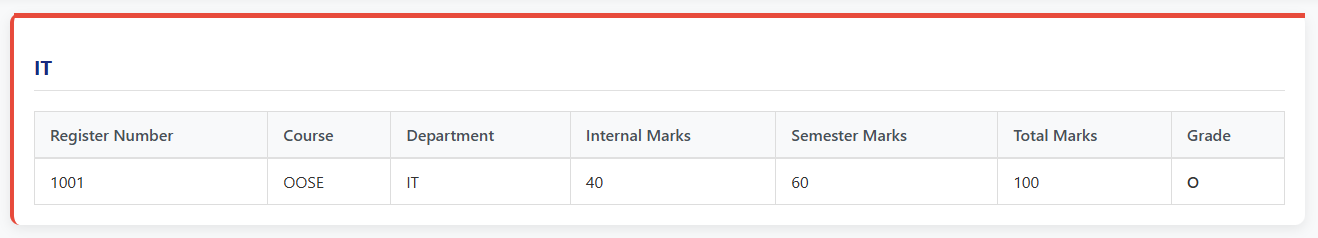
* **Test Case ID:** TC\_SYS\_U2\_001
* **Unit to be Tested:** Admin Dashboard - Student Management
* **Assumptions:** The Admin has valid login credentials
* **Test Data:** Student Name, ID, Course Details
* **Steps to be Executed:**
  1. Login as Admin
  2. Navigate to "Add Student" section
  3. Enter student details and submit
  4. Navigate to "Delete Student" section
  5. Select a student and delete
* **Expected Result:** Student added successfully, Student removed successfully
* **Actual Result:**

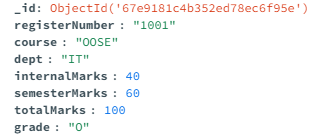


* **Pass / Fail:** Pass

#### **Test Case 2: Faculty Dashboard - Report Management**

* **Test Case ID:** TC\_SYS\_U3\_001
* **Unit to be Tested:** Faculty Dashboard - Report Management
* **Assumptions:** The Faculty has valid login credentials
* **Test Data:** Report Title, Report Content
* **Steps to be Executed:**
  1. Login as Faculty
  2. Navigate to "Add Report"
  3. Enter report details and submit
  4. Navigate to "View Reports"
  5. Verify if the added report is displayed
* **Expected Result:** Report added successfully, Reports displayed correctly
* **Actual Result:**

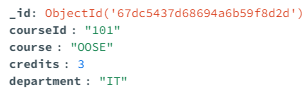
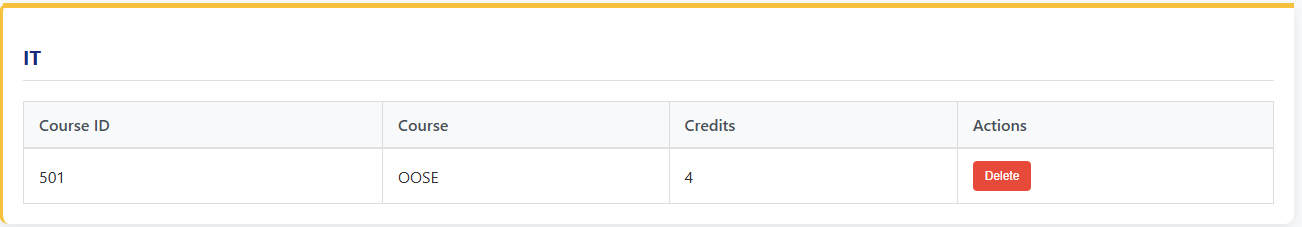




* **Pass / Fail:** Pass

#### **Test Case 3: Student Dashboard - Course Registration**

* **Test Case ID:** TC\_SYS\_U4\_001
* **Unit to be Tested:** Student Dashboard - Course Registration
* **Assumptions:** The student has valid login credentials
* **Test Data:** Available Course List
* **Steps to be Executed:**
  1. Login as Student
  2. Navigate to "Course Registration"
  3. Select a course and submit
  4. Verify that the course is added to the student's profile
* **Expected Result:** Course successfully registered for the student
* **Actual Result:**



* **Pass / Fail:** Pass

**RESULT:**