



**Ganpat  
University**

॥ विद्यया समाजोत्कर्षः ॥

**Institute of  
Computer  
Technology**

**Name: Tushar Panchal**

**En.No: 21162101014**

**Sub: EADC ( Enterprise Application Development for Cloud)**

**Branch: CBA**

**Batch:61**

## **PRACTICAL 04**

### **❖ Question :**

Create REST API using NodeJS and apply REST method to perform CRUD operations on resources at server regarding universities or Industries scenario and deploy it over AWS Cloud

1. Create a Rest API for universities or industries using NodeJS on the cloud platform
2. Perform CRUD operation on the server using Postman.
3. Integrate it with HTML form, where it provides the option to POST the data of new employees, Get information of any employee based on ID, Get information of all employees, Update information of any employee based on their id, and delete employee records based on their id

➤ **Below I provided step by step solution for above practical .**

## ✓ Source Code(server.js) :

```
const express = require('express');
const bodyParser = require('body-parser');
const path = require('path'); // Import the path module to work with
file paths

const app = express();
const PORT = process.env.PORT || 8080;

app.use(bodyParser.json());

// Sample data for universities or industries
let employees = [
  { id: 1, name: 'John Doe', role: 'Professor', department: 'Computer
Science' },
  { id: 2, name: 'Jane Smith', role: 'Engineer', department:
'Mechanical Engineering' },
];

// Serve static files from the "public" directory
app.use(express.static(path.join(__dirname, 'public')));

// Route to get all employees
app.get('/employees', (req, res) => {
  res.json(employees);
});

// Route to get an employee by ID
app.get('/employees/:id', (req, res) => {
  const id = parseInt(req.params.id);
  const employee = employees.find((emp) => emp.id === id);
  if (employee) {
    res.json(employee);
  } else {
    res.status(404).json({ error: 'Employee not found' });
  }
});

// Route to add a new employee
app.post('/employees', (req, res) => {
  const { id, name, role, department } = req.body;
  if (!id || !name || !role || !department) {
    return res.status(400).json({ error: 'Please provide all required
fields' });
  }

  const newEmployee = { id, name, role, department };
  employees.push(newEmployee);
});
```

```

    console.log("New employee added:", newEmployee);
    res.status(201).json(newEmployee);
  });

  // Route to update an existing employee by ID
  // Route to update an existing employee by ID
  app.put('/employees/:id', (req, res) => {
    const id = parseInt(req.params.id);
    const { name, role, department } = req.body;
    const employeeIndex = employees.findIndex((emp) => emp.id === id);

    if (employeeIndex !== -1) {
      // Employee with the provided ID exists, update the data
      employees[employeeIndex].name = name ||
employees[employeeIndex].name;
      employees[employeeIndex].role = role ||
employees[employeeIndex].role;
      employees[employeeIndex].department = department ||
employees[employeeIndex].department;
      console.log("Employee updated:", employees[employeeIndex]);
      res.json(employees[employeeIndex]);
    } else {
      // Employee with the provided ID doesn't exist, create a new
employee
      const newEmployee = { id, name, role, department };
      employees.push(newEmployee);
      console.log("New employee added:", newEmployee);
      res.status(201).json(newEmployee);
    }
  });

  // Route to delete an employee by NAME
  app.delete('/employees/:name', (req, res) => {
    const name = req.params.name;
    employees = employees.filter((emp) => emp.name !== name);
    res.json({ message: 'Employee deleted successfully' });
  });

  // Serve the index.html file for the root path
  app.get('/', (req, res) => {
    res.sendFile(path.join(__dirname, 'index.html'));
  });

  // Start the server
  app.listen(PORT, () => {
    console.log(`Server is running on http://localhost:${PORT}`);
  });

```

## » Initialize EB in our project folder:

Command: **eb init**

```

PowerShell
> eb init

Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : EU (Ireland)
5) eu-central-1 : EU (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
8) ap-southeast-2 : Asia Pacific (Sydney)
9) ap-northeast-1 : Asia Pacific (Tokyo)
10) ap-northeast-2 : Asia Pacific (Seoul)
11) sa-east-1 : South America (Sao Paulo)
12) cn-north-1 : China (Beijing)
13) cn-northwest-1 : China (Ningxia)
14) us-east-2 : US East (Ohio)
15) ca-central-1 : Canada (Central)
16) eu-west-2 : EU (London)
17) eu-west-3 : EU (Paris)
18) eu-north-1 : EU (Stockholm)
19) eu-south-1 : EU (Milano)
20) ap-east-1 : Asia Pacific (Hong Kong)
21) me-south-1 : Middle East (Bahrain)
22) il-central-1 : Middle East (Israel)
23) af-south-1 : Africa (Cape Town)
24) ap-southeast-3 : Asia Pacific (Jakarta)
25) ap-northeast-3 : Asia Pacific (Osaka)
(default is 3): 6

Enter Application Name
(default is "Practical-4"):
Application Practical-4 has been created.

It appears you are using Node.js. Is this correct?
(Y/n):
Select a platform branch.
1) Node.js 20 running on 64bit Amazon Linux 2023
2) Node.js 18 running on 64bit Amazon Linux 2023
3) Node.js 18 running on 64bit Amazon Linux 2

```

## » This shows that our Application is created :

```

6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
8) ap-southeast-2 : Asia Pacific (Sydney)
9) ap-northeast-1 : Asia Pacific (Tokyo)
10) ap-northeast-2 : Asia Pacific (Seoul)
11) sa-east-1 : South America (Sao Paulo)
12) cn-north-1 : China (Beijing)
13) cn-northwest-1 : China (Ningxia)
14) us-east-2 : US East (Ohio)
15) ca-central-1 : Canada (Central)
16) eu-west-2 : EU (London)
17) eu-west-3 : EU (Paris)
18) eu-north-1 : EU (Stockholm)
19) eu-south-1 : EU (Milano)
20) ap-east-1 : Asia Pacific (Hong Kong)
21) me-south-1 : Middle East (Bahrain)
22) il-central-1 : Middle East (Israel)
23) af-south-1 : Africa (Cape Town)
24) ap-southeast-3 : Asia Pacific (Jakarta)
25) ap-northeast-3 : Asia Pacific (Osaka)
(default is 3): 6

Enter Application Name
(default is "Practical-4"):
Application Practical-4 has been created.

It appears you are using Node.js. Is this correct?
(Y/n):
Select a platform branch.
1) Node.js 20 running on 64bit Amazon Linux 2023
2) Node.js 18 running on 64bit Amazon Linux 2023
3) Node.js 18 running on 64bit Amazon Linux 2
4) Node.js 16 running on 64bit Amazon Linux 2 (Deprecated)
5) Node.js 14 running on 64bit Amazon Linux 2 (Deprecated)
(default is 1):

Cannot setup CodeCommit because there is no Source Control setup, continuing with initialization
Do you want to set up SSH for your instances?
(Y/n): n

```

Application name	Environments	Date created	Last modified	AR
Practical-4	-	February 13, 2024 20:01:01 (UT...)	February 13, 2024 20:01:01 (UT...)	

## » Now we have to create environment using command **eb create (name):**

**Command:** `eb create Tushar-Practical-4`

```

PowerShell
> eb create Tushar-Practical-4
Creating application version archive "app-240213_200358582061".
Uploading Practical-4/app-240213_200358582061.zip to S3. This may take a while.
Upload Complete.
Environment details for: Tushar-Practical-4
  Application name: Practical-4
  Region: ap-south-1
  Deployed Version: app-240213_200358582061
  Environment ID: e-myjgethphh
  Platform: arn:aws:elasticbeanstalk:ap-south-1::platform/Node.js 20 running on 64bit Amazon Linux 2023/6.1.0
  Tier: WebServer-Standard-1.0
  CNAME: UNKNOWN
  Updated: 2024-02-13 14:34:20.017000+00:00
Printing Status:
2024-02-13 14:34:18 INFO createEnvironment is starting.
2024-02-13 14:34:20 INFO Using elasticbeanstalk-ap-south-1-767398120148 as Amazon S3 storage bucket for environment data.
2024-02-13 14:34:39 INFO Created security group named: sg-029750d38fffaaa3f
2024-02-13 14:34:39 INFO Created security group named: awseb-e-myjgethphh-stack-AWSEBSecurityGroup-L080EOTJW1S6
2024-02-13 14:34:55 INFO Created Auto Scaling launch configuration named: awseb-e-myjgethphh-stack-AWSEBAutoScalingLaunchConfiguration-9AojmBLat1KF
2024-02-13 14:34:55 INFO Created target group named: arn:aws:elasticloadbalancing:ap-south-1:767398120148:targetgroup/awseb-AWSEB-X2Z3PNTASOJR/44a0bdd87cfd05de
2024-02-13 14:35:10 INFO Created Auto Scaling group named: awseb-e-myjgethphh-stack-AWSEBAutoScalingGroup-QPfnvHjIBMx0
2024-02-13 14:35:10 INFO Waiting for EC2 instances to launch. This may take a few minutes.
2024-02-13 14:35:10 INFO Created Auto Scaling group policy named: arn:aws:autoscaling:ap-south-1:767398120148:scalingPolicy:fdda974a-a8d7-48e6-80b0-594de9d197ec:autoScalingGroupName/awseb-e-myjgethphh-stack-AWSEBAutoScalingGroup-QPfnvHjIBMx0:policyName/awseb-e-myjgethphh-stack-AWSEBAutoScalingScaleDownPolicy-VpD1cUJ602z5
2024-02-13 14:35:10 INFO Created Auto Scaling group policy named: arn:aws:autoscaling:ap-south-1:767398120148:scalingPolicy:00223145-e606-47d7-877a-8e9a3926eb94:autoScalingGroupName/awseb-e-myjgethphh-stack-AWSEBAutoScalingGroup-QPfnvHjIBMx0:policyName/awseb-e-myjgethphh-stack-AWSEBAutoScalingScaleUpPolicy-6TuzUFluNIof
2024-02-13 14:35:10 INFO Created CloudWatch alarm named: awseb-e-myjgethphh-stack-AWSEBCloudwatchAlarmLow-TkN9jrD5a33T
2024-02-13 14:35:10 INFO Created CloudWatch alarm named: awseb-e-myjgethphh-stack-AWSEBCloudwatchAlarmHigh-XrLPQZ1wE7IA
2024-02-13 14:37:16 INFO Created load balancer named: arn:aws:elasticloadbalancing:ap-south-1:767398120148:loadbalancer/app/awseb--AWSEB-hFLJnUUXqRfR/5469286771289e7d
2024-02-13 14:37:16 INFO Created Load Balancer listener named: arn:aws:elasticloadbalancing:ap-south-1:767398120148:listener/app/awseb--AWSEB-hFLJnUUXqRfR/5469286771289e7d/54b6da0ab748bd05
2024-02-13 14:37:24 INFO Instance deployment completed successfully.
2024-02-13 14:38:29 INFO Successfully launched environment: Tushar-Practical-4
  
```

The screenshot shows the AWS Elastic Beanstalk console interface. The left sidebar contains navigation links for Applications, Environments, and Change history. The main content area displays the 'Tushar-Practical-4' environment overview. The 'Environment overview' section shows a 'Warning' icon and details for the environment ID 'e-myjgethphh', domain 'Tushar-Practical-4.elasticbeanstalk.com', and application name 'Practical-4'. The 'Platform' section shows the platform as 'Node.js 20 running on 64bit Amazon Linux 2023/6.1.0' and the running version as 'app-240213\_200358582061'. The 'Platform state' is 'Supported'. Below the overview, there are tabs for Events, Health, Logs, Monitoring, Alarms, Managed updates, and Tags. The 'Events' tab is selected, showing a list of events with a search bar and pagination controls.

## » Output of my REST API node-js application (on AWS Cloud) :

# EMPLOYEE MANAGEMENT

### ADD EMPLOYEE

ID:

Name:

Role:

Department:

**ADD EMPLOYEE**

Employee ID:

**GET EMPLOYEE DETAILS**

### UPDATE EMPLOYEE

Employee ID:

New Name:

New Role:

New Department:

**UPDATE EMPLOYEE**

### DELETE EMPLOYEE

Employee Name:

Employee Name:

**DELETE EMPLOYEE**

### All Employees

ID: 1  
Name: John Doe  
Role: Professor  
Department: Computer Science

ID: 2  
Name: Jane Smith  
Role: Engineer  
Department: Mechanical Engineering

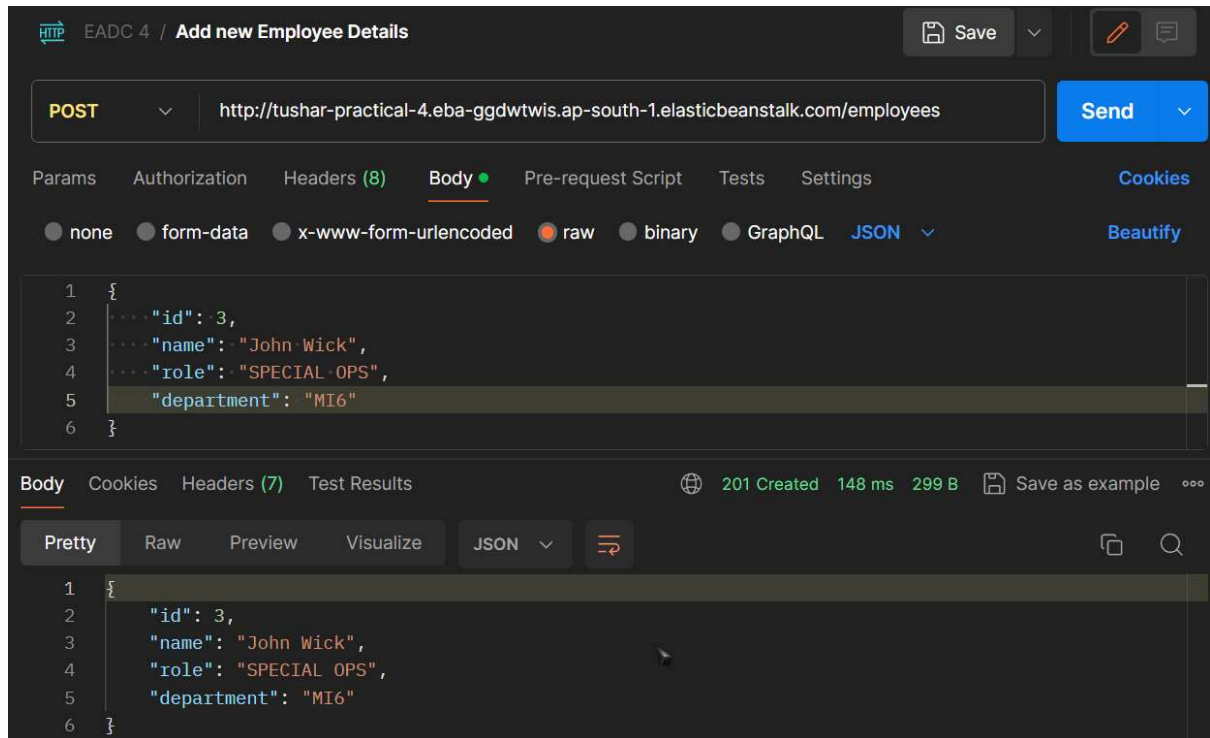
**GET ALL EMPLOYEE DETAILS**



## » Output of performing CRUD operations on the server using postman :

### » Add Employee :

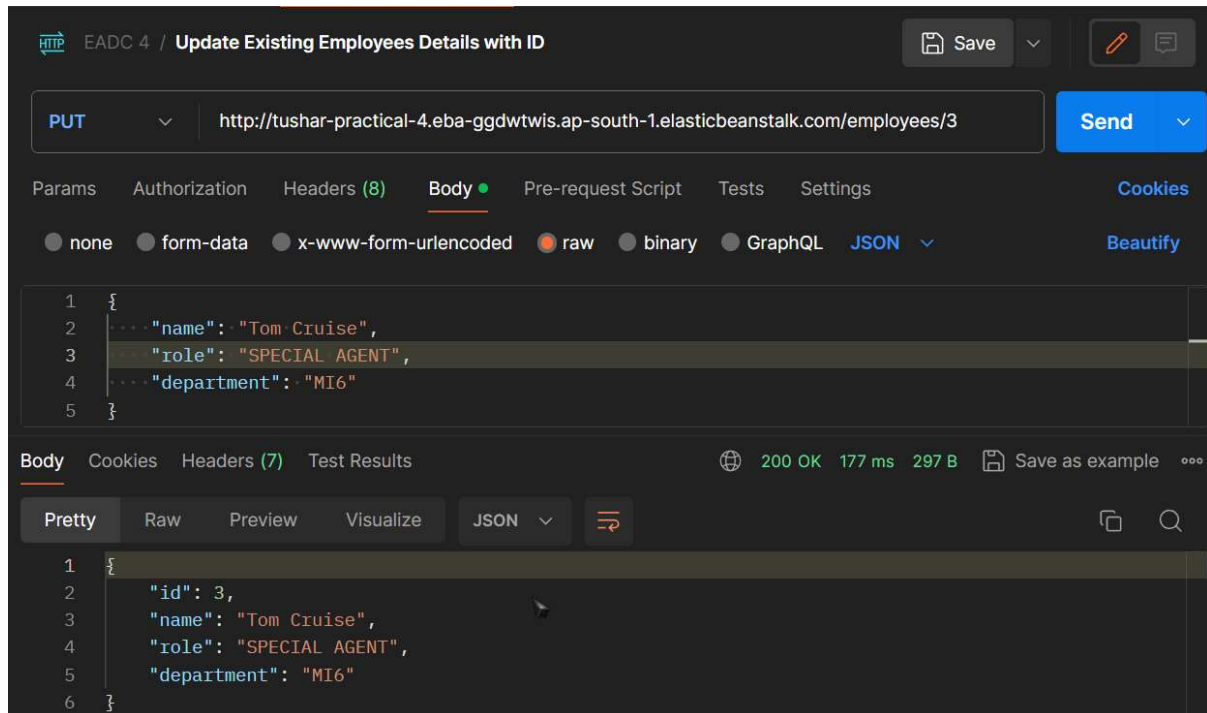
<http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees>



## ⇒ Update Employee :

<http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees/3>

here I updating John wick and her id is 3 :



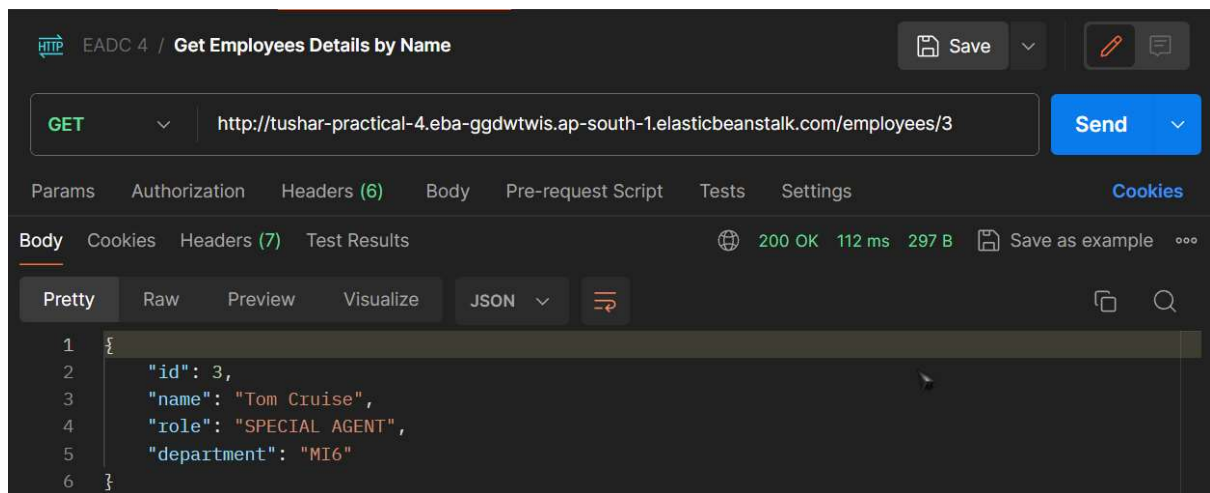
Here you can see John wick is now updated :





## ⇒ Get Employee by id :

<http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees/3>

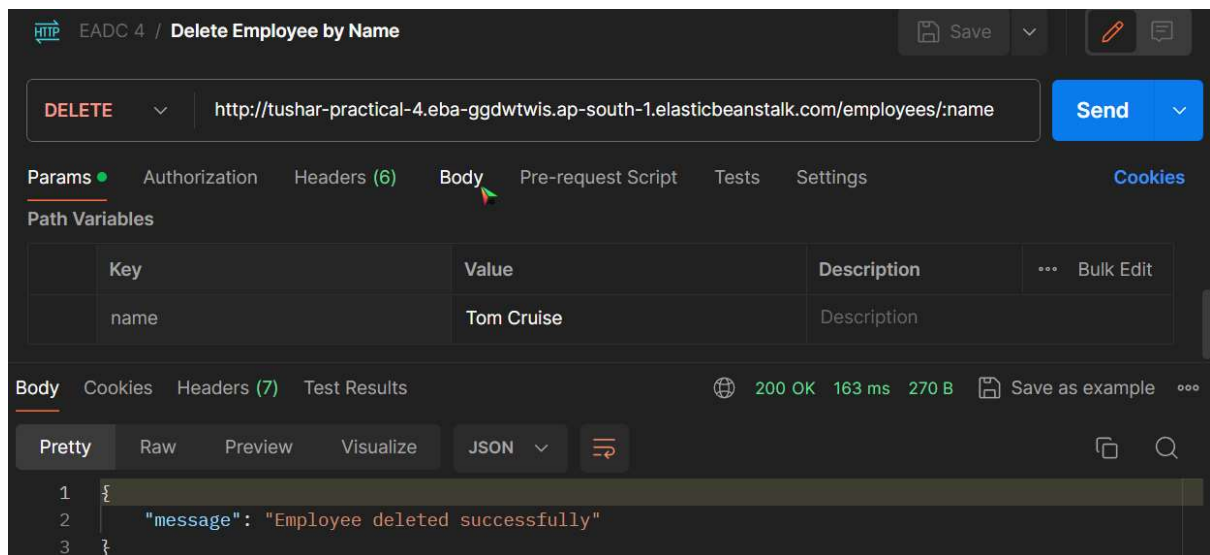


## ⇒ Delete Employee by Name :

<http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees/:name>

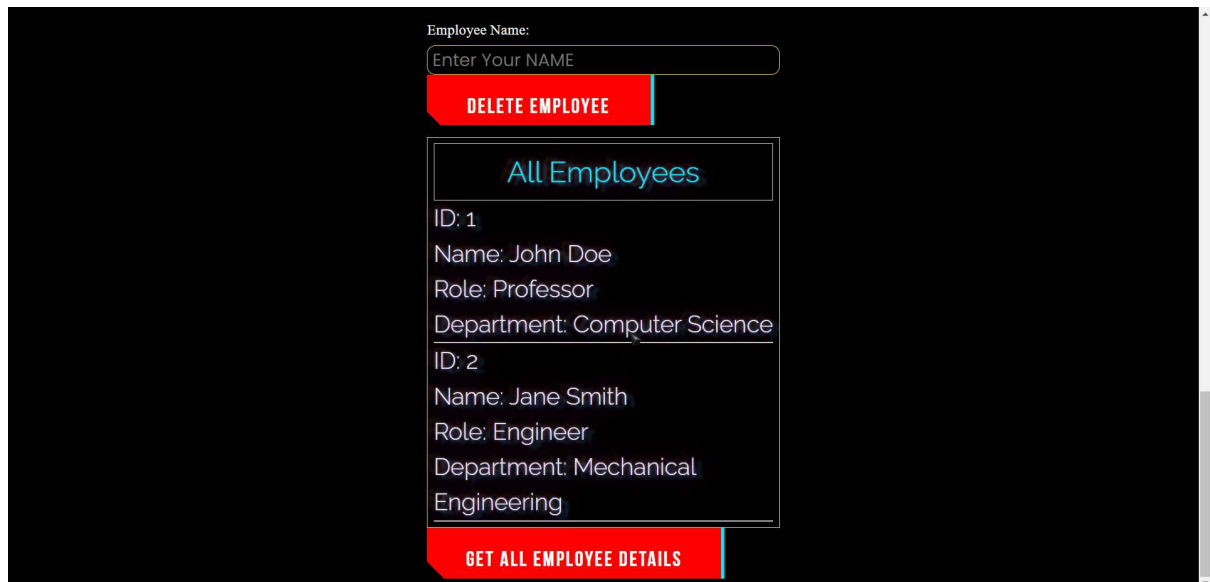
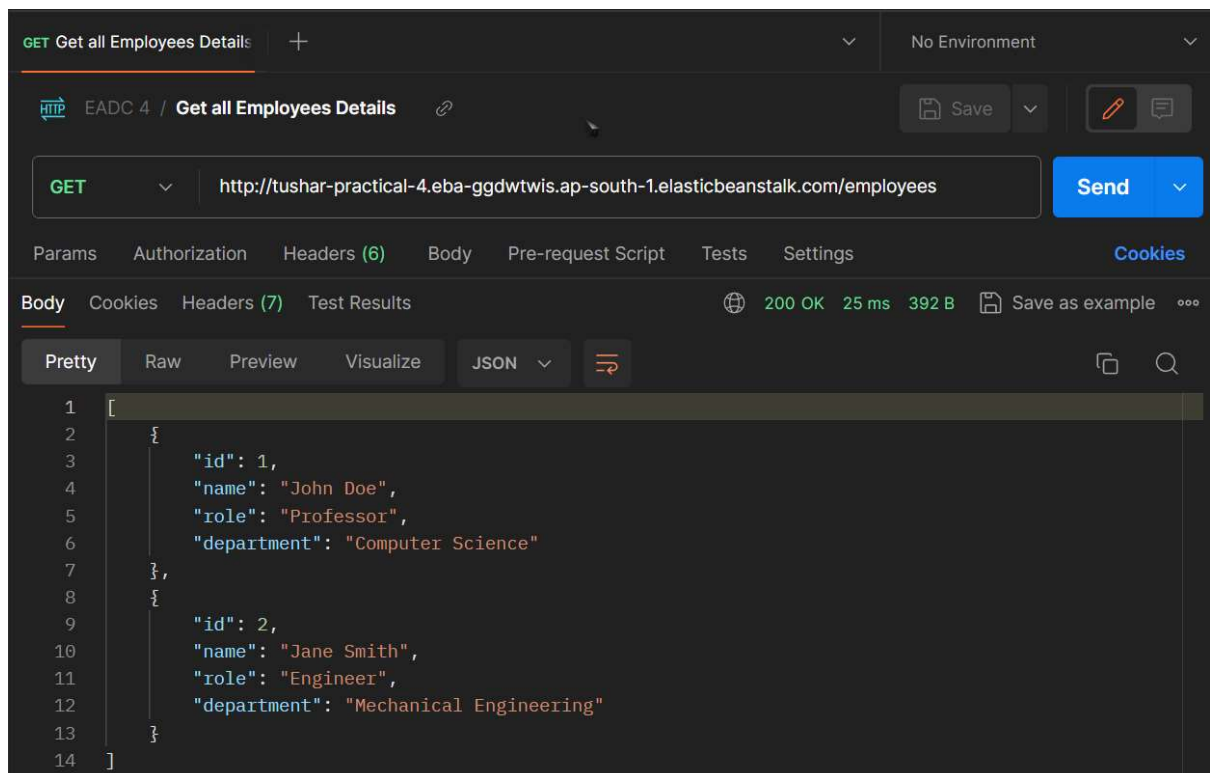
Here I delete an existing employee by Name :

I passed value Tom Cruise in name key to delete that employee



## ➡ Get All Employees Details :

<http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees>



That's it I finally performed all CRUD operations on the AWS server using postman .