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 .**

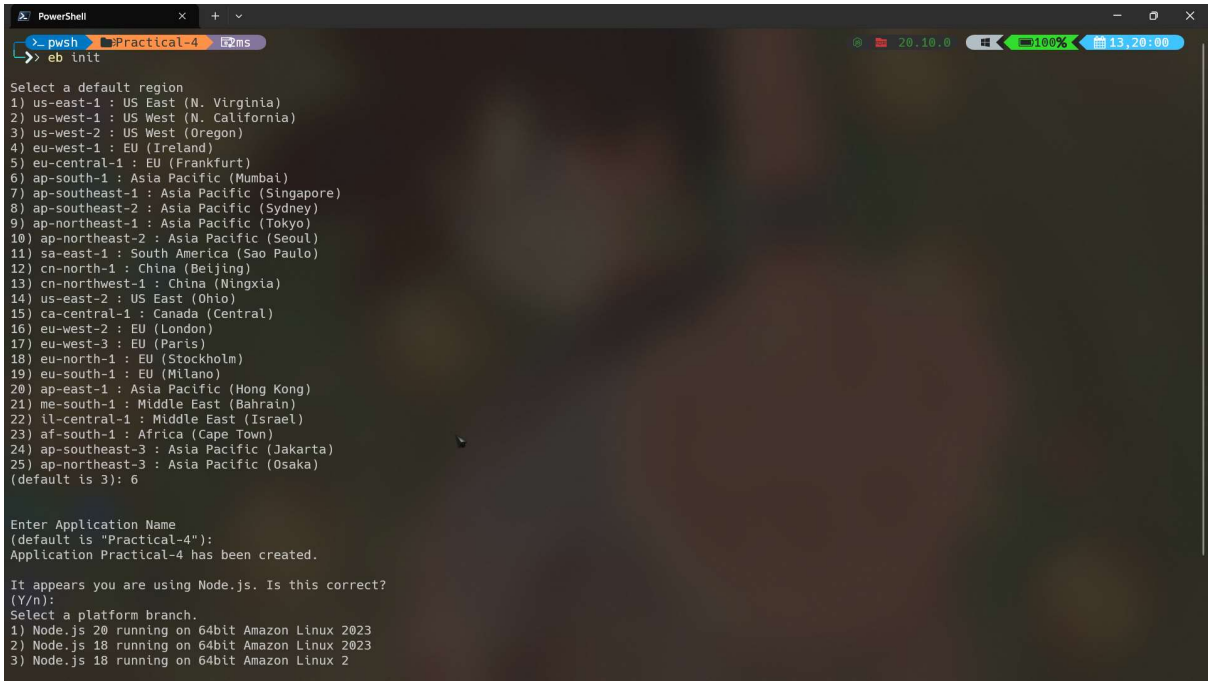
## ✓ Source Code(server.js) :

```
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);
```

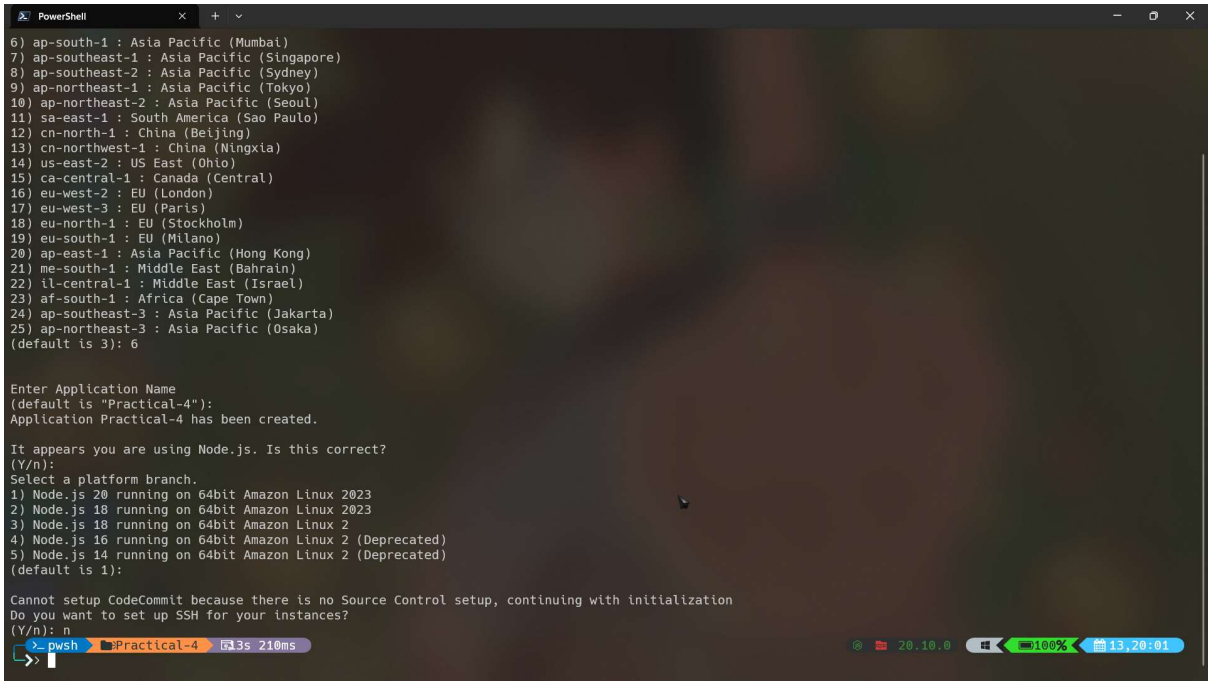
```
  console.log("New employee added:", newEmployee);
  res.status(201).json(newEmployee);
});

// 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}`);
});
```

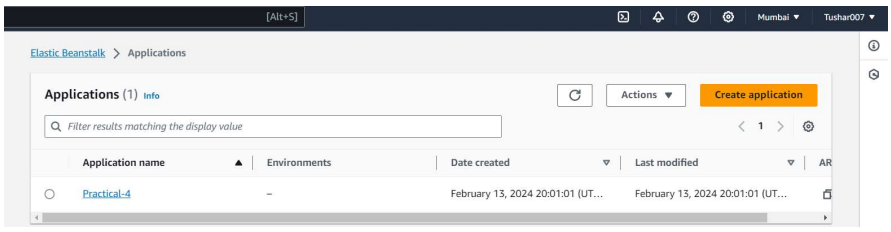
## » Initialize EB in our project folder:

**Command : `eb init`**



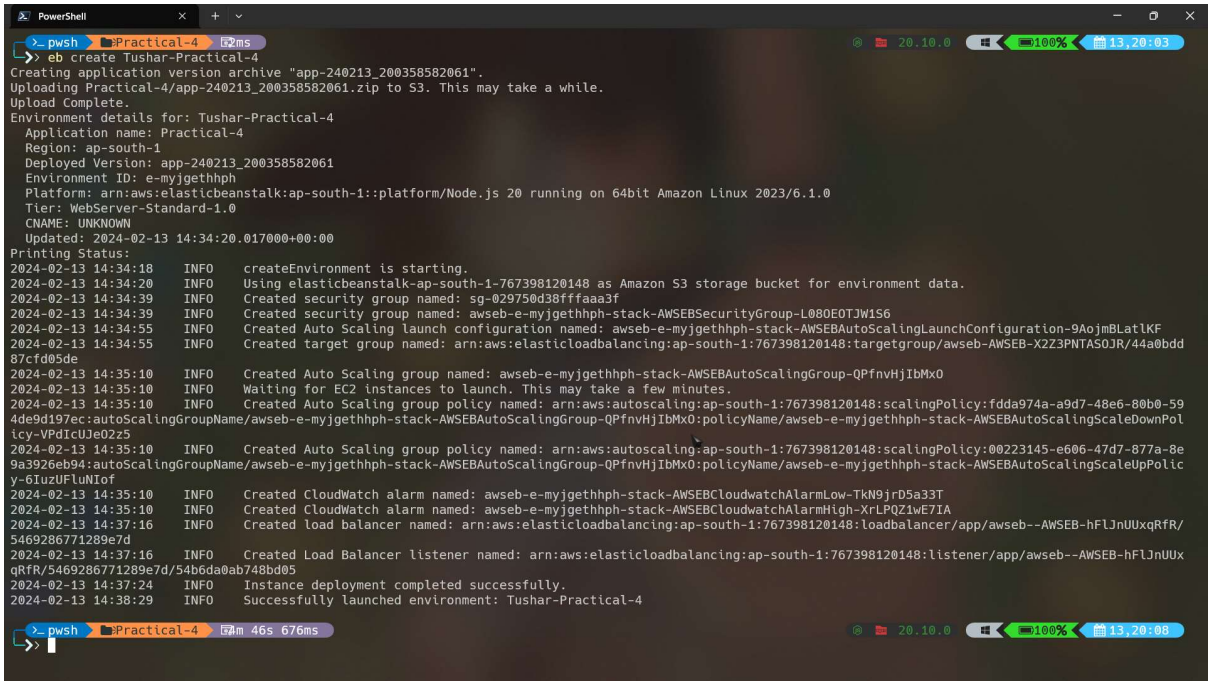

## » This shows that our Application is created :

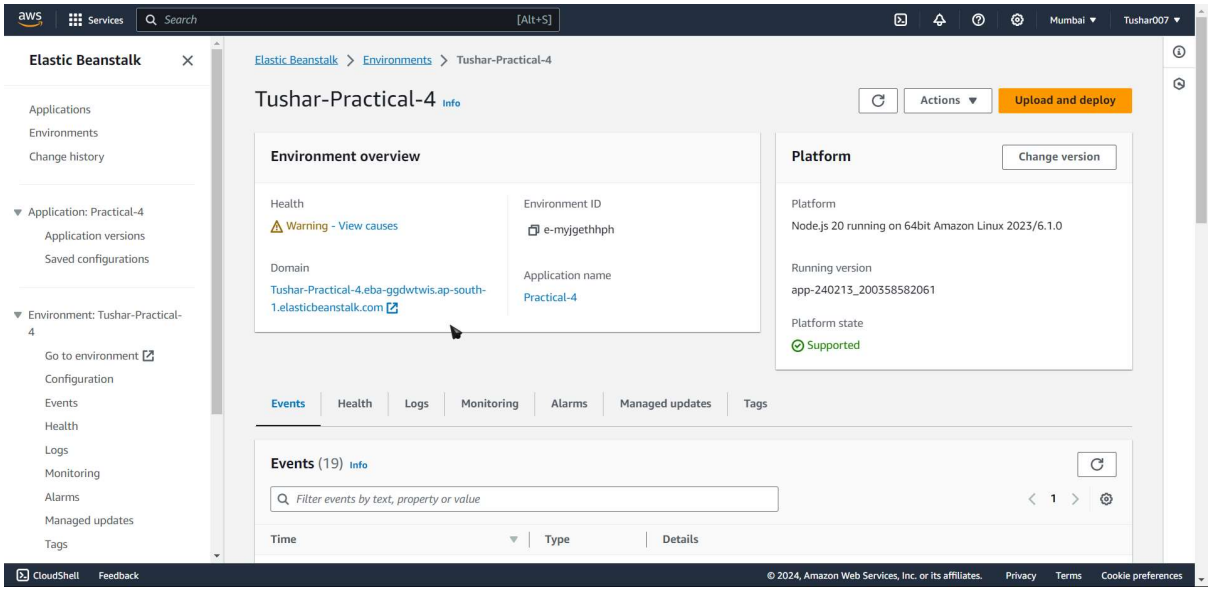


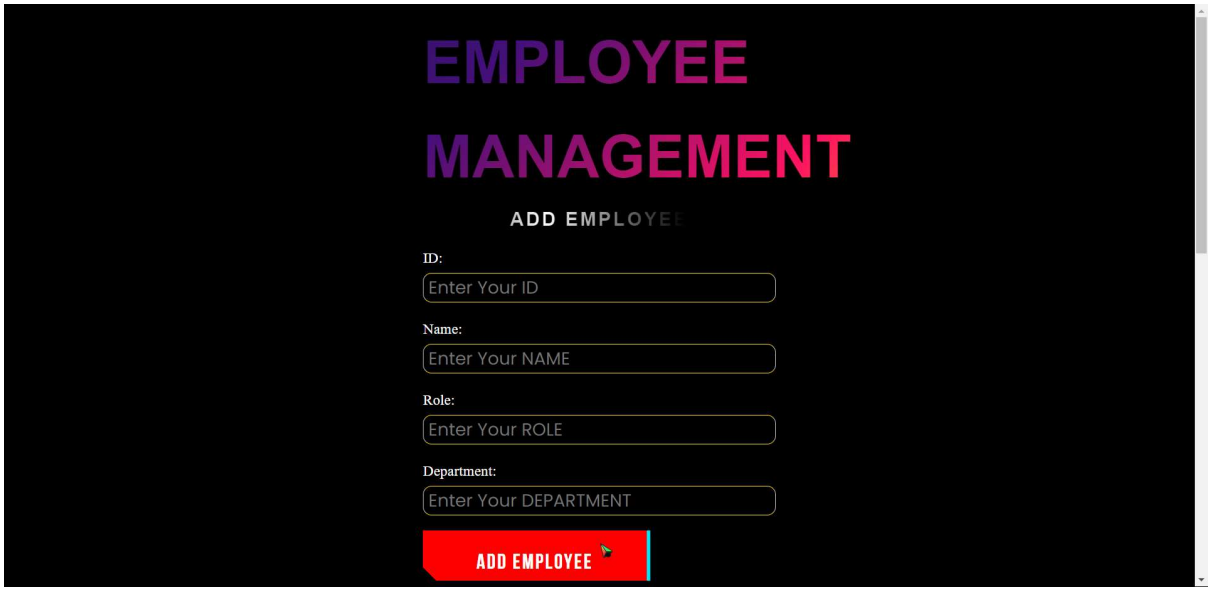
## » Now we have to create environment using command eb create (name) :

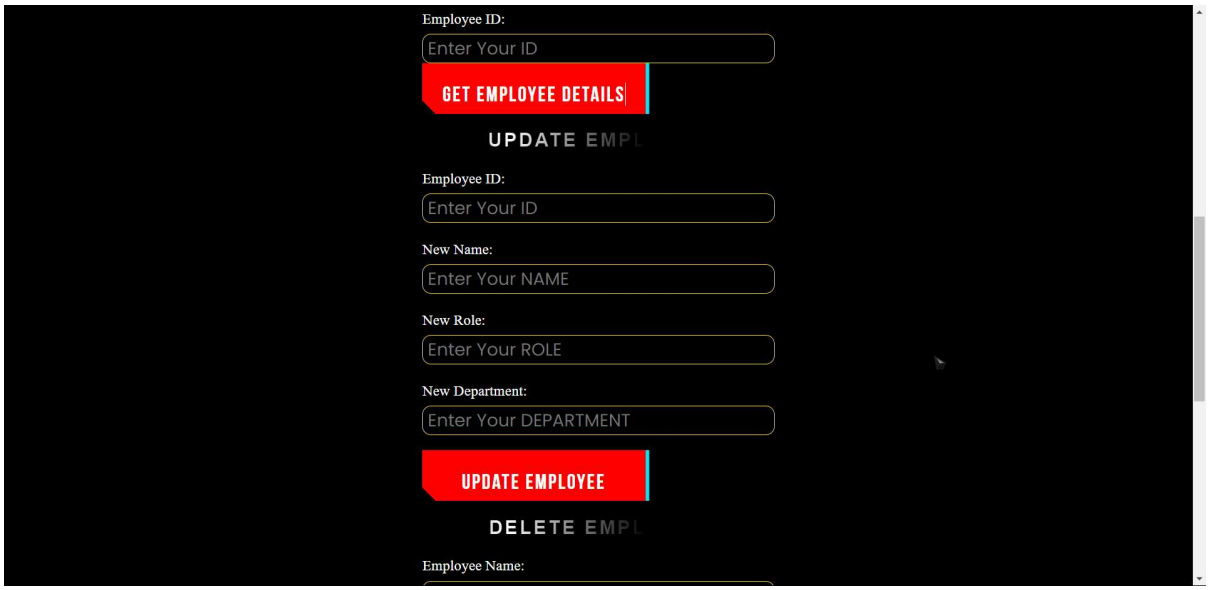
**Command : `eb create Tushar-Practical-4`**

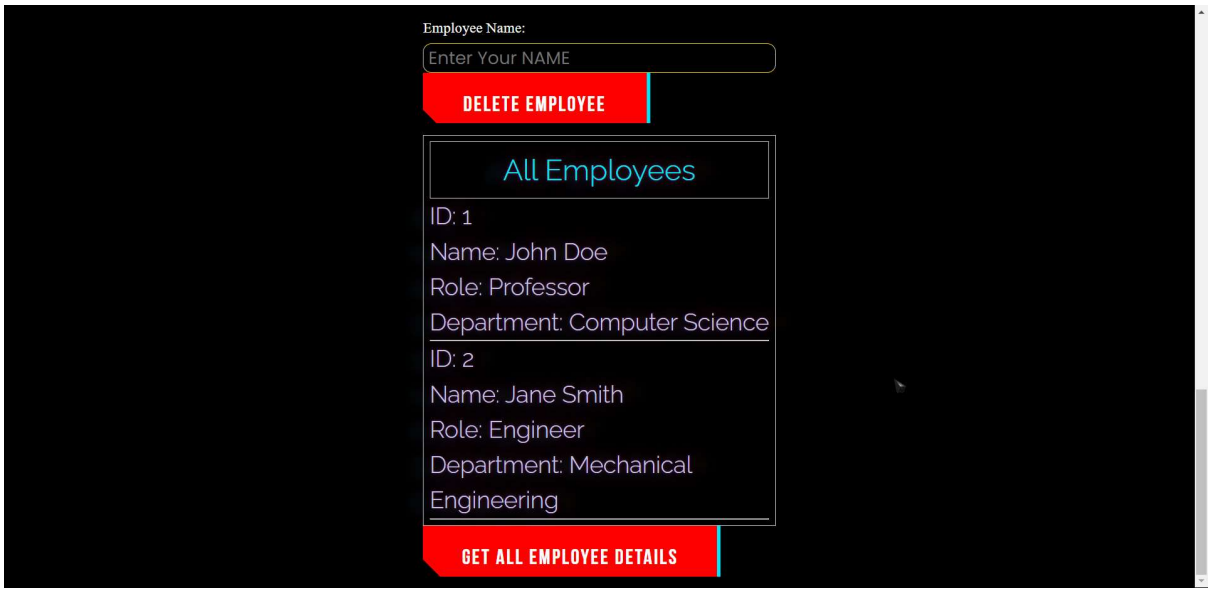


## » Output of my REST API node-js application (on AWS Cloud) :



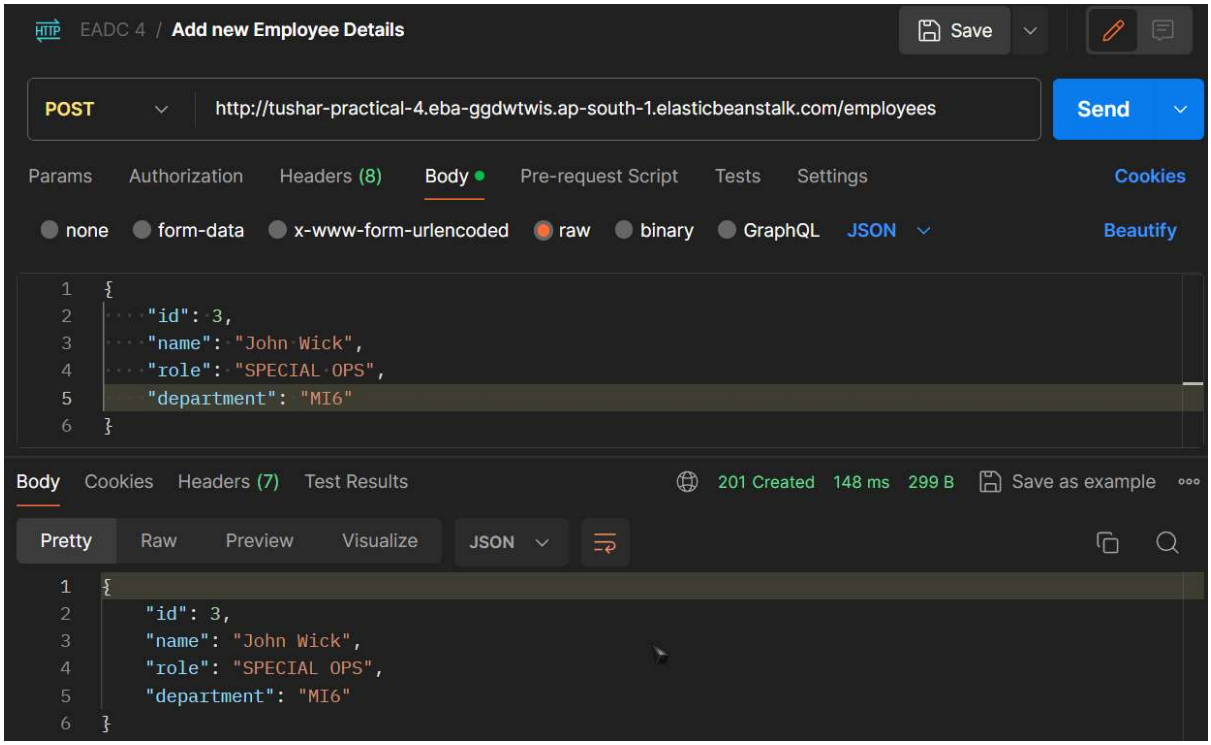



## » Output of performing CRUD operations on the server using postman :

### ➥ Add Employee :

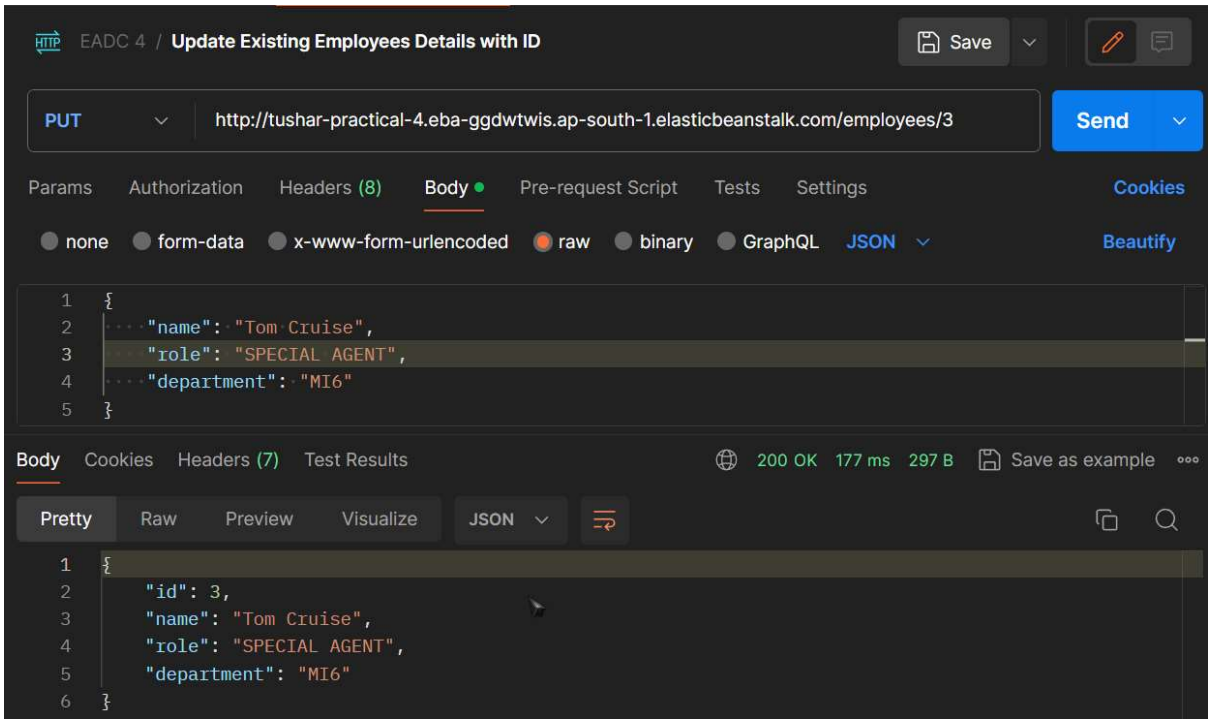
http://tushar-practical-4.eba-ggdwtwis.ap-south-1.elasticbeanstalk.com/employees




## 👉 Update Employee :

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

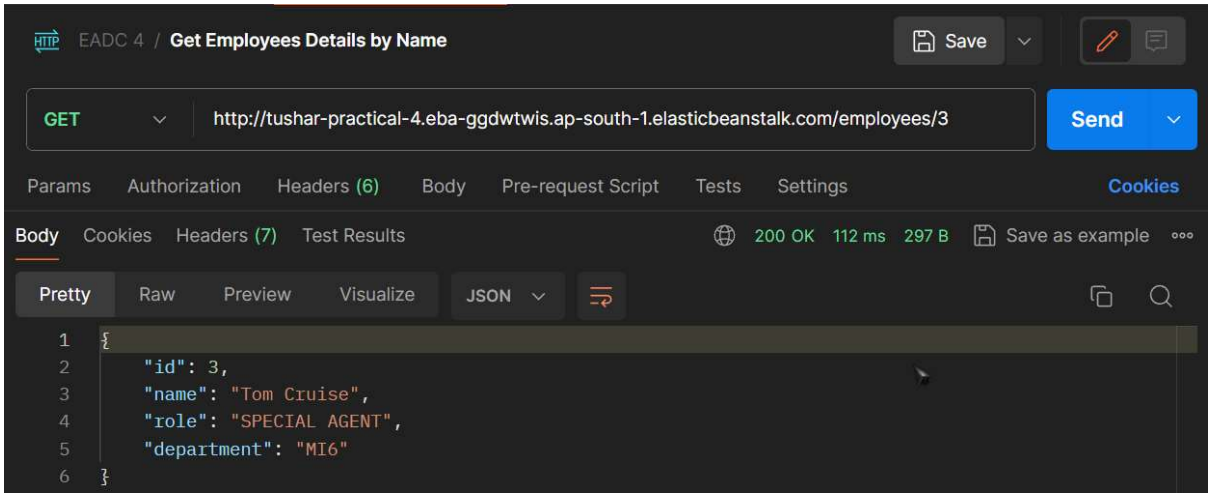
here I updating John wick and her id is 3 :




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

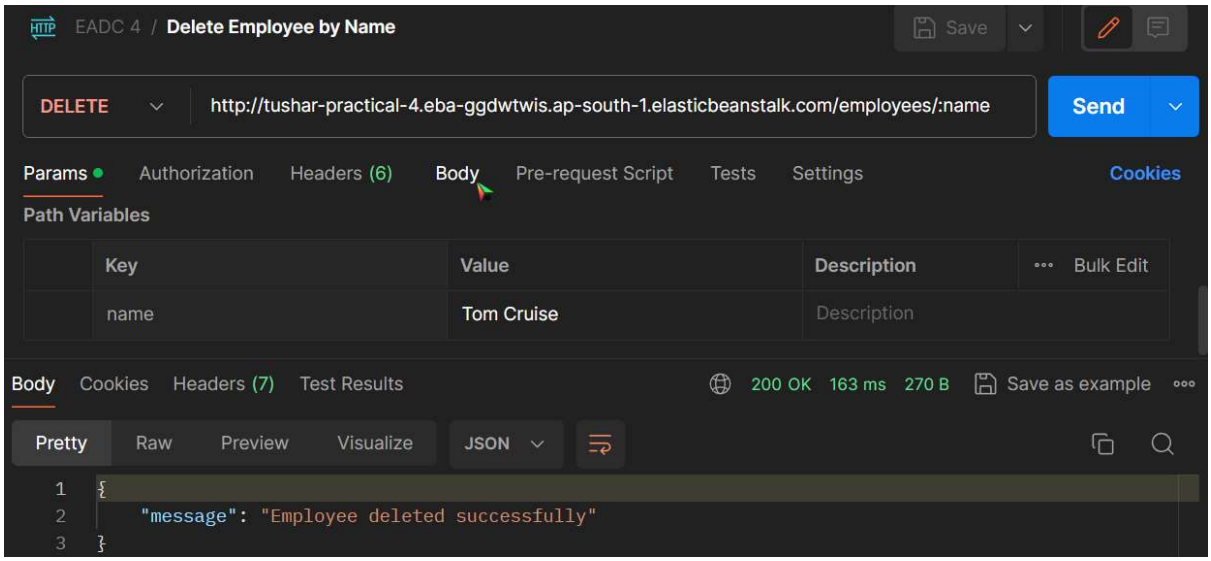



## ➥ Delete Employee by Name :

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

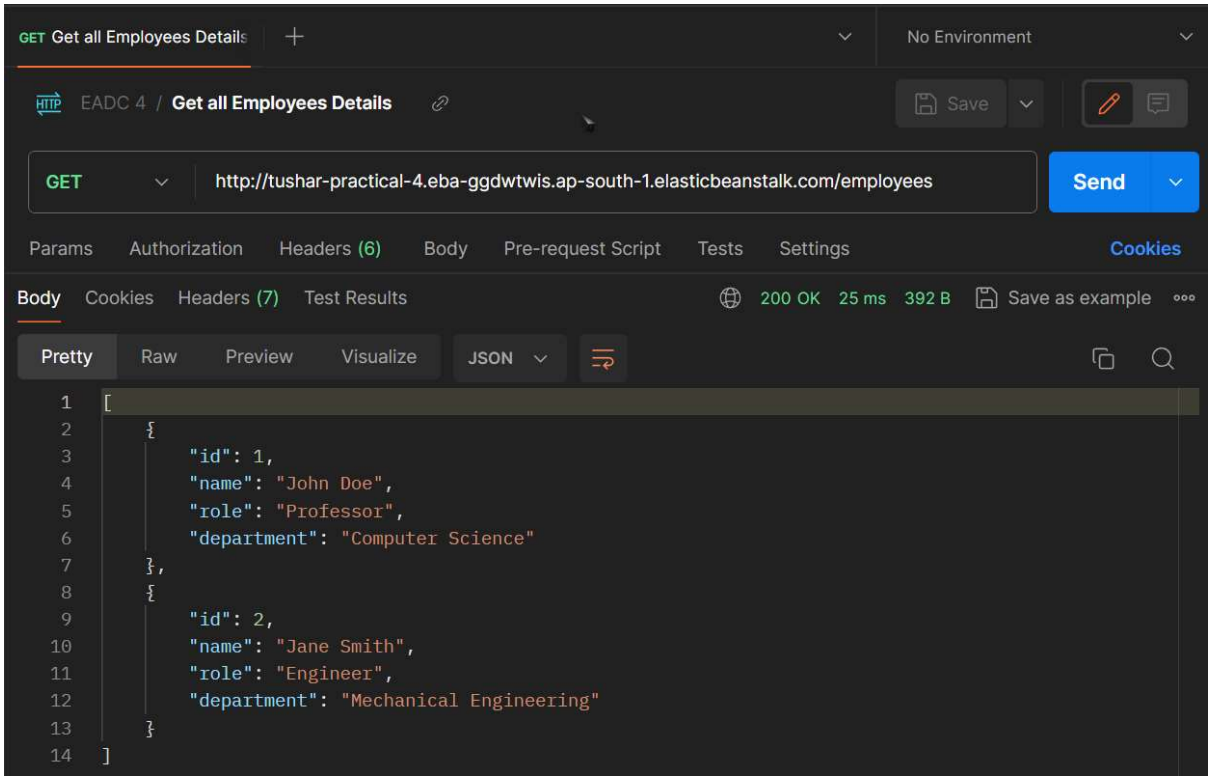
Here I delete an existing employee by Name :

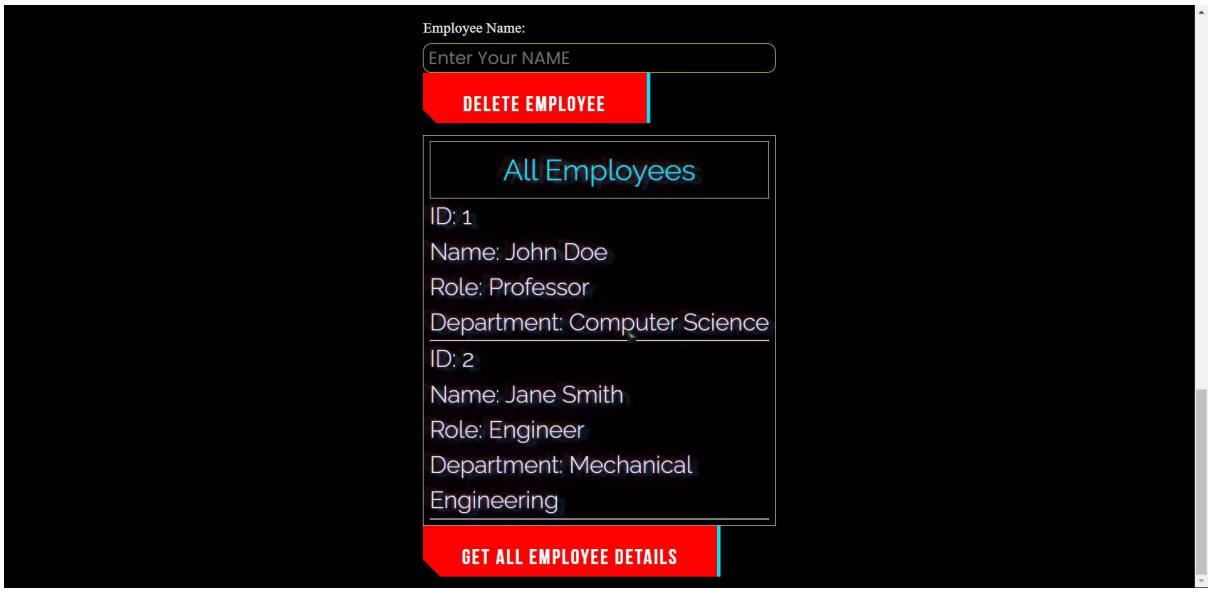
I passed value Tom Cruise in name key to delete that employee



## ➥ 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 .