PROGRAM 4: Design an employee Management system using RESTFULL APIs

**server.js**

**const express = require('express');**

**const mongoose = require('mongoose');**

**const bodyParser = require('body-parser');**

**const cors = require('cors');**

**const app = express();**

**const PORT = process.env.PORT || 3001;**

**app.use(cors());**

**app.use(bodyParser.json());**

**// MongoDB connection**

**mongoose.connect('mongodb://localhost:27017/employee-management', {**

**useNewUrlParser: true,**

**useUnifiedTopology: true,**

**});**

**// Define Employee Schema**

**const employeeSchema = new mongoose.Schema({**

**name: String,**

**position: String,**

**department: String,**

**});**

**const Employee = mongoose.model('Employee', employeeSchema);**

**// API routes**

**app.get('/api/employees', async (req, res) => {**

**try {**

**const employees = await Employee.find();**

**res.json(employees);**

**} catch (error) {**

**res.status(500).json({ error: 'Internal Server Error' });**

**}**

**});**

**app.post('/api/employees', async (req, res) => {**

**const { name, position, department } = req.body;**

**try {**

**const newEmployee = new Employee({ name, position, department });**

**await newEmployee.save();**

**res.status(201).json(newEmployee);**

**} catch (error) {**

**res.status(500).json({ error: 'Internal Server Error' });**

**}**

**});**

**app.listen(PORT, () => {**

**console.log(`Server is running on http://localhost:${PORT}`);**

**});**

**AddEmployeeForm.js**

import React from 'react';

const AddEmployeeForm = ({ newEmployee, onInputChange, onAddEmployee }) => {

  return (

    <div>

      <h2>Add Employee</h2>

      <label>Name:</label>

      <input type="text" name="name" value={newEmployee.name} onChange={onInputChange} />

      <label>Position:</label>

      <input type="text" name="position" value={newEmployee.position} onChange={onInputChange} />

      <label>Department:</label>

      <input type="text" name="department" value={newEmployee.department} onChange={onInputChange} />

      <button onClick={onAddEmployee}>Add Employee</button>

    </div>

  );

};

export default AddEmployeeForm;

**EmployeeList.js**

// components/EmployeeList.js

import React from 'react';

const EmployeeList = ({ employees }) => {

  return (

    <div>

      <h2>Employee List</h2>

      <ul>

        {employees.map(employee => (

          <li key={employee.\_id}>

            {employee.name} - {employee.position} - {employee.department}

          </li>

        ))}

      </ul>

    </div>

  );

};

export default EmployeeList;

**OUTPUT**

**Local: http://localhost:3000**

**On Your Network: http://192.168.1.8:3000**

**styles.css**

/\* styles.css \*/

body {

    font-family: 'Arial', sans-serif;

  }

  .app {

    max-width: 800px;

    margin: 0 auto;

    padding: 20px;

  }

  h1, h2 {

    color: #333;

  }

  ul {

    list-style-type: none;

    padding: 0;

  }

  li {

    margin-bottom: 10px;

  }

  label {

    display: block;

    margin-bottom: 5px;

  }

  input {

    width: 100%;

    padding: 8px;

    margin-bottom: 10px;

  }

  button {

    background-color: #4caf50;

    color: white;

    padding: 10px;

    border: none;

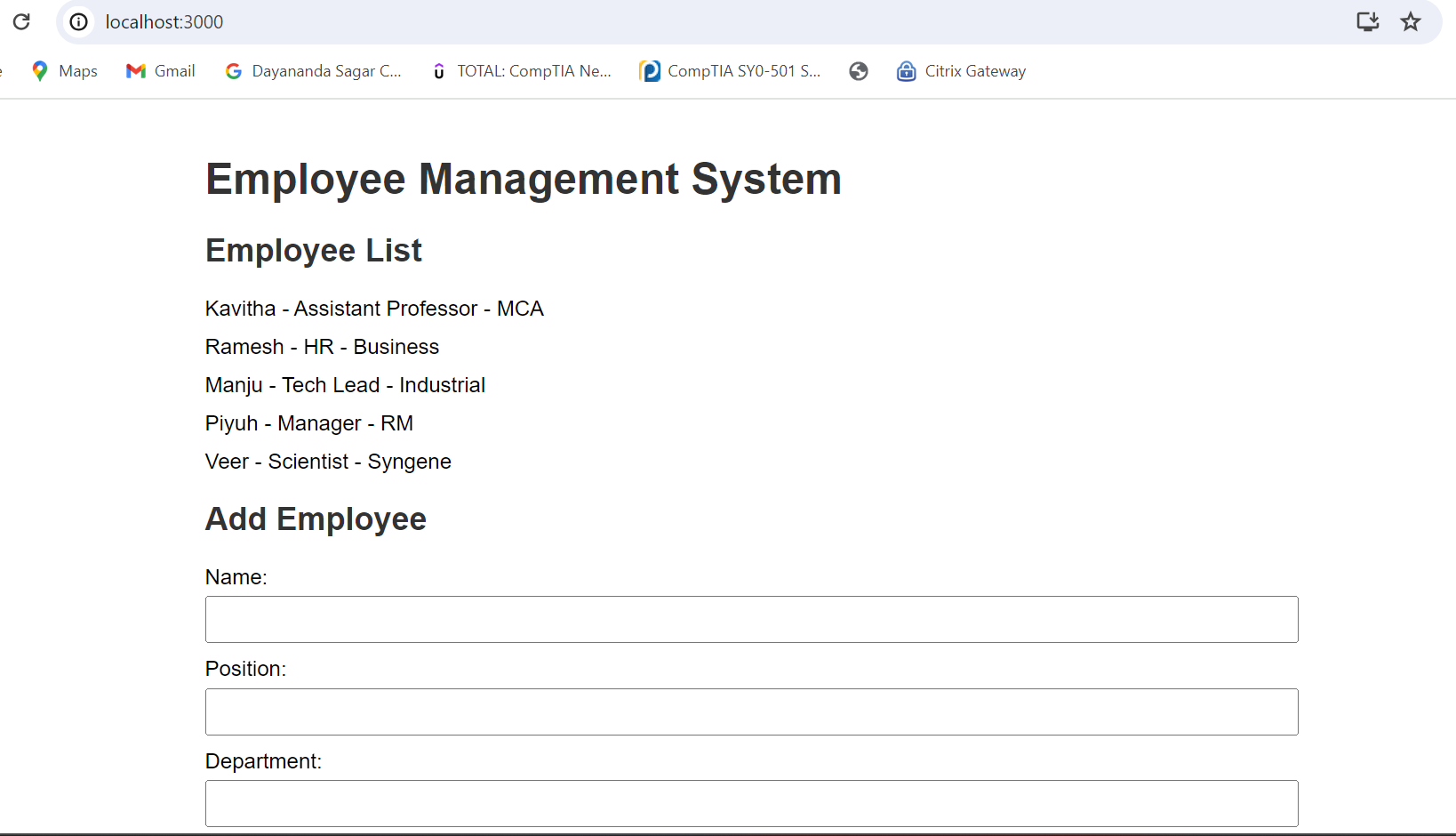
    cursor: pointer;

  }

  button:hover {

    background-color: #45a049;

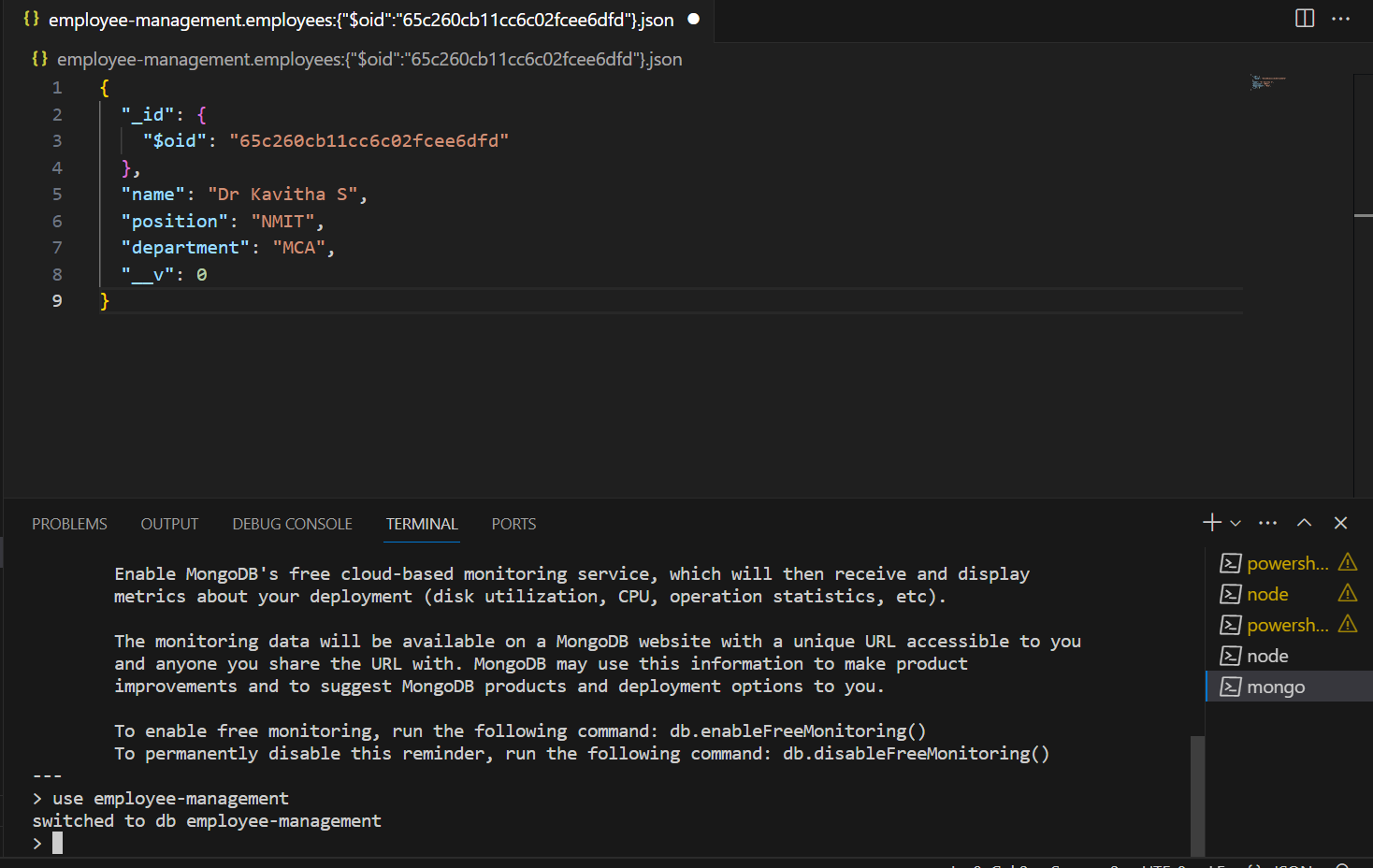
  }

****

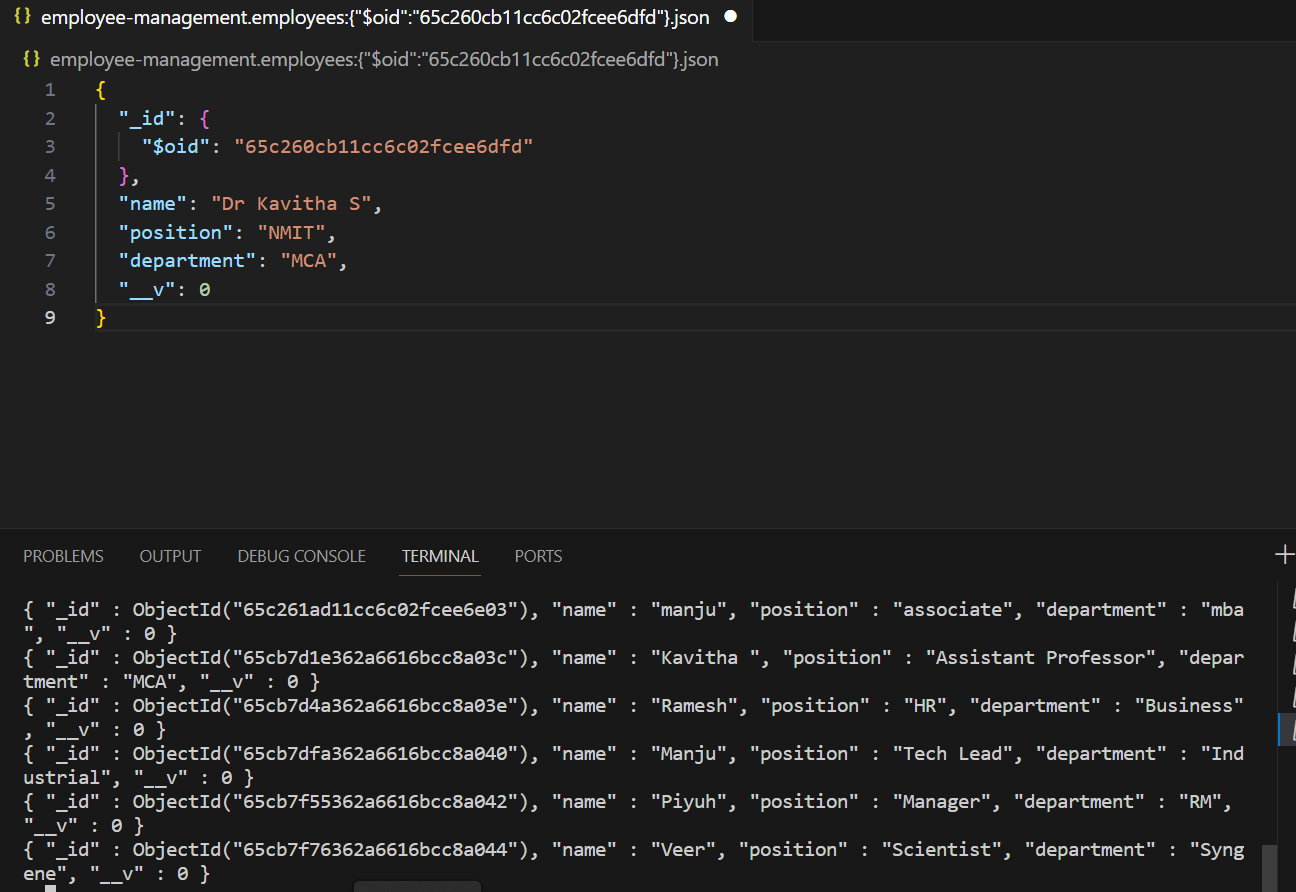
### Using MongoDB Command-Line Interface:

1. Open a new terminal.
2. Start the MongoDB shell by typing the following command:

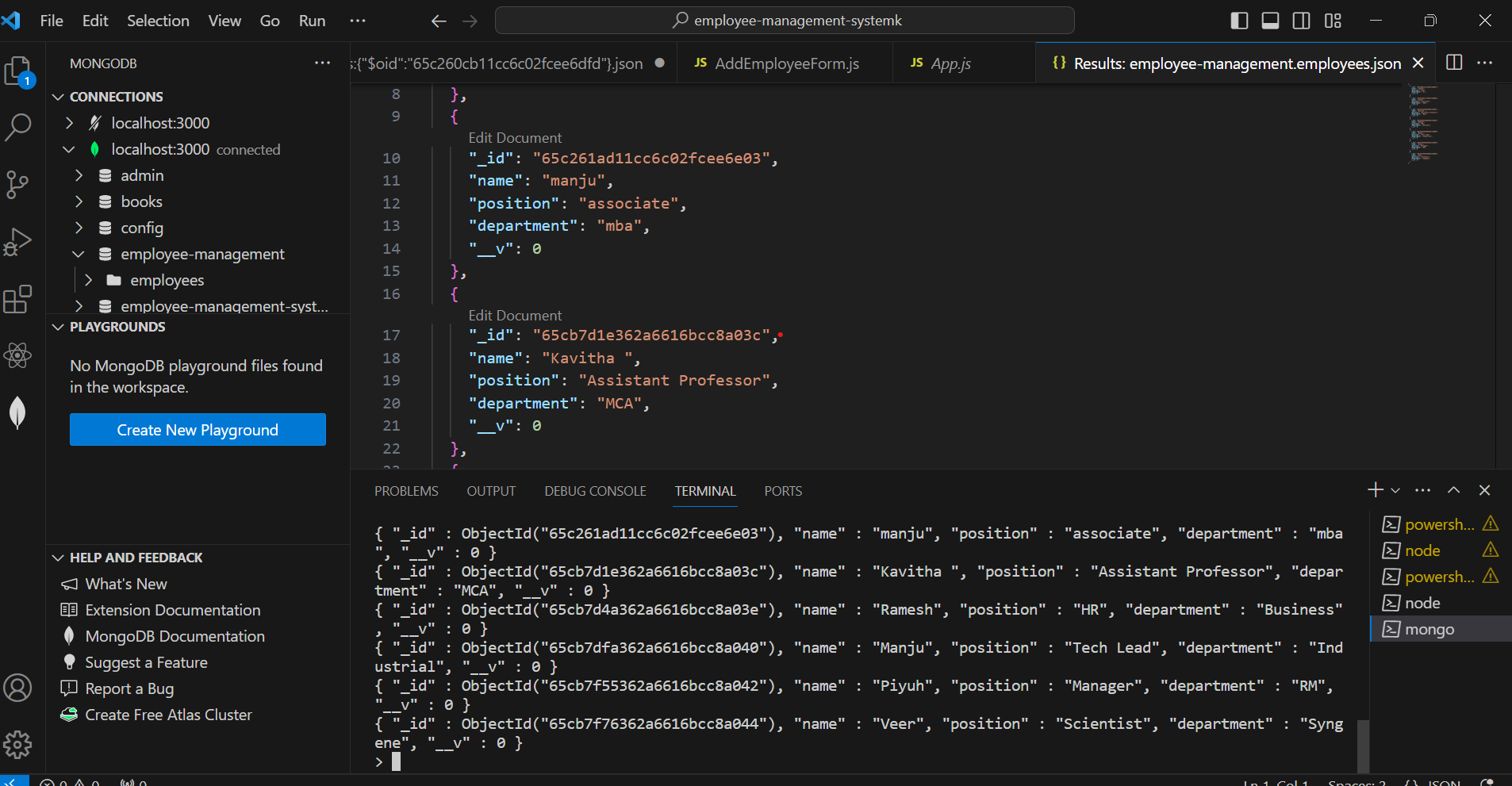
**Output for use employee-management**

****

Display all documents in the **employees** collection

****

**Collection of Document in Employee-Management**

****