**Question**

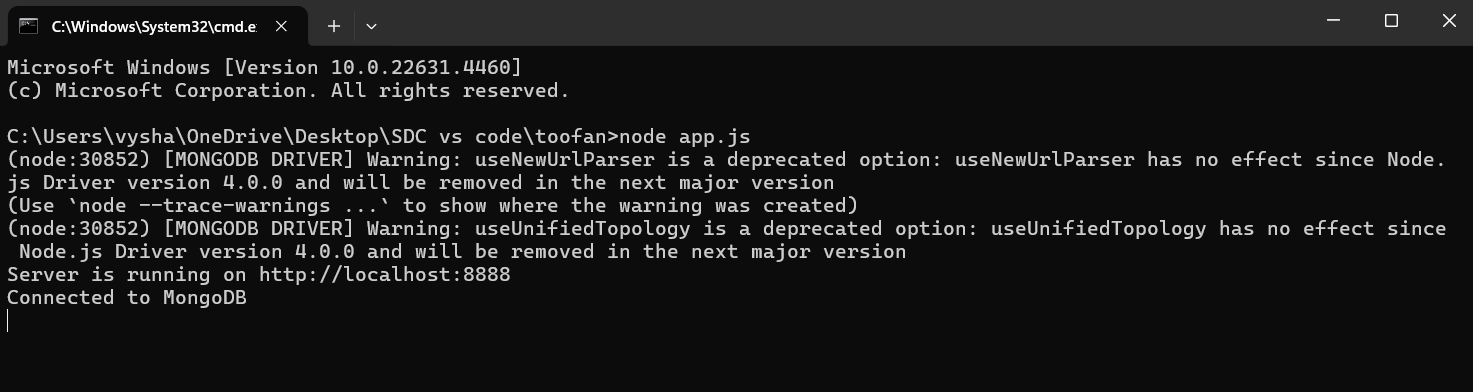
Design a schema for an Employee collection in MongoDB. Each document in the collection should have the following fields(5marks):

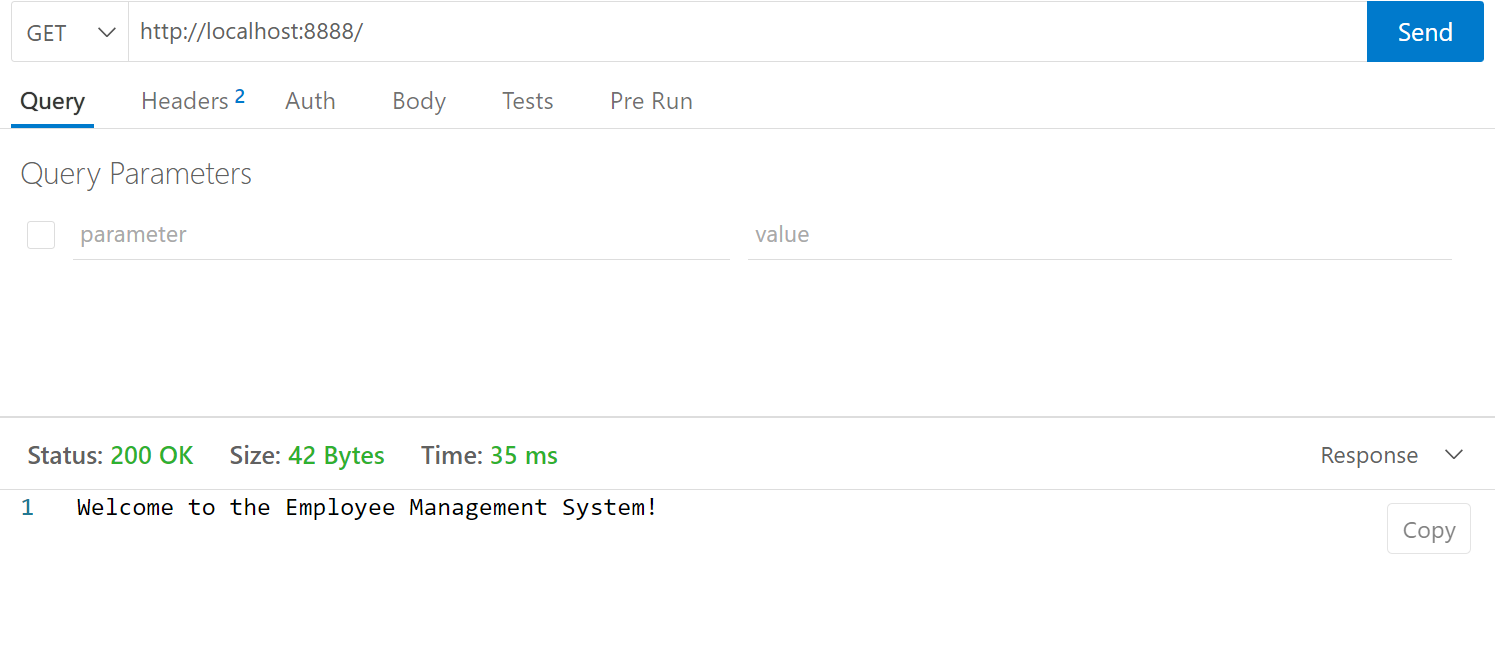
1. **employeeId**: A unique number for identifying the employee (e.g., 101).
2. **name**: The full name of the employee (e.g., "John Doe").
3. **position**: The job title or role of the employee (e.g., "Software Engineer").
4. **department**: The department the employee works in (e.g., "IT").
5. **email**: The official email address of the employee (e.g., "john.doe@example.com").
6. **salary**: The annual salary of the employee in USD (e.g., 60000).
7. **joinDate**: The date when the employee joined the organization (e.g., "2022-01-15").

**Questions for CRUD Operations (Total: 30 Marks)**

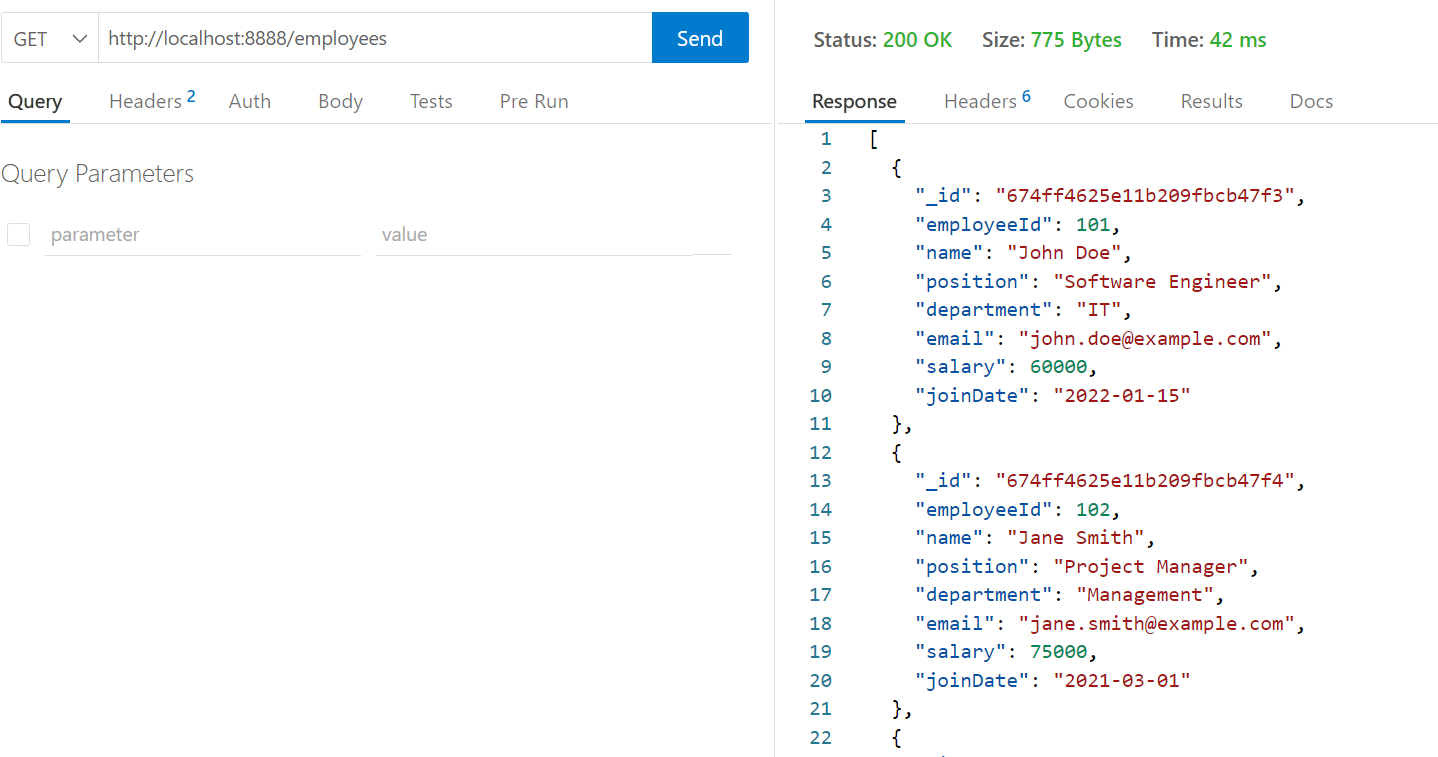
1. **LISTEN Operation**  
   Implement a Node.js server that listens on a specific port and logs incoming requests for the Employee API. (5 Marks)
2. **GET Operation**  
   Write a JavaScript API to fetch all employees from the database. (5 Marks)
3. **POST Operation**  
   Write a JavaScript API to add a new employee document to the collection. (5 Marks)
4. **PUT Operation**  
   Write a JavaScript API to update the salary of an employee based on their employeeId. (5 Marks)
5. **DELETE Operation**  
   Write a JavaScript API to delete an employee document by their employeeId. (5 Marks)

1.Server Started

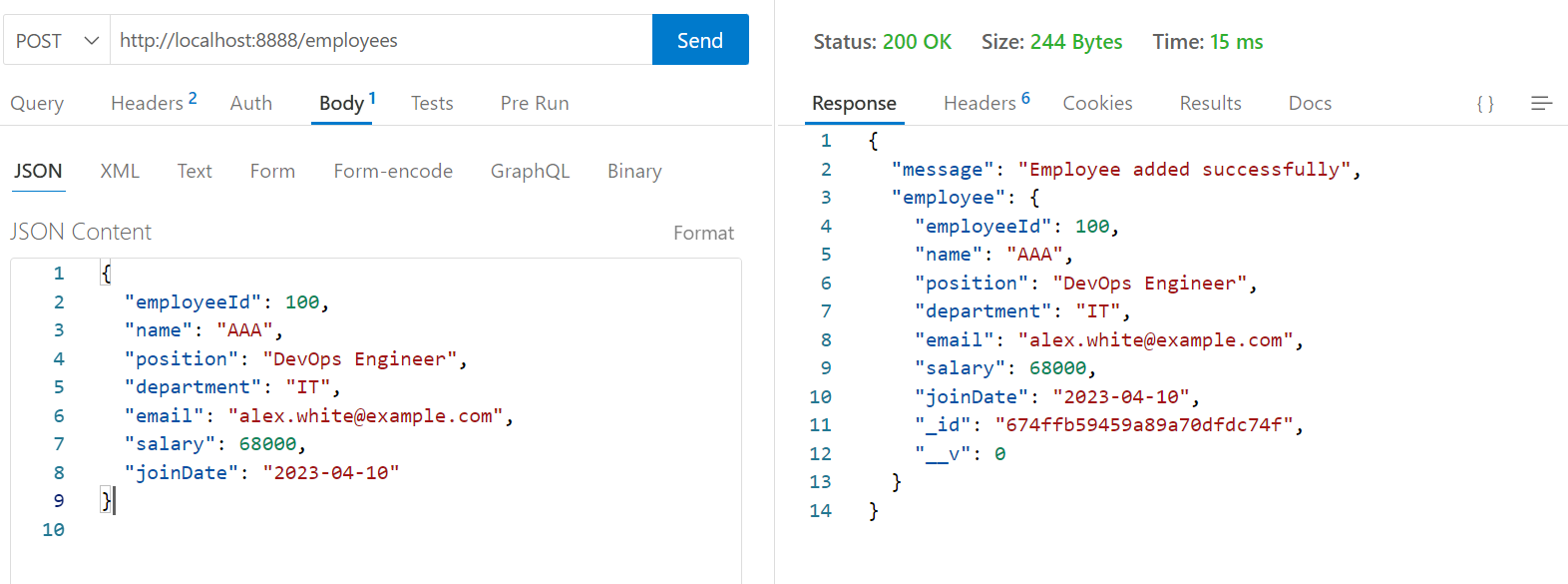


2.

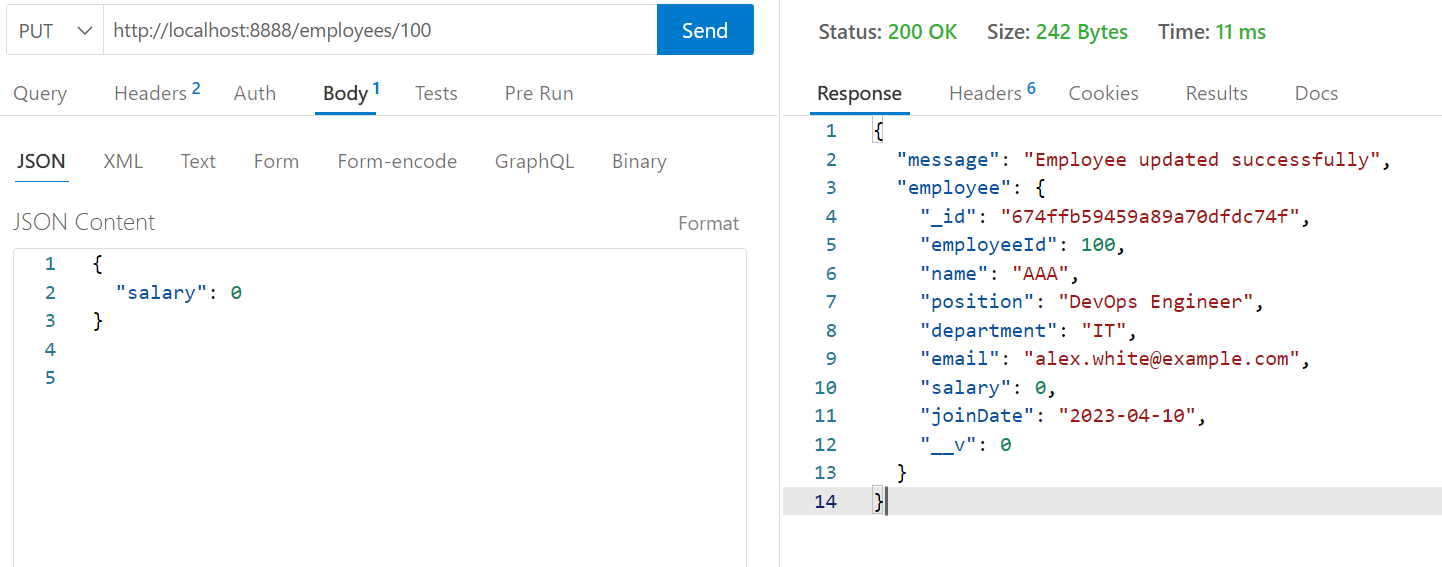
3.Fetching Employee data by GET

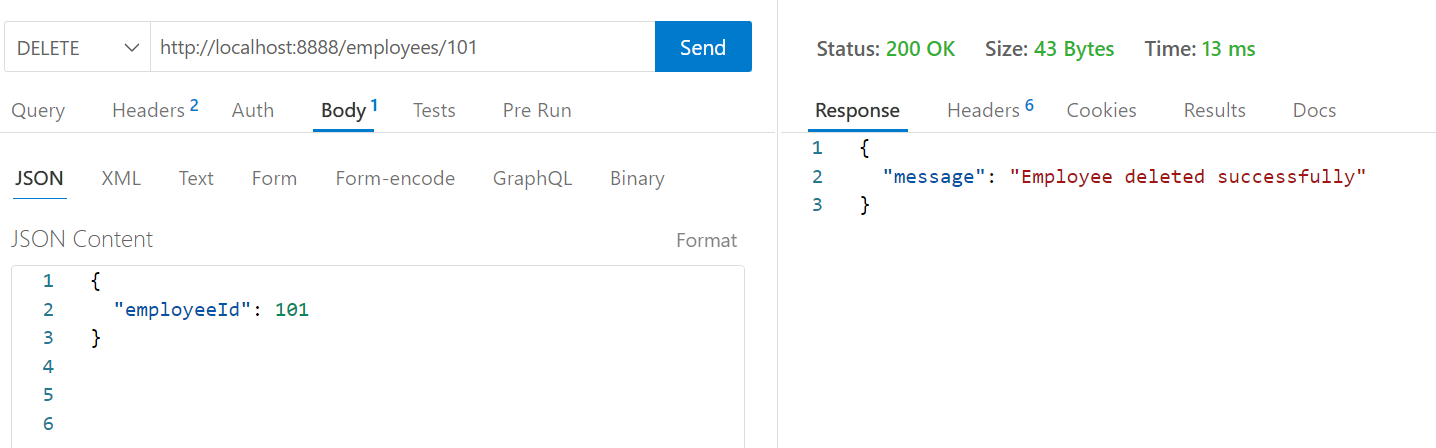


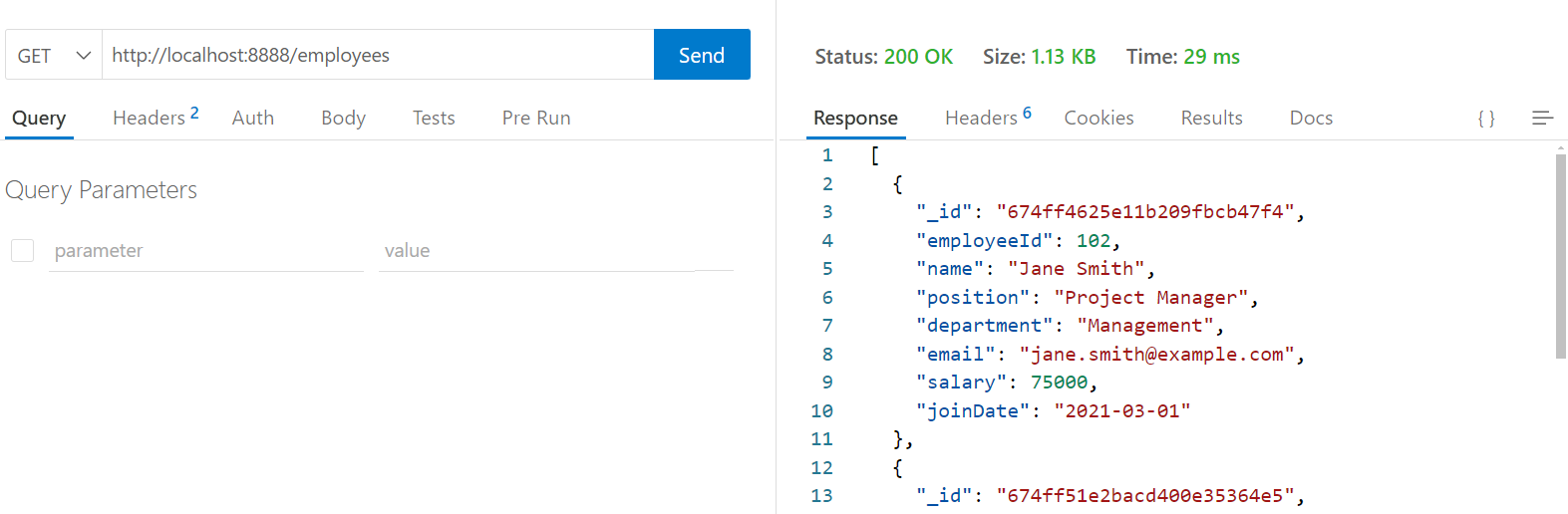
4.Adding a new Employee.



5.Updating a Salary of Employee 100 to 0.



6.Deleting Employee 101

7.After Deletion Verifying the Employee data by GET.

**App.js code**

const express = require('express');

const mongoose = require('mongoose');

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

const app = express();

app.use(bodyParser.json());

// Connect to MongoDB

mongoose.connect('mongodb://127.0.0.1:27017/employeeDB', { useNewUrlParser: true, useUnifiedTopology: true })

  .then(() => console.log('Connected to MongoDB'))

  .catch(err => console.error('Connection failed:', err));

// Define schema and model

const employeeSchema = new mongoose.Schema({

  employeeId: Number,

  name: String,

  position: String,

  department: String,

  email: String,

  salary: Number,

  joinDate: String

});

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

app.get('/', (req, res) => {

    res.send('Welcome to the Employee Management System!');

});

// GET: Fetch all employees

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

  const employees = await Employee.find();

  res.json(employees);

});

// POST: Add a new employee

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

  const newEmployee = new Employee(req.body);

  await newEmployee.save();

  res.json({ message: 'Employee added successfully', employee: newEmployee });

});

// PUT: Update employee salary

app.put('/employees/:employeeId', async (req, res) => {

  const { employeeId } = req.params;

  const { salary } = req.body;

  const updatedEmployee = await Employee.findOneAndUpdate(

    { employeeId: employeeId },

    { salary: salary },

    { new: true }

  );

  res.json({ message: 'Employee updated successfully', employee: updatedEmployee });

});

// DELETE: Delete an employee

app.delete('/employees/:employeeId', async (req, res) => {

  const { employeeId } = req.params;

  await Employee.findOneAndDelete({ employeeId: employeeId });

  res.json({ message: 'Employee deleted successfully' });

});

// Start server

const PORT = 8888;

app.listen(PORT, () => {

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

});

**MongoDB Employee Collection**

[

{

"employeeId": 101,

"name": "John Doe",

"position": "Software Engineer",

"department": "IT",

"email": "john.doe@example.com",

"salary": 60000,

"joinDate": "2022-01-15"

},

{

"employeeId": 102,

"name": "Jane Smith",

"position": "Project Manager",

"department": "Management",

"email": "jane.smith@example.com",

"salary": 75000,

"joinDate": "2021-03-01"

},

{

"employeeId": 103,

"name": "Emily Brown",

"position": "QA Analyst",

"department": "Quality Assurance",

"email": "emily.brown@example.com",

"salary": 55000,

"joinDate": "2022-06-10"

},

{

"employeeId": 104,

"name": "Michael Green",

"position": "HR Executive",

"department": "Human Resources",

"email": "michael.green@example.com",

"salary": 45000,

"joinDate": "2020-09-25"

},

{

"employeeId": 105,

"name": "Sarah Johnson",

"position": "Marketing Specialist",

"department": "Marketing",

"email": "sarah.johnson@example.com",

"salary": 50000,

"joinDate": "2021-11-05"

}

]