TASK: Build a small backend using Node.js (Express) and PostgreSQL that performs CRUD operations.

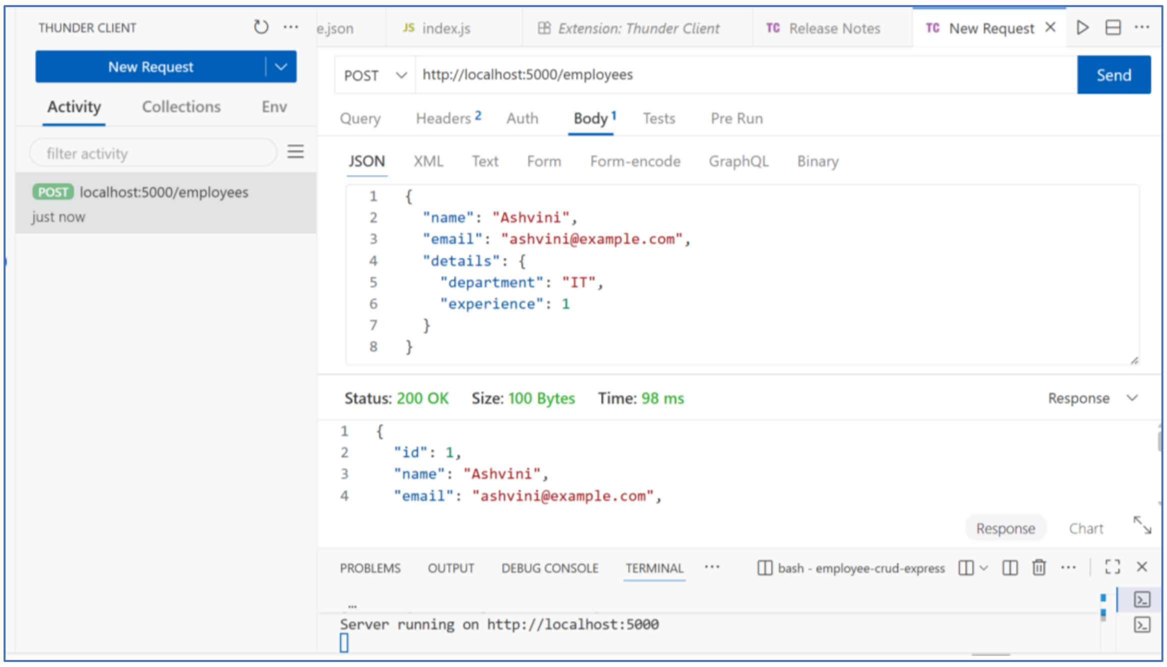
OUTPUT:

- **POST /employees → Add a new employee:**

  Tool, I used to send POST data is Thunder Client – VS Code extension

  3 records are added using POST.

  1) Ashvini user data



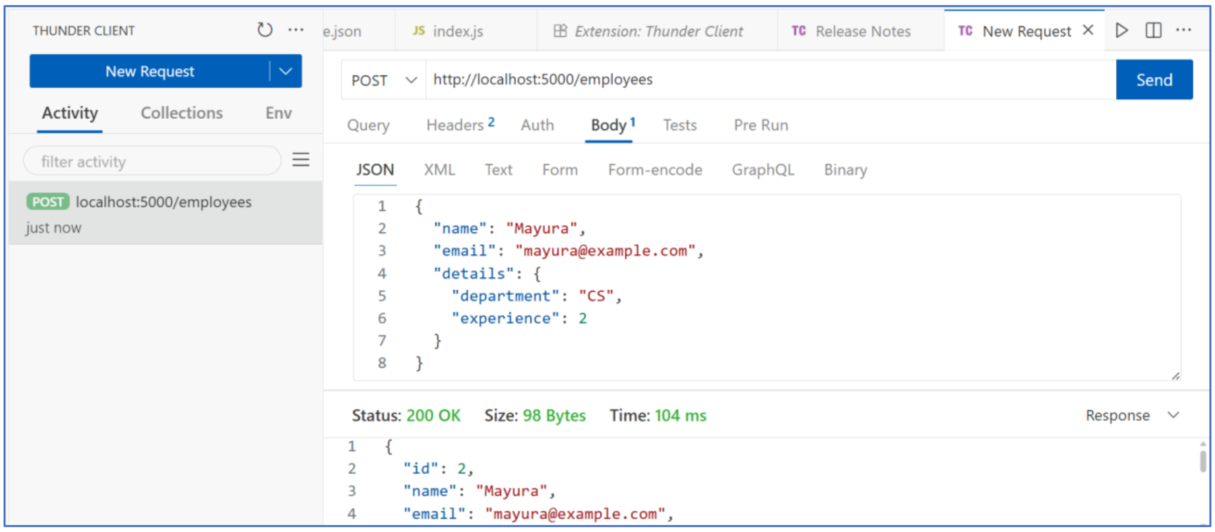
```
{
  "name": "Ashvini",
  "email": "ashvini@example.com",
  "details": {
    "department": "IT",
    "experience": 1
  }
}
```

Status: 200 OK Size: 100 Bytes Time: 98 ms

```
{
  "id": 1,
  "name": "Ashvini",
  "email": "ashvini@example.com",
```

```
Server running on http://localhost:5000
```

  2) Mayura user data



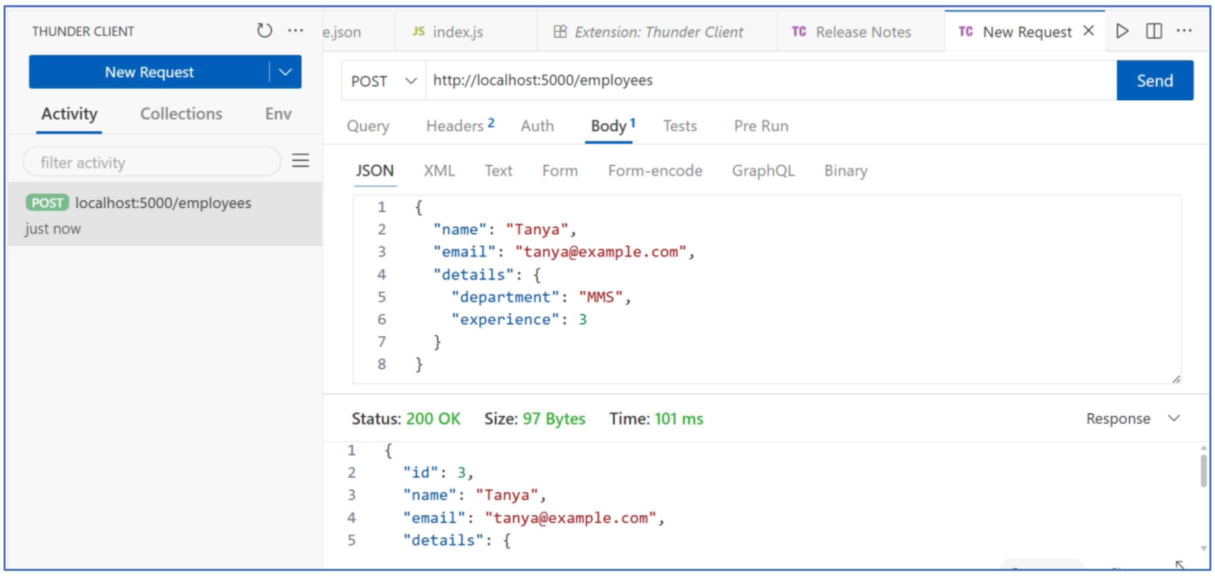
```
{
  "name": "Mayura",
  "email": "mayura@example.com",
  "details": {
    "department": "CS",
    "experience": 2
  }
}
```

Status: 200 OK Size: 98 Bytes Time: 104 ms

```
{
  "id": 2,
  "name": "Mayura",
  "email": "mayura@example.com",
```

s

3) Tanya user data



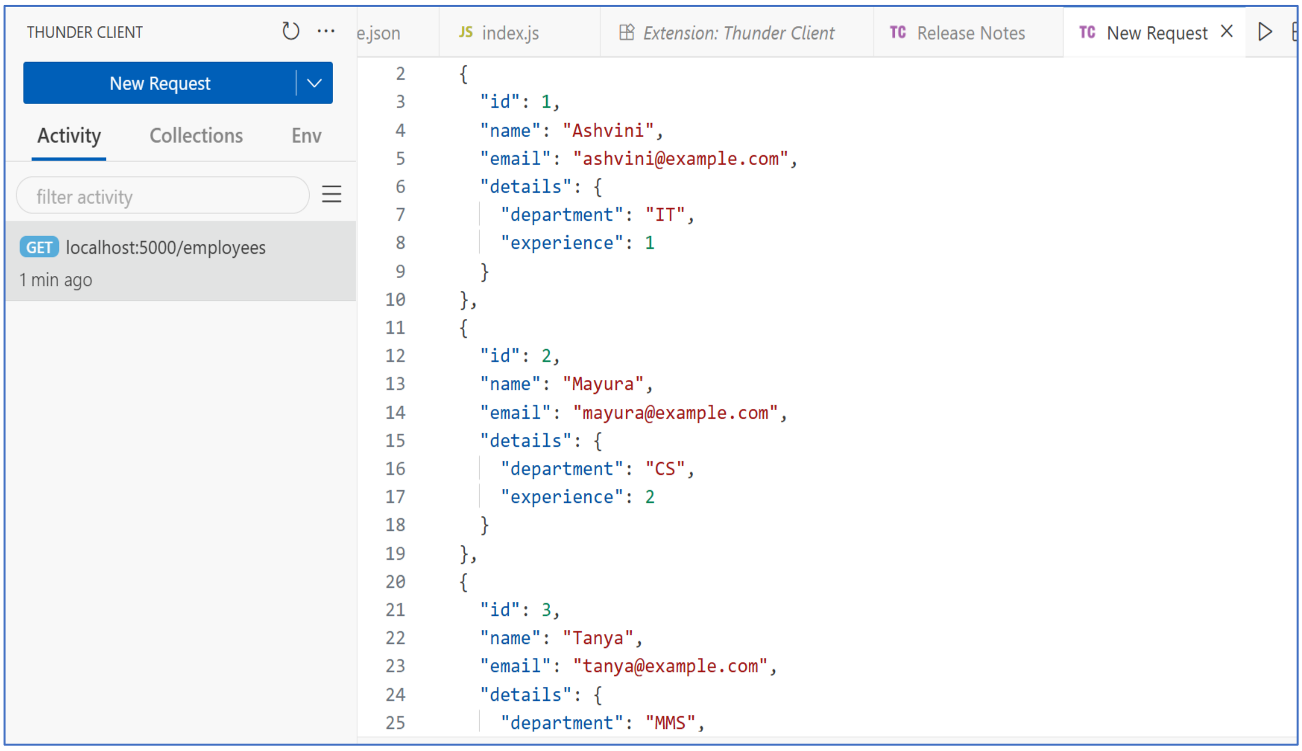
POST http://localhost:5000/employees

```
{
  "name": "Tanya",
  "email": "tanya@example.com",
  "details": {
    "department": "MMS",
    "experience": 3
  }
}
```

Status: 200 OK Size: 97 Bytes Time: 101 ms

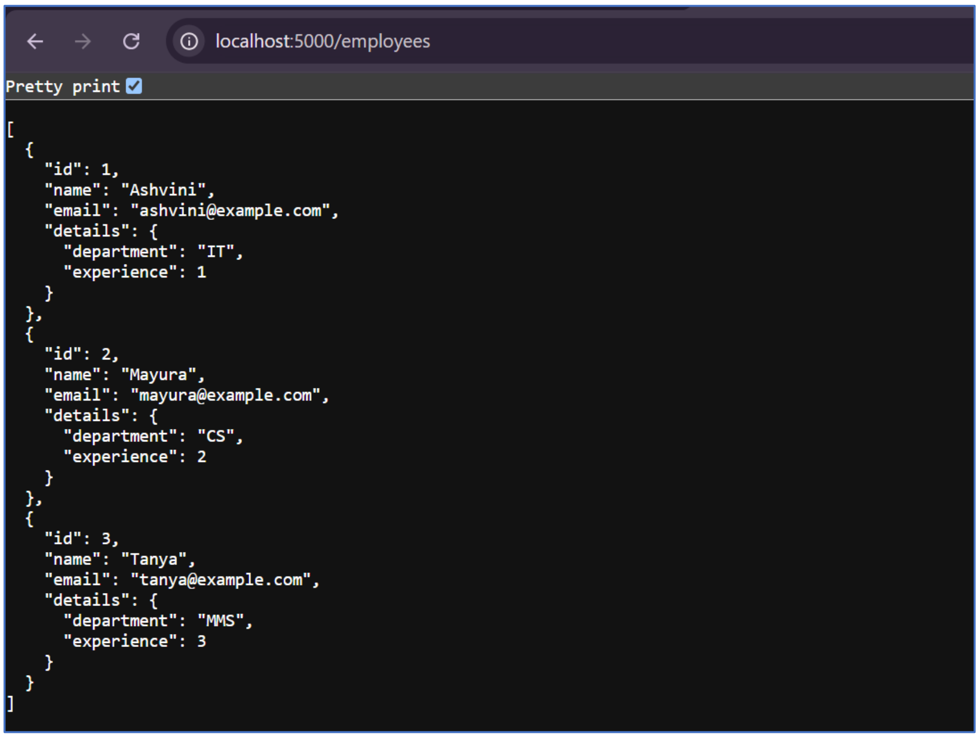
```
{
  "id": 3,
  "name": "Tanya",
  "email": "tanya@example.com",
  "details": {
```

- **GET /employees → Fetch all employees:**



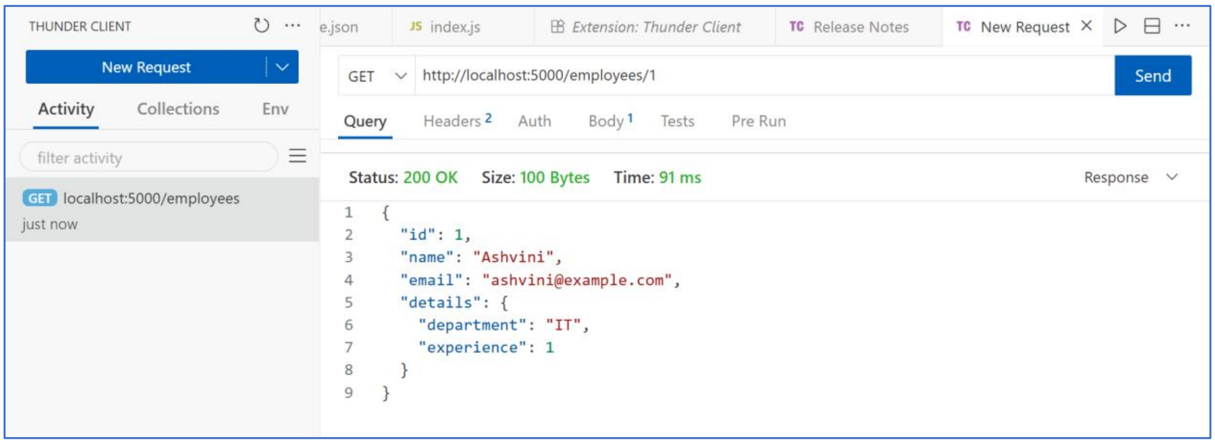
```
  {
    "id": 1,
    "name": "Ashvini",
    "email": "ashvini@example.com",
    "details": {
      "department": "IT",
      "experience": 1
    }
  },
  {
    "id": 2,
    "name": "Mayura",
    "email": "mayura@example.com",
    "details": {
      "department": "CS",
      "experience": 2
    }
  },
  {
    "id": 3,
    "name": "Tanya",
    "email": "tanya@example.com",
    "details": {
      "department": "MMS",
```

Accessing From Chrome:



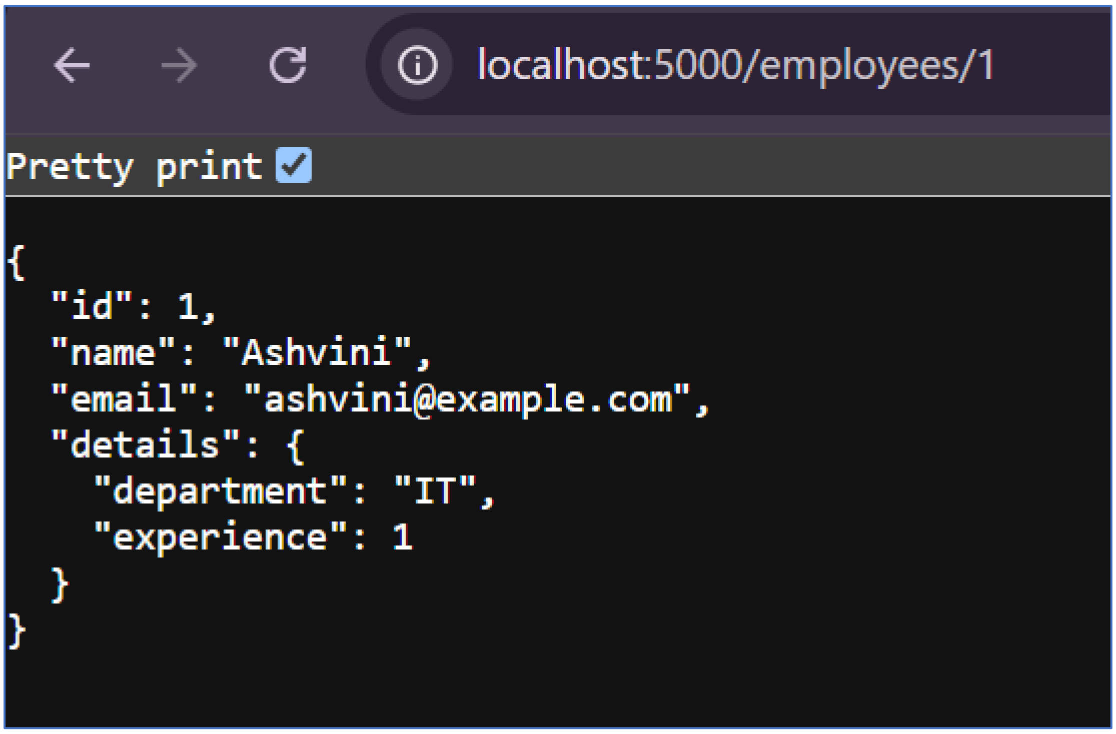
```
[
  {
    "id": 1,
    "name": "Ashvini",
    "email": "ashvini@example.com",
    "details": {
      "department": "IT",
      "experience": 1
    }
  },
  {
    "id": 2,
    "name": "Mayura",
    "email": "mayura@example.com",
    "details": {
      "department": "CS",
      "experience": 2
    }
  },
  {
    "id": 3,
    "name": "Tanya",
    "email": "tanya@example.com",
    "details": {
      "department": "MMS",
      "experience": 3
    }
  }
]
```

- **GET /employees/:id → Fetch one employee by id:**



Status: 200 OK Size: 100 Bytes Time: 91 ms

```
{
  "id": 1,
  "name": "Ashvini",
  "email": "ashvini@example.com",
  "details": {
    "department": "IT",
    "experience": 1
  }
}
```



localhost:5000/employees/1

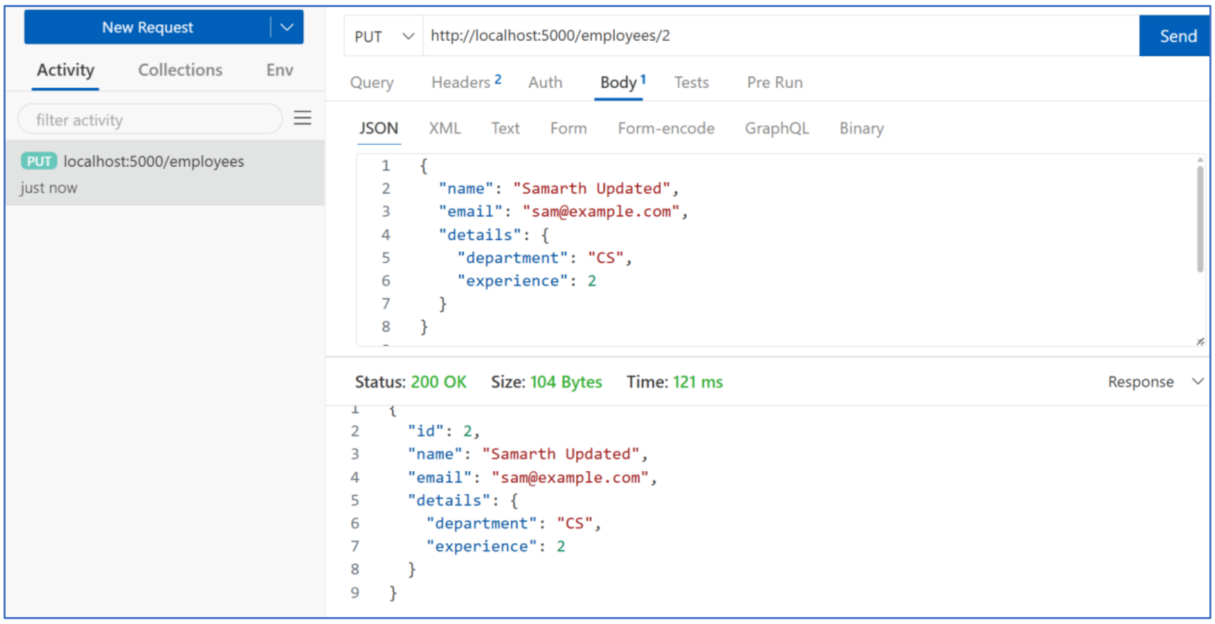
Pretty print ✓

```
{
  "id": 1,
  "name": "Ashvini",
  "email": "ashvini@example.com",
  "details": {
    "department": "IT",
    "experience": 1
  }
}
```

- **PUT /employees/:id → Update an employee's details:**

>> Update user with id 2

```
{
 "name": "Samarth Updated",
 "email": "sam@example.com",
 "details": {
   "department": "CS",
   "experience": 2
 }
}
```


New Request

Activity | Collections | Env

PUT localhost:5000/employees
just now

PUT http://localhost:5000/employees/2 Send

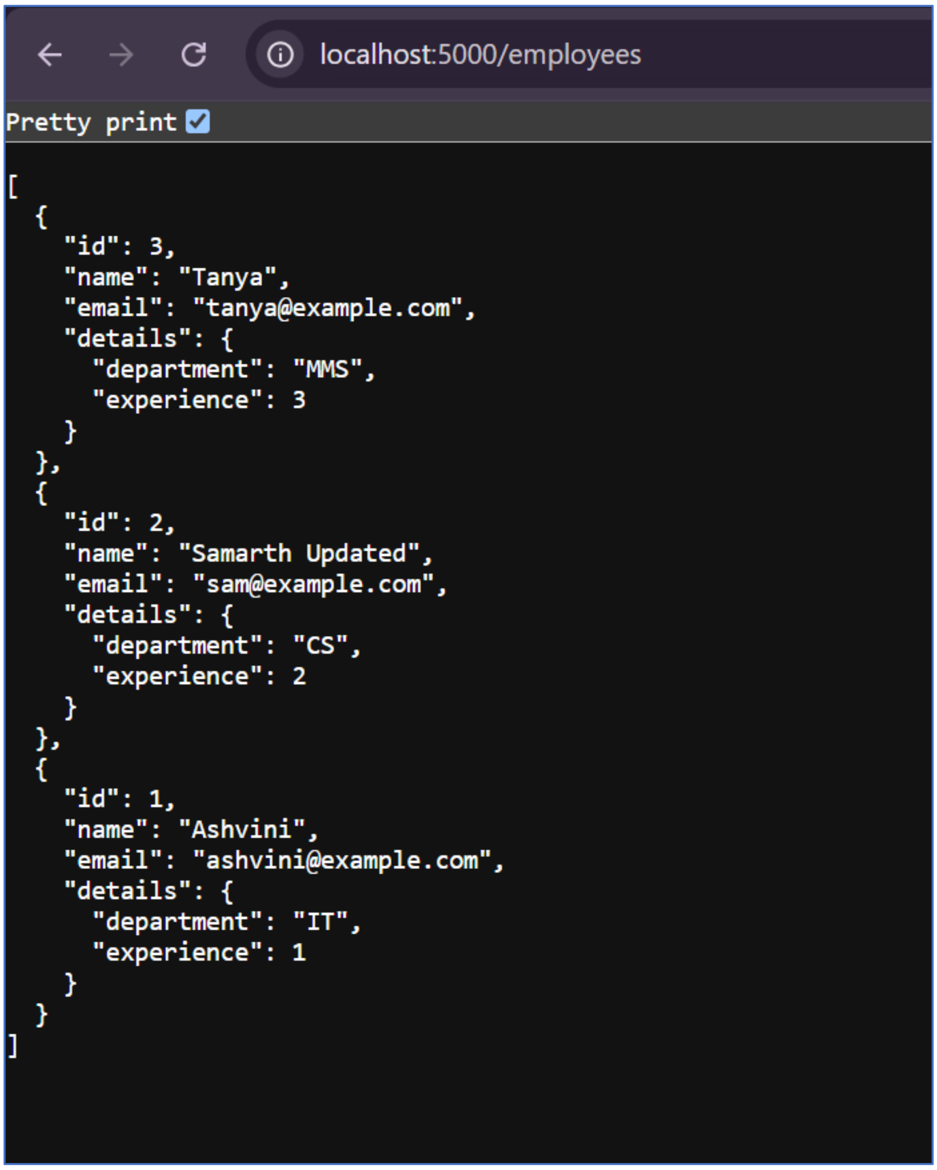
Query | Headers 2 | Auth | Body 1 | Tests | Pre Run

JSON | XML | Text | Form | Form-encode | GraphQL | Binary

```
{
  "name": "Samarth Updated",
  "email": "sam@example.com",
  "details": {
    "department": "CS",
    "experience": 2
  }
}
```

Status: 200 OK Size: 104 Bytes Time: 121 ms Response

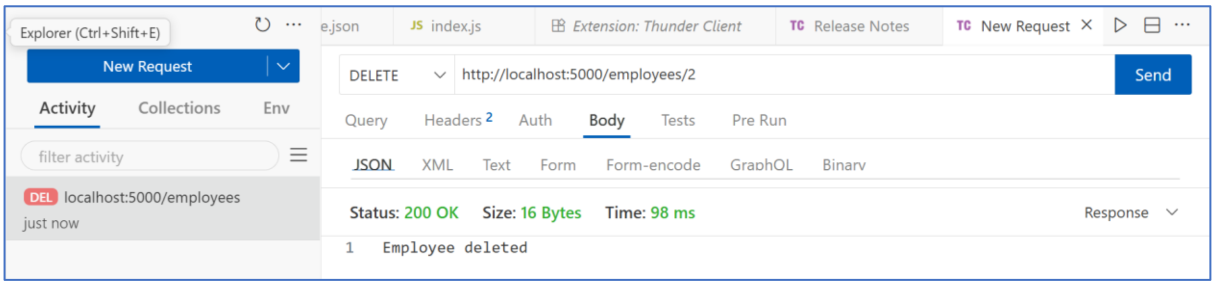
```
{
  "id": 2,
  "name": "Samarth Updated",
  "email": "sam@example.com",
  "details": {
    "department": "CS",
    "experience": 2
  }
}
```


localhost:5000/employees

Pretty print ✓

```
[
  {
    "id": 3,
    "name": "Tanya",
    "email": "tanya@example.com",
    "details": {
      "department": "MMS",
      "experience": 3
    }
  },
  {
    "id": 2,
    "name": "Samarth Updated",
    "email": "sam@example.com",
    "details": {
      "department": "CS",
      "experience": 2
    }
  },
  {
    "id": 1,
    "name": "Ashvini",
    "email": "ashvini@example.com",
    "details": {
      "department": "IT",
      "experience": 1
    }
  }
]
```

- **DELETE /employees/:id → Delete an employee:**
-

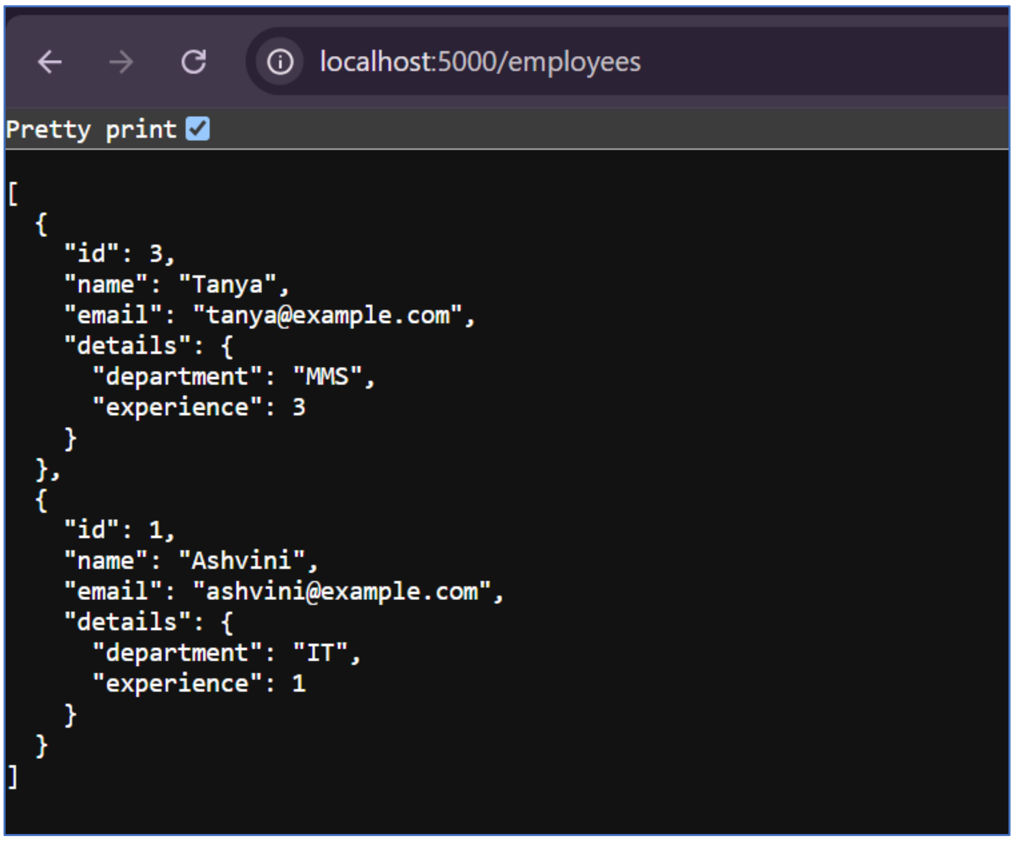


DEL localhost:5000/employees
just now

DELETE http://localhost:5000/employees/2

Status: 200 OK Size: 16 Bytes Time: 98 ms

```
Employee deleted
```

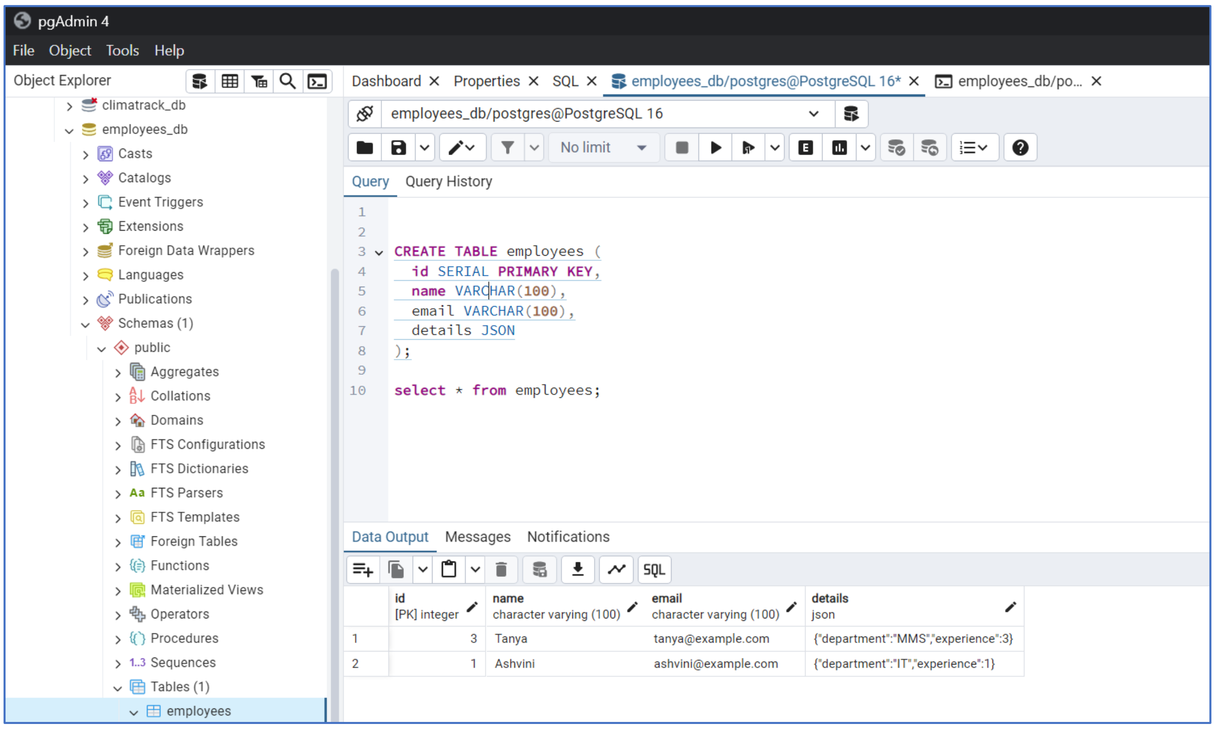


localhost:5000/employees

Pretty print ✓

```
[
  {
    "id": 3,
    "name": "Tanya",
    "email": "tanya@example.com",
    "details": {
      "department": "MMS",
      "experience": 3
    }
  },
  {
    "id": 1,
    "name": "Ashvini",
    "email": "ashvini@example.com",
    "details": {
      "department": "IT",
      "experience": 1
    }
  }
]
```

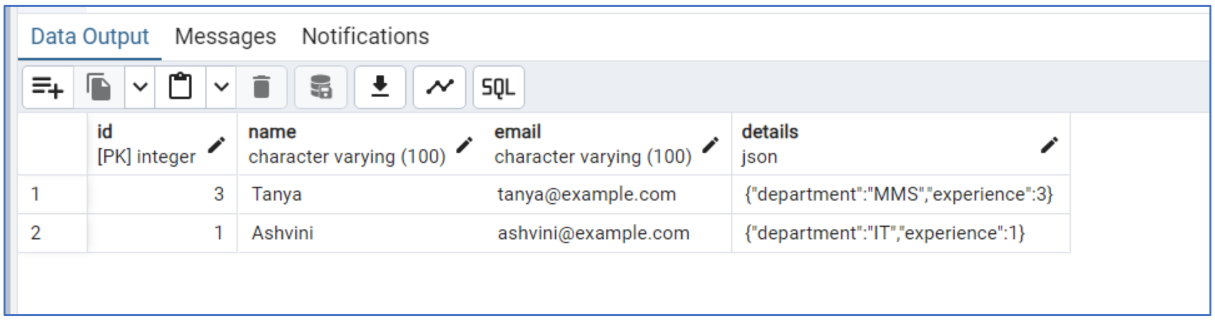
- **In PostgreSQL Final Data Left:**



```
CREATE TABLE employees (
  id SERIAL PRIMARY KEY,
  name VARCHAR(100),
  email VARCHAR(100),
  details JSON
);

select * from employees;
```

	id [PK] integer	name character varying (100)	email character varying (100)	details json
1	3	Tanya	tanya@example.com	{"department":"MMS","experience":3}
2	1	Ashvini	ashvini@example.com	{"department":"IT","experience":1}



	id [PK] integer	name character varying (100)	email character varying (100)	details json
1	3	Tanya	tanya@example.com	{"department":"MMS","experience":3}
2	1	Ashvini	ashvini@example.com	{"department":"IT","experience":1}