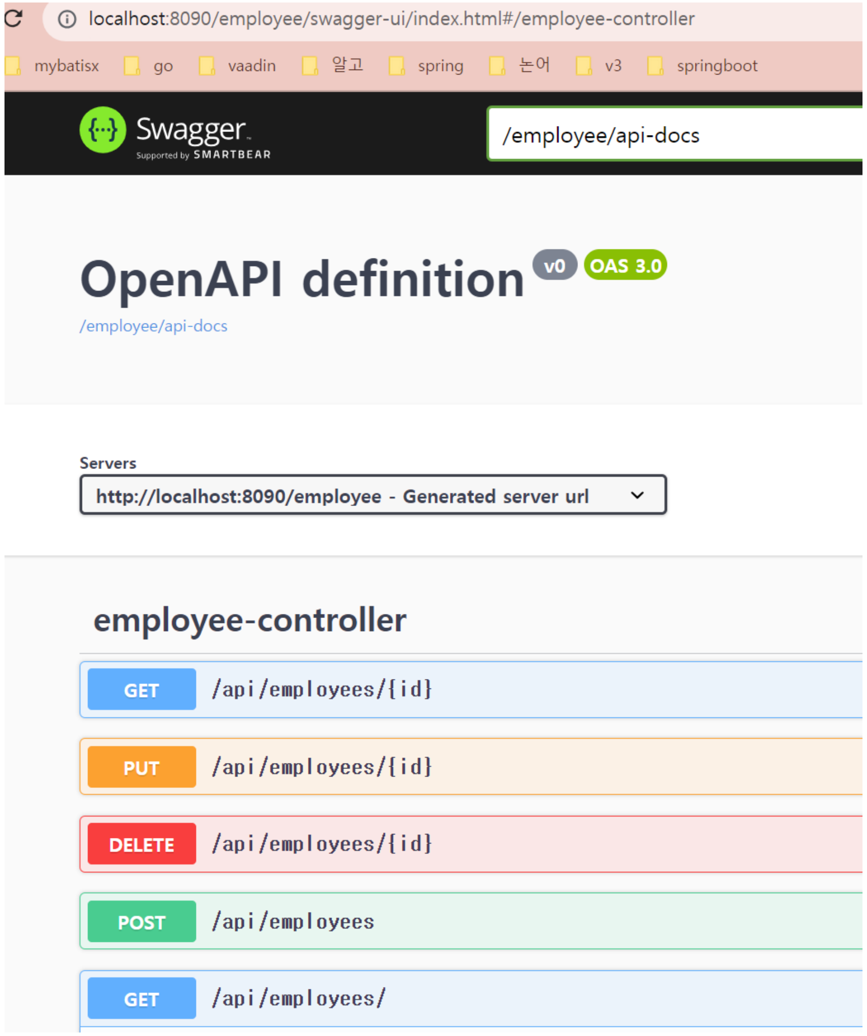
EmployeeRestApi

EmployeeRestApi

- swagger

http://localhost:8090/employee/swagger-ui/index.html



localhost:8090/employee/swagger-ui/index.html#/employee-controller

/employee/api-docs

OpenAPI definition v0 OAS 3.0

/employee/api-docs

Servers

http://localhost:8090/employee - Generated server url

employee-controller

```
GET     /api/employees/{id}
PUT     /api/employees/{id}
DELETE  /api/employees/{id}
POST    /api/employees
GET     /api/employees/
```

http://localhost:8090/employee/api/employees/

```
[
  {
    "id": 1,
    "name": "hyo",
    "phone": "010-1234-1234",
    "email": "hyo@naver.com",
    "position": "it",
    "bio": "java fluent, db fluent"
  },
  {
    "id": 2,
    "name": "kang",
    "phone": "010-1234-1235",
    "email": "kang@naver.com",
    "position": "it",
    "bio": "java fluent, db fluent, math fluent"
  }
]
```

EmployeeRestApi

- swagger

Methods	Urls	Actions
POST	/api/tutorials	create new Tutorial
GET	/api/tutorials	retrieve all Tutorials
GET	/api/tutorials/:id	retrieve a Tutorial by :id
PUT	/api/tutorials/:id	update a Tutorial by :id
DELETE	/api/tutorials/:id	delete a Tutorial by :id
DELETE	/api/tutorials	delete all Tutorials
GET	/api/tutorials/published	find all published Tutorials
GET	/api/tutorials?title=[keyword]	find all Tutorials which title contain s keyword

EmployeeRestApi

- Hibernate

```
<dependencies>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>
<dependency>
<groupId>com.mysql</groupId>
<artifactId>mysql-connector-j</artifactId>
<scope>runtime</scope>
</dependency>
<dependency>
<groupId>org.projectlombok</groupId>
<artifactId>lombok</artifactId>
<optional>true</optional>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
</dependency>
</dependencies>
```

EmployeeRestApi

Hibernate

```
server.servlet.context-path=/employee
server.port=8090

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/jwtspring?useUnicode=true&serverTimezone=Asia/Seoul&char
acterEncoding=utf8
spring.datasource.username=ssafy
spring.datasource.password=ssafy

spring.jpa.database=mysql
spring.jpa.properties.hibernate.format_sql=true
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.MySQLDialect
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql: true
spring.jpa.hibernate.naming.strategy=org.hibernate.cfg.ImprovedNamingStrategy
spring.jpa.hibernate.naming.physical-
strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl

logging.level.com.zaxxer.hikari.HikariConfig=DEBUG
logging.level.com.zaxxer.hikari=TRACE

spring.main.allow-circular-references=true
```

EmployeeRestApi

- Hibernate

```
package com.howtodoinjava.app.entity;

import jakarta.persistence.*;

@Entity
public class Employee {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
private String name;
private String phone;
private String email;
private String position;

@Column(length = 1000)
private String bio;
```

EmployeeRestApi

- Hibernate

```
package com.howtodoinjava.app.repository;
import com.howtodoinjava.app.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

@Repository
public interface EmployeeRepository extends JpaRepository<Employee, Long> {

}
```

EmployeeRestApi

- Hibernate

```
package com.howtodoinjava.app.service;

import com.howtodoinjava.app.entity.Employee;
import com.howtodoinjava.app.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

public interface EmployeeService {

  List<Employee> getAllEmployees() ;

  Optional<Employee> getEmployeeById(Long id) ;

  Employee addEmployee(Employee employee) ;

  Optional<Employee> updateEmployee(Long id, Employee employee) ;

  void deleteEmployee(Long id) ;
}
```

EmployeeRestApi

- Hibernate

```
package com.howtodoinjava.app.service;

import com.howtodoinjava.app.entity.Employee;
import com.howtodoinjava.app.repository.EmployeeRepository;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import java.util.List;
import java.util.Optional;

@Service
public class EmployeeServiceImpl implements EmployeeService{

  @Autowired
  private EmployeeRepository employeeRepository;

  public List<Employee> getAllEmployees() {
    return employeeRepository.findAll();
  }
```

EmployeeRestApi

- Hibernate

```
 public Optional<Employee> getEmployeeById(Long id) {
   return employeeRepository.findById(id);
 }
 public Employee addEmployee(Employee employee) {
   return employeeRepository.save(employee);
 }
 public Optional<Employee> updateEmployee(Long id, Employee employee) {
   Optional<Employee> existingEmployee = employeeRepository.findById(id);
   if (existingEmployee.isPresent()) {
     Employee updatedEmployee = existingEmployee.get();
     updatedEmployee.setName(employee.getName());
     updatedEmployee.setPhone(employee.getPhone());
     updatedEmployee.setEmail(employee.getEmail());
     updatedEmployee.setPosition(employee.getPosition());
     updatedEmployee.setBio(employee.getBio());
     employeeRepository.save(updatedEmployee);
   }
   return existingEmployee;
 }
 public void deleteEmployee(Long id) {
   employeeRepository.deleteById(id);
 }
}
```

EmployeeRestApi

- Hibernate

```
package com.howtodoinjava.app.web;

import com.howtodoinjava.app.entity.Employee;
import com.howtodoinjava.app.service.EmployeeService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import java.net.URI;
import java.util.List;
import java.util.Optional;
//http://localhost:8090/employee/api/users
@RestController
@RequestMapping("/api/employees")
@CrossOrigin
public class EmployeeController {
  @Autowired
  private EmployeeService employeeService;
  @GetMapping("/")
  public List<Employee> getAllEmployees() {
    return employeeService.getAllEmployees();
  }
```

EmployeeRestApi

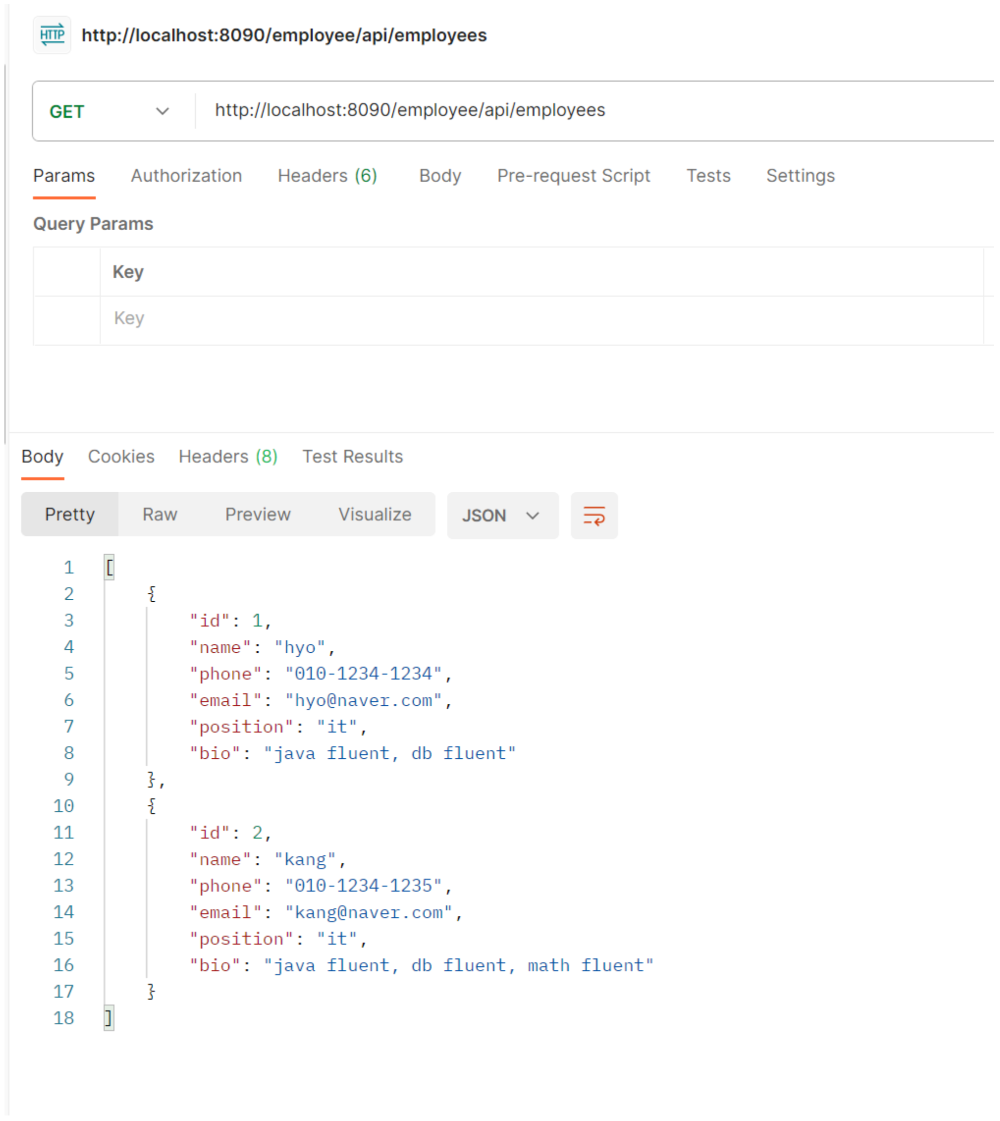
- Hibernate

```
@GetMapping("/{id}")
public ResponseEntity<Employee> getEmployeeById(@PathVariable Long id) {
  Optional<Employee> employee = employeeService.getEmployeeById(id);
  return employee.map(ResponseEntity::ok).orElseGet(() -> ResponseEntity.notFound().build());
}

@PostMapping
public ResponseEntity<Employee> addEmployee(@RequestBody Employee employee) {
  Employee savedEmployee = employeeService.addEmployee(employee);
  return ResponseEntity.created(URI.create("/api/employees/" + savedEmployee.getId())).body(savedEmployee);
}

@PutMapping("/{id}")
public ResponseEntity<Employee> updateEmployee(@PathVariable Long id,
                                                @RequestBody Employee employee) {
  Optional<Employee> existingEmployee = employeeService.updateEmployee(id,
      employee);
  return existingEmployee.map(ResponseEntity::ok).orElseGet(() -> ResponseEntity.notFound().build());
}

@DeleteMapping("/{id}")
public ResponseEntity<Void> deleteEmployee(@PathVariable Long id) {
  employeeService.deleteEmployee(id);
  return ResponseEntity.noContent().build();
}
}
```

EmployeeRestApi

- Hibernate

http://localhost:8090/employee/api/employees

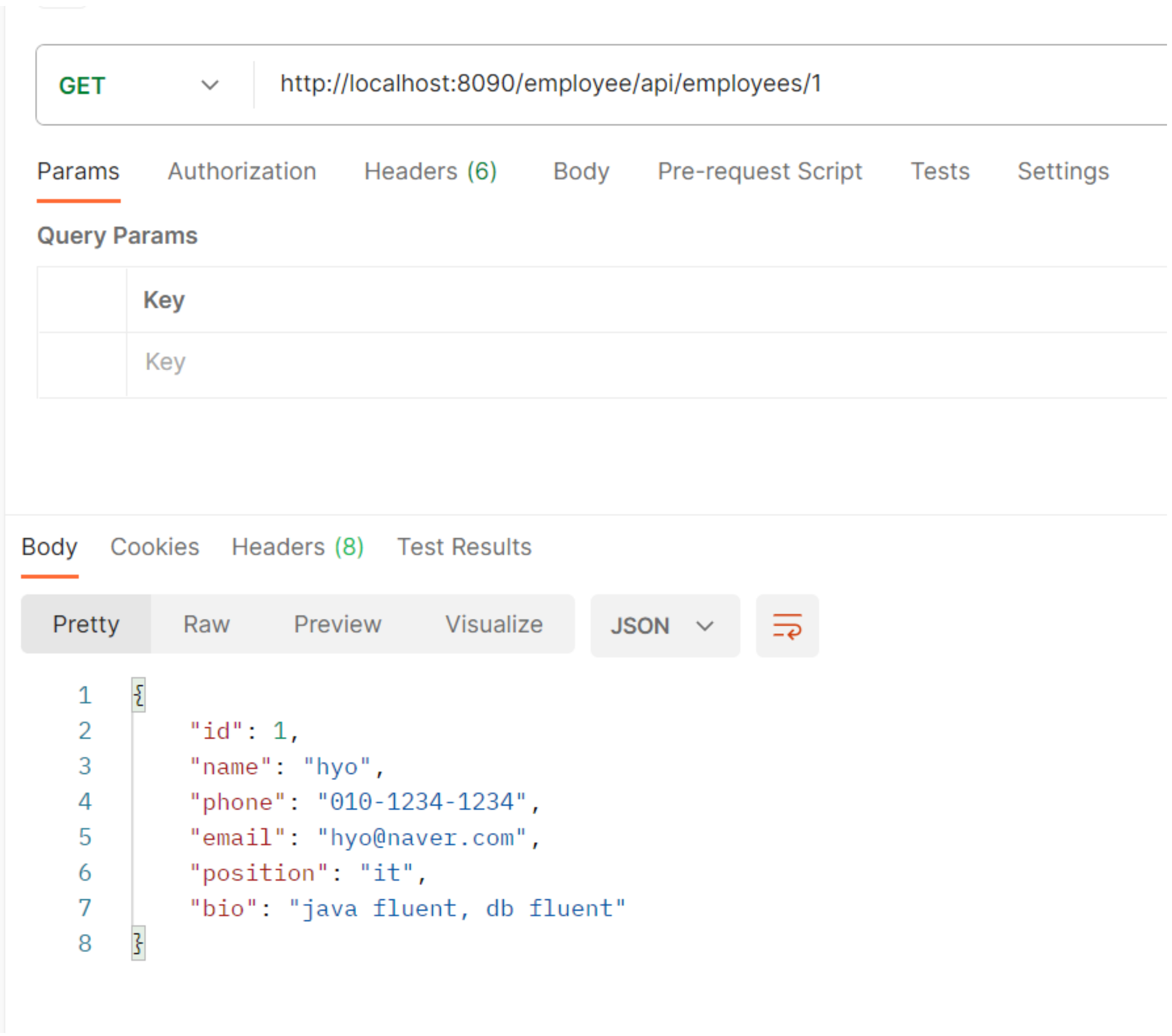


```
[
    {
        "id": 1,
        "name": "hyo",
        "phone": "010-1234-1234",
        "email": "hyo@naver.com",
        "position": "it",
        "bio": "java fluent, db fluent"
    },
    {
        "id": 2,
        "name": "kang",
        "phone": "010-1234-1235",
        "email": "kang@naver.com",
        "position": "it",
        "bio": "java fluent, db fluent, math fluent"
    }
]
```

EmployeeRestApi

- Hibernate

http://localhost:8090/employee/api/employees/1



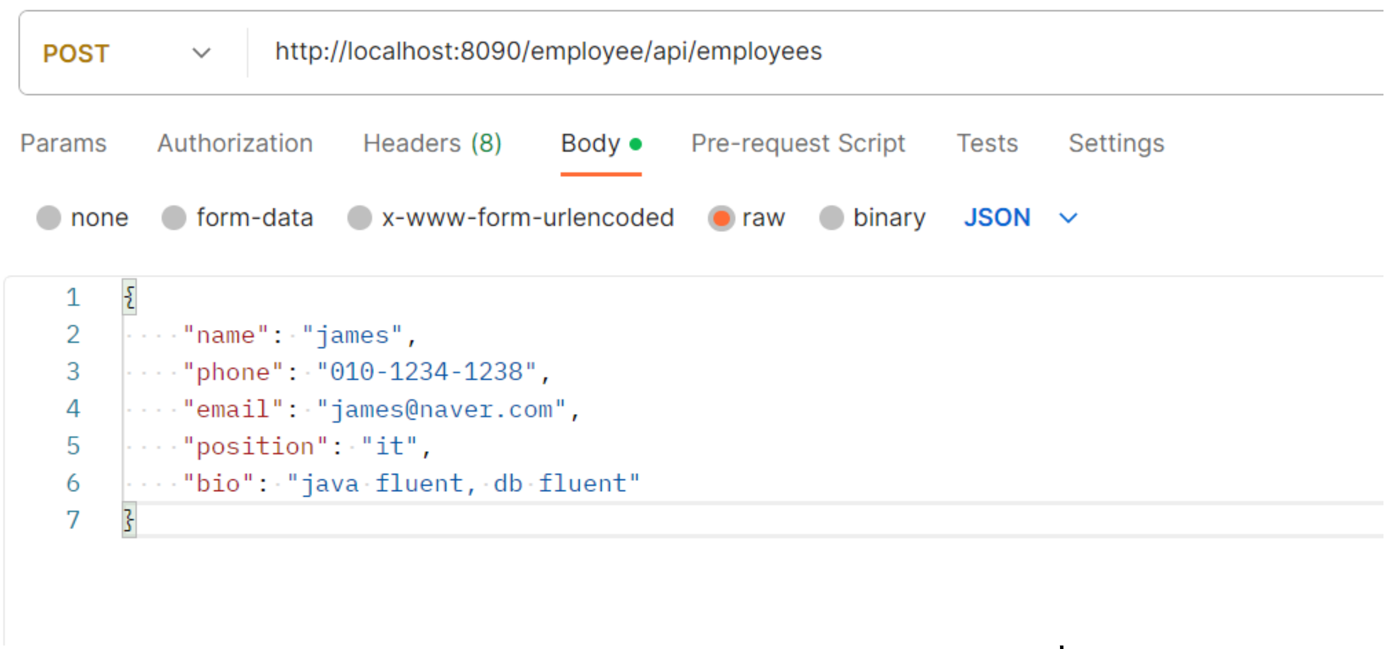
http://localhost:8090/employee/api/employees/1

```
{
    "id": 1,
    "name": "hyo",
    "phone": "010-1234-1234",
    "email": "hyo@naver.com",
    "position": "it",
    "bio": "java fluent, db fluent"
}
```

EmployeeRestApi

- Hibernate

http://localhost:8090/employee/api/employees



http://localhost:8090/employee/api/employees

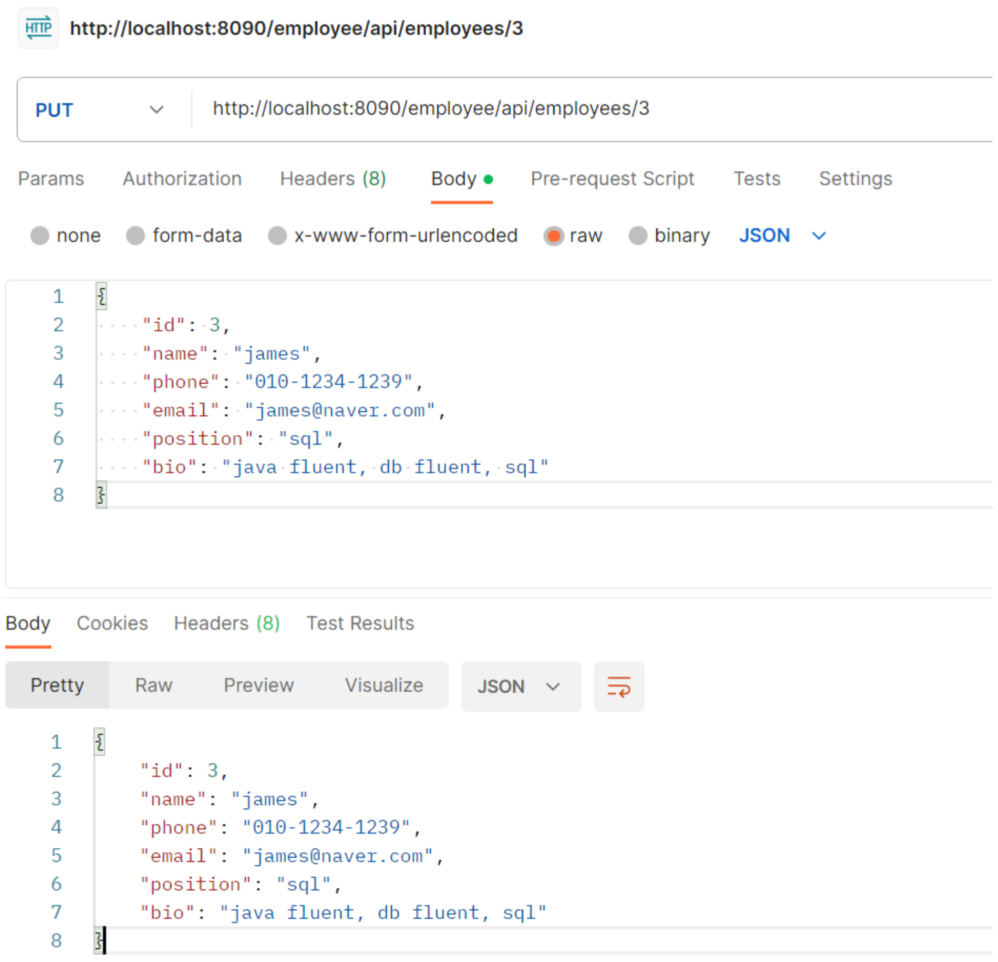
Body: [✓ raw] JSON

```
{
    "name": "james",
    "phone": "010-1234-1238",
    "email": "james@naver.com",
    "position": "it",
    "bio": "java fluent, db fluent"
}
```

```
{
    "name": "james",
    "phone": "010-1234-1238",
    "email": "james@naver.com",
    "position": "it",
    "bio": "java fluent, db fluent"
}
```

EmployeeRestApi

- Hibernate



http://localhost:8090/employee/api/employees/3

http://localhost:8090/employee/api/employees/3

```
{
    "id": 3,
    "name": "james",
    "phone": "010-1234-1239",
    "email": "james@naver.com",
    "position": "sql",
    "bio": "java fluent, db fluent, sql"
}
```

```
{
    "id": 3,
    "name": "james",
    "phone": "010-1234-1239",
    "email": "james@naver.com",
    "position": "sql",
    "bio": "java fluent, db fluent, sql"
}
```

```
{
    "id": 3,
    "name": "james",
    "phone": "010-1234-1239",
    "email": "james@naver.com",
    "position": "sql",
    "bio": "java fluent, db fluent, sql"
}
```

EmployeeRestApi

- properties

```
springdoc.api-docs.path=/api-docs
# http://localhost:8090/employee/swagger-ui/index.html
```

- pom

```
<dependency>
<groupId>org.springdoc</groupId>
<artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
<version>2.2.0</version>
</dependency>
```