# RestFul Web Service Part1

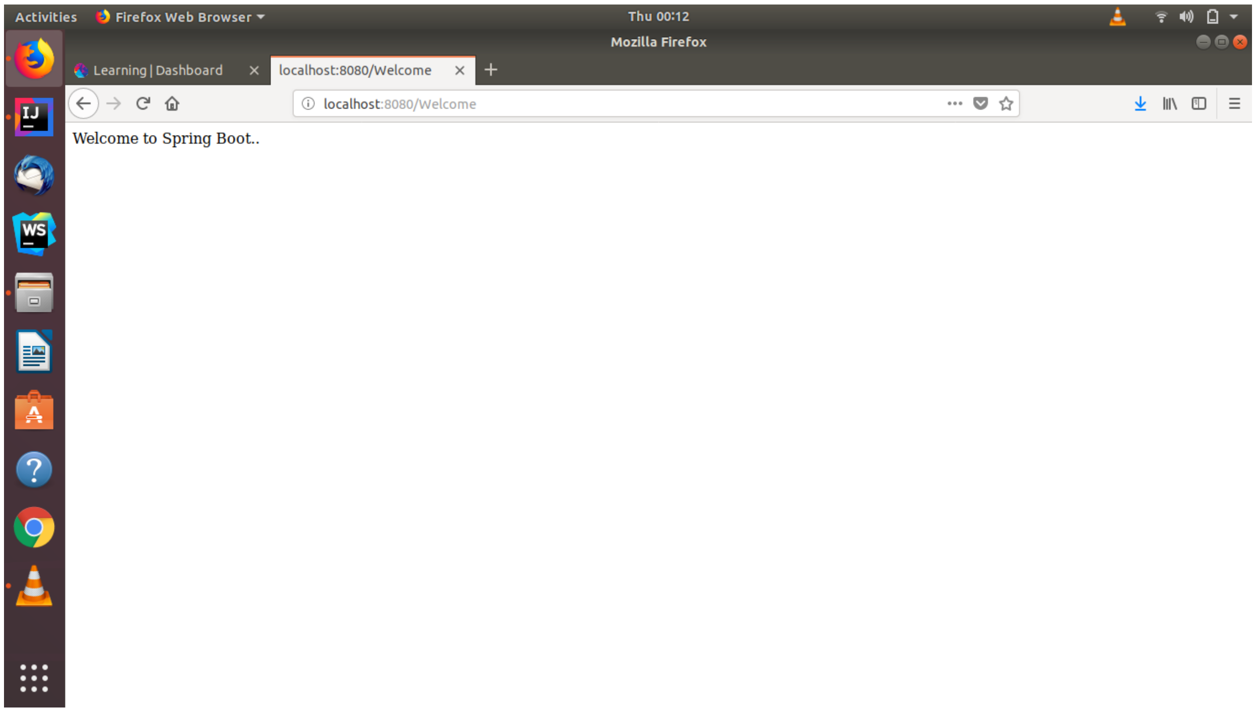
1. Create a simple RESTful service in Spring Boot which returns the Response "Welcome to spring boot".

**File=RestfulWebServiceApplication.java**

```
package com.example.RestfulWebService;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class RestfulWebServiceApplication {

  public static void main(String[] args) {
     SpringApplication.run(RestfulWebServiceApplication.class, args);
  }

}
```

**File=Welcome.java**

```
package com.example.RestfulWebService;

import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class Welcome {
   @GetMapping(path = "/Welcome")
   public String helloWorld(){
       return "Welcome to Spring Boot..";
   }

}
```



```
Welcome to Spring Boot..
```

------------------------------------------------------------------------------------------------------------------------------------------

2. Create an Employee Bean(id, name, age) and service to  perform different operations related to employee.

**File=EmployeBean**

```java
package com.example.RestfulWebService.Employee;

public class EmployeeBean {
  private Integer id;
  private String name;
  private int age;

  protected EmployeeBean(){

  }
```

```
    public EmployeeBean(int id, String name, int age) {
        this.id = id;
        this.name = name;
        this.age = age;
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    @Override
    public String toString() {
        return "EmployeeBean{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", age=" + age +
                '}';
    }
}
```

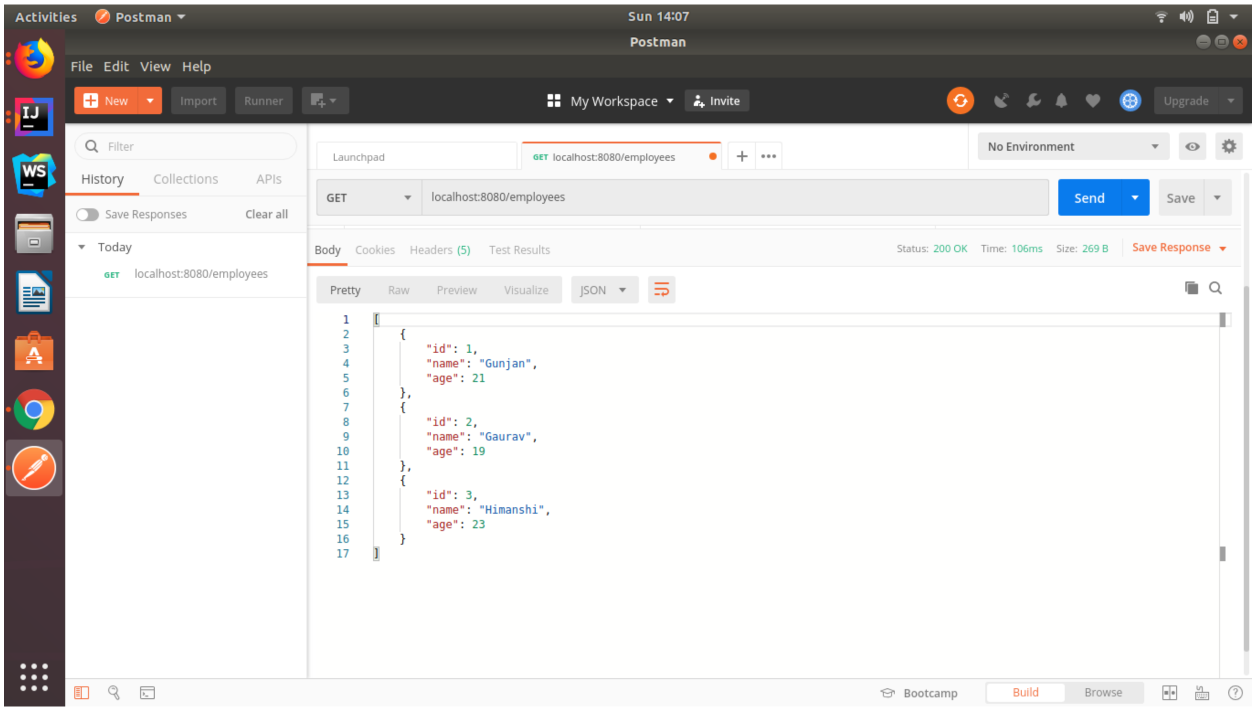
**File=EmployeDaoService**

```
package com.example.RestfulWebService.Employee;

import org.springframework.stereotype.Repository;

import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

@Repository
public class EmployeeDaoService {

  private static List<EmployeeBean> emp=new ArrayList<>();
  private static int empCount=3;

  static {
    emp.add(new EmployeeBean(1,"Gunjan",21));
    emp.add(new EmployeeBean(2, "Gaurav", 19));
    emp.add(new EmployeeBean(3, "Himanshi", 23));
  }

  //To return a list of employees back
  public List<EmployeeBean> findAll(){
    return emp;
  }

  //To add employee to the list
  public EmployeeBean save(EmployeeBean employeeBean){
    if(employeeBean.getId()==null){
      employeeBean.setId(++empCount);
    }
    emp.add(employeeBean);
    return employeeBean;
  }

  //To find a particular employee
  public EmployeeBean findOne(int id){
    for(EmployeeBean employeeBean:emp){
      if(employeeBean.getId()==id){
        return employeeBean;
      }
    }
    return null;
  }

  //To delete a particualer employee by id
  public EmployeeBean deleteOne(int id){
    Iterator<EmployeeBean> iterator=emp.iterator();
    while (iterator.hasNext()){
            EmployeeBean employeeBean=iterator.next();
            if(employeeBean.getId()==id){
                iterator.remove();
                return employeeBean;
            }
        }
        return null;
    }

}
```

3. Implement GET http request for Employee to get list of employees.

**File=EmployeeController.java**

```java
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
    return service.findAll();
  }

}
```



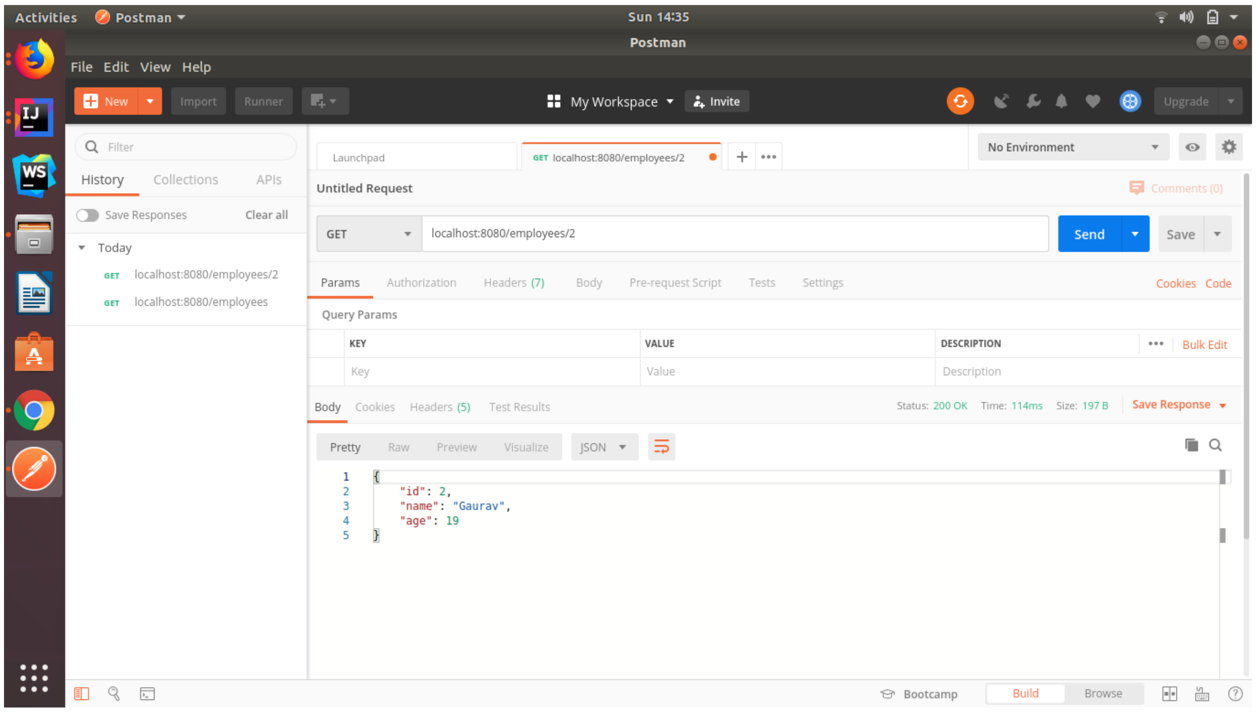
GET localhost:8080/employees

Status: 200 OK Time: 106ms Size: 269 B

```json
[
    {
        "id": 1,
        "name": "Gunjan",
        "age": 21
    },
    {
        "id": 2,
        "name": "Gaurav",
        "age": 19
    },
    {
        "id": 3,
        "name": "Himanshi",
        "age": 23
    }
]
```

4. Implement GET http request using path variable top get one employee

```java
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
      return service.findAll();
  }

  //Ques4 GET http request using path variable to get one employee
  @GetMapping("employees/{id}")
  public EmployeeBean retrieveUser(@PathVariable int id){
      EmployeeBean employeeBean=service.findOne(id);
      return employeeBean;

  }
}
```



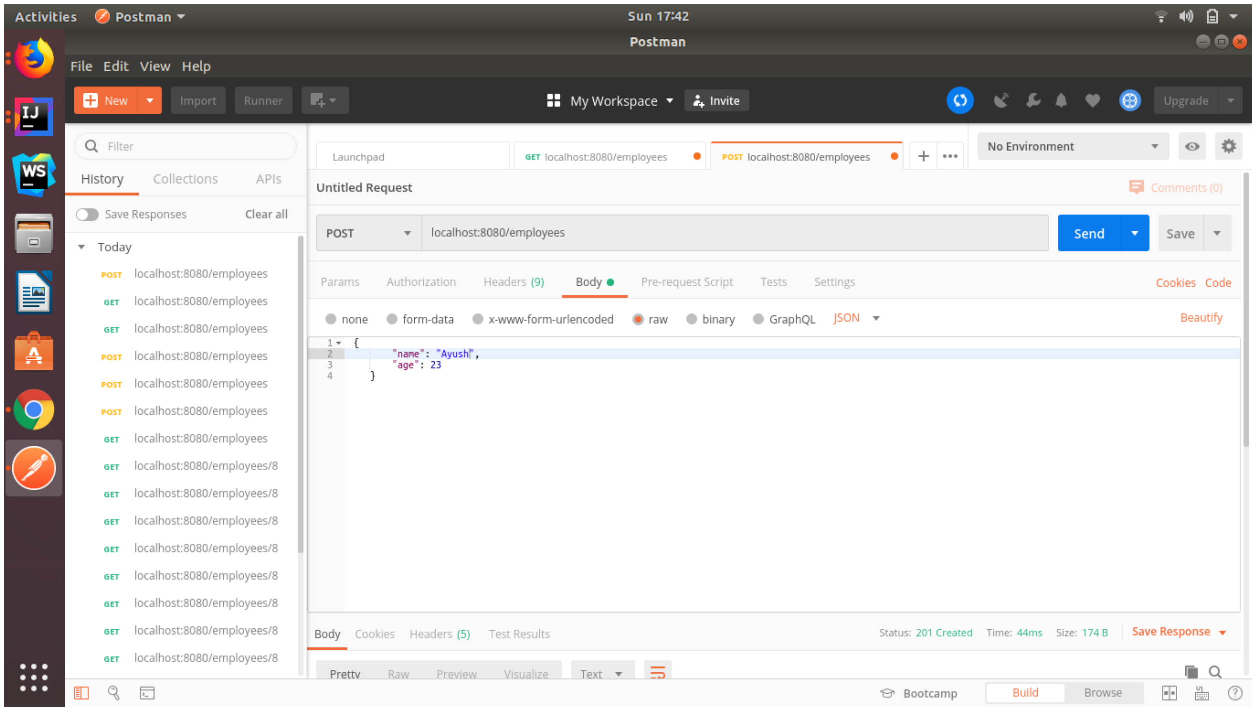
GET localhost:8080/employees/2

Status: 200 OK Time: 114ms Size: 197 B

```
{
    "id": 2,
    "name": "Gaurav",
    "age": 19
}
```

5. Implement POST http request for Employee to create a new employee.

```
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.method.annotation.ResponseEntityExceptionHandler;
import org.springframework.web.servlet.support.ServletUriComponentsBuilder;

import java.net.URI;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
      return service.findAll();
  }

  //Ques4 GET http request using path variable to get one employee
  @GetMapping("employees/{id}")
  public EmployeeBean retrieveUser(@PathVariable int id) {
      EmployeeBean employeeBean = service.findOne(id);
      //Ques6 - throwing exception for resource not found
      if (employeeBean == null)
          throw new EmpNotFoundException("id" + id);
      return employeeBean;

  }


  //Ques5 POST http request for Employee to create a new employee.
  @PostMapping("/employees")
  public ResponseEntity<Object> createEmployee(@RequestBody EmployeeBean employeeBean) {
      EmployeeBean addemp = service.save(employeeBean);
      URI location= ServletUriComponentsBuilder
              .fromCurrentRequest()
              .path("/id")
              .buildAndExpand(addemp.getId()).toUri();

          return ResponseEntity.created(location).build();
  }
}
```

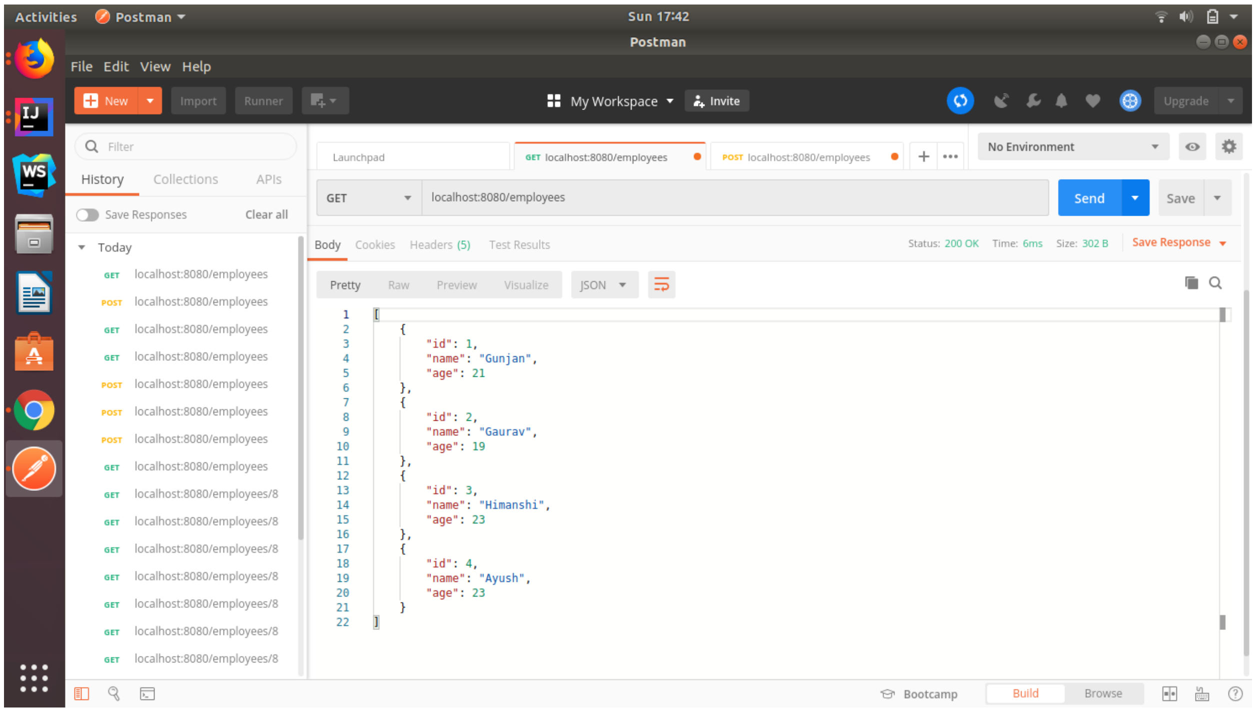


POST localhost:8080/employees

```
{
    "name": "Ayush",
    "age": 23
}
```

Status: 201 Created   Time: 44ms   Size: 174 B

| | |
|---|---|
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |



GET localhost:8080/employees

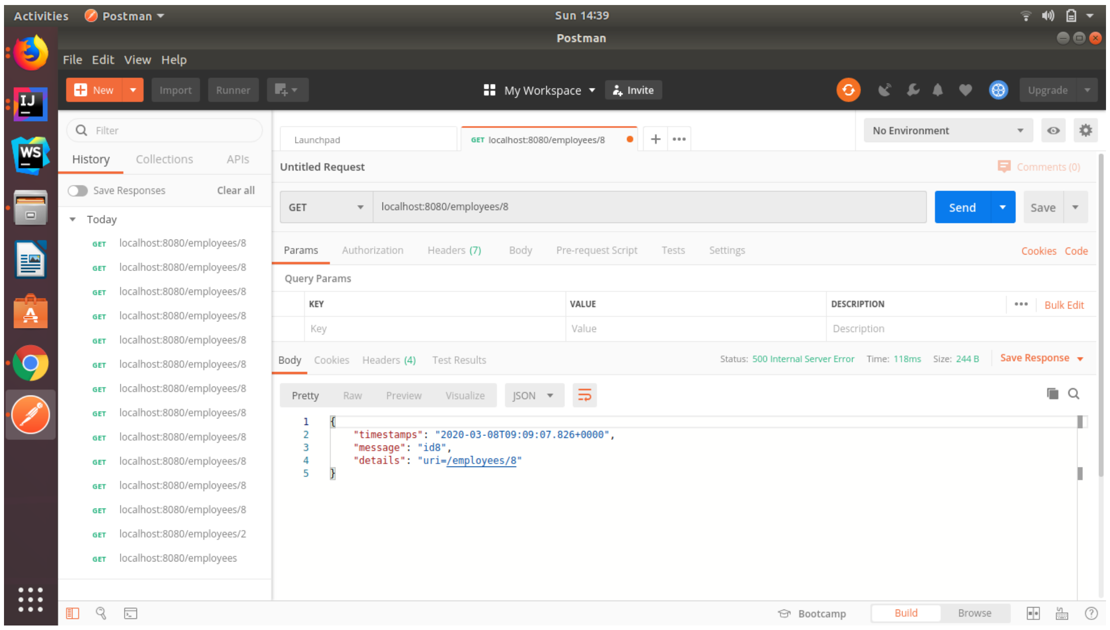
Status: 200 OK   Time: 6ms   Size: 302 B

```
[
    {
        "id": 1,
        "name": "Gunjan",
        "age": 21
    },
    {
        "id": 2,
        "name": "Gaurav",
        "age": 19
    },
    {
        "id": 3,
        "name": "Himanshi",
        "age": 23
    },
    {
        "id": 4,
        "name": "Ayush",
        "age": 23
    }
]
```

| | |
|---|---|
| GET | localhost:8080/employees |
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |
| GET | localhost:8080/employees/8 |

6. Implement Exception Handling for resource not found

```
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
      return service.findAll();
  }

  //Ques4 GET http request using path variable to get one employee
  @GetMapping("employees/{id}")
  public EmployeeBean retrieveUser(@PathVariable int id){
      EmployeeBean employeeBean=service.findOne(id);
      //Ques6 - throwing exception for resource not found
      if (employeeBean==null)
          throw new EmpNotFoundException("id"+id);
      return employeeBean;

  }
```



GET localhost:8080/employees/8

History:
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/8
GET localhost:8080/employees/2
GET localhost:8080/employees

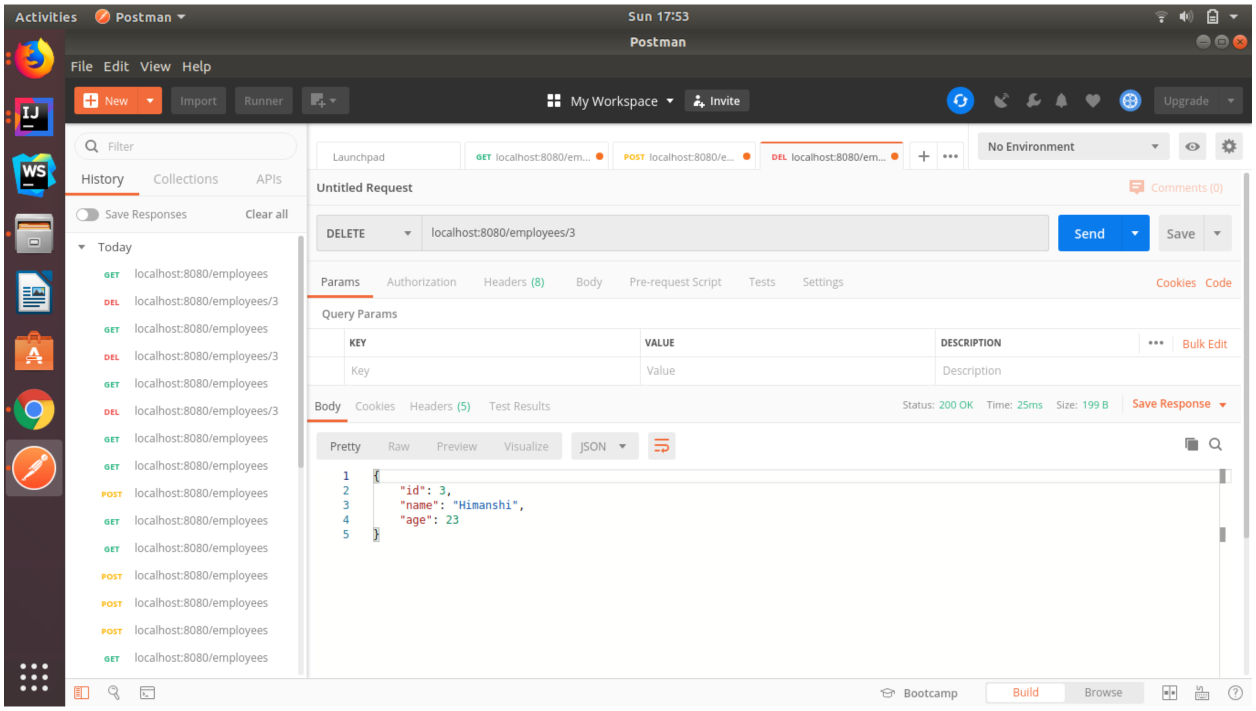
Status: 500 Internal Server Error Time: 118ms Size: 244 B

```
{
    "timestamps": "2020-03-08T09:09:07.826+0000",
    "message": "id8",
    "details": "uri=/employees/8"
}
```

}

7. Implement DELETE http request for Employee to delete employee

```
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.method.annotation.ResponseEntityExceptionHandler;
import org.springframework.web.servlet.support.ServletUriComponentsBuilder;

import java.net.URI;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
     return service.findAll();
  }

  //Ques4 GET http request using path variable to get one employee
  @GetMapping("employees/{id}")
  public EmployeeBean retrieveUser(@PathVariable int id) {
     EmployeeBean employeeBean = service.findOne(id);
     //Ques6 - throwing exception for resource not found
     if (employeeBean == null)
        throw new EmpNotFoundException("id" + id);
     return employeeBean;

  }


  //Ques5 POST http request for Employee to create a new employee.
  @PostMapping("/employees")
  public ResponseEntity<Object> createEmployee(@RequestBody EmployeeBean employeeBean) {
     EmployeeBean addemp = service.save(employeeBean);
     URI location= ServletUriComponentsBuilder
           .fromCurrentRequest()
           .path("/id")
           .buildAndExpand(addemp.getId()).toUri();

        return ResponseEntity.created(location).build();
  }

  //Ques7 DELETE http request for Employee to delete employee
  @DeleteMapping("/employees/{id}")
public EmployeeBean deleteEmployee(@PathVariable int id){
    EmployeeBean employeeBean=service.deleteOne(id);
    if (employeeBean==null)
        throw new EmpNotFoundException("id-"+id);
    return employeeBean;
  }
}
```



| | |
|---|---|
| GET | localhost:8080/employees |
| DEL | localhost:8080/employees/3 |
| GET | localhost:8080/employees |
| DEL | localhost:8080/employees/3 |
| GET | localhost:8080/employees |
| DEL | localhost:8080/employees/3 |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees |
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |
| GET | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| POST | localhost:8080/employees |
| GET | localhost:8080/employees |

DELETE localhost:8080/employees/3

| KEY | VALUE | DESCRIPTION |
|---|---|---|
| Key | Value | Description |

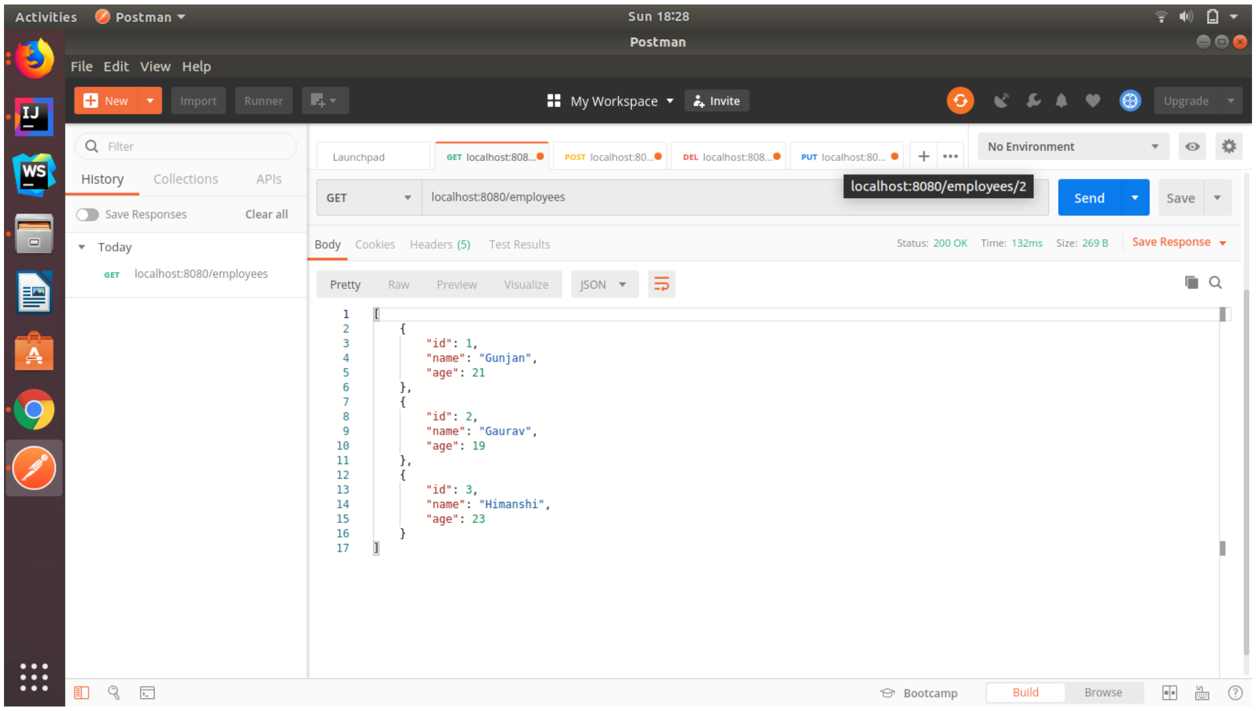
Status: 200 OK Time: 25ms Size: 199 B

```
{
    "id": 3,
    "name": "Himanshi",
    "age": 23
}
```

8. Implement PUT http request for Employee to update employee

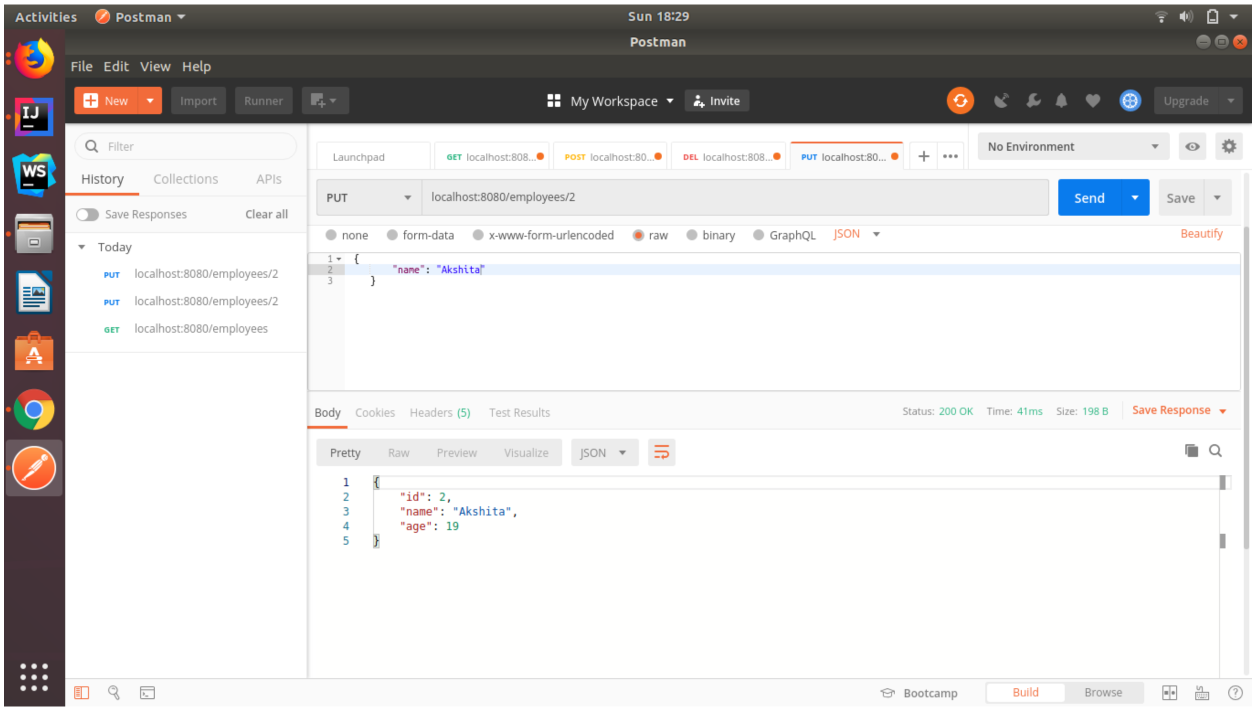
```
package com.example.RestfulWebService.Employee;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
import org.springframework.web.servlet.mvc.method.annotation.ResponseEntityExceptionHandler;
import org.springframework.web.servlet.support.ServletUriComponentsBuilder;

import java.net.URI;
import java.util.List;

@RestController
public class EmployeeController {
  @Autowired
  private EmployeeDaoService service;

  //Ques3 GET method to retrieve list of all employees
  @GetMapping("/employees")
  public List<EmployeeBean> retrieveAllEmployees() {
     return service.findAll();
  }

  //Ques4 GET http request using path variable to get one employee
  @GetMapping("employees/{id}")
  public EmployeeBean retrieveUser(@PathVariable int id) {
     EmployeeBean employeeBean = service.findOne(id);
     //Ques6 - throwing exception for resource not found
     if (employeeBean == null)
        throw new EmpNotFoundException("id" + id);
     return employeeBean;

  }


  //Ques5 POST http request for Employee to create a new employee.
  @PostMapping("/employees")
  public ResponseEntity<Object> createEmployee(@RequestBody EmployeeBean employeeBean) {
     EmployeeBean addemp = service.save(employeeBean);
     URI location = ServletUriComponentsBuilder
           .fromCurrentRequest()
           .path("/id")
           .buildAndExpand(addemp.getId()).toUri();

     return ResponseEntity.created(location).build();
  }

  //Ques7 DELETE http request for Employee to delete employee
  @DeleteMapping("/employees/{id}")
```

```
    public EmployeeBean deleteEmployee(@PathVariable int id) {
        EmployeeBean employeeBean = service.deleteOne(id);
        if (employeeBean == null)
            throw new EmpNotFoundException("id-" + id);
        return employeeBean;
    }

    //Ques8 PUT http request for Employee to update employee
    @PutMapping("/employees/{id}")
    public EmployeeBean updateEmployee(@PathVariable Integer id, @RequestBody EmployeeBean employee) {
        EmployeeBean employeeBean = service.findOne(id);
        if (employeeBean == null)
            throw new EmpNotFoundException("id-" + id);

        else {
            if (employee.getId() != null) {
                employeeBean.setId(employee.getId());
            }

            if (employee.getName() != null) {
                employeeBean.setName(employee.getName());
            }
            if (employee.getAge() != null) {
                employeeBean.setAge(employee.getAge());
            }
        }
        return employeeBean;
    }
}
```


GET localhost:8080/employees

Status: 200 OK Time: 132ms Size: 269 B

```
[
    {
        "id": 1,
        "name": "Gunjan",
        "age": 21
    },
    {
        "id": 2,
        "name": "Gaurav",
        "age": 19
    },
    {
        "id": 3,
        "name": "Himanshi",
        "age": 23
    }
]
```


PUT localhost:8080/employees/2

```
{
    "name": "Akshita"
}
```

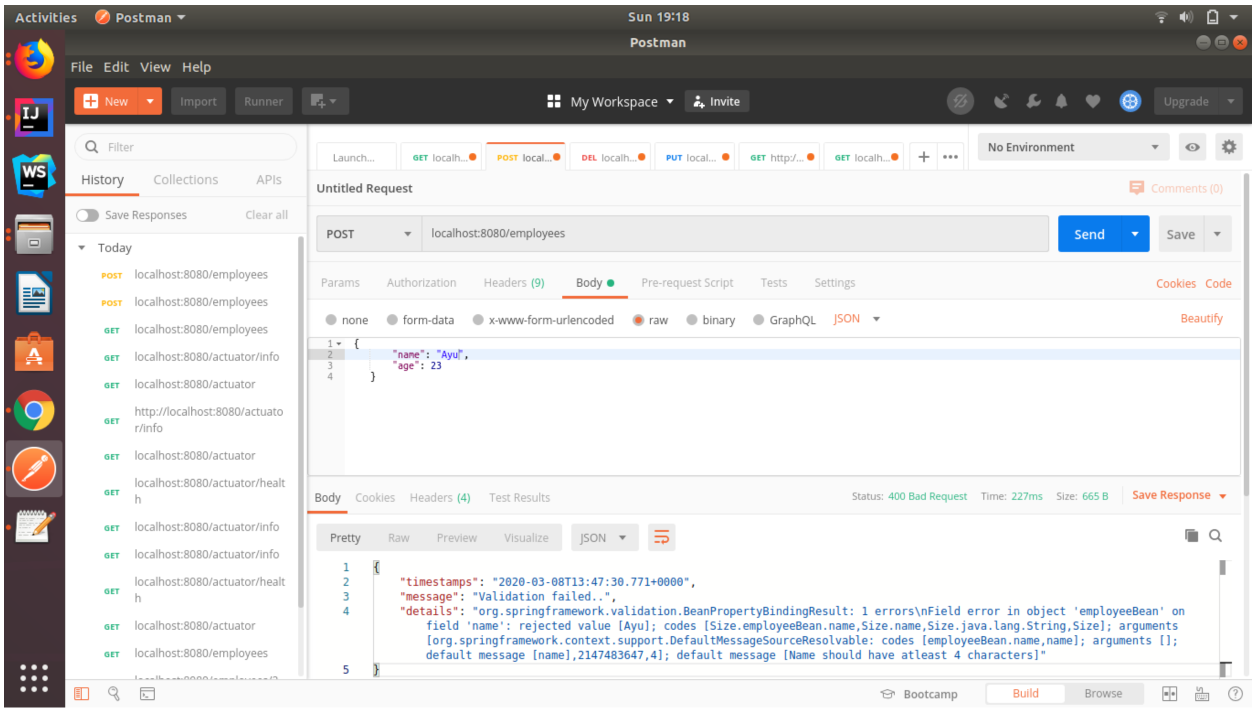
Status: 200 OK Time: 41ms Size: 198 B

```
{
    "id": 2,
    "name": "Akshita",
    "age": 19
}
```

9. Apply validation while create a new employee using POST http Request.

```
package com.example.RestfulWebService.Employee;

import org.hibernate.validator.constraints.Range;

import javax.validation.constraints.Size;

public class EmployeeBean {

   private Integer id;

   @Size(min = 4,message = "Name should have atleast 4 characters")
   private String name;

   @Range(max = 50,min = 18)
   private Integer age;

   protected EmployeeBean(){

   }
   public EmployeeBean(int id, String name, int age) {
       this.id = id;
       this.name = name;
       this.age = age;
   }

   public Integer getId() {
       return id;
   }

   public void setId(int id) {
       this.id = id;
   }

   public String getName() {
       return name;
   }

   public void setName(String name) {
       this.name = name;
   }

   public Integer getAge() {
       return age;
   }

   public void setAge(int age) {
       this.age = age;
   }
```

```java
    @Override
    public String toString() {
        return "EmployeeBean{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", age=" + age +
                '}';
    }
}
```



POST localhost:8080/employees

Request Body (raw, JSON):

```json
{
    "name": "Ayu",
    "age": 23
}
```

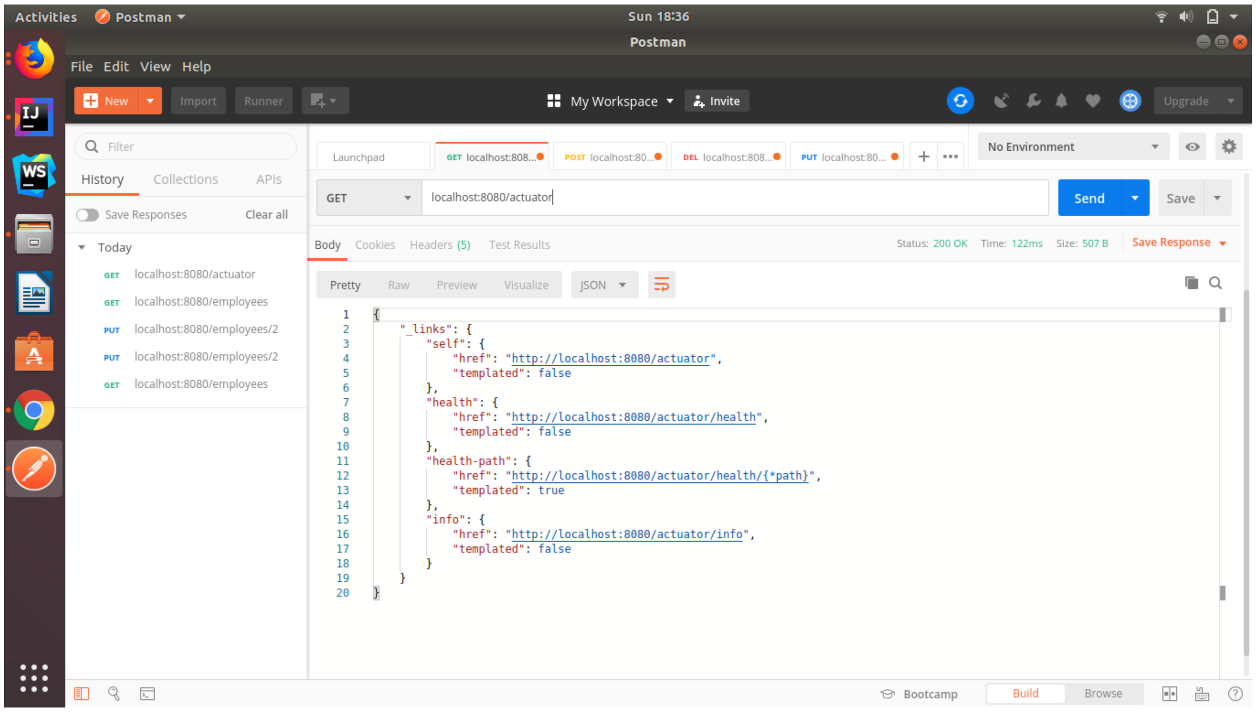
Status: 400 Bad Request   Time: 227ms   Size: 665 B

```json
{
    "timestamps": "2020-03-08T13:47:30.771+0000",
    "message": "Validation failed..",
    "details": "org.springframework.validation.BeanPropertyBindingResult: 1 errors\nField error in object 'employeeBean' on
        field 'name': rejected value [Ayu]; codes [Size.employeeBean.name,Size.name,Size.java.lang.String,Size]; arguments
        [org.springframework.context.support.DefaultMessageSourceResolvable: codes [employeeBean.name,name]; arguments [];
        default message [name],2147483647,4]; default message [Name should have atleast 4 characters]"
}
```

10. Configure actuator in your project to check the health of application and get the information about various beans configured in your application

***File=build.gradle***

```
plugins {
  id 'org.springframework.boot' version '2.2.5.RELEASE'
  id 'io.spring.dependency-management' version '1.0.9.RELEASE'
  id 'java'
}

group = 'com.example'
version = '0.0.1-SNAPSHOT'
sourceCompatibility = '1.8'

repositories {
  mavenCentral()
}

dependencies {

  implementation 'org.springframework.boot:spring-boot-starter-web'
  compile 'com.fasterxml.jackson.dataformat:jackson-dataformat-xml'
  //Ques10 Configure actuator in your project to check the health of application
  compile group: 'org.springframework.boot', name: 'spring-boot-starter-actuator'
  testImplementation('org.springframework.boot:spring-boot-starter-test') {
    exclude group: 'org.junit.vintage', module: 'junit-vintage-engine'
  }
}

test {
  useJUnitPlatform()
}
```



History

Save Responses — Clear all

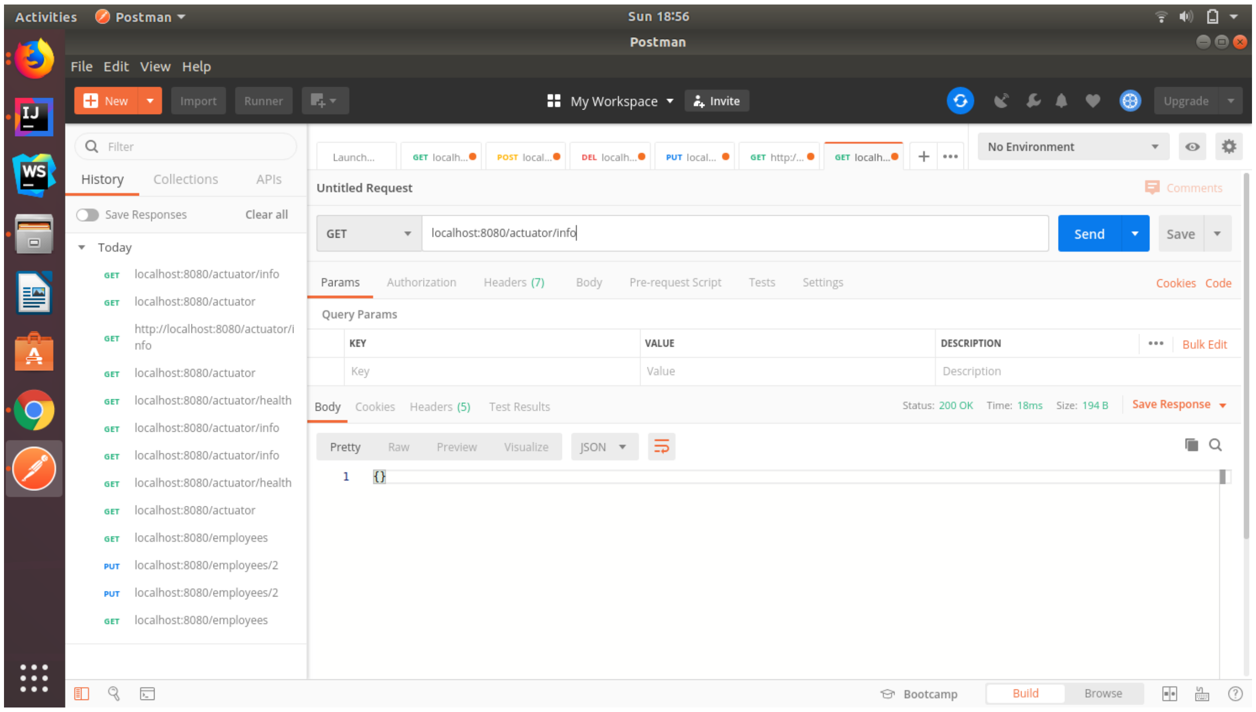
Today

GET localhost:8080/actuator
GET localhost:8080/employees
PUT localhost:8080/employees/2
PUT localhost:8080/employees/2
GET localhost:8080/employees

GET localhost:8080/actuator

Status: 200 OK Time: 122ms Size: 507 B

```
{
    "_links": {
        "self": {
            "href": "http://localhost:8080/actuator",
            "templated": false
        },
        "health": {
            "href": "http://localhost:8080/actuator/health",
            "templated": false
        },
        "health-path": {
            "href": "http://localhost:8080/actuator/health/{*path}",
            "templated": true
        },
        "info": {
            "href": "http://localhost:8080/actuator/info",
            "templated": false
        }
    }
}
```



History

Save Responses — Clear all

Today

GET localhost:8080/actuator/info
GET localhost:8080/actuator
GET http://localhost:8080/actuator/info
GET localhost:8080/actuator
GET localhost:8080/actuator/health
GET localhost:8080/actuator/info
GET localhost:8080/actuator/info
GET localhost:8080/actuator/health
GET localhost:8080/actuator
GET localhost:8080/employees
PUT localhost:8080/employees/2
PUT localhost:8080/employees/2
GET localhost:8080/employees

Untitled Request

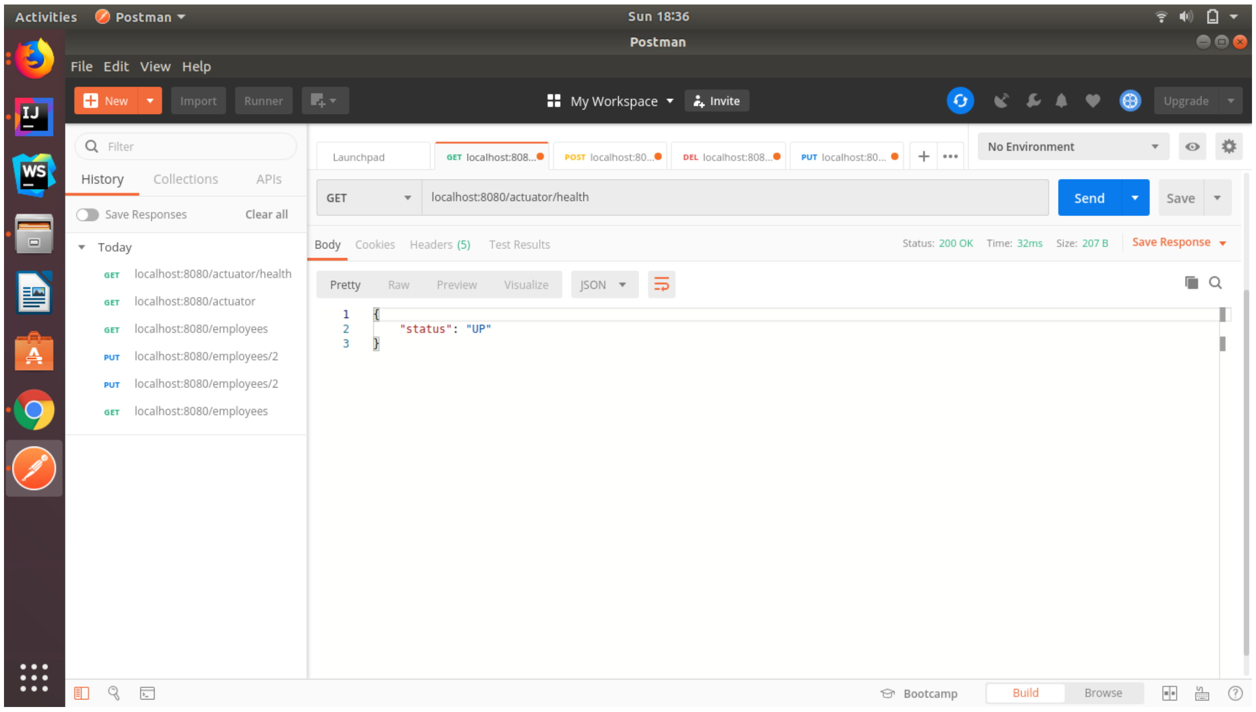
GET localhost:8080/actuator/info

Query Params

| KEY | VALUE | DESCRIPTION |
|---|---|---|
| Key | Value | Description |

Status: 200 OK Time: 18ms Size: 194 B

```
{}
```



History:

GET localhost:8080/actuator/health
GET localhost:8080/actuator
GET localhost:8080/employees
PUT localhost:8080/employees/2
PUT localhost:8080/employees/2
GET localhost:8080/employees

GET localhost:8080/actuator/health

Status: 200 OK Time: 32ms Size: 207 B

```
{
    "status": "UP"
}
```