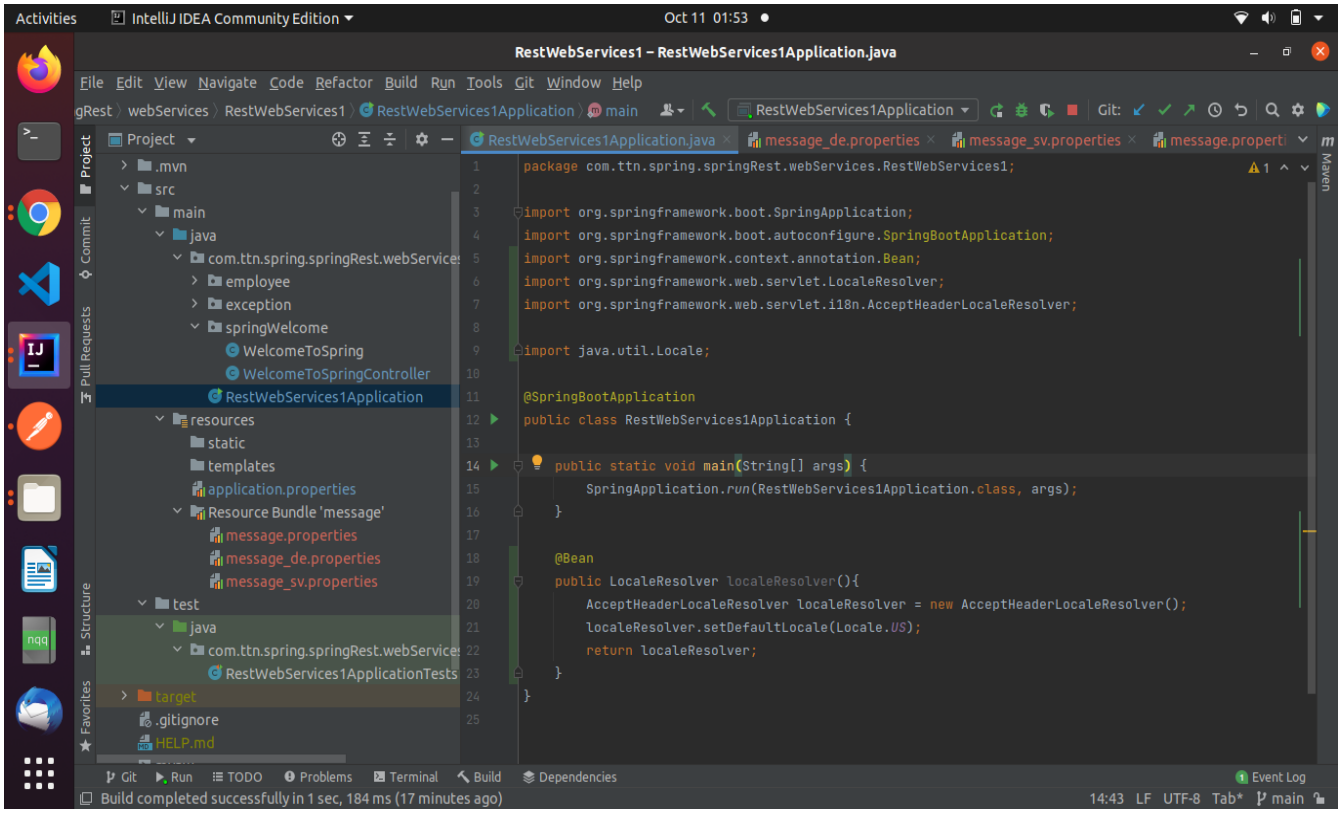
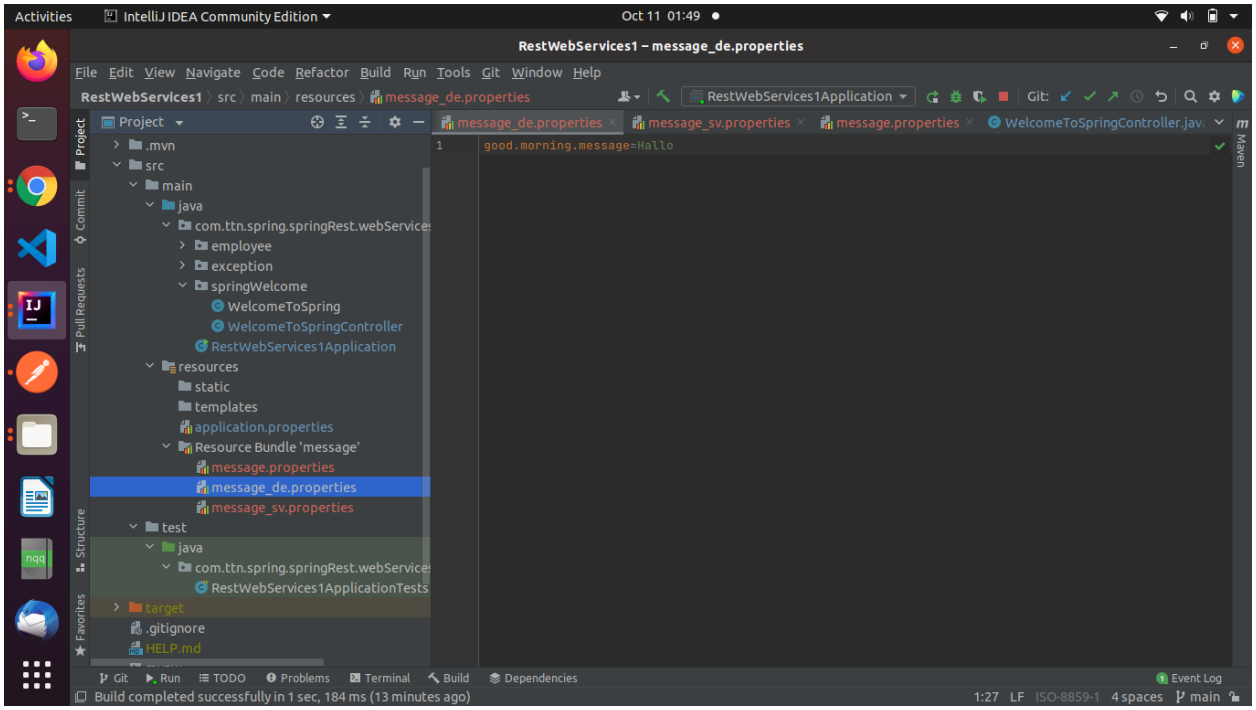
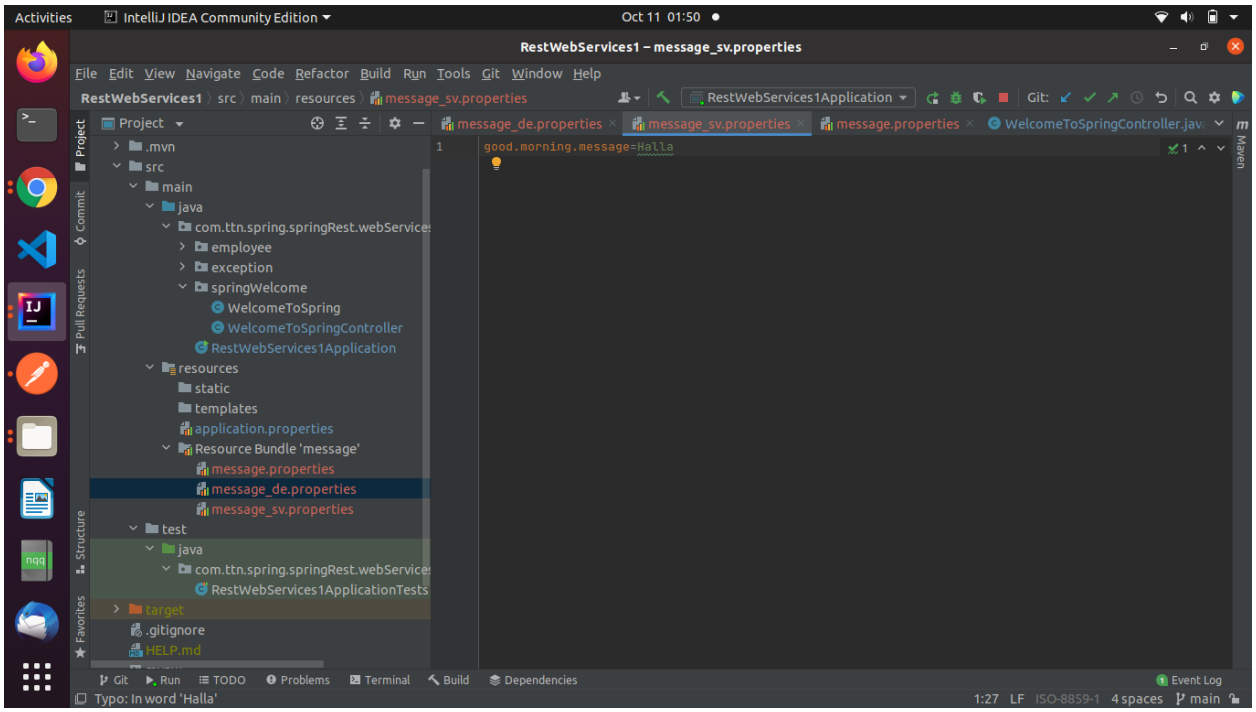
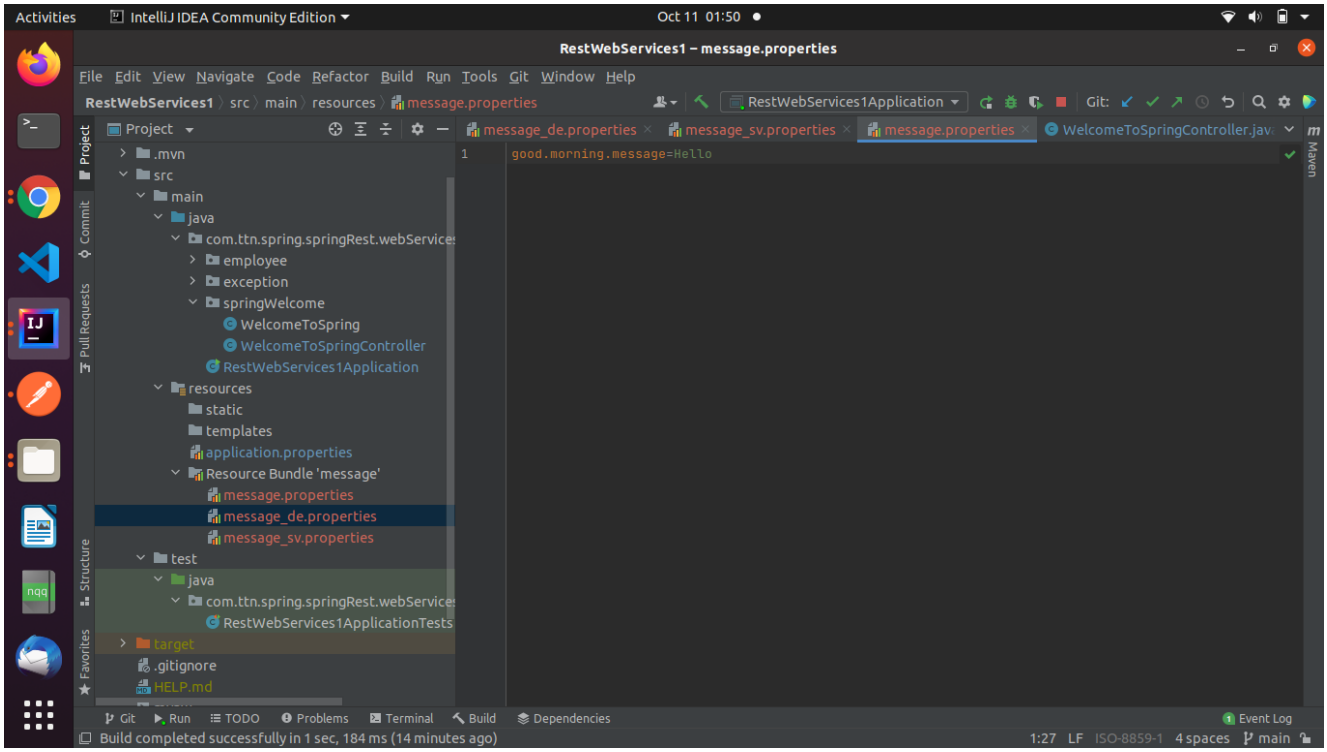
*Internationalization

1. **Add support for Internationalization in your application allowing messages to be shown in English, German and Swedish, keeping English as default.**




Activities
IntelliJ IDEA Community Edition
Oct 11 01:49
RestWebServices1 – message_de.properties
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RestWebServices1 › src › main › resources › message_de.properties
RestWebServices1Application
Project
.mvn
src
main
java
com.ttn.spring.springRest.webServices
employee
exception
springWelcome
WelcomeToSpring
WelcomeToSpringController
RestWebServices1Application
resources
static
templates
application.properties
Resource Bundle 'message'
message.properties
message_de.properties
message_sv.properties
test
java
com.ttn.spring.springRest.webServices
RestWebServices1ApplicationTests
target
.gitignore
HELP.md
message_de.properties
message_sv.properties
message.properties
WelcomeToSpringController.jav
good.morning.message=Hallo
Git Run TODO Problems Terminal Build Dependencies
Event Log
Build completed successfully in 1 sec, 184 ms (13 minutes ago)
1:27 LF ISO-8859-1 4 spaces main


Activities
IntelliJ IDEA Community Edition
Oct 11 01:50
RestWebServices1 – message_sv.properties
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
RestWebServices1 › src › main › resources › message_sv.properties
RestWebServices1Application
Project
.mvn
src
main
java
com.ttn.spring.springRest.webServices
employee
exception
springWelcome
WelcomeToSpring
WelcomeToSpringController
RestWebServices1Application
resources
static
templates
application.properties
Resource Bundle 'message'
message.properties
message_de.properties
message_sv.properties
test
java
com.ttn.spring.springRest.webServices
RestWebServices1ApplicationTests
target
.gitignore
HELP.md
message_de.properties
message_sv.properties
message.properties
WelcomeToSpringController.jav
good.morning.message=Halla
Git Run TODO Problems Terminal Build Dependencies
Event Log
Typo: In word 'Halla'
1:27 LF ISO-8859-1 4 spaces main



2. **Create a GET request which takes "username" as param and shows a localized message "Hello Username". (Use parameters in message properties)**

Activities IntelliJ IDEA Community Edition Oct 11 01:51

RestWebServices1 – WelcomeToSpringController.java

File Edit View Navigate Code Refactor Build Run Tools Git Window Help

webServices 〉 RestWebServices1 〉 springWelcome 〉 WelcomeToSpringController

RestWebServices1Application Git:

Project

- .mvn
- src
  - main
    - java
      - com.ttn.spring.springRest.webServices
        - employee
        - exception
        - springWelcome
          - WelcomeToSpring
          - WelcomeToSpringController
        - RestWebServices1Application
    - resources
      - static
      - templates
      - application.properties
      - Resource Bundle 'message'
        - message.properties
        - message_de.properties
        - message_sv.properties
  - test
    - java
      - com.ttn.spring.springRest.webServices
        - RestWebServices1ApplicationTests
- target
- .gitignore
- HELP.md

message_de.properties | message_sv.properties | message.properties | WelcomeToSpringController.java

```java
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class WelcomeToSpringController {

    @Autowired
    private MessageSource messageSource;

    @GetMapping("/welcome-spring")
    public String SpringWorld(){
        return "Welcome to spring boot";
    }

    @GetMapping("/welcome-spring-bean")
    public WelcomeToSpring springBootWorld(){
        return new WelcomeToSpring( msg: "Welcome to spring boot");
    }

    @GetMapping(path = "/hello-employee-itn")
    public String helloWorldInternationalized(@RequestParam String username){
        return messageSource.getMessage( code: "good.morning.message", args: null,
                LocaleContextHolder.getLocale()) + " " + username;
    }
}
```

Git Run TODO Problems Terminal Build Dependencies Event Log

Build completed successfully in 1 sec, 184 ms (15 minutes ago) 26:1 LF UTF-8 4 spaces main

http://localhost:8080/hello-employee-itn?username=vaishali Save

GET http://localhost:8080/hello-employee-itn?username=vaishali Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Headers 6 hidden

KEY	VALUE	DESCRIPTION
Accept-Language	de	
Key	Value	Description

Bulk Edit Presets

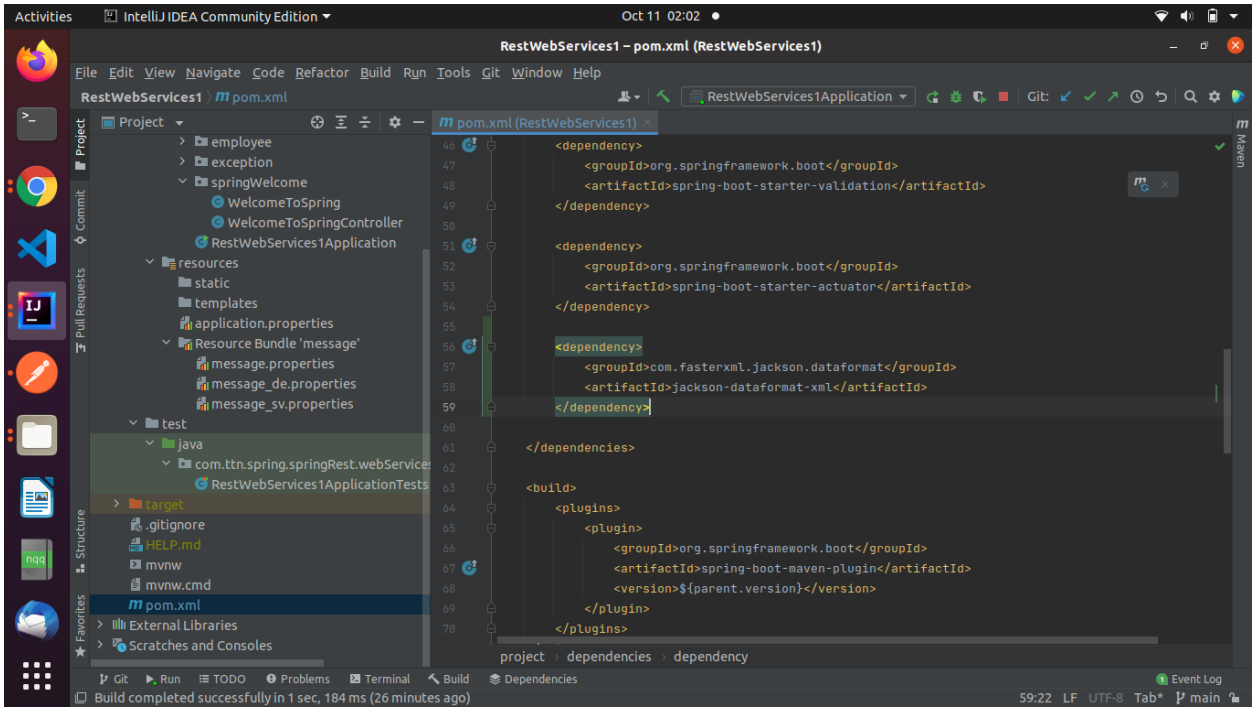
Body Cookies Headers (5) Test Results 200 OK 9 ms 178 B Save Response

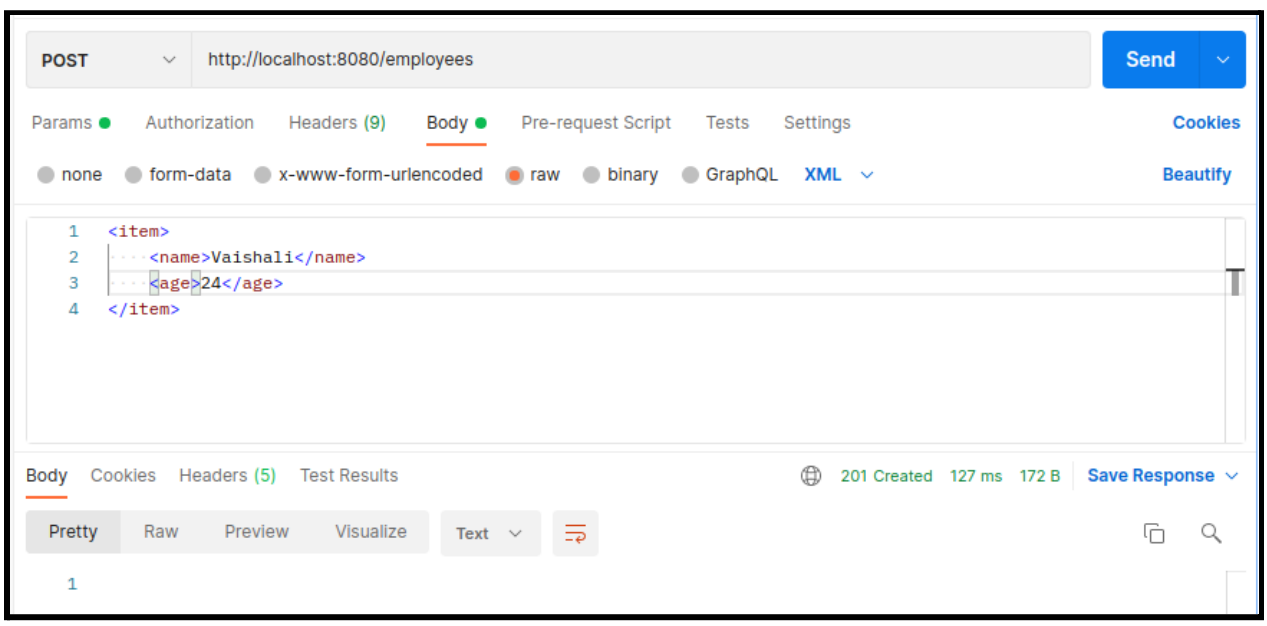
Pretty Raw Preview Visualize Text

```
1  Hallo vaishali
```

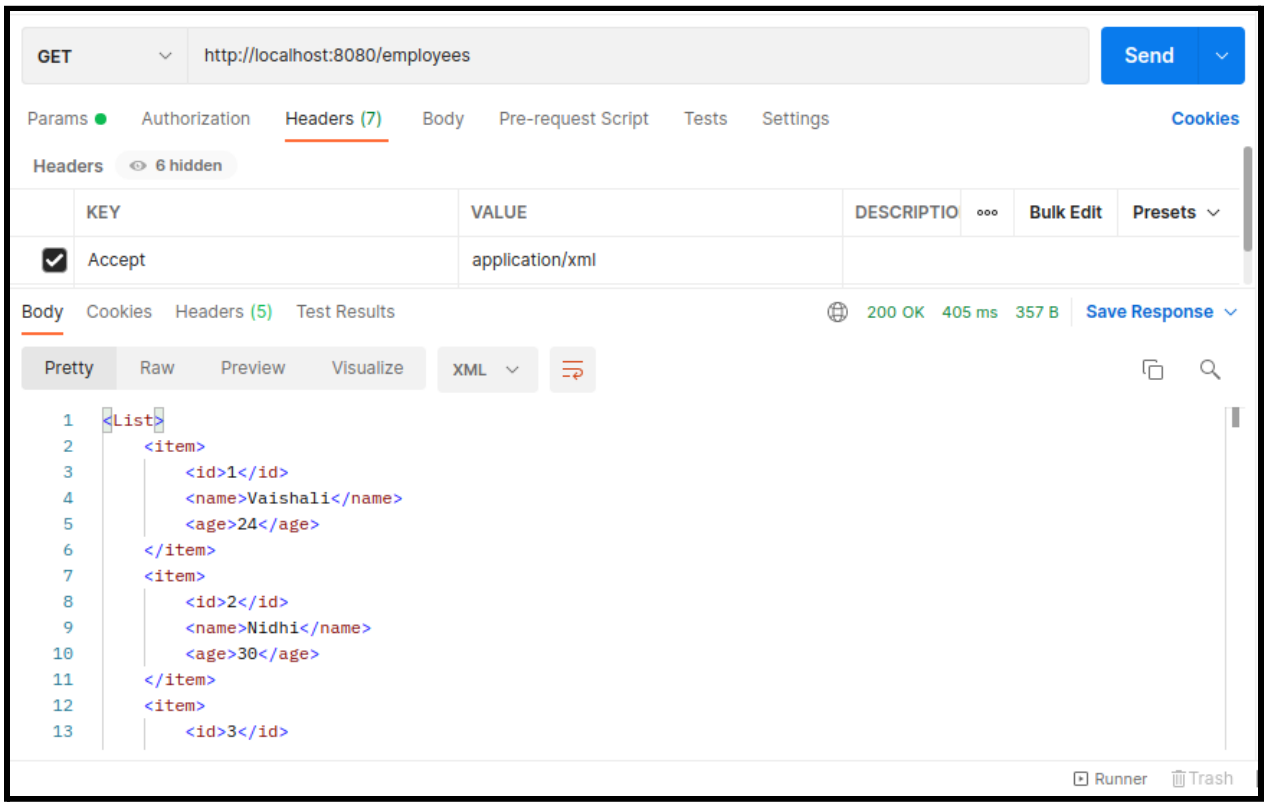
*Content Negotiation

3. Create POST Method to create user details which can accept XML for user creation.



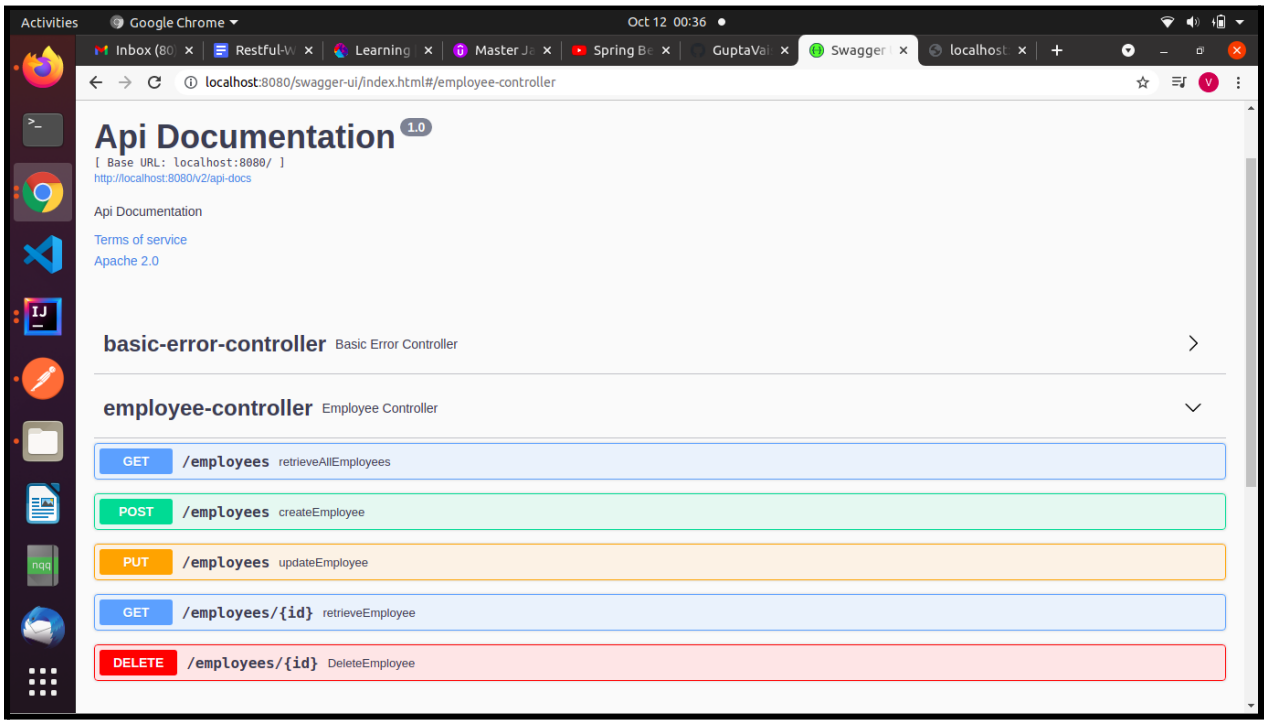


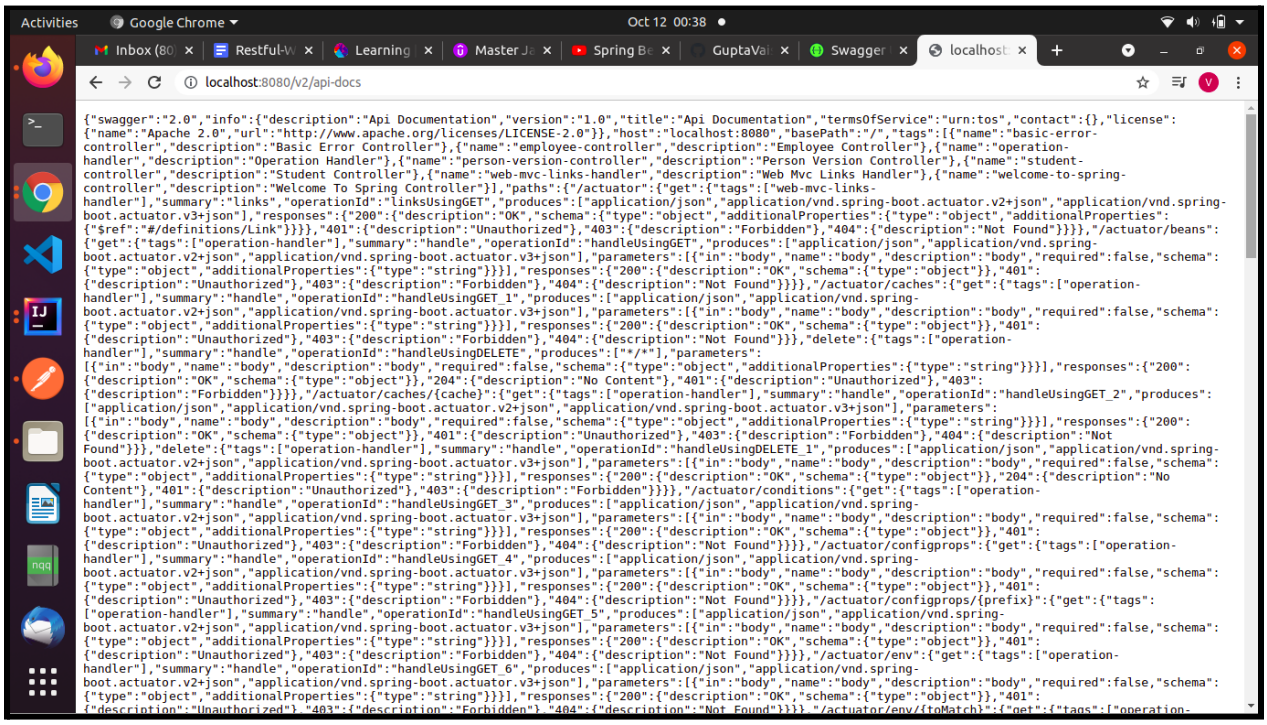
4. **Create GET Method to fetch the list of users in XML format.**



***Swagger**

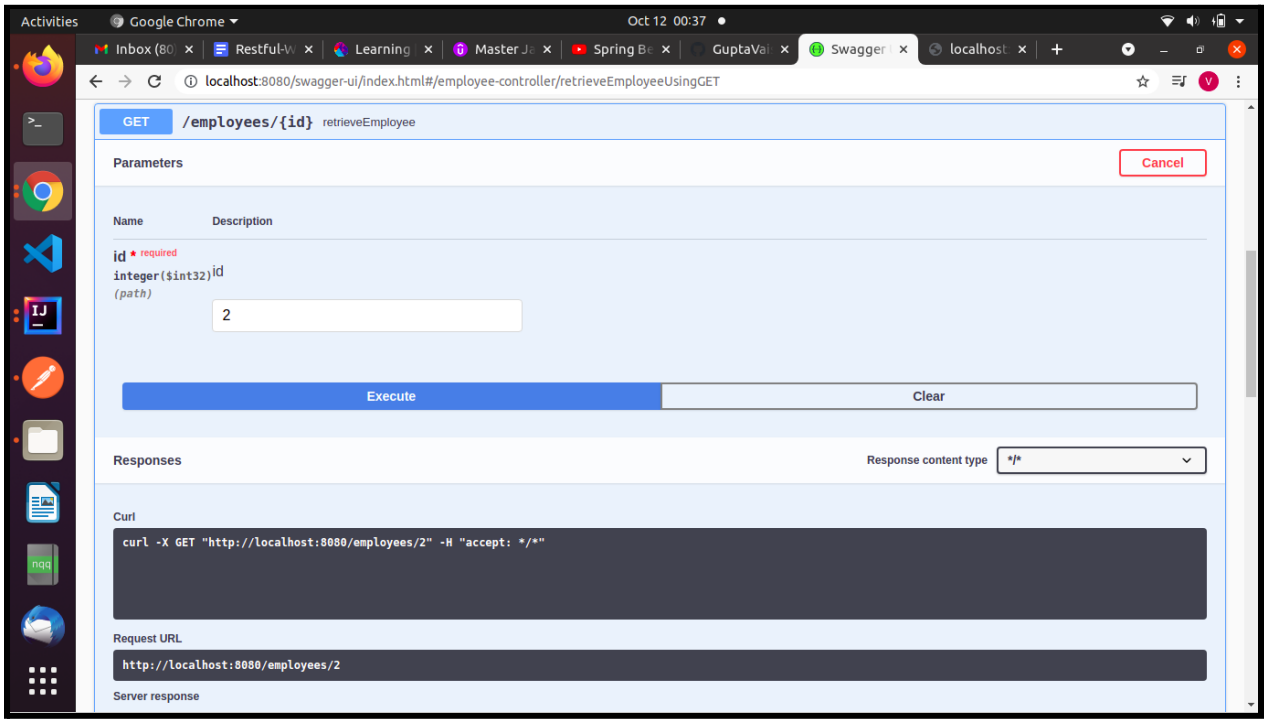
5. **Configure swagger plugin and create document of following methods:**

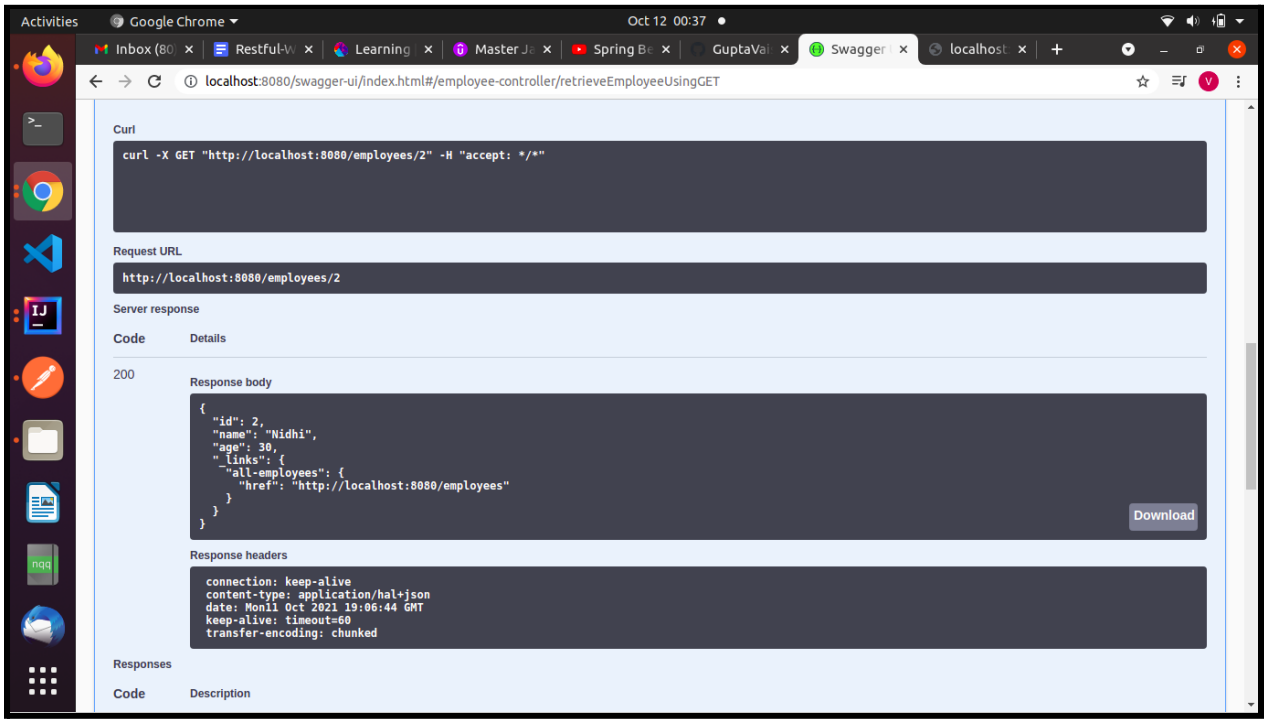


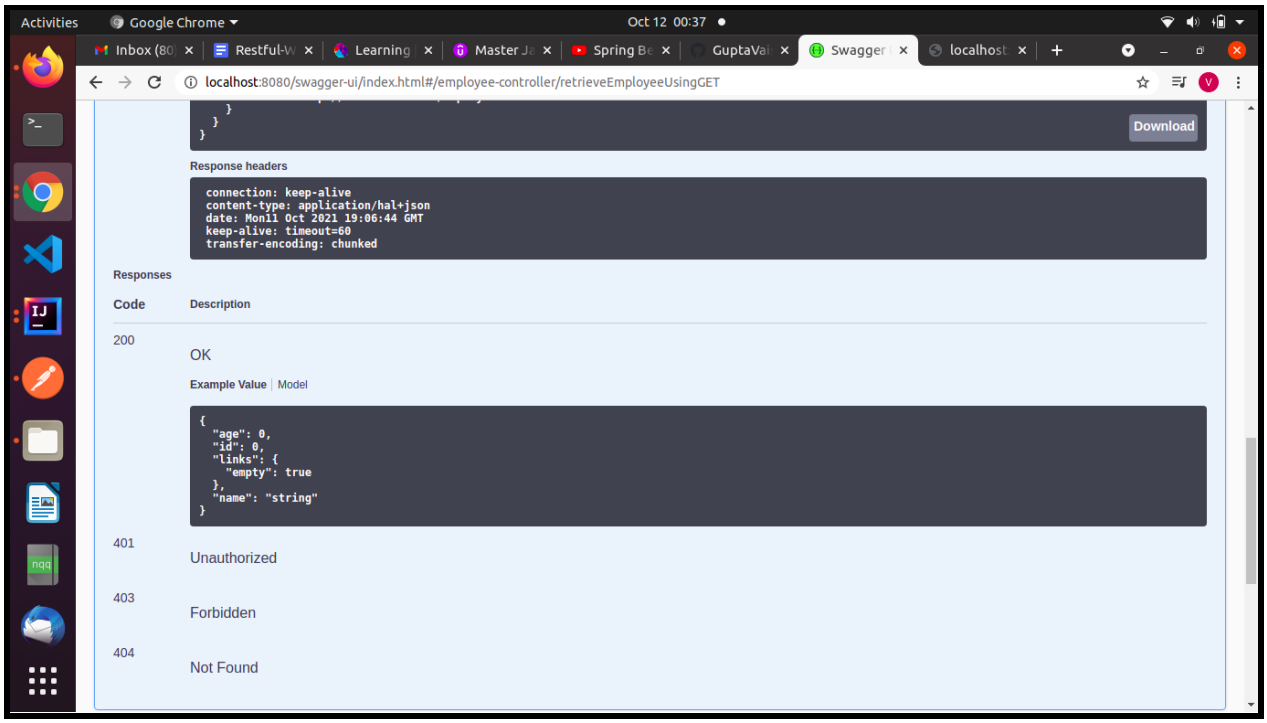


{"swagger":"2.0","info":{"description":"Api Documentation","version":"1.0","title":"Api Documentation","termsOfService":"urn:tos","contact":{},"license":{"name":"Apache 2.0","url":"http://www.apache.org/licenses/LICENSE-2.0"}},"host":"localhost:8080","basePath":"/","tags":[{"name":"basic-error-controller","description":"Basic Error Controller"},{"name":"employee-controller","description":"Employee Controller"},{"name":"operation-handler","description":"Operation Handler"},{"name":"person-version-controller","description":"Person Version Controller"},{"name":"student-controller","description":"Student Controller"},{"name":"web-mvc-links-handler","description":"Web Mvc Links Handler"},{"name":"welcome-to-spring-controller","description":"Welcome To Spring Controller"}],"paths":{"/actuator":{"get":{"tags":["web-mvc-links-handler"],"summary":"links","operationId":"linksUsingGET","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"responses":{"200":{"description":"OK","schema":{"type":"object","additionalProperties":{"type":"object","additionalProperties":{"$ref":"#/definitions/Link"}}}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/beans":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/caches":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_1","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingDELETE","produces":["*/*"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/actuator/caches/{cache}":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_2","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}},"delete":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingDELETE_1","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"204":{"description":"No Content"},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"}}}},"/actuator/conditions":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_3","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/configprops":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_4","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/configprops/{prefix}":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_5","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/env":{"get":{"tags":["operation-handler"],"summary":"handle","operationId":"handleUsingGET_6","produces":["application/json","application/vnd.spring-boot.actuator.v2+json","application/vnd.spring-boot.actuator.v3+json"],"parameters":[{"in":"body","name":"body","description":"body","required":false,"schema":{"type":"object","additionalProperties":{"type":"string"}}}],"responses":{"200":{"description":"OK","schema":{"type":"object"}},"401":{"description":"Unauthorized"},"403":{"description":"Forbidden"},"404":{"description":"Not Found"}}}},"/actuator/env/{toMatch}":{"get":{"tags":["operation-

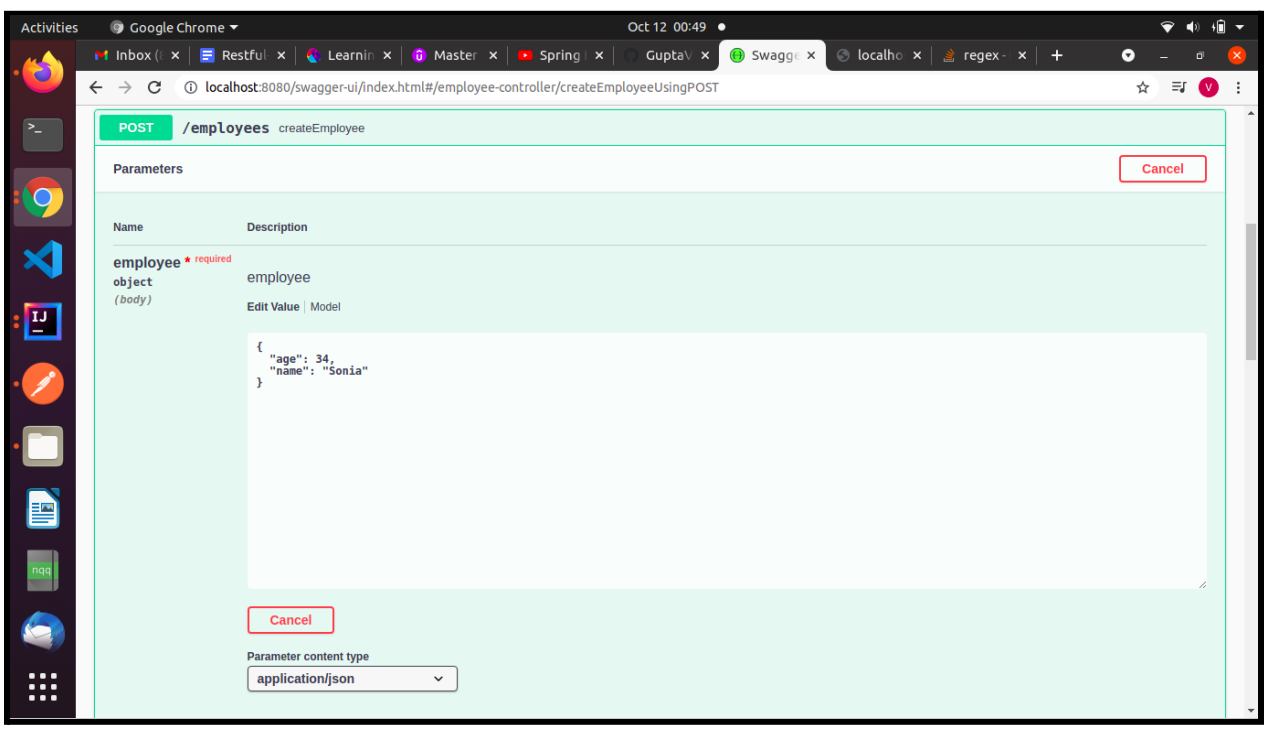
Get details of User using GET request.

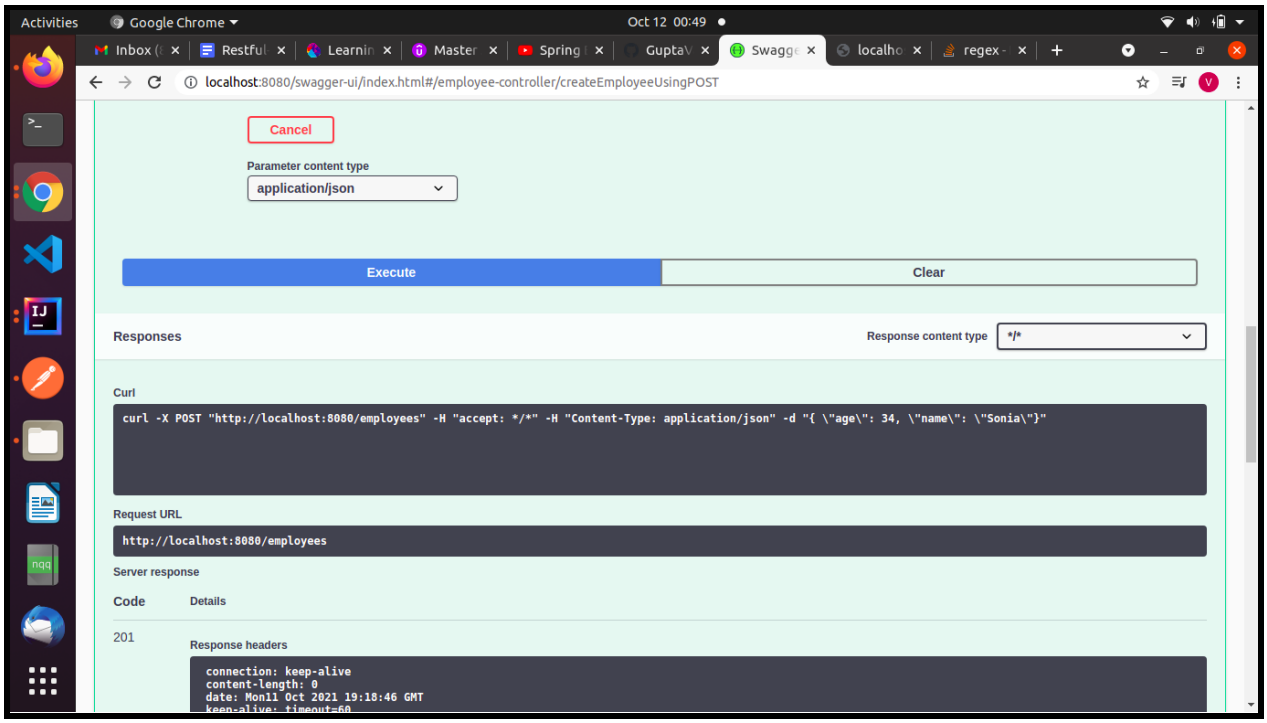


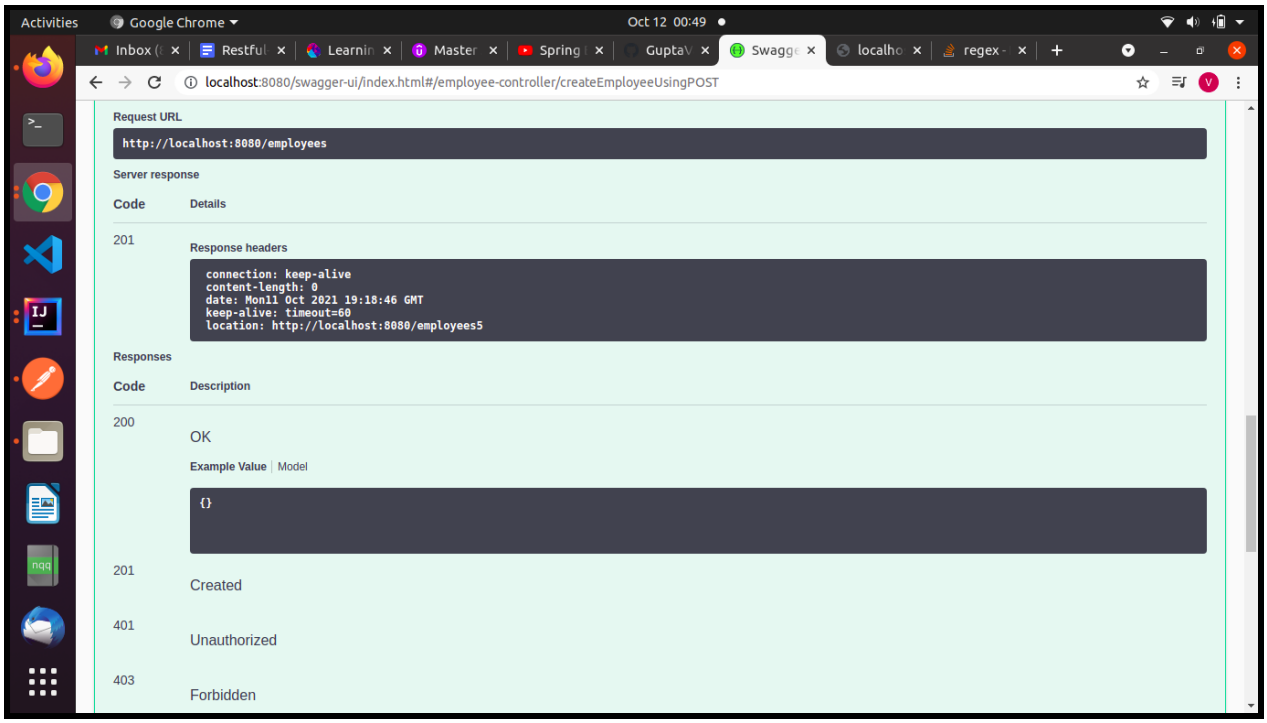




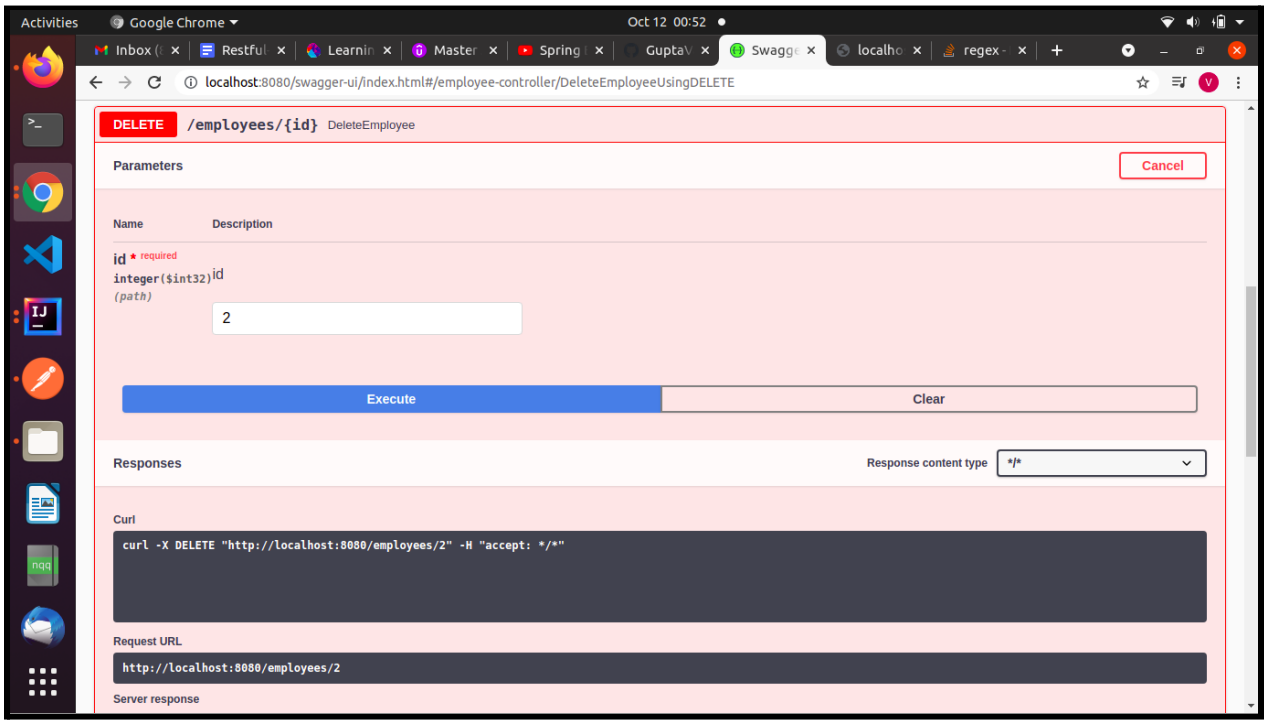
Save details of the user using POST request.

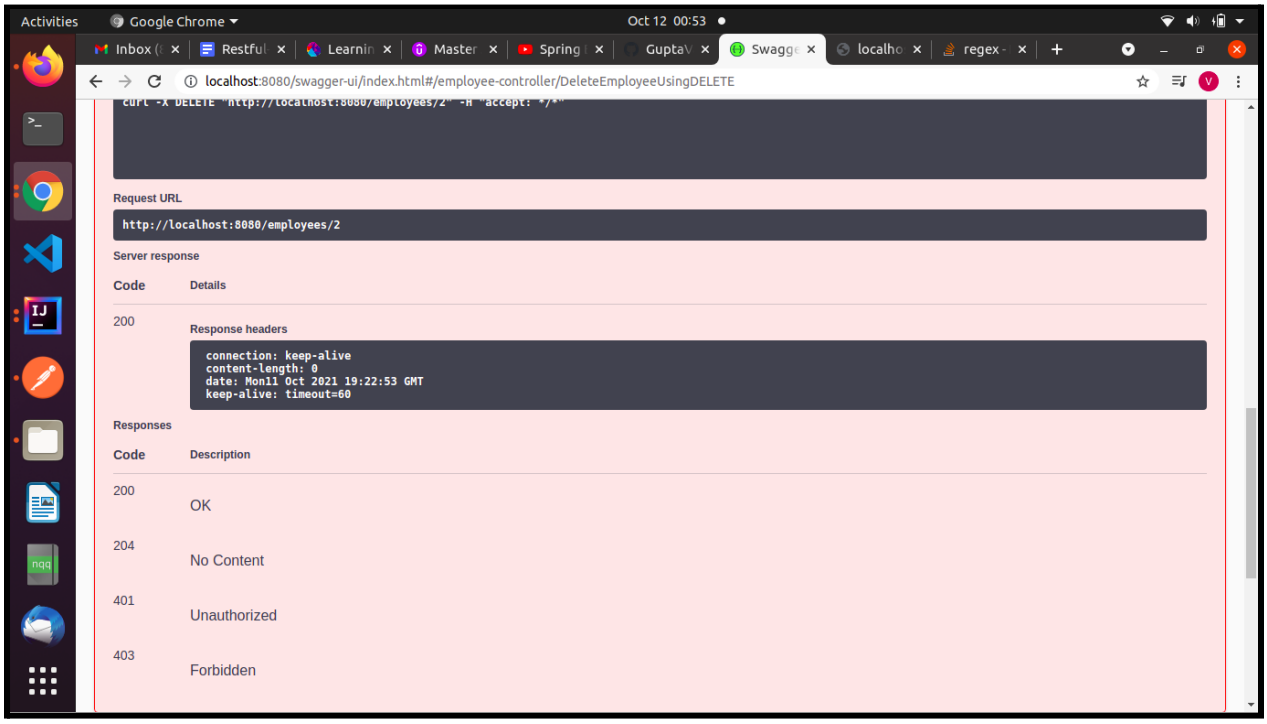






Delete a user using a DELETE request.


Activities
Google Chrome
Oct 12 00:52
localhost:8080/swagger-ui/index.html#/employee-controller/DeleteEmployeeUsingDELETE
DELETE
/employees/{id}
DeleteEmployee
Parameters
Cancel
Name
Description
id * required
integer($int32)
(path)
id
2
Execute
Clear
Responses
Response content type
/
Curl
curl -X DELETE "http://localhost:8080/employees/2" -H "accept: */*"
Request URL
http://localhost:8080/employees/2
Server response


Activities
Google Chrome
Oct 12 00:53
localhost:8080/swagger-ui/index.html#/employee-controller/DeleteEmployeeUsingDELETE
Request URL
http://localhost:8080/employees/2
Server response
Code
Details
200
Response headers
connection: keep-alive
content-length: 0
date: Mon11 Oct 2021 19:22:53 GMT
keep-alive: timeout=60
Responses
Code
Description
200
OK
204
No Content
401
Unauthorized
403
Forbidden

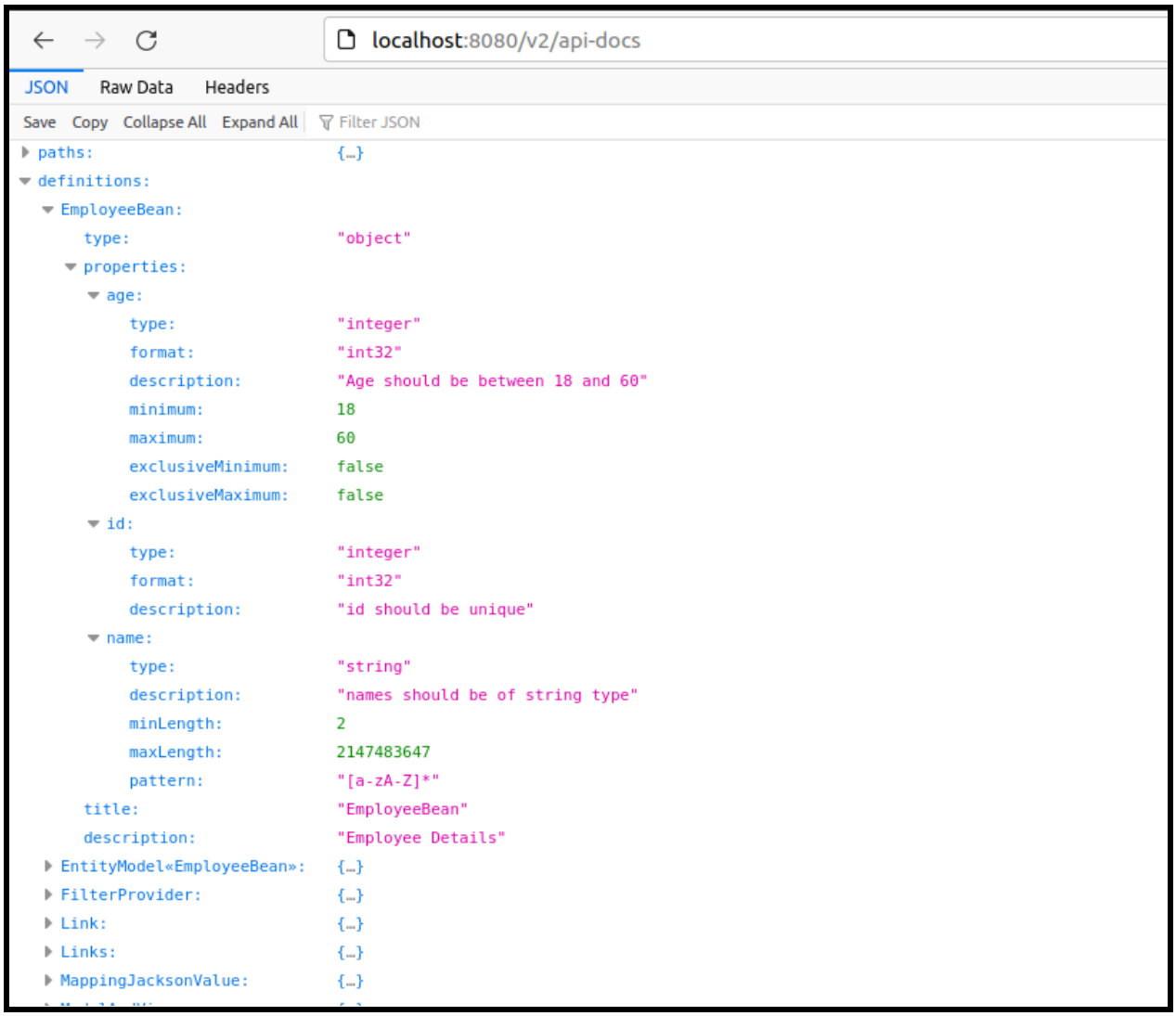
7. In swagger documentation, add the description of each class and URI so that in swagger UI the purpose of class and URI is clear.

EmployeeBean.java

```java
import javax.validation.constraints.Min;
import javax.validation.constraints.Pattern;
import javax.validation.constraints.Size;

@ApiModel(description = "Employee Details")
public class EmployeeBean {

    @ApiModelProperty(notes = "id should be unique")
    private Integer id;


    @ApiModelProperty(notes = "names should be of string type")
    @Pattern(regexp = "[a-zA-Z]*", message = "Name should contain alphabets only")
    @Size(min = 2, message = "Name length should be greater than 2")
    private String name;

    @ApiModelProperty(notes = "Age should be between 18 and 60")
    @Min(value = 18, message = "Minimum age of employee should be 18")
    @Max(value = 60, message = "Maximum age of employee should be 60")
    private Integer age;

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

EmployeeBean.java × EmployeeController.java × WelcomeToSpringController.java × WelcomeToSp

```java
    private EmployeeDaoService service;

    @ApiModelProperty(notes = "All employee details")
    @GetMapping("/employees")
    public List<EmployeeBean> retrieveAllEmployees() { return service.findAll(); }

    /*
    @GetMapping("/employees/{id}")
    public EmployeeBean retrieveEmployee(@PathVariable int id){
        EmployeeBean employee = service.findOne(id);
        if(employee == null)
            throw new EmployeeNotFoundException("id- " + id);
        return employee;
    }
     */


    @ApiModelProperty(notes = "Employee detail")
    @GetMapping("/employees/{id}")
    public EntityModel<EmployeeBean> retrieveEmployee(@PathVariable int id){
        EmployeeBean employee = service.findOne(id);
        if(employee == null)
            throw new EmployeeNotFoundException("id- " + id);

        //using hateoas
        EntityModel<EmployeeBean> resource = EntityModel.of(employee);
```



***Static and Dynamic filtering**

8. Create API which saves details of User (along with the password) but on successfully saving returns only non-critical data. (Use static filtering).

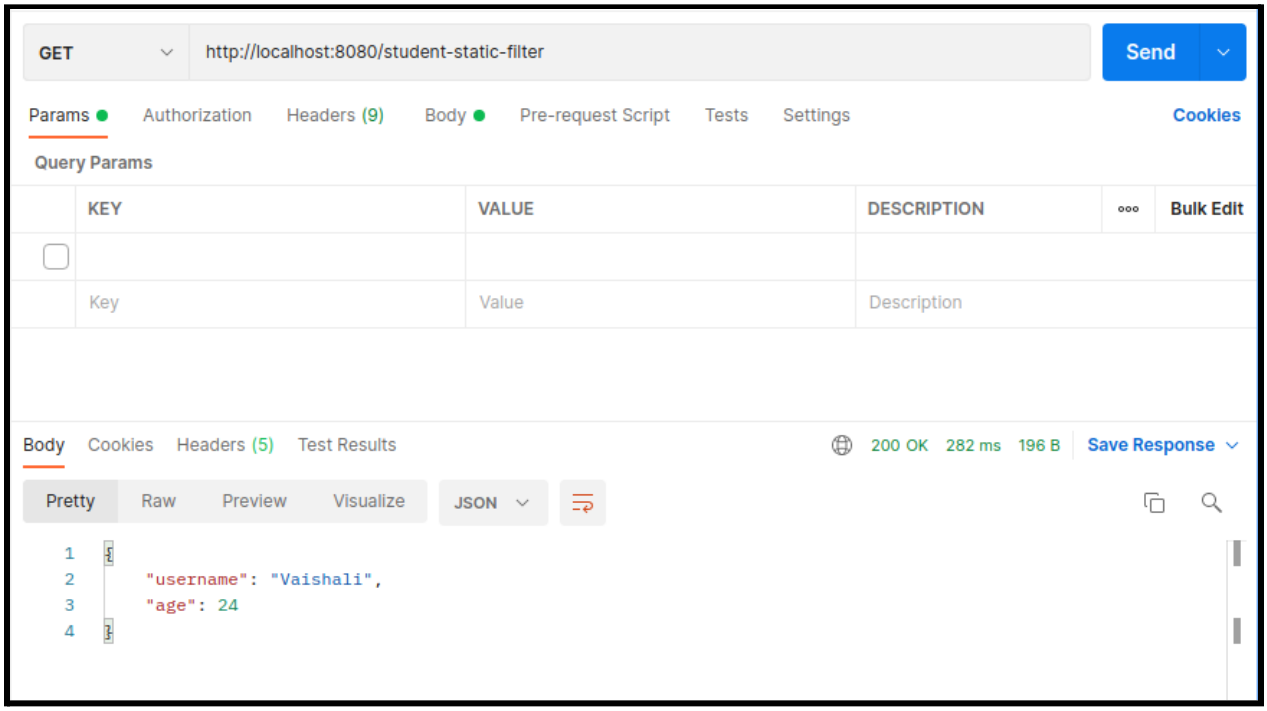
<Student.java>

```
package com.ttn.spring.springRest.webServices.RestWebServices1.filtering;

import com.fasterxml.jackson.annotation.JsonFilter;
import com.fasterxml.jackson.annotation.JsonIgnore;

//use for dynamic filter
```

```
@JsonFilter("StudentFilter")
public class Student {

   private String username;

   //use for static filter
  // @JsonIgnore
   private String password;
   private Integer age;

   public Student() {
   }

   public Student(String username, String password, Integer age) {
       this.username = username;
       this.password = password;
       this.age = age;
   }

   public String getUsername() {
       return username;
   }

   public void setUsername(String username) {
       this.username = username;
   }

   public String getPassword() {
       return password;
   }

   public void setPassword(String password) {
       this.password = password;
   }

   public Integer getAge() {
       return age;
   }

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

<StudentController.java>

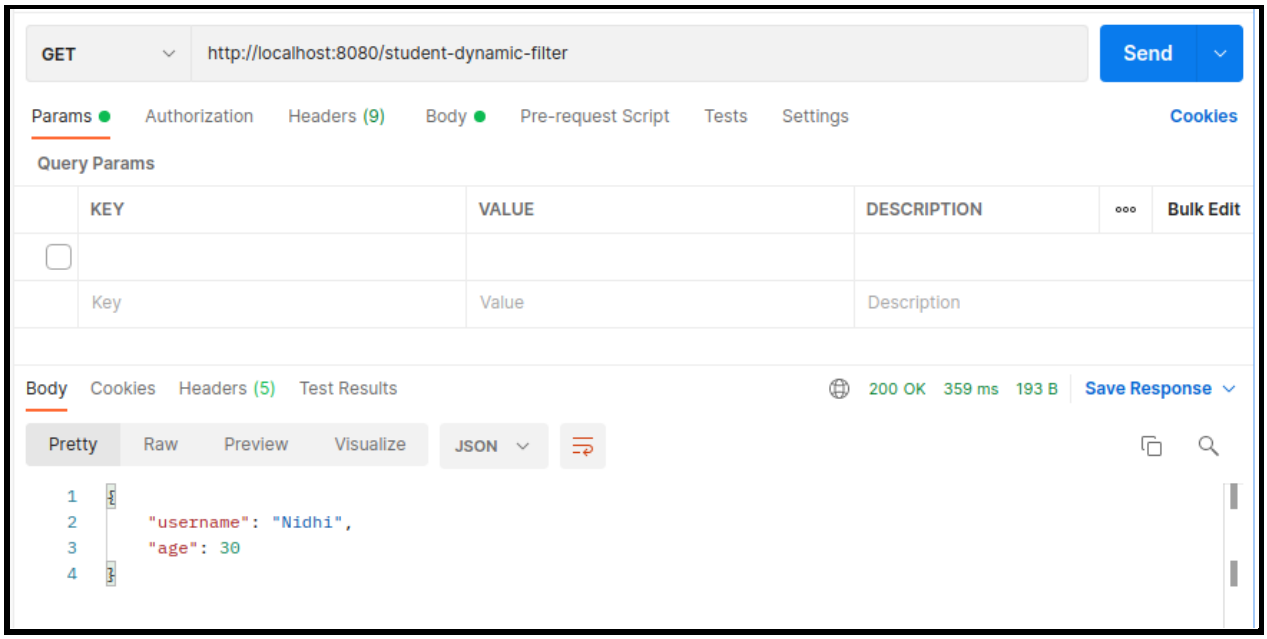
```
package com.ttn.spring.springRest.webServices.RestWebServices1.filtering;

import com.fasterxml.jackson.databind.ser.FilterProvider;
import com.fasterxml.jackson.databind.ser.impl.SimpleBeanPropertyFilter;
import com.fasterxml.jackson.databind.ser.impl.SimpleFilterProvider;
import org.springframework.http.converter.json.MappingJacksonValue;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class StudentController {

  /***** Static Filtering *********/
  @GetMapping("/student-static-filter")
  public Student findStudent(){
     return new Student("Vaishali","vaishali123",24);
  }
}
```



9. Create another API that does the same by using Dynamic Filtering.

<StudentController.java>

```
package com.ttn.spring.springRest.webServices.RestWebServices1.filtering;

import com.fasterxml.jackson.databind.ser.FilterProvider;
import com.fasterxml.jackson.databind.ser.impl.SimpleBeanPropertyFilter;
import com.fasterxml.jackson.databind.ser.impl.SimpleFilterProvider;
import org.springframework.http.converter.json.MappingJacksonValue;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RestController;

@RestController
public class StudentController {

  /***** Static Filtering *********/
  @GetMapping("/student-static-filter")
  public Student findStudent(){
     return new Student("Vaishali","vaishali123",24);
  }

  /****** Dynamic Filtering  ******/
  @GetMapping("/student-dynamic-filter")
  public MappingJacksonValue retrieveSomeBean(){
     Student student = new Student("Nidhi","Nidhi123",30);
     SimpleBeanPropertyFilter filter = SimpleBeanPropertyFilter
            .filterOutAllExcept("username","age");
     FilterProvider filters = new SimpleFilterProvider()
            .addFilter("StudentFilter",filter);
     MappingJacksonValue mapping = new MappingJacksonValue(student);
     mapping.setFilters(filters);
     return mapping;
  }

}
```



***Versioning Restful APIs**

10. Create 2 API for showing user details. The first api should return only basic details of the user and the other API should return more/enhanced details of the user,

Now apply versioning using the following methods:

Name.java × PersonV1.java × PersonV2.java × PersonVersionController.java ×

```java
package com.ttn.spring.springRest.webServices.RestWebServices1.versioning;

public class PersonV1 {
    private String name;

    PersonV1(){
    }

    public PersonV1(String name) { this.name = name; }

    public String getName() { return name; }
}
```

Name.java × PersonV1.java × PersonV2.java × PersonVersionController.java ×

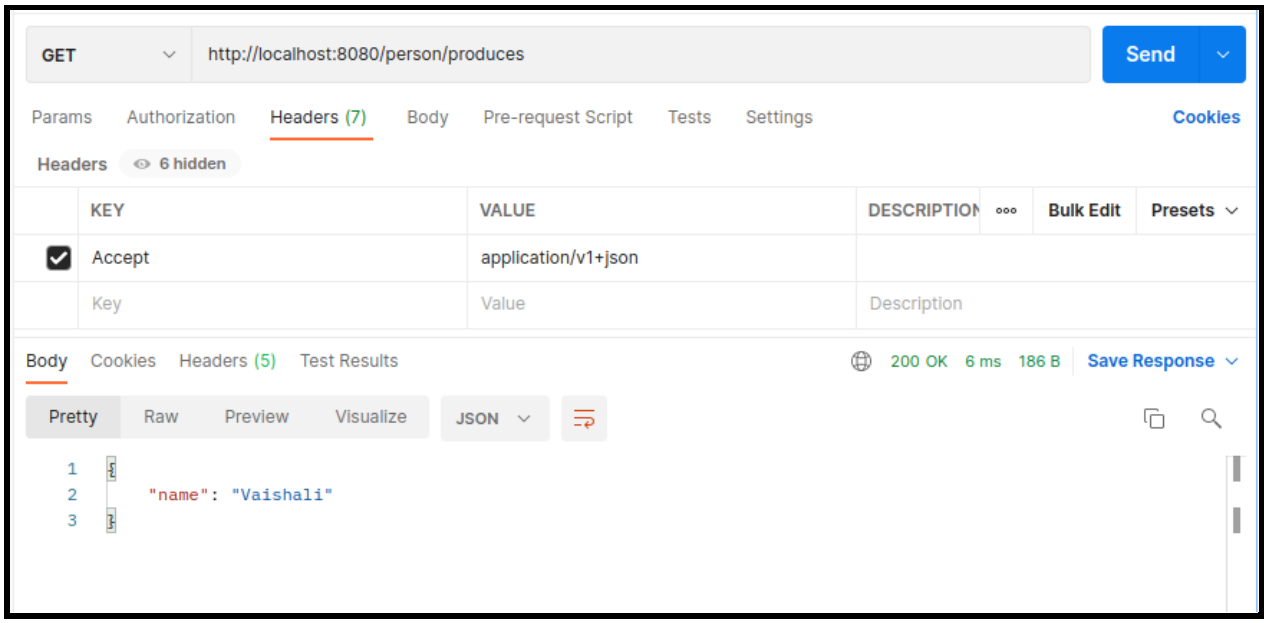
```java
package com.ttn.spring.springRest.webServices.RestWebServices1.versioning;

public class PersonV2 {
    private Name name;

    public PersonV2(Name name) { this.name = name; }

    public Name getName() {
        return name;
    }

}
```

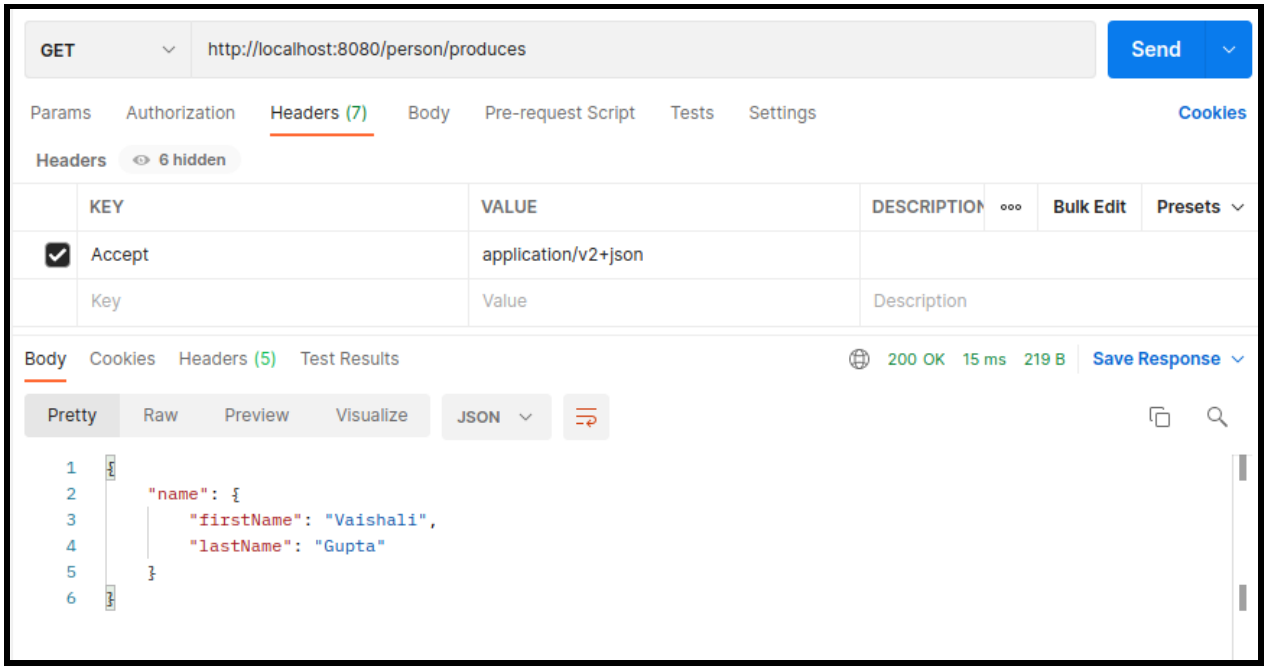
```java
package com.ttn.spring.springRest.webServices.RestWebServices1.versioning;

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

@RestController
public class PersonVersionController {

    //Content Negotiation or Accept versioning or MIME type versioning or producer versioning

    @GetMapping(value = "person/produces",produces = "application/v1+json")
    public PersonV1 personAcceptV1(){
        return new PersonV1( name: "Vaishali");
    }

    @GetMapping(value = "person/produces",produces = "application/v2+json")
    public PersonV2 personAcceptV2(){
        return new PersonV2(new Name("Vaishali","Gupta"));
    }


    //header versioning
    @GetMapping(value = "person/header",headers = "VERSION=1")
    public PersonV1 personHeaderV1(){
        return new PersonV1( name: "Vaishali");
    }
```

- **MimeType Versioning**

//Content Negotiation or Accept versioning or MIME type versioning or producer versioning

```
@GetMapping(value = "person/produces",produces = "application/v1+json")
public PersonV1 personAcceptV1(){
  return new PersonV1("Vaishali");
}

@GetMapping(value = "person/produces",produces = "application/v2+json")
public PersonV2 personAcceptV2(){
  return new PersonV2(new Name("Vaishali","Gupta"));
}
```

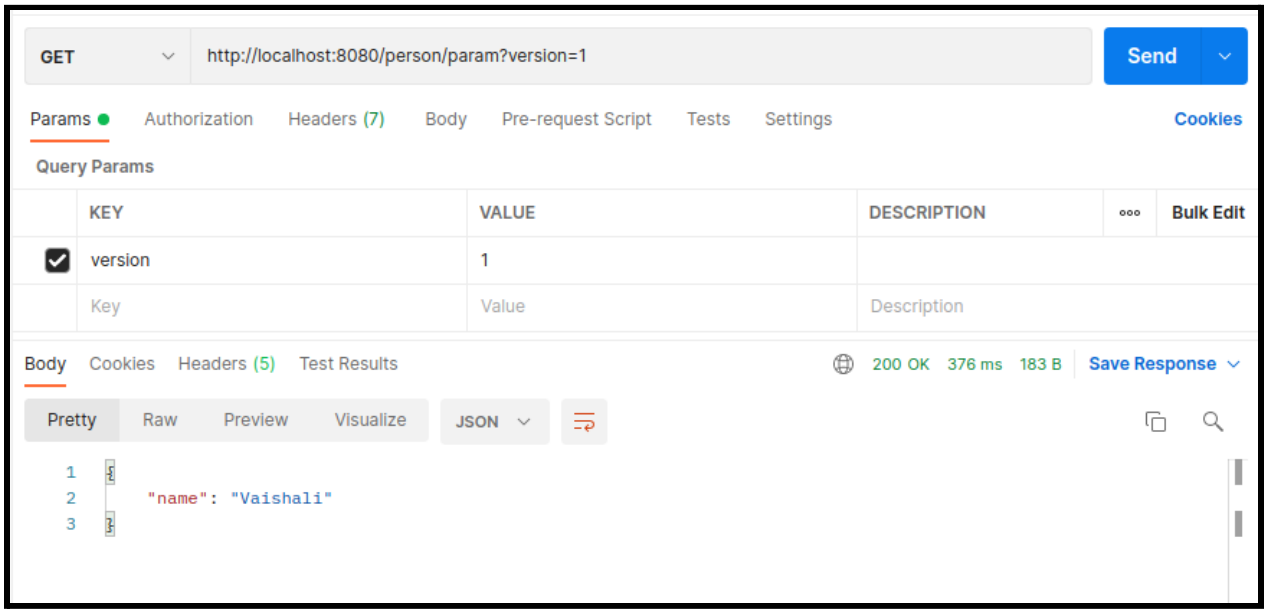


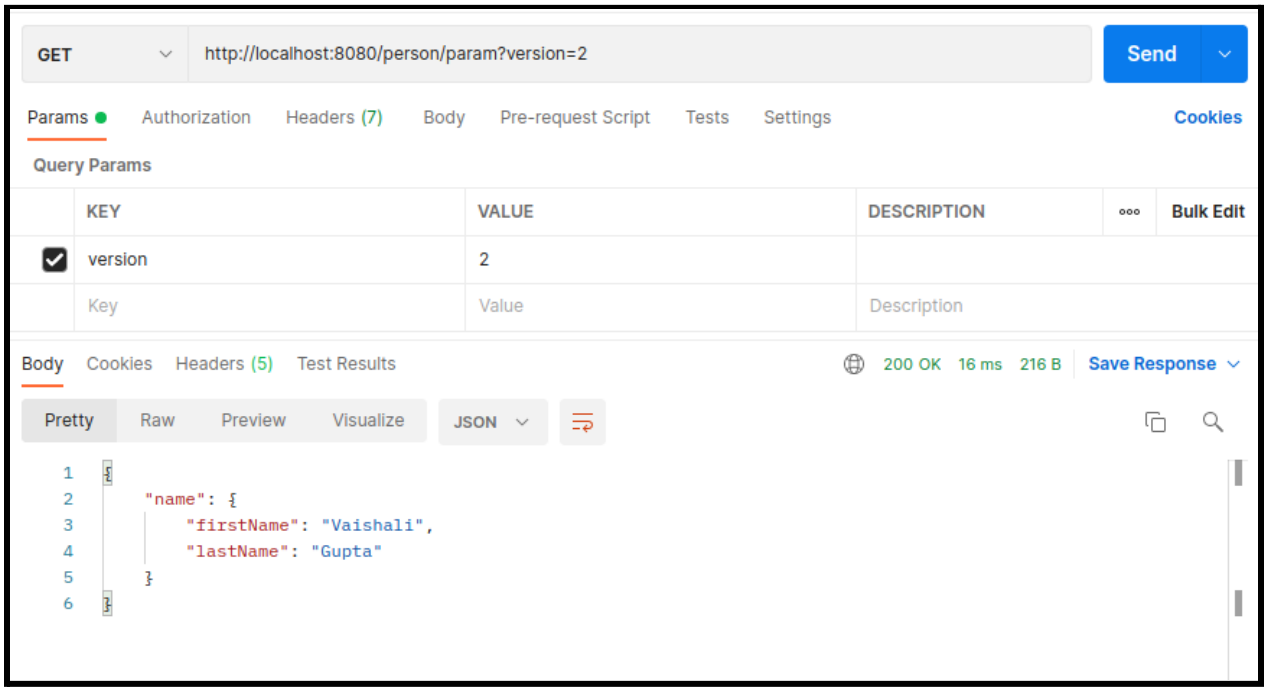


- **Request Parameter versioning**

```
//Param versioning
@GetMapping(value = "person/param",params = "version=1")
public PersonV1 personParamV1(){
  return new PersonV1("Vaishali");
}
```

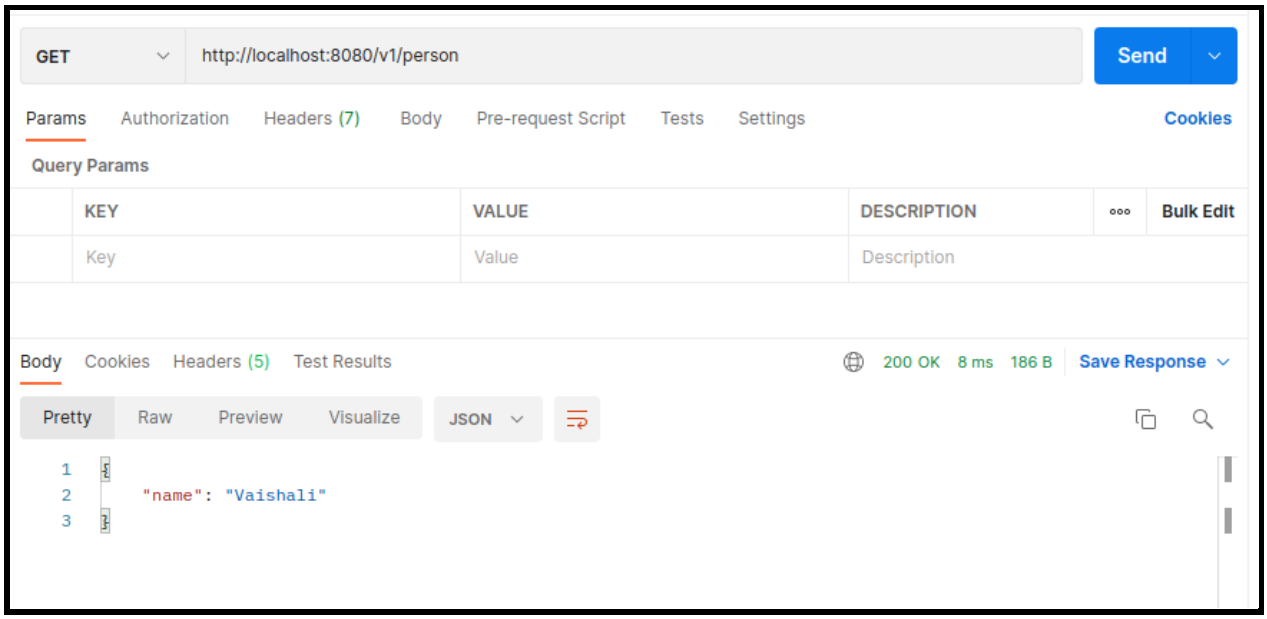
```
@GetMapping(value = "person/param",params = "version=2")
public PersonV2 personParamV2(){
  return new PersonV2(new Name("Vaishali","Gupta"));
}
```

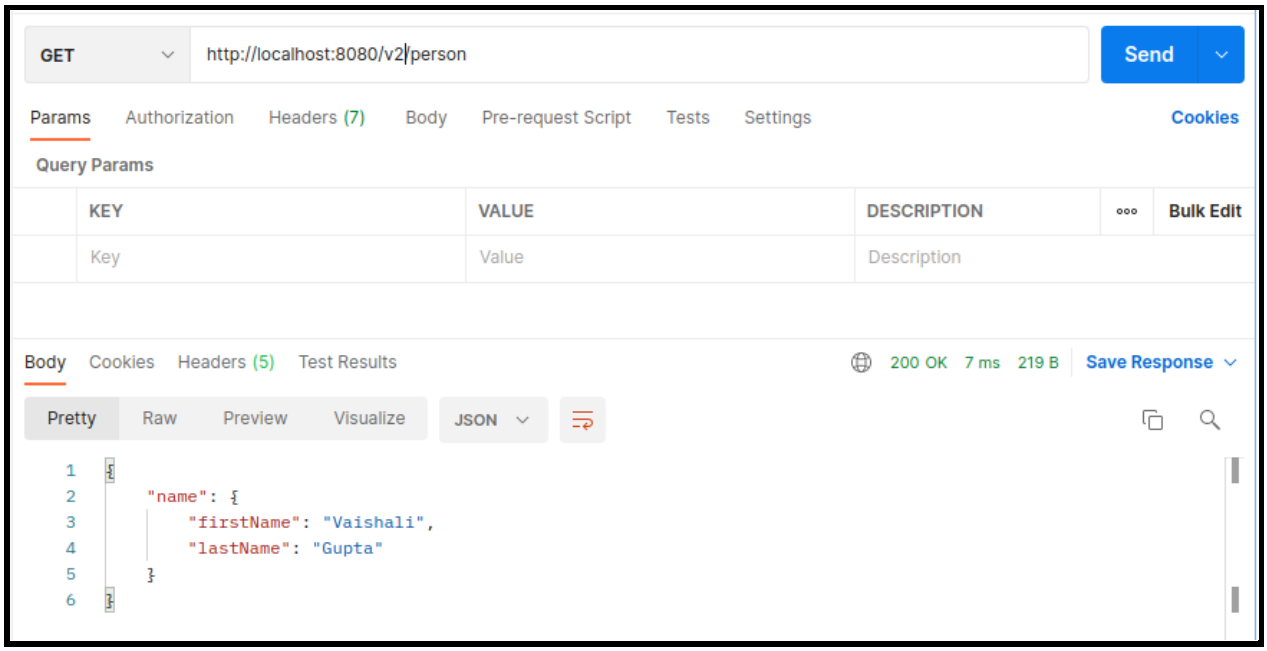




- **URI versioning**

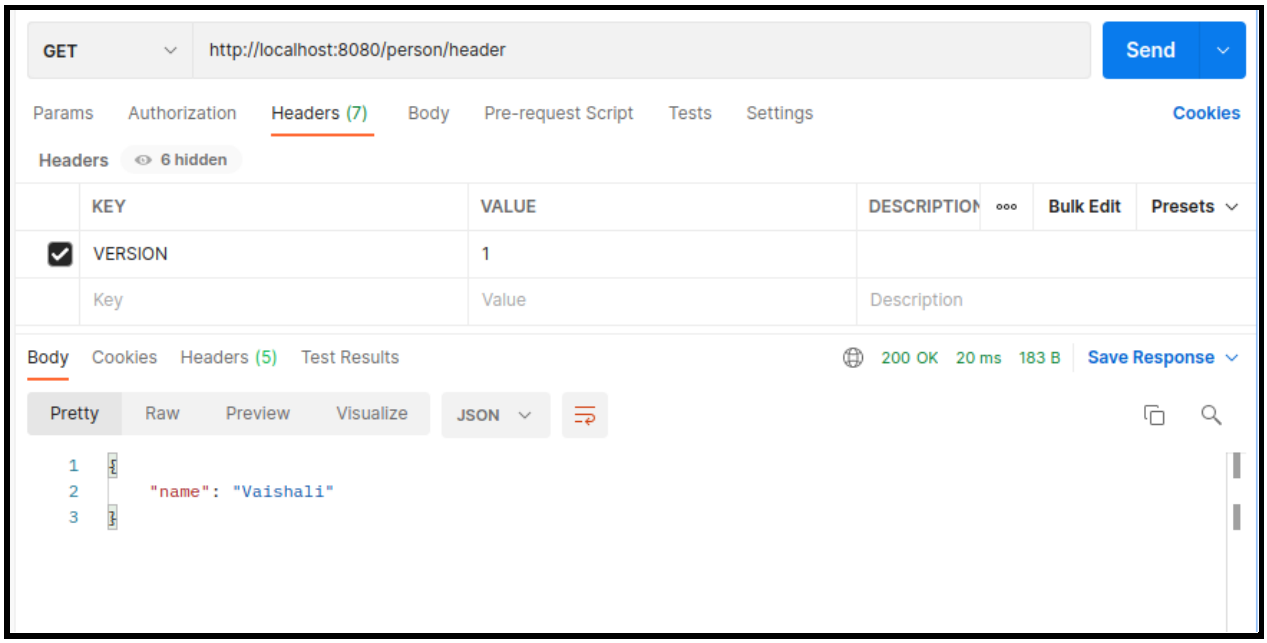
```
//url versioning
@GetMapping("/v1/person")
public PersonV1 personV1(){
   return new PersonV1("Vaishali");
}

@GetMapping("/v2/person")
public PersonV2 personV2(){
   return new PersonV2(new Name("Vaishali","Gupta"));
}
```

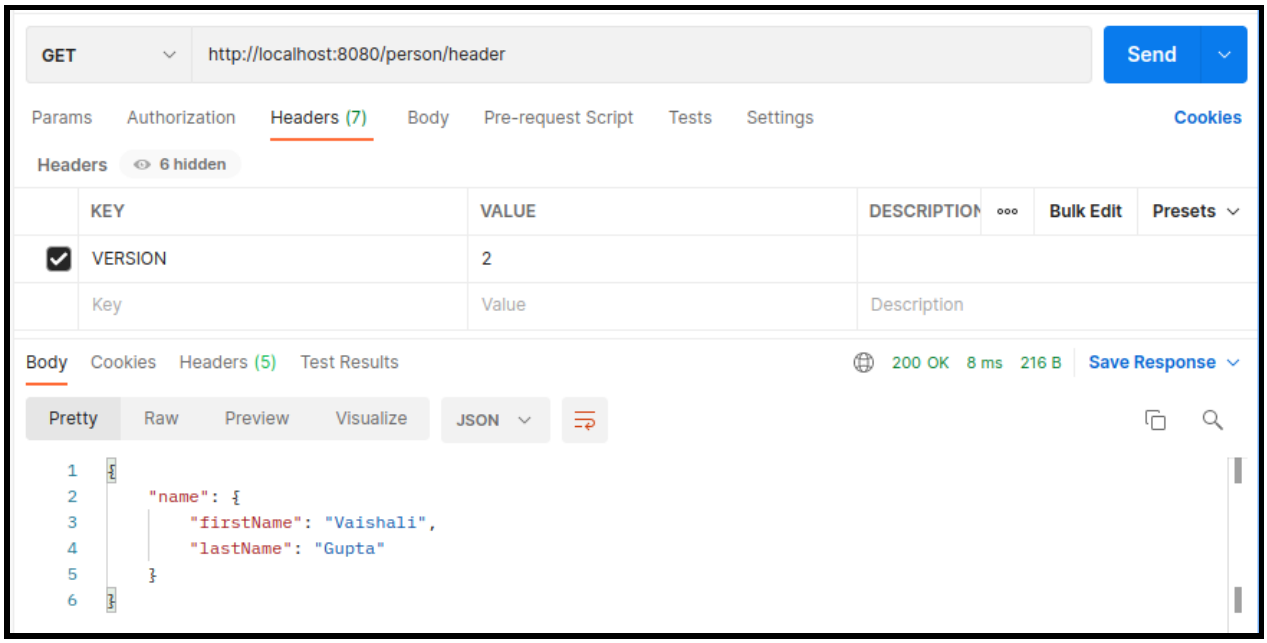




- **Custom Header Versioning**

```
//header versioning
@GetMapping(value = "person/header",headers = "VERSION=1")
public PersonV1 personHeaderV1(){
  return new PersonV1("Vaishali");
}

@GetMapping(value = "person/header",headers = "VERSION=2")
public PersonV2 personHeaderV2(){
  return new PersonV2(new Name("Vaishali","Gupta"));
}
```





*HATEOAS

11. Configure hateoas with your springboot application. Create an api which returns User Details along with url to show all topics.

@GetMapping("/employees/{id}")

```
public EntityModel<EmployeeBean> retrieveEmployee(@PathVariable int id){
    EmployeeBean employee = service.findOne(id);
    if(employee == null)
        throw new EmployeeNotFoundException("id- " + id);

    //using hateoas
    EntityModel<EmployeeBean> resource = EntityModel.of(employee);
    WebMvcLinkBuilder linkTo = linkTo(methodOn(this.getClass()).retrieveAllEmployees());
    resource.add(linkTo.withRel("all-employees"));
    return resource;
}
```

pom.xml (RestWebServices1)

```
44          </dependency>
45
46          <dependency>
47              <groupId>org.springframework.boot</groupId>
48              <artifactId>spring-boot-starter-validation</artifactId>
49          </dependency>
50
51          <dependency>
52              <groupId>org.springframework.boot</groupId>
53              <artifactId>spring-boot-starter-actuator</artifactId>
54          </dependency>
55
56  <!--        <dependency>-->
57  <!--            <groupId>com.fasterxml.jackson.dataformat</groupId>-->
58  <!--            <artifactId>jackson-dataformat-xml</artifactId>-->
59  <!--        </dependency>-->
60
61          <dependency>
62              <groupId>org.springframework.boot</groupId>
63              <artifactId>spring-boot-starter-hateoas</artifactId>
64          </dependency>
65
66      </dependencies>
67
68      <build>
```

GET http://localhost:8080/employees/1 Send

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	ooo Bulk Edit
Key	Value	Description	

Body Cookies Headers (5) Test Results 200 OK 13 ms 273 B Save Response

Pretty Raw Preview Visualize JSON


```
{
    "id": 1,
    "name": "Vaishali",
    "age": 24,
    "_links": {
        "all-employees": {
            "href": "http://localhost:8080/employees"
        }
    }
}
```