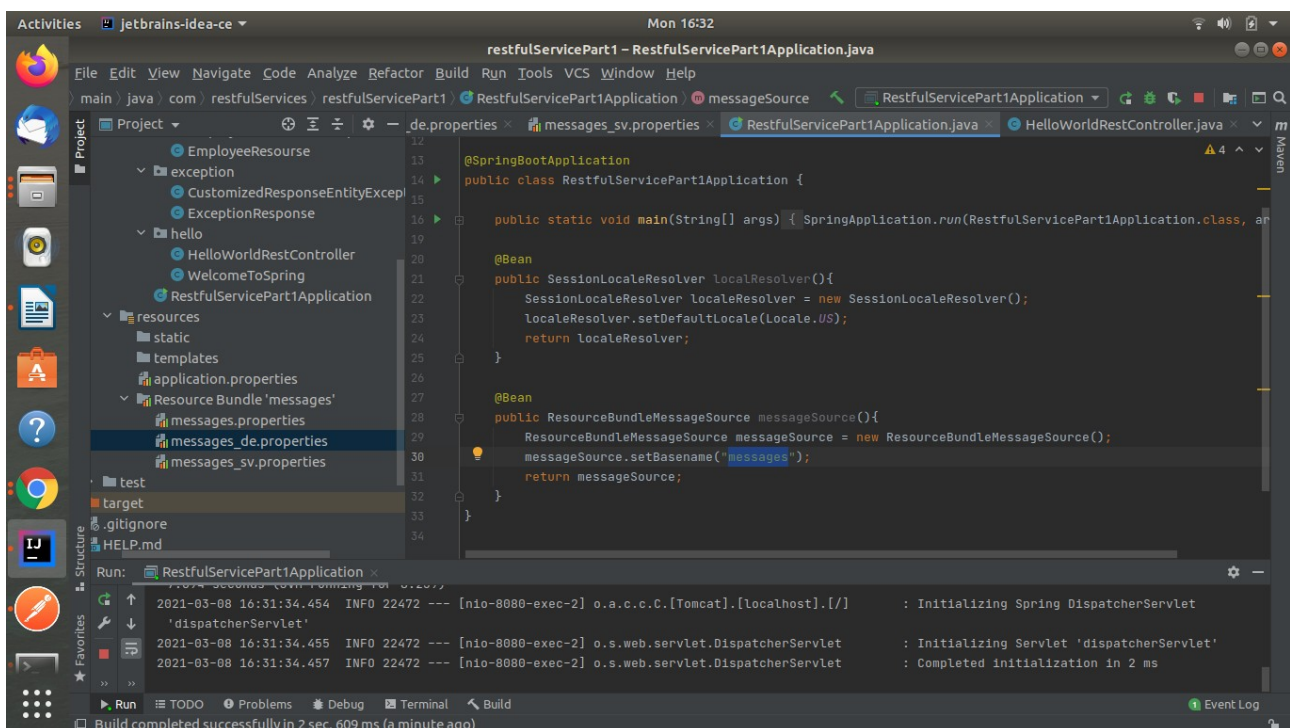
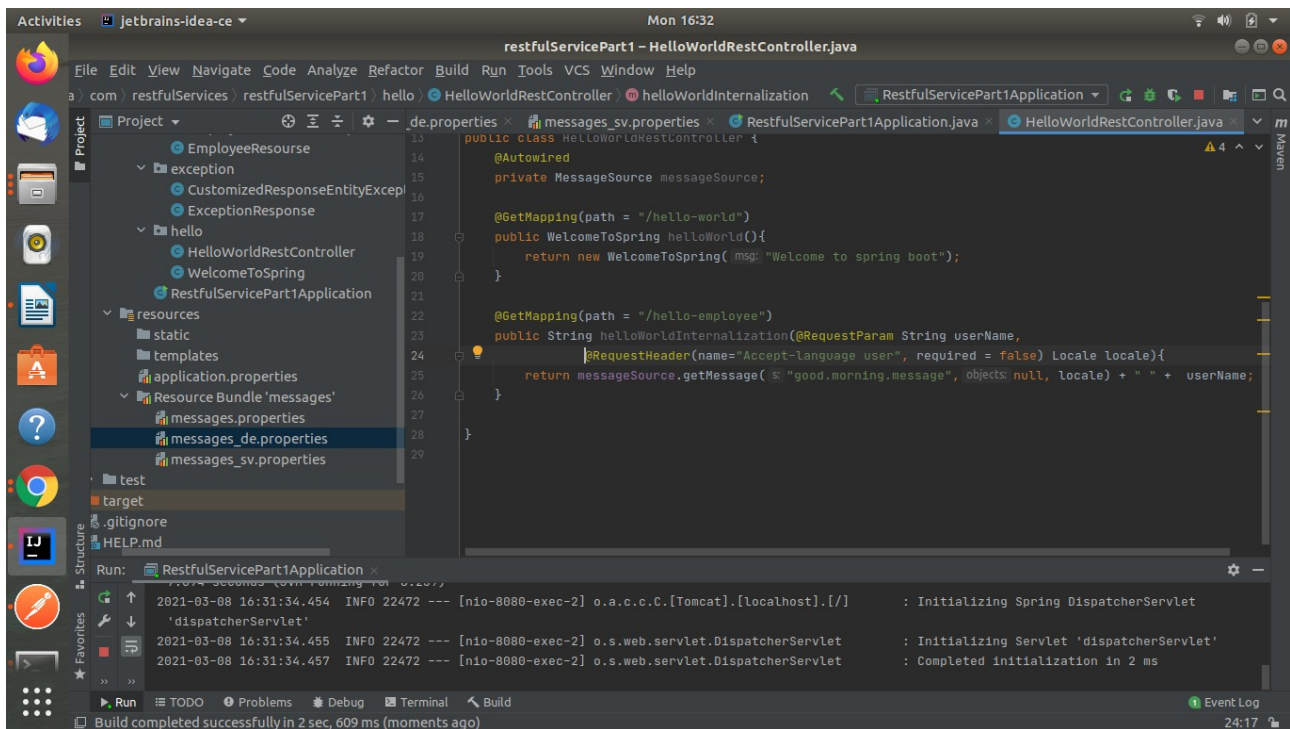
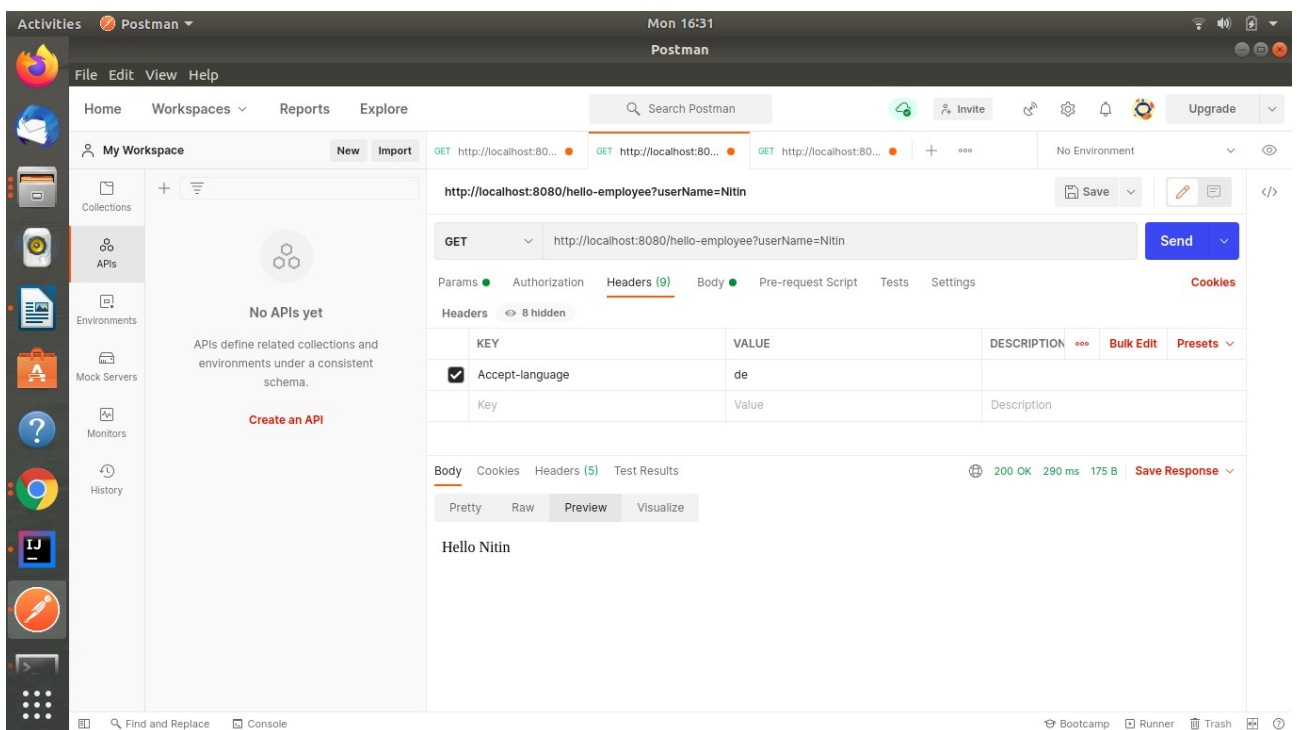
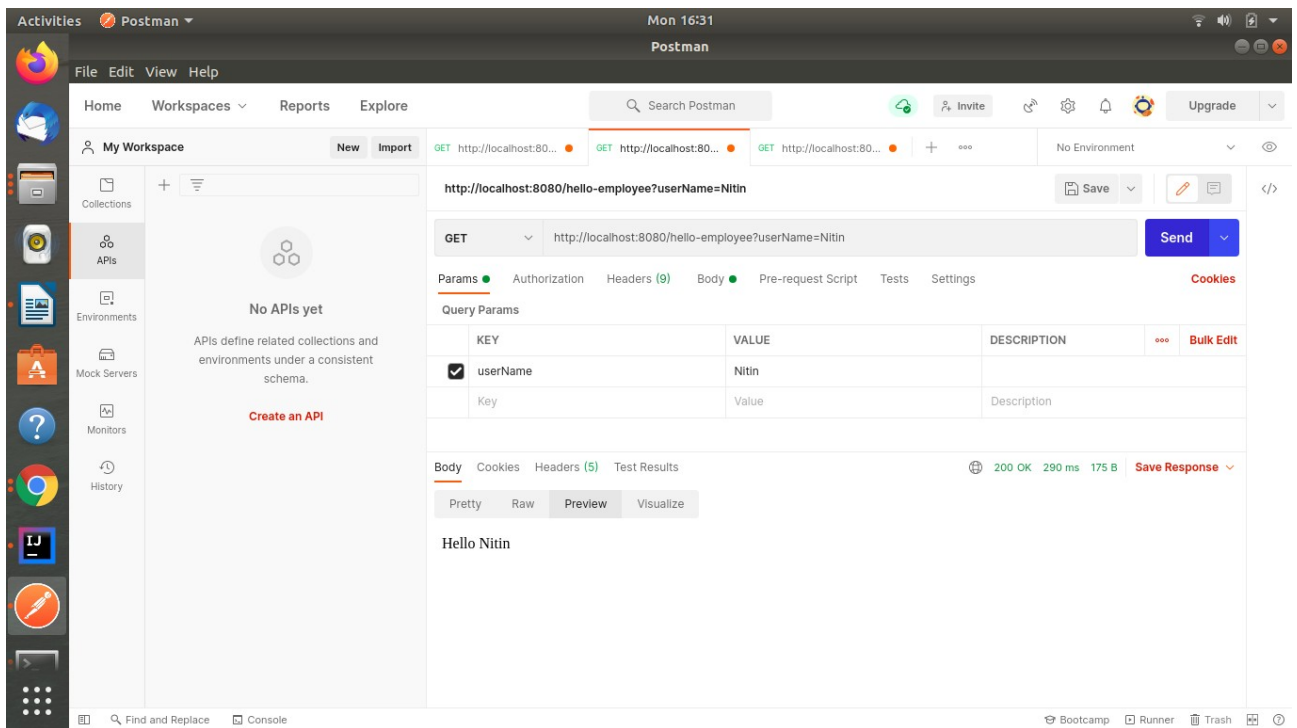


*Internationalization

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

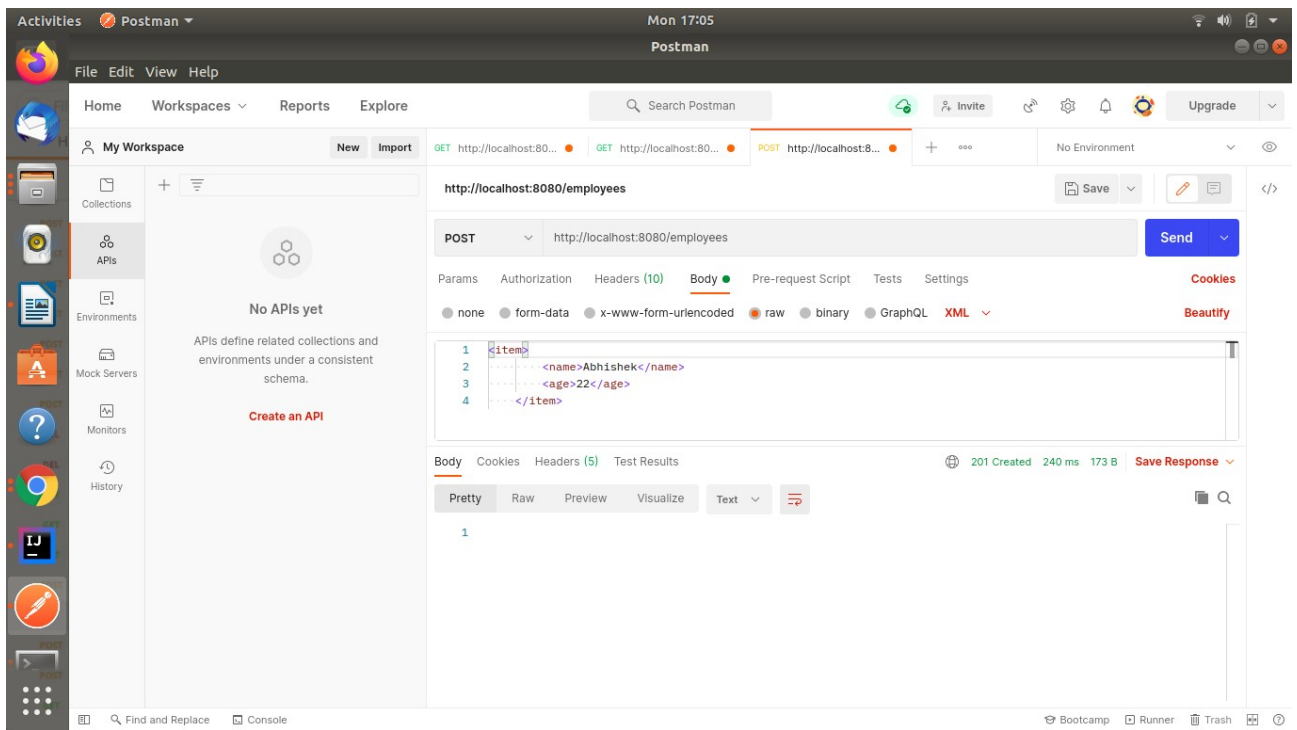


2. Create a GET request which takes "username" as param and shows a localized message "Hello Username". (Use parameters in message properties)

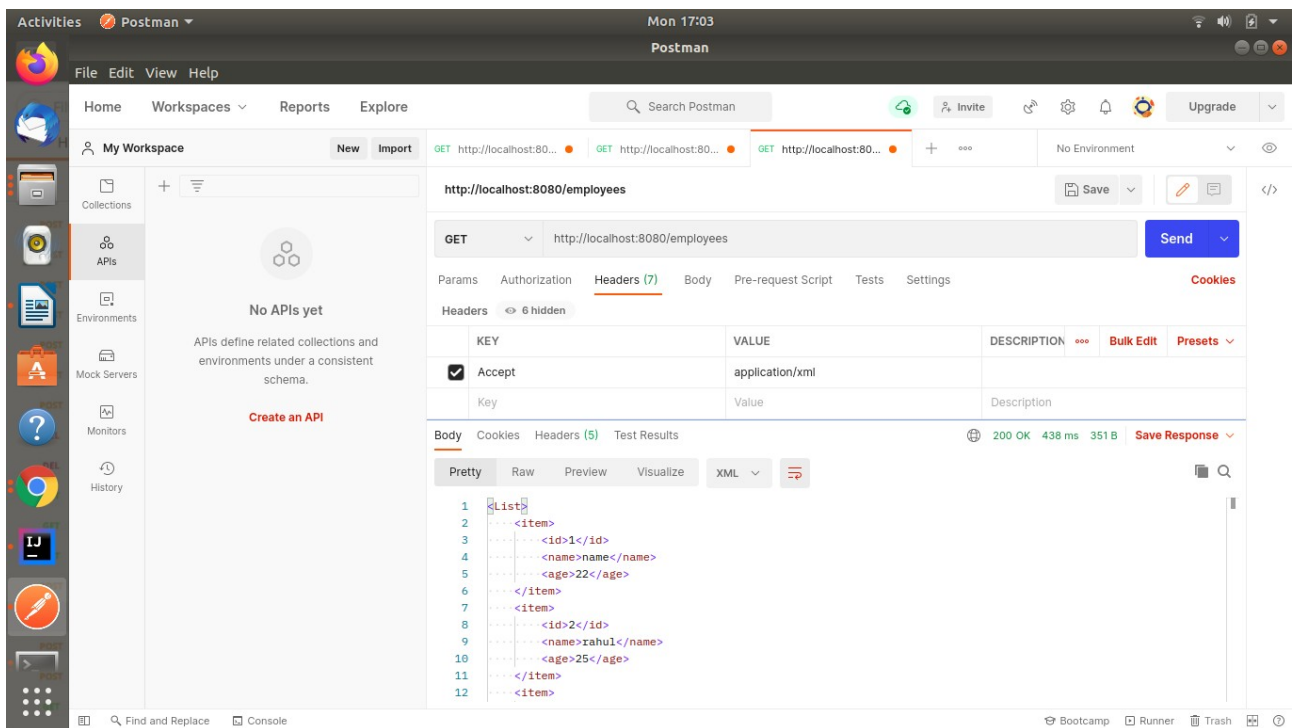


*Content Negotiation

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.



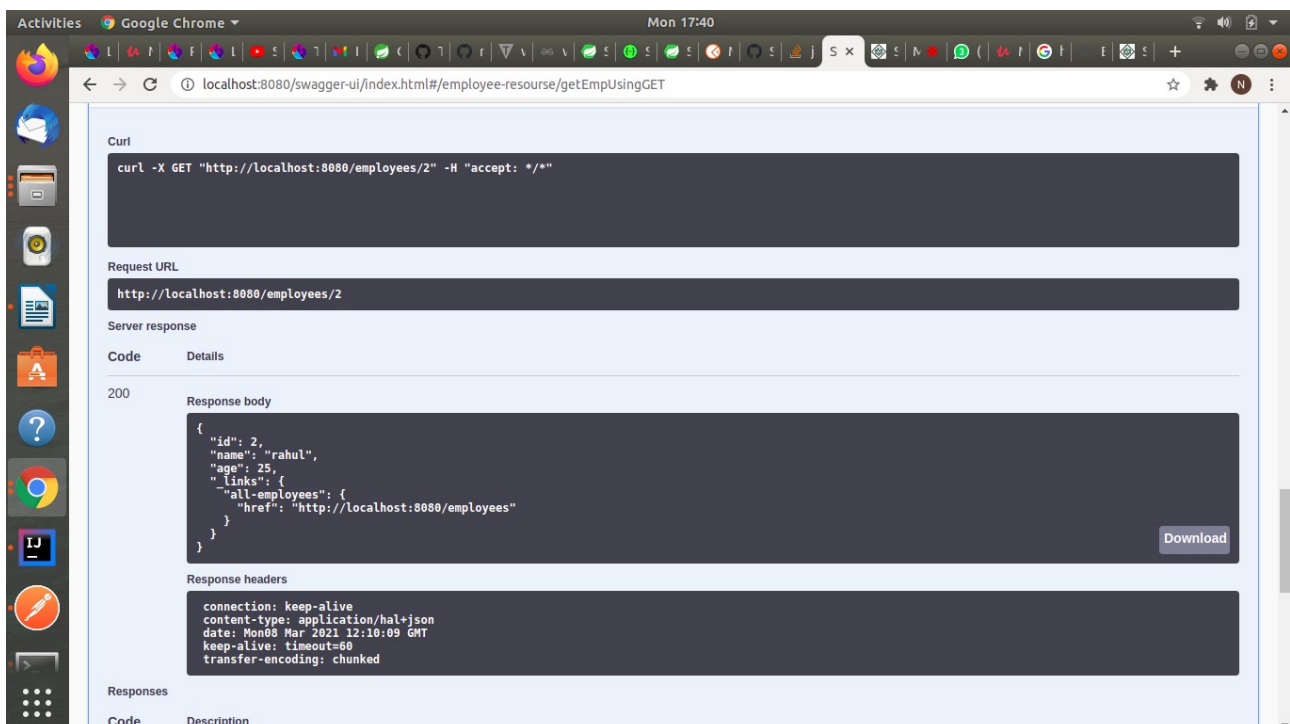
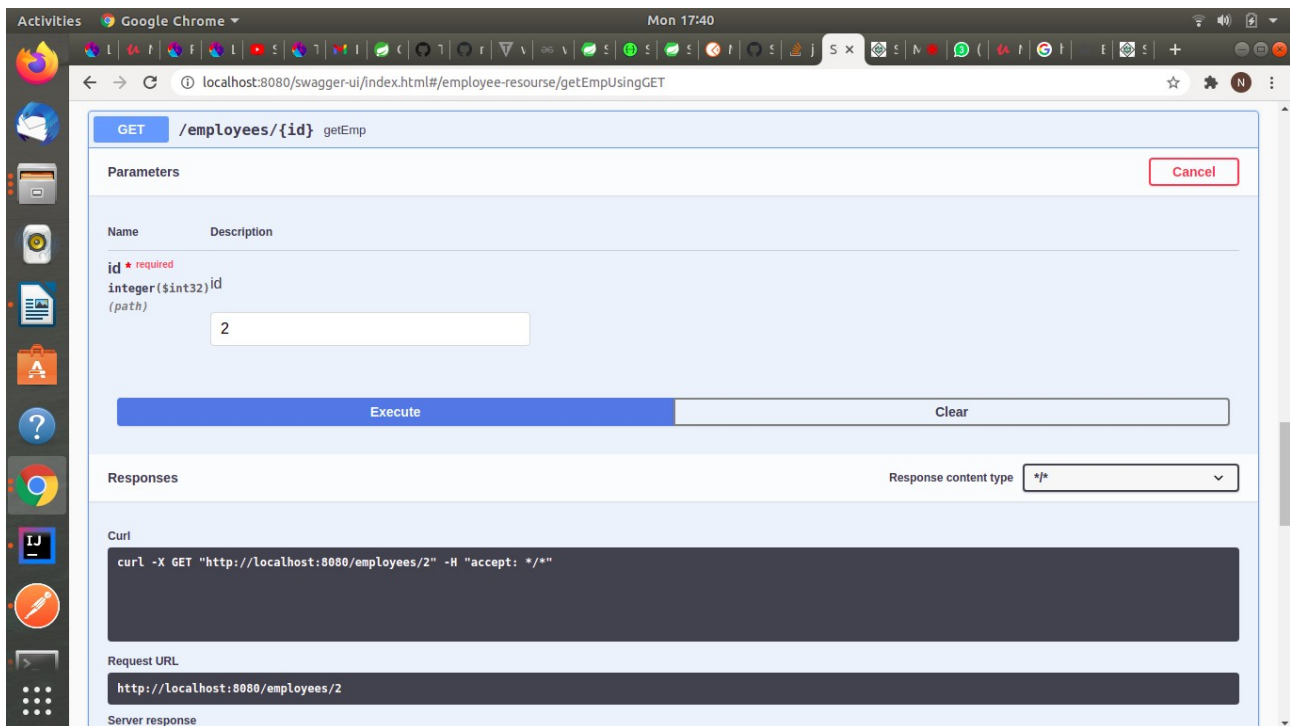
*Swagger

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

Get details of User using GET request.

Save details of the user using POST request.

Delete a user using DELETE request.



Activities Google Chrome Mon 17:40

localhost:8080/swagger-ui/index.html#/employee-resource/getEmpUsingGET

Download

Response headers

```
connection: keep-alive
content-type: application/hal+json
date: Mon08 Mar 2021 12:10:09 GMT
keep-alive: timeout=60
transfer-encoding: chunked
```

Responses

Code	Description
200	OK
401	Unauthorized
403	Forbidden
404	Not Found

Example Value | Model

```
{
  "_links": {
    "empty": true
  },
  "age": 0,
  "id": 0,
  "name": "string"
}
```

Activities Google Chrome Mon 17:45

localhost:8080/swagger-ui/index.html#/employee-resource/getAllEmpUsingGET

Name Description

emp * required

object

(body)

Edit Value | Model

```
{
  "age": 20,
  "name": "Nitin"
}
```

Cancel

Parameter content type

application/json

Execute Clear

Activities Google Chrome Mon 17:45

localhost:8080/swagger-ui/index.html#/employee-resource/getAllEmpUsingGET

Curl

```
curl -X POST "http://localhost:8080/employees" -H "accept: */*" -H "Content-Type: application/json" -d '{"age": 20, "name": "Nitin"}'
```

Request URL

http://localhost:8080/employees

Server response

Code	Details
201	<p>Response headers</p> <pre>connection: keep-alive content-length: 0 date: Mon08 Mar 2021 12:14:51 GMT keep-alive: timeout=60 location: http://localhost:8080/employees/5</pre>

Responses

Code	Description
200	<p>OK</p> <p>Example Value Model</p> <pre>{}</pre>
201	<p>Created</p>

Activities Google Chrome Mon 17:45

localhost:8080/swagger-ui/index.html#/employee-resource/getAllEmpUsingGET

Server response

Code	Details
201	<p>Response headers</p> <pre>connection: keep-alive content-length: 0 date: Mon08 Mar 2021 12:14:51 GMT keep-alive: timeout=60 location: http://localhost:8080/employees/5</pre>

Responses

Code	Description
200	<p>OK</p> <p>Example Value Model</p> <pre>{}</pre>
201	<p>Created</p>
401	<p>Unauthorized</p>
403	<p>Forbidden</p>
404	<p>Not Found</p>

Activities Google Chrome Mon 17:46

localhost:8080/swagger-ui/index.html#/employee-resource/deleteEmpUsingDELETE

DELETE /employees/{id} deleteEmp

Parameters Cancel

Name	Description
id <small>* required</small>	
integer(\$int32)id	
(path)	

5

Execute Clear

Responses Response content type */*

Curl

```
curl -X DELETE "http://localhost:8080/employees/5" -H "accept: */*"
```

Request URL

```
http://localhost:8080/employees/5
```

Server response

Activities Google Chrome Mon 17:46

localhost:8080/swagger-ui/index.html#/employee-resource/deleteEmpUsingDELETE

Curl

```
curl -X DELETE "http://localhost:8080/employees/5" -H "accept: */*"
```

Request URL

```
http://localhost:8080/employees/5
```

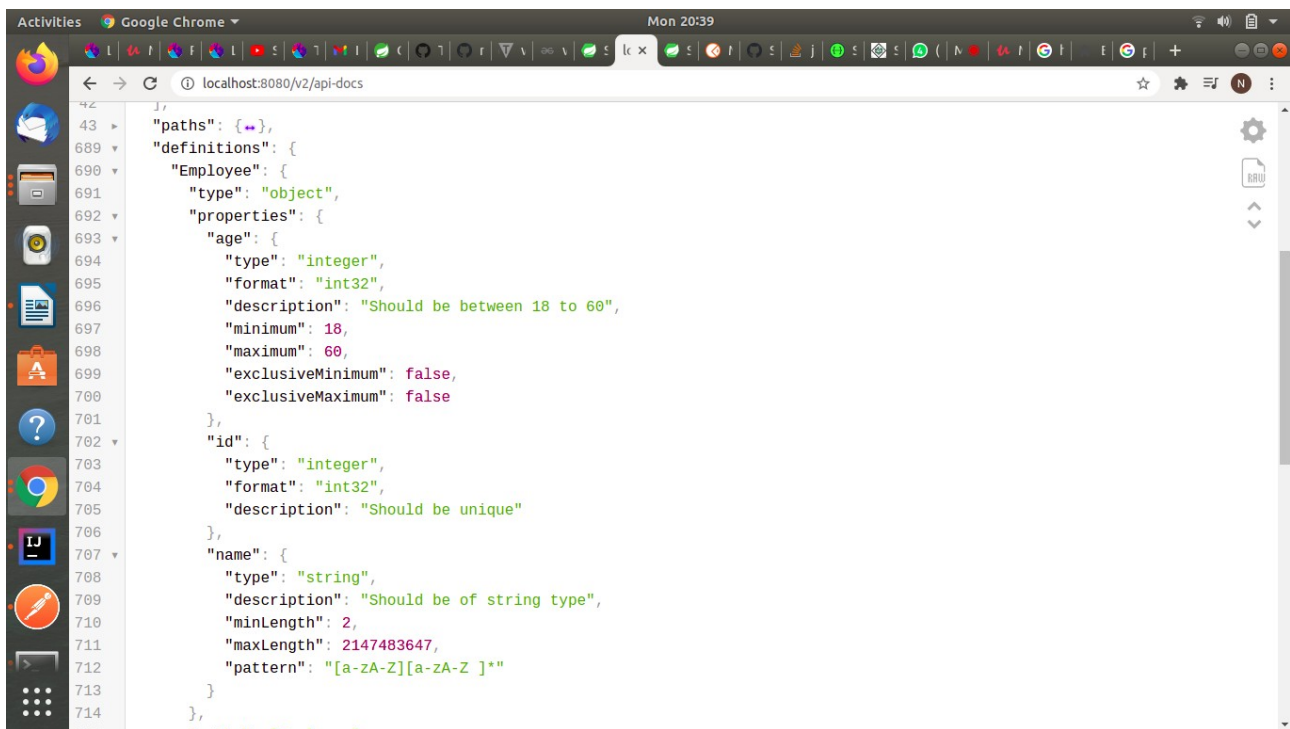
Server response

Code	Details
200	<p>Response headers</p> <pre>connection: keep-alive content-length: 0 date: Mon03 Mar 2021 12:16:15 GMT keep-alive: timeout=60</pre>

Responses

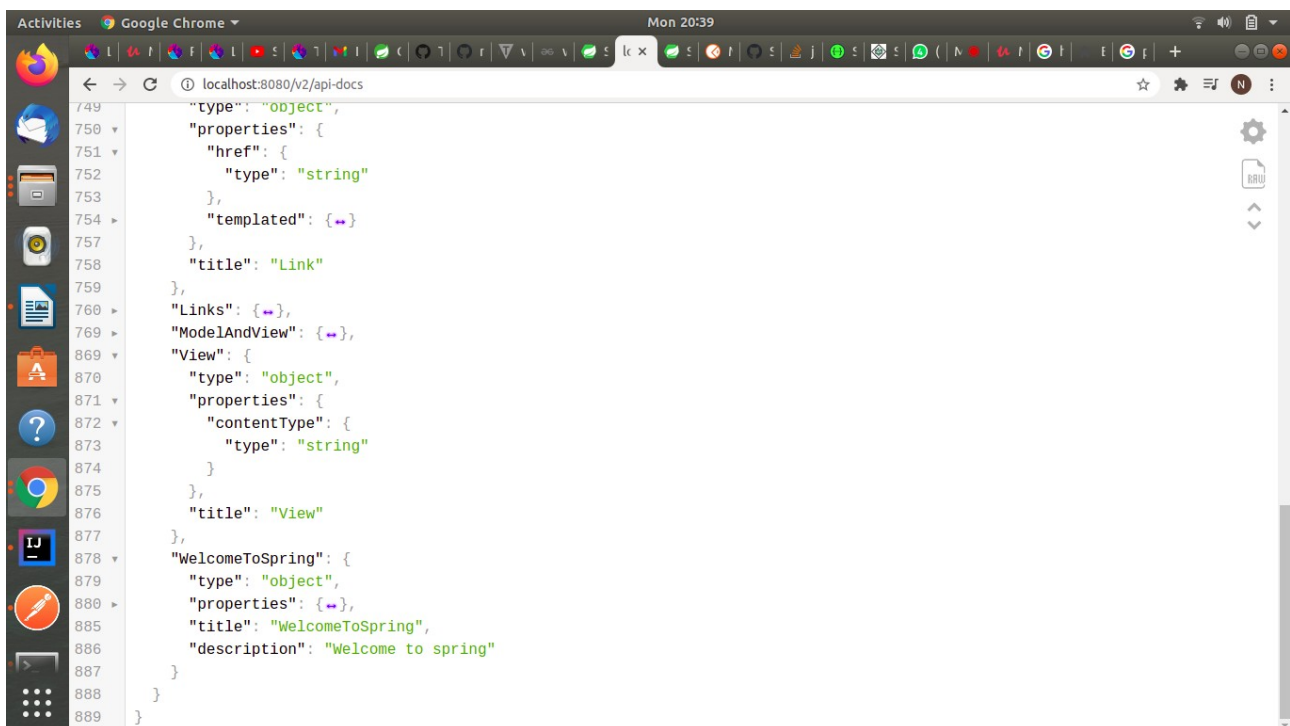
Code	Description
200	OK
204	No Content
401	Unauthorized
403	Forbidden

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.



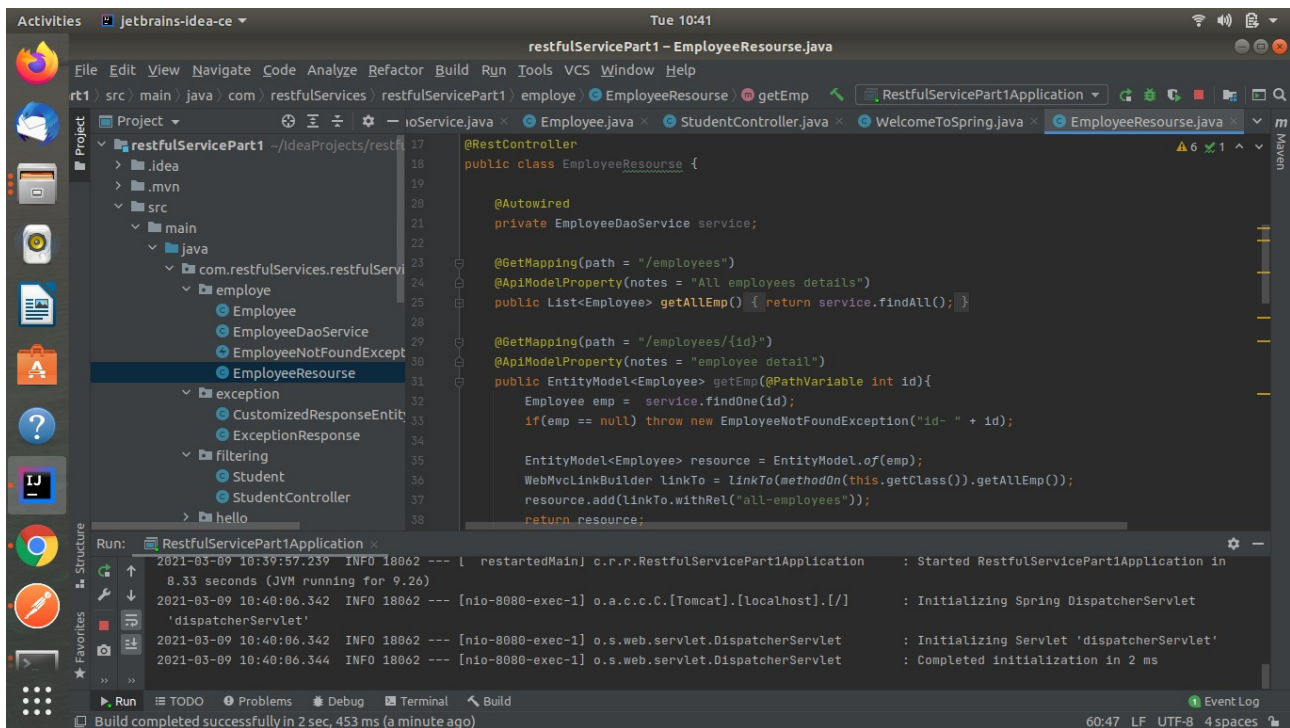
The screenshot shows a Google Chrome browser window with the address bar displaying 'localhost:8080/v2/api-docs'. The page content is a Swagger JSON specification. The 'definitions' section defines an 'Employee' class. The 'Employee' class is an object with two properties: 'age' and 'id'. The 'age' property is an integer with a minimum value of 18 and a maximum value of 60. The 'id' property is an integer that should be unique. The 'name' property is a string with a minimum length of 2 and a maximum length of 2147483647, with a pattern of '[a-zA-Z][a-zA-Z]*'.

```
742 },
743 "paths": { },
744 "definitions": {
745   "Employee": {
746     "type": "object",
747     "properties": {
748       "age": {
749         "type": "integer",
750         "format": "int32",
751         "description": "Should be between 18 to 60",
752         "minimum": 18,
753         "maximum": 60,
754         "exclusiveMinimum": false,
755         "exclusiveMaximum": false
756       },
757       "id": {
758         "type": "integer",
759         "format": "int32",
760         "description": "Should be unique"
761       },
762       "name": {
763         "type": "string",
764         "description": "Should be of string type",
765         "minLength": 2,
766         "maxLength": 2147483647,
767         "pattern": "[a-zA-Z][a-zA-Z]*"
768       }
769     }
770   }
771 },
772 }
```



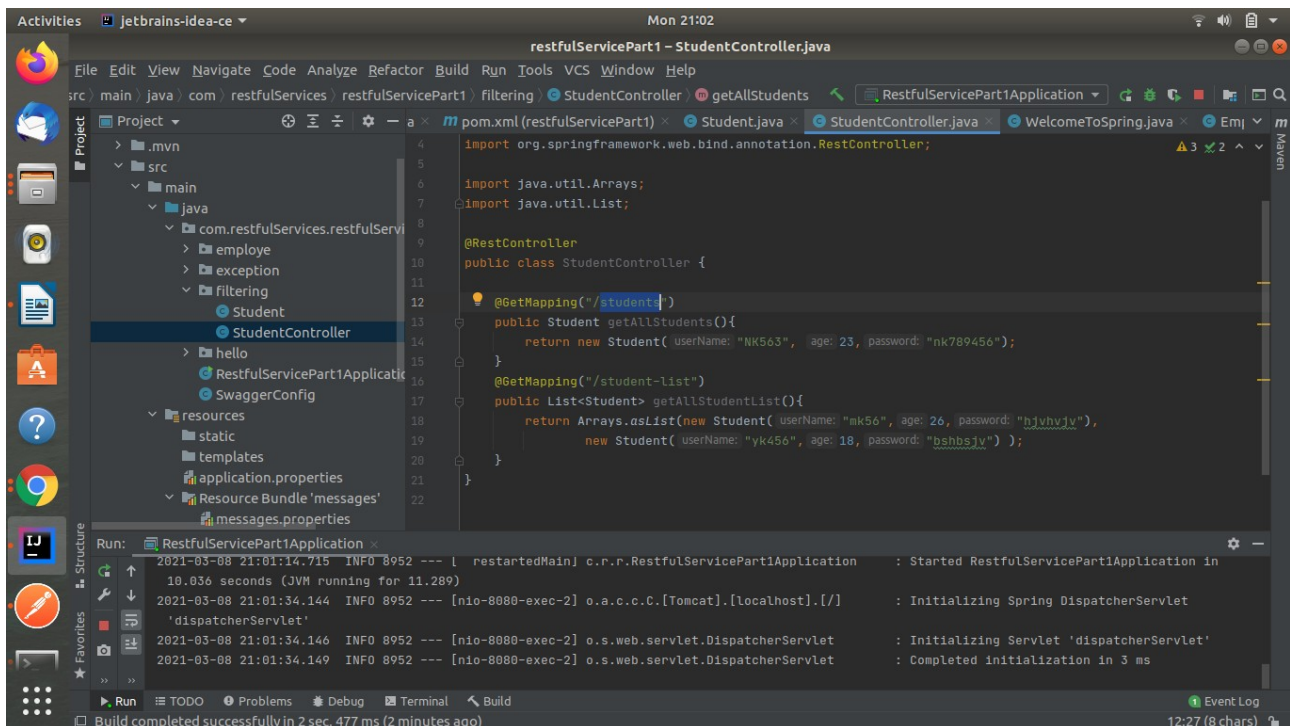
The screenshot shows a Google Chrome browser window with the address bar displaying 'localhost:8080/v2/api-docs'. The page content is a Swagger JSON specification. The 'definitions' section defines a 'WelcomeToSpring' class. The 'WelcomeToSpring' class is an object with two properties: 'contentType' and 'title'. The 'contentType' property is a string. The 'title' property is a string. The 'WelcomeToSpring' class is also associated with a 'View' class. The 'View' class is an object with two properties: 'contentType' and 'title'. The 'contentType' property is a string. The 'title' property is a string. The 'WelcomeToSpring' class is also associated with a 'Link' class. The 'Link' class is an object with two properties: 'href' and 'title'. The 'href' property is a string. The 'title' property is a string. The 'WelcomeToSpring' class is also associated with a 'ModelAndView' class. The 'ModelAndView' class is an object with two properties: 'contentType' and 'title'. The 'contentType' property is a string. The 'title' property is a string. The 'WelcomeToSpring' class is also associated with a 'Links' class. The 'Links' class is an object with two properties: 'href' and 'title'. The 'href' property is a string. The 'title' property is a string.

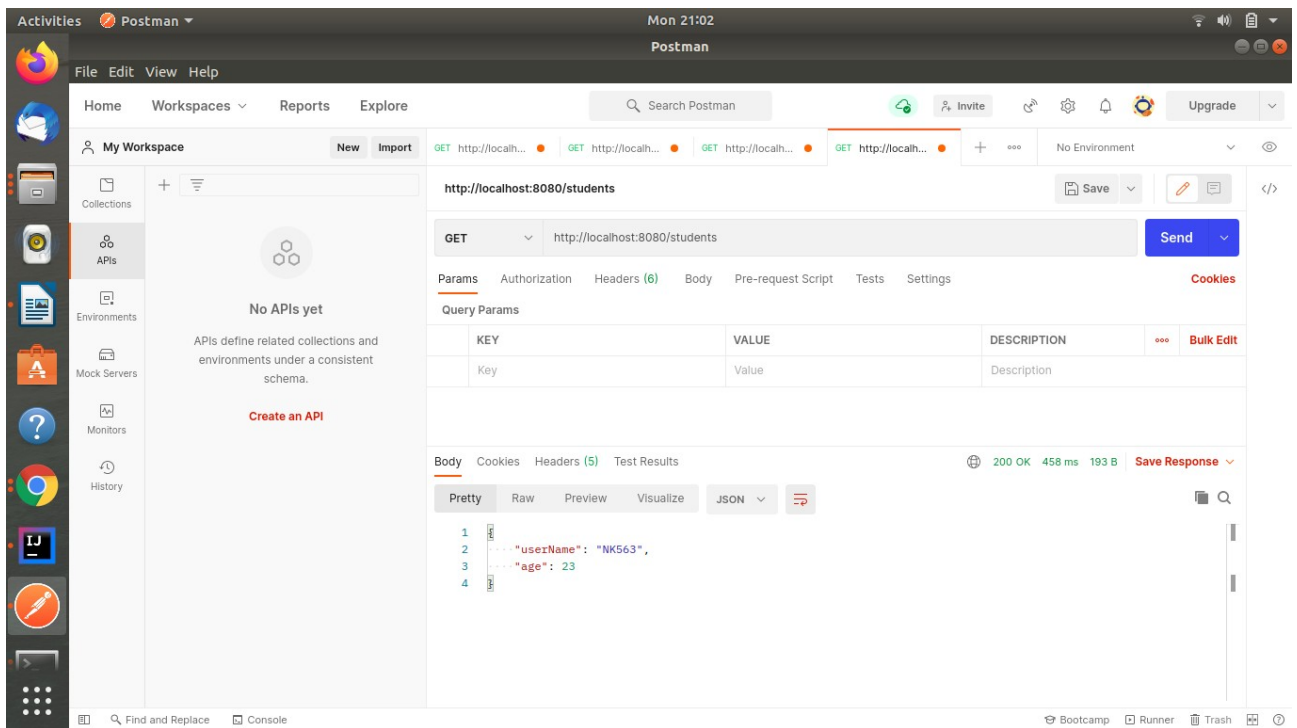
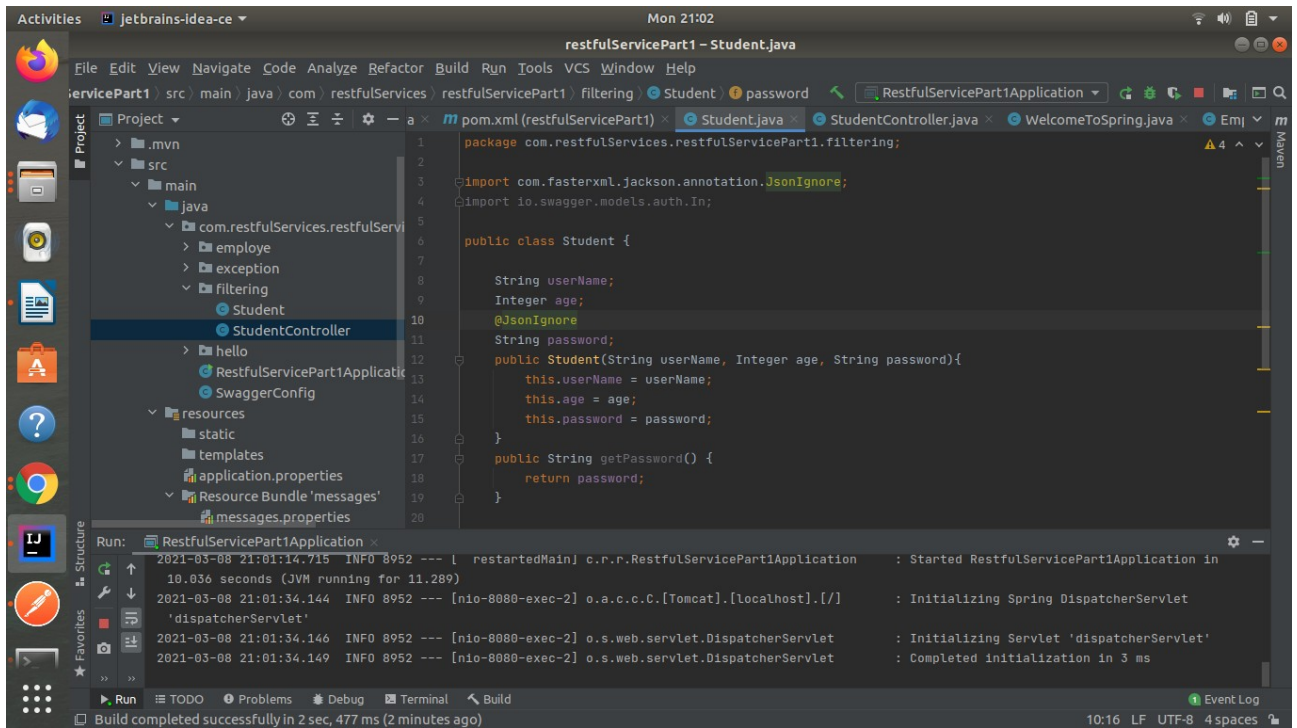
```
749 "type": "object",
750 "properties": {
751   "href": {
752     "type": "string"
753   },
754   "templated": { }
755 },
756 "title": "Link"
757 },
758 "Links": { },
759 "ModelAndView": { },
760 "View": {
761   "type": "object",
762   "properties": {
763     "contentType": {
764       "type": "string"
765     }
766   },
767   "title": "View"
768 },
769 "WelcomeToSpring": {
770   "type": "object",
771   "properties": {
772     "contentType": {
773       "type": "string"
774     },
775     "title": "WelcomeToSpring",
776     "description": "Welcome to spring"
777   }
778 },
779 }
```

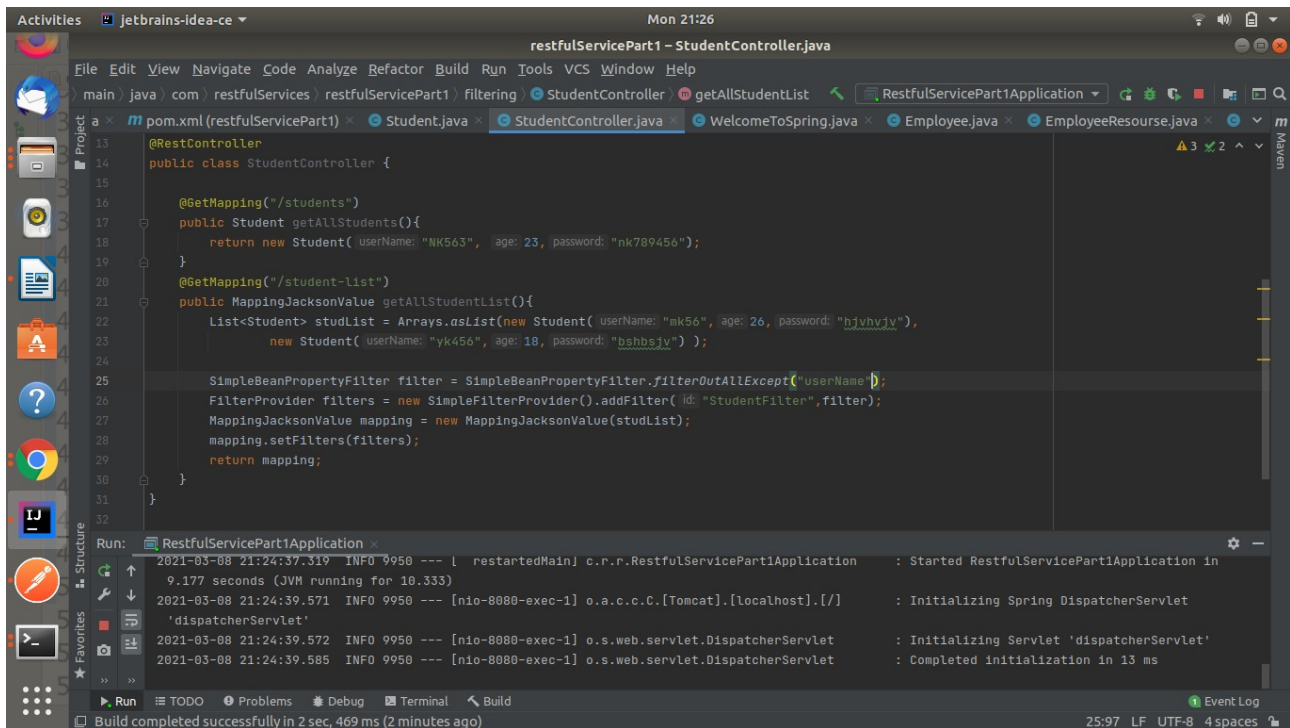
*Static and Dynamic filtering

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





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

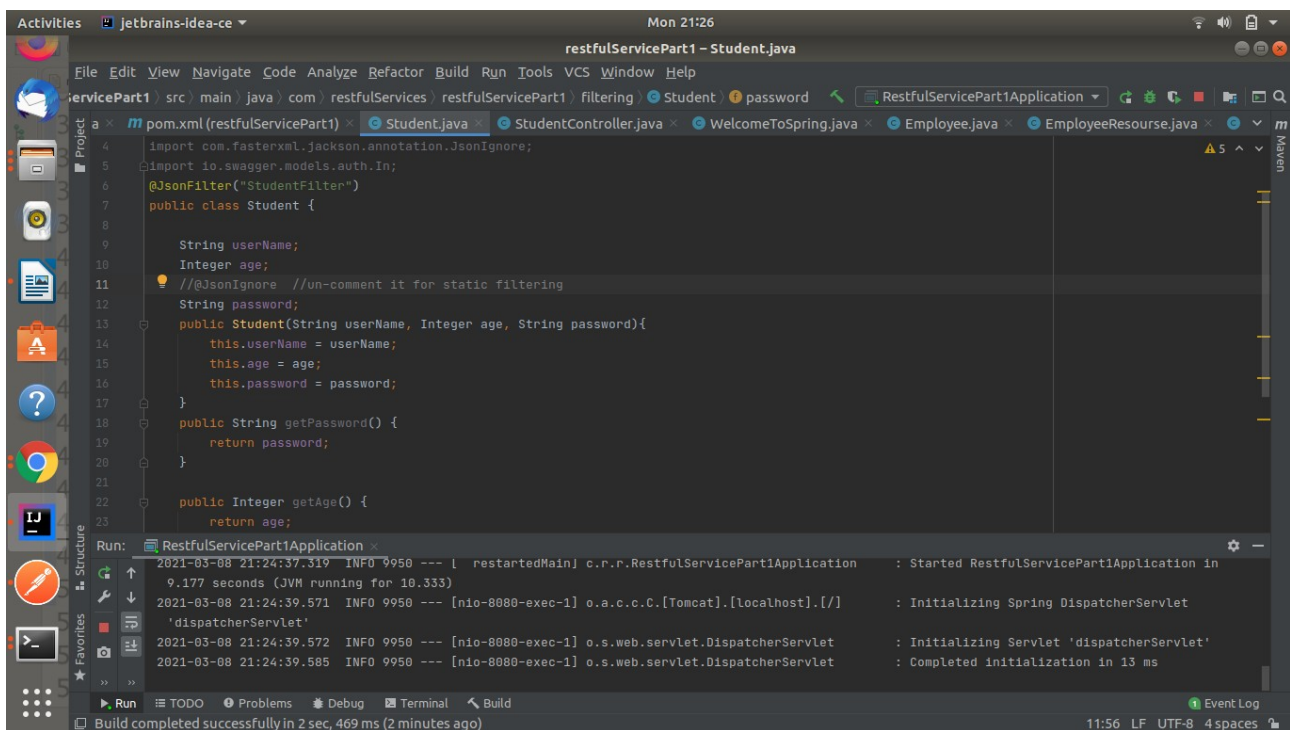


The screenshot shows the IntelliJ IDEA IDE with the `restfulServicePart1 - StudentController.java` file open. The code defines a `RestController` with two endpoints: `getAllStudents()` and `getAllStudentList()`. The `getAllStudentList()` method uses dynamic filtering to exclude the `password` field from the JSON response.

```
13 @RestController
14 public class StudentController {
15
16     @GetMapping("/students")
17     public Student getAllStudents(){
18         return new Student( userName: "NK563", age: 23, password: "nk789456");
19     }
20
21     @GetMapping("/student-list")
22     public MappingJacksonValue getAllStudentList(){
23         List<Student> studList = Arrays.asList(new Student( userName: "mk56", age: 26, password: "hjhvhvjv"),
24             new Student( userName: "yk456", age: 18, password: "bshbsjv" ));
25
26         SimpleBeanPropertyFilter filter = SimpleBeanPropertyFilter.filterOutAllExcept("userName");
27         FilterProvider filters = new SimpleFilterProvider().addFilter( id: "StudentFilter",filter);
28         MappingJacksonValue mapping = new MappingJacksonValue(studList);
29         mapping.setFilters(filters);
30         return mapping;
31     }
32 }
```

The Run console shows the application starting successfully:

```
2021-03-08 21:24:37.319 INFO 9950 --- [ restartedMain] c.r.r.RestfulServicePart1Application : Started RestfulServicePart1Application in
9.177 seconds (JVM running for 10.333)
2021-03-08 21:24:39.571 INFO 9950 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet
'dispatcherServlet'
2021-03-08 21:24:39.572 INFO 9950 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2021-03-08 21:24:39.585 INFO 9950 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 13 ms
```

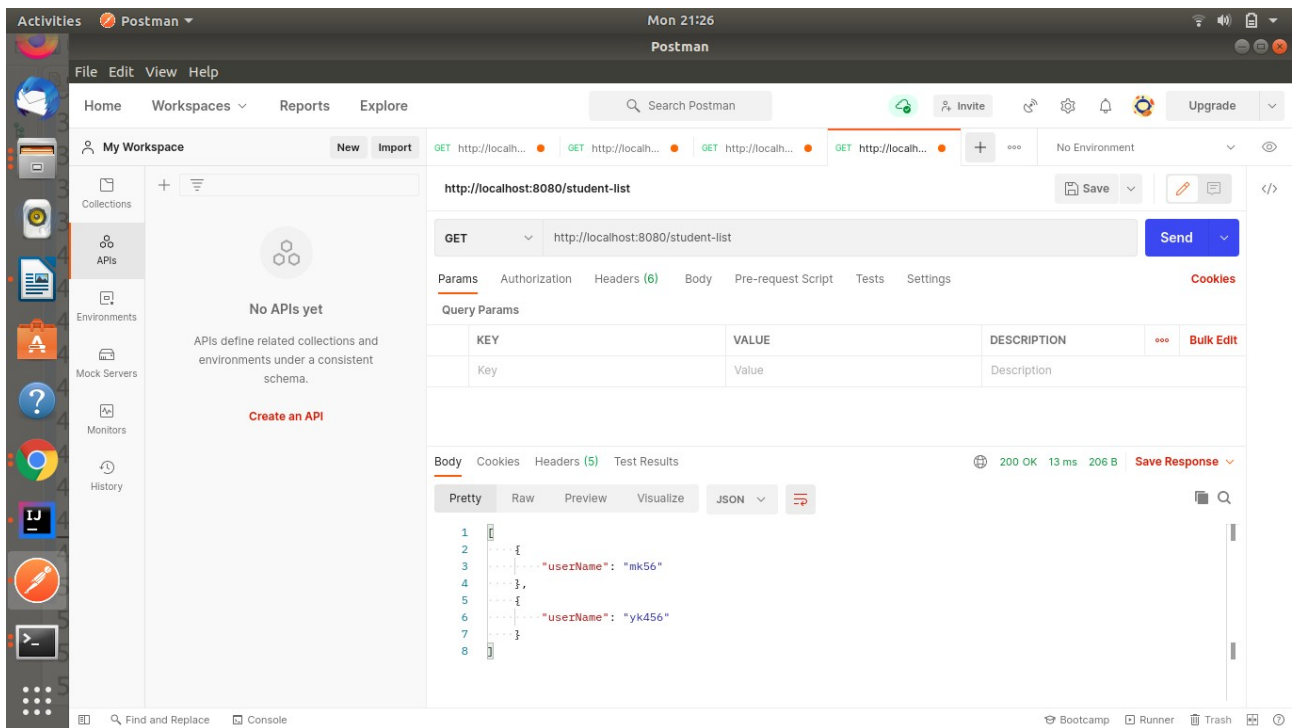


The screenshot shows the IntelliJ IDEA IDE with the `restfulServicePart1 - Student.java` file open. The code defines the `Student` class with fields `userName`, `age`, and `password`. The `password` field is annotated with `@JsonIgnore` to exclude it from the JSON response.

```
4 import com.fasterxml.jackson.annotation.JsonIgnore;
5 import io.swagger.models.auth.In;
6 @JsonIgnore("StudentFilter")
7 public class Student {
8
9     String userName;
10     Integer age;
11     // @JsonIgnore //un-comment it for static filtering
12     String password;
13     public Student(String userName, Integer age, String password){
14         this.userName = userName;
15         this.age = age;
16         this.password = password;
17     }
18     public String getPassword() {
19         return password;
20     }
21
22     public Integer getAge() {
23         return age;
24     }
25 }
```

The Run console shows the application starting successfully:

```
2021-03-08 21:24:37.319 INFO 9950 --- [ restartedMain] c.r.r.RestfulServicePart1Application : Started RestfulServicePart1Application in
9.177 seconds (JVM running for 10.333)
2021-03-08 21:24:39.571 INFO 9950 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring DispatcherServlet
'dispatcherServlet'
2021-03-08 21:24:39.572 INFO 9950 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Initializing Servlet 'dispatcherServlet'
2021-03-08 21:24:39.585 INFO 9950 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : Completed initialization in 13 ms
```



*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:

- MimeType Versioning
- Request Parameter versioning
- URI versioning
- Custom Header Versioning

Activities Postman Mon 23:47

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace New Import

GET http://localhost:8080/person/param?version=1

Save Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
<input checked="" type="checkbox"/> version	1		
Key	Value	Description	

Body Cookies Headers (5) Test Results 200 OK 294 ms 180 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": "Nitin"
3 }
```

Find and Replace Console Bootcamp Runner Trash

Activities Postman Mon 23:47

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace New Import

GET http://localhost:8080/person/param?version=2

Save Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

Query Params

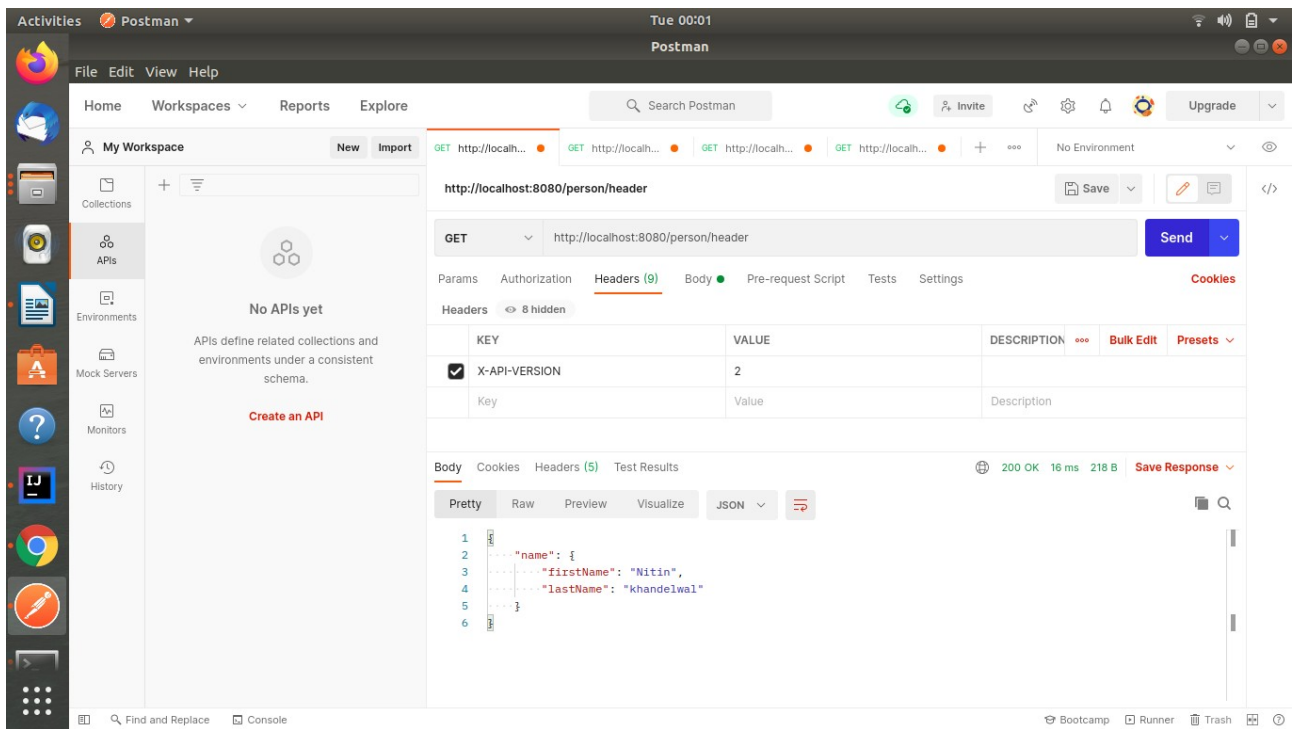
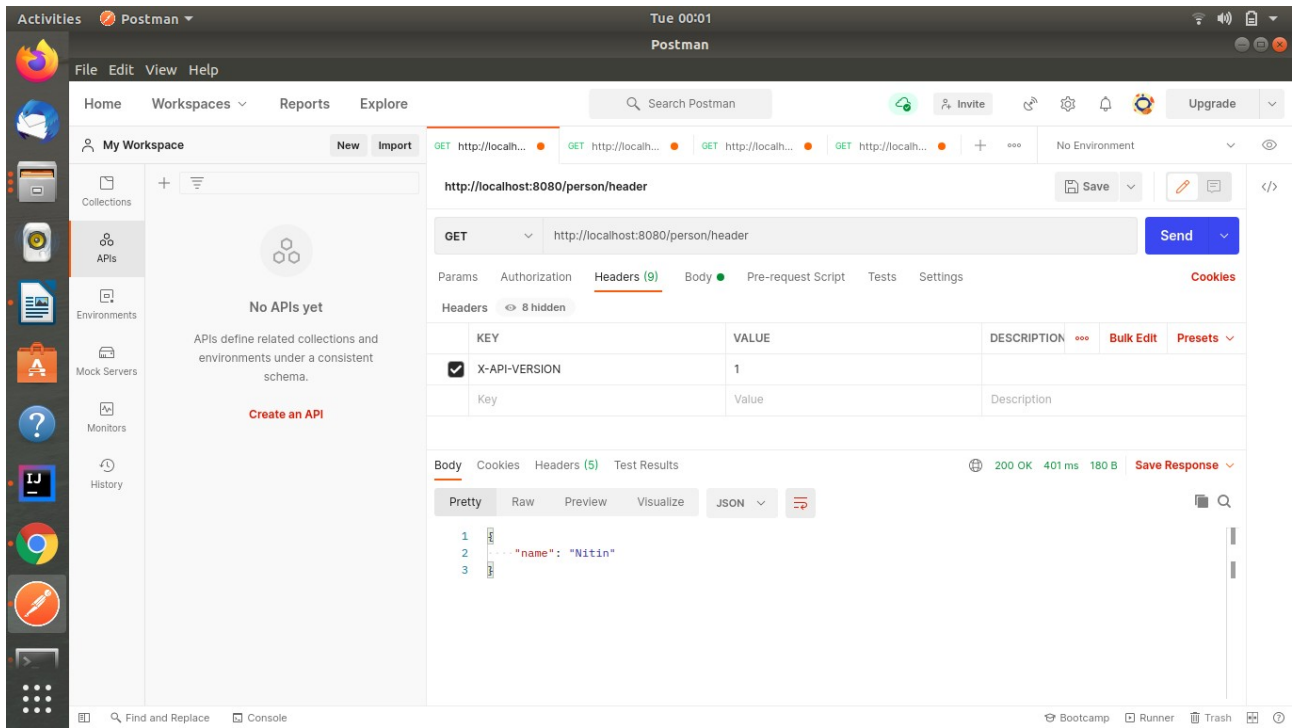
KEY	VALUE	DESCRIPTION	Bulk Edit
<input checked="" type="checkbox"/> version	2		
Key	Value	Description	

Body Cookies Headers (5) Test Results 200 OK 19 ms 218 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": {
3     "firstName": "Nitin",
4     "lastName": "khandelwal"
5   }
6 }
```

Find and Replace Console Bootcamp Runner Trash



Activities Postman Tue 00:37

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace New Import

GET http://localhost:8080/person/produces?Accept=application/vnd.company.app-v1+json

GET http://localhost:8080/person/produces?Accept=application/vnd.company.app-v1+json

Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

KEY	VALUE	DESCRIPTION	Bulk Edit
<input checked="" type="checkbox"/> Accept	application/vnd.company.app-v1+json		
Key	Value	Description	

Body Cookies Headers (5) Test Results 200 OK 11 ms 199 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": "Nitin"
3 }
```

Find and Replace Console Bootcamp Runner Trash

Activities Postman Tue 00:44

File Edit View Help

Home Workspaces Reports Explore

Search Postman

My Workspace New Import

GET http://localhost:8080/person/produces

GET http://localhost:8080/person/produces

Send

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

Headers 6 hidden

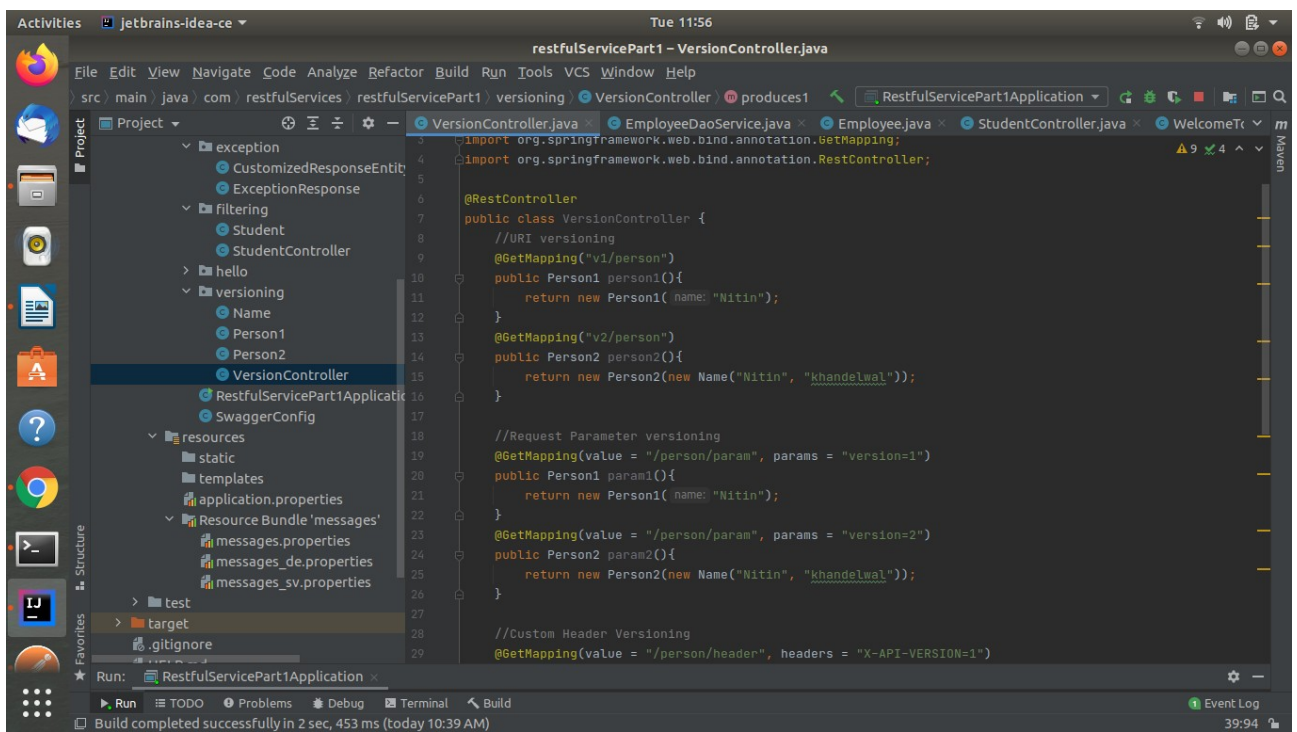
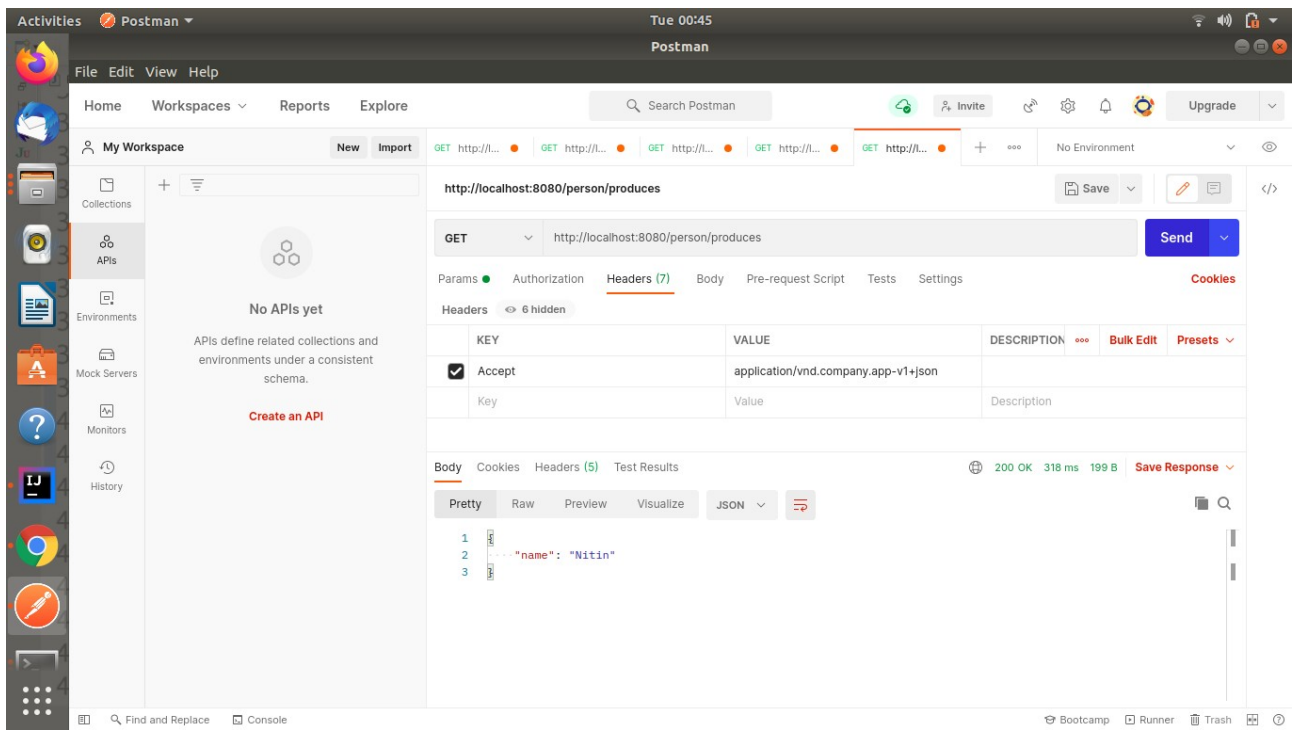
KEY	VALUE	DESCRIPTION	Bulk Edit	Presets
<input checked="" type="checkbox"/> Accept	application/vnd.company.app-v2+json			
Key	Value	Description		

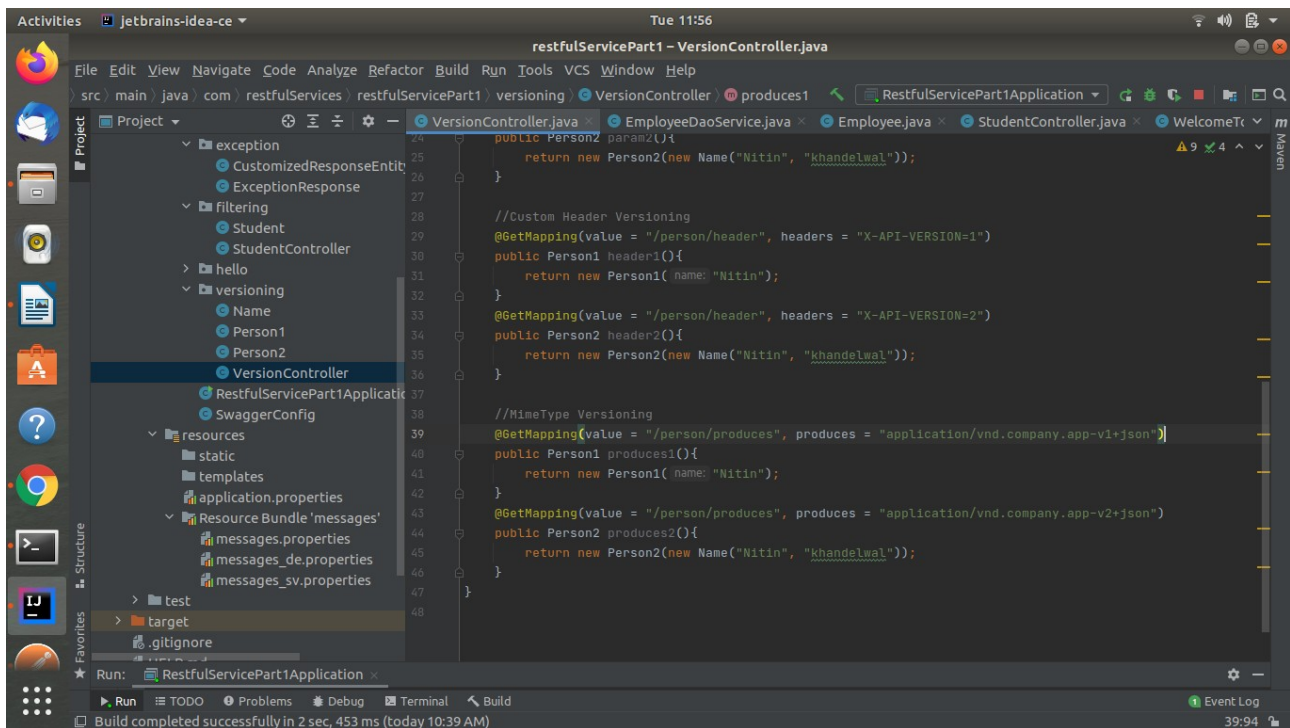
Body Cookies Headers (5) Test Results 200 OK 267 ms 237 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "name": {
3     "firstName": "Nitin",
4     "lastName": "khandelwal"
5   }
6 }
```

Find and Replace Console Bootcamp Runner Trash





*HATEOAS

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

