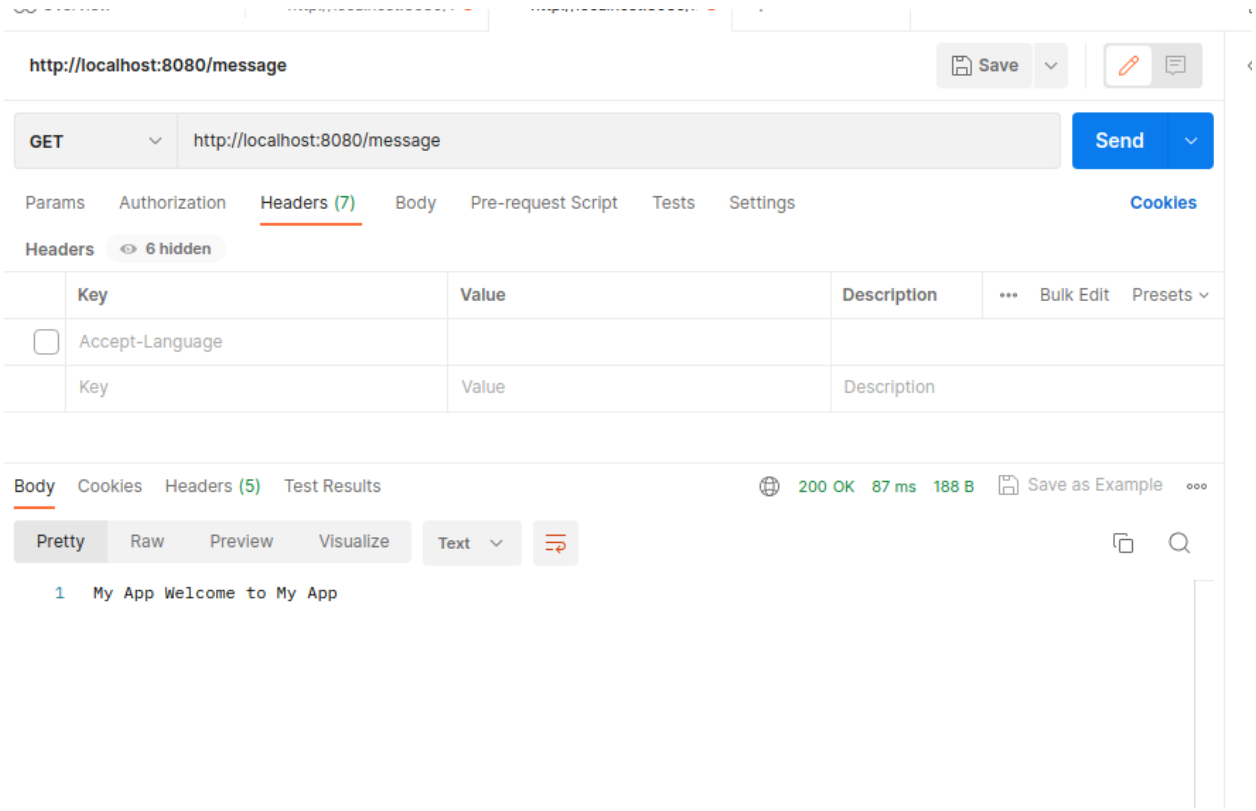
Restful WebServices Part 2

*Internationalization

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



http://localhost:8080/message

Save

GET http://localhost:8080/message

Send

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

Headers 6 hidden

Key	Value	Description	Bulk Edit Presets
Accept-Language	de		
Key	Value	Description	

Body Cookies Headers (5) Test Results 200 OK 9 ms 198 B Save as Example

Pretty Raw Preview Visualize Text

```
1  Meine App Willkommen bei Meine App
```

http://localhost:8080/message

Save

GET http://localhost:8080/message

Send

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

Headers 6 hidden

Key	Value	Description	Bulk Edit Presets
Accept-Language	sv		
Key	Value	Description	

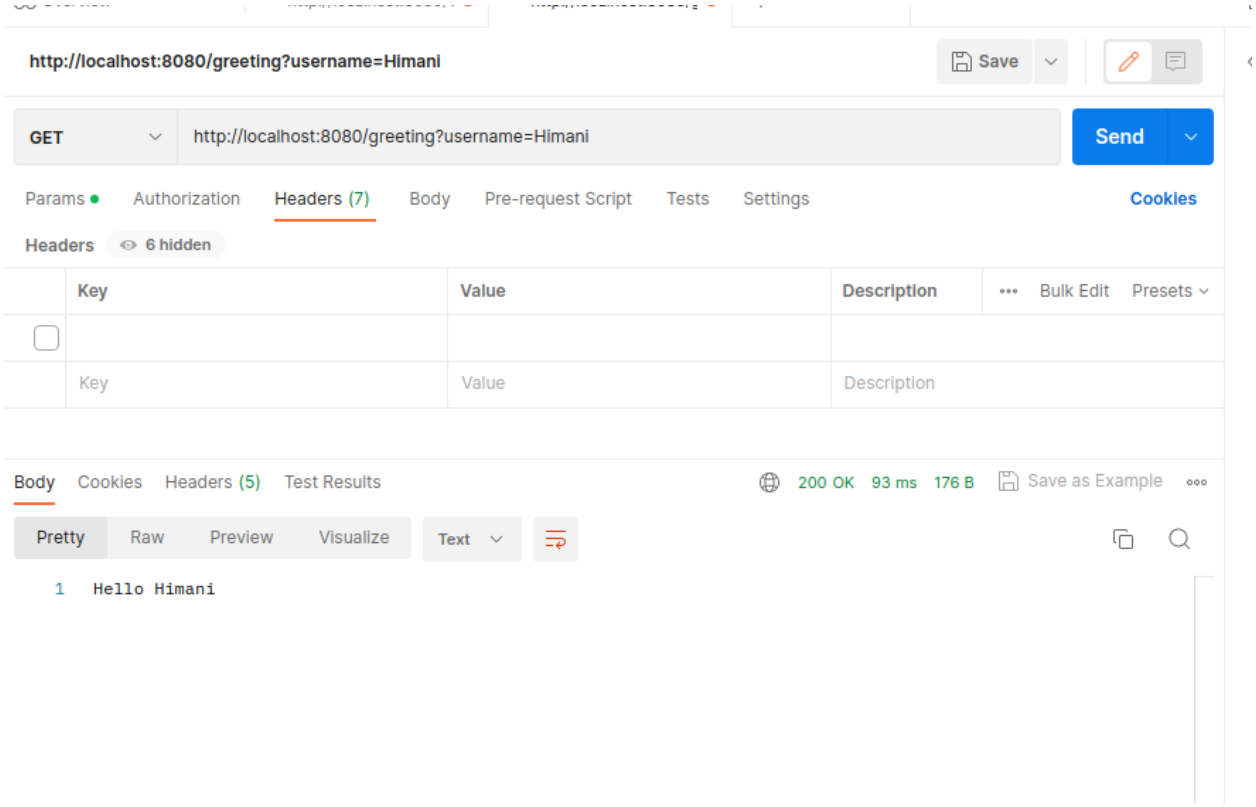
Body Cookies Headers (5) Test Results 200 OK 124 ms 194 B Save as Example

Pretty Raw Preview Visualize Text

```
1  Min App Valkommen till Min App
```

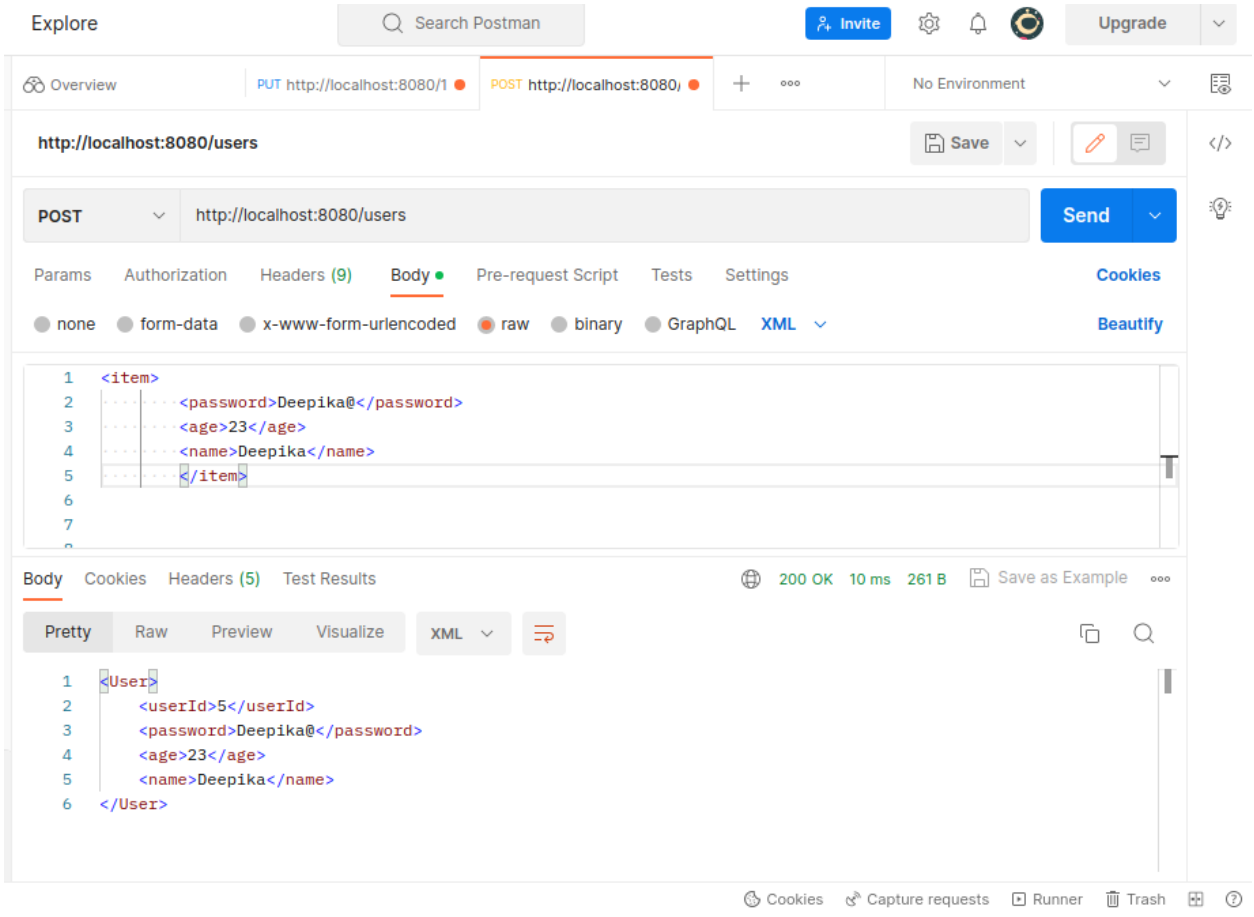
2.

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

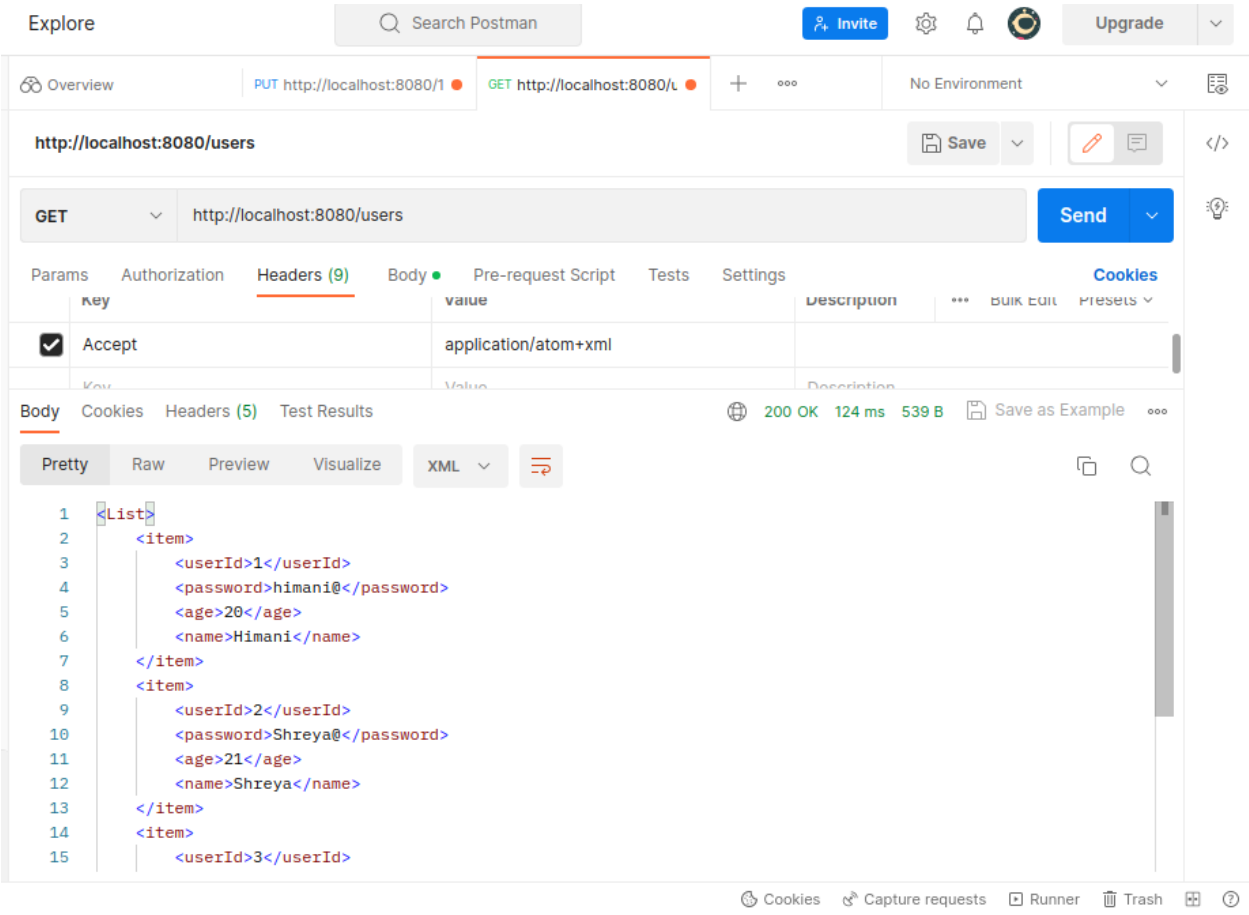


*Content Negotiation

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.



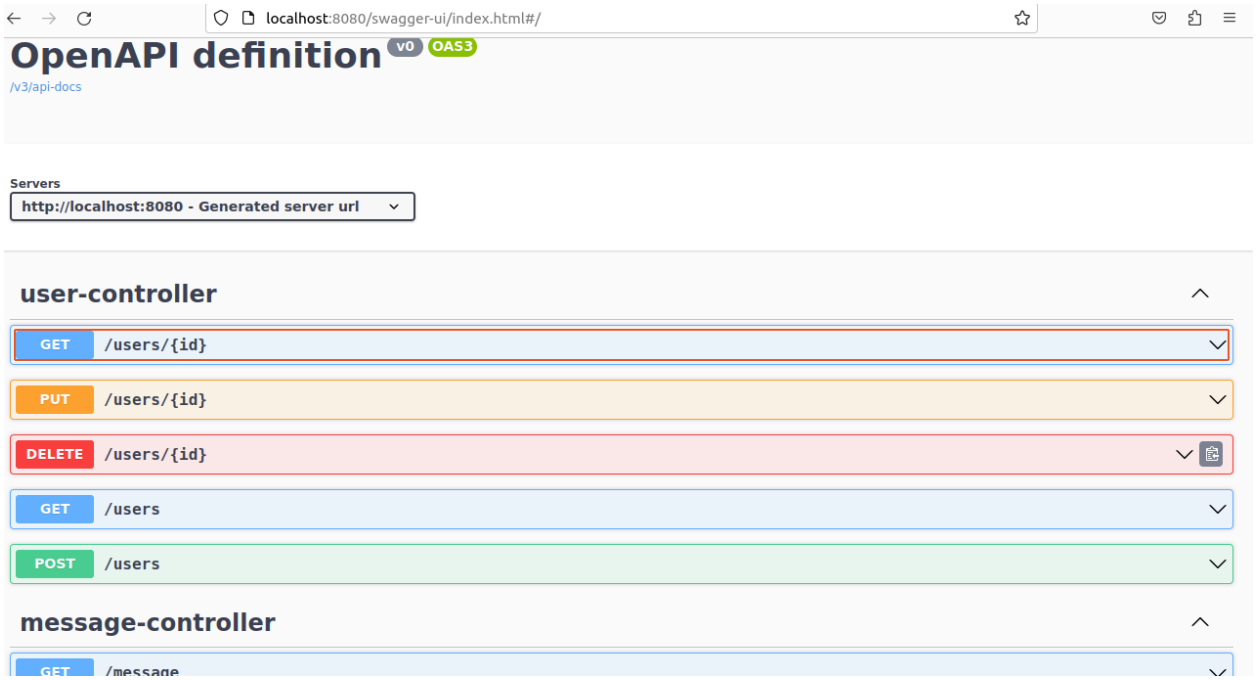
*Swagger

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

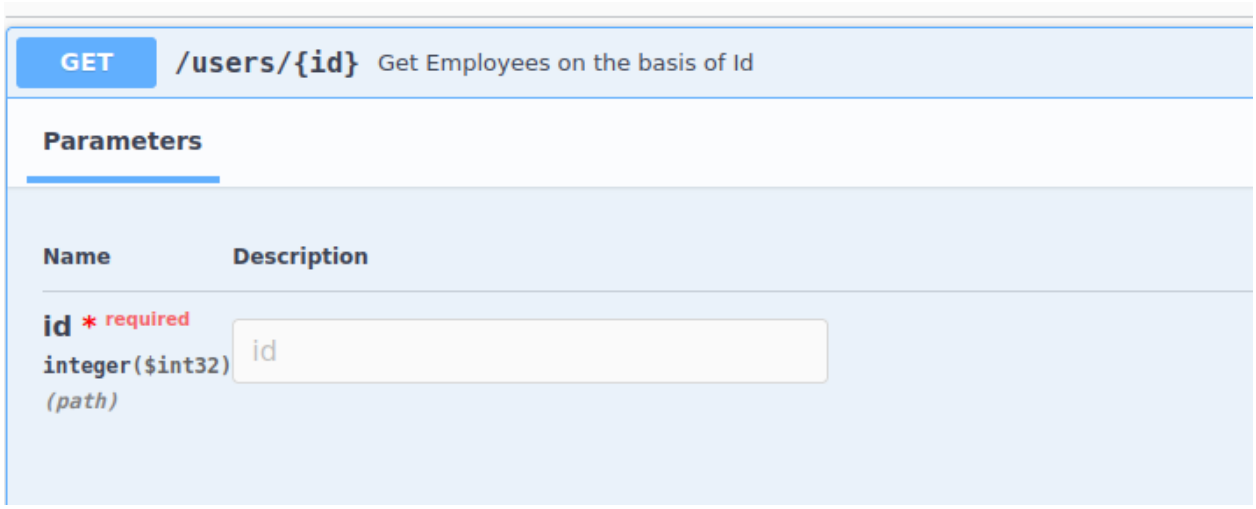
   Get details of User using GET request.

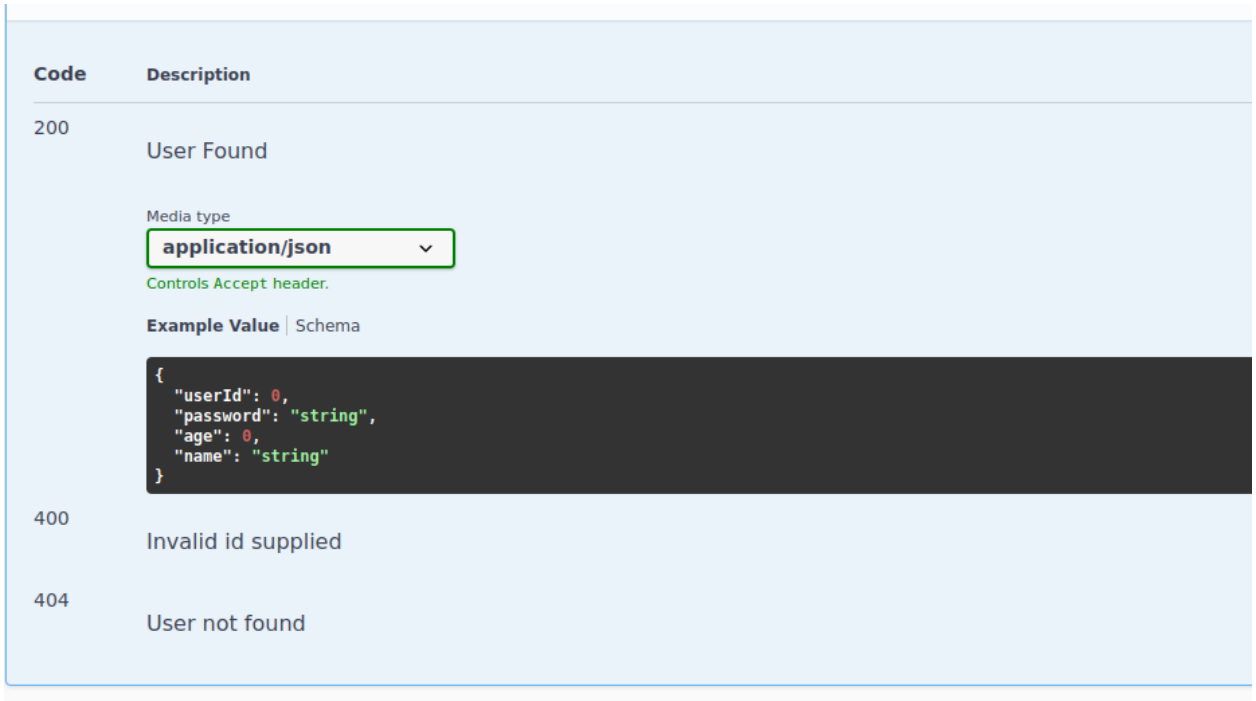
   Save details of the user using POST request.

   Delete a user using DELETE request.



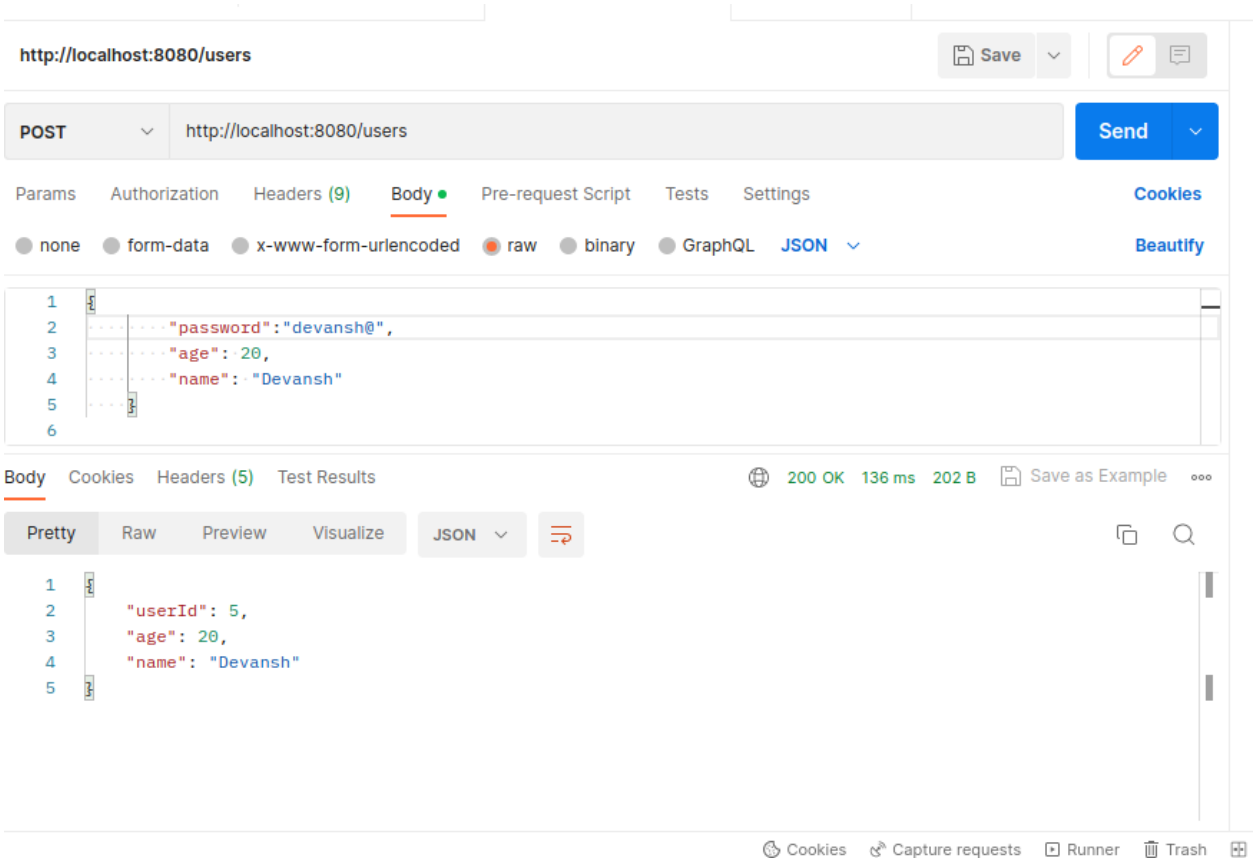
6. In swagger documentation, add the description of each class and URI so that in swagger UI the purpose of class and URI is clear.



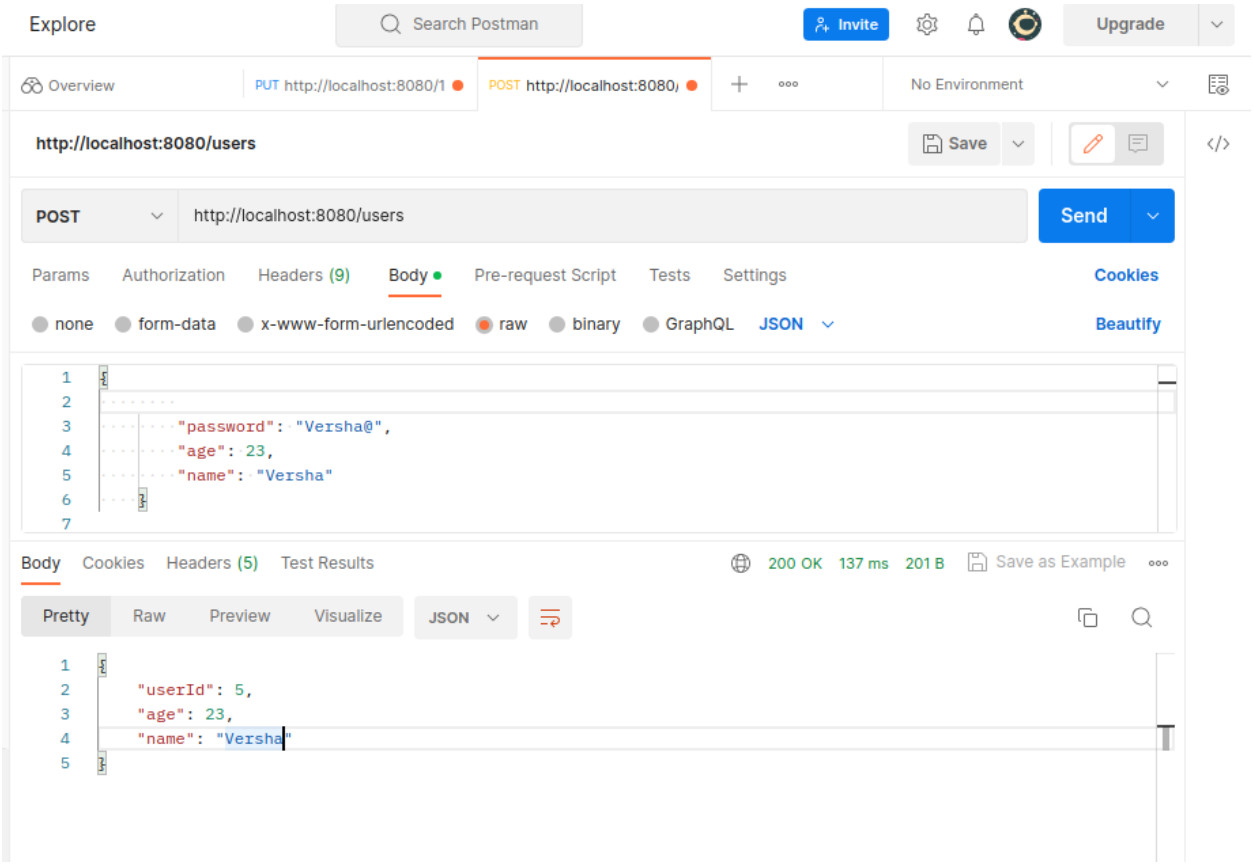


*Static and Dynamic filtering

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



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

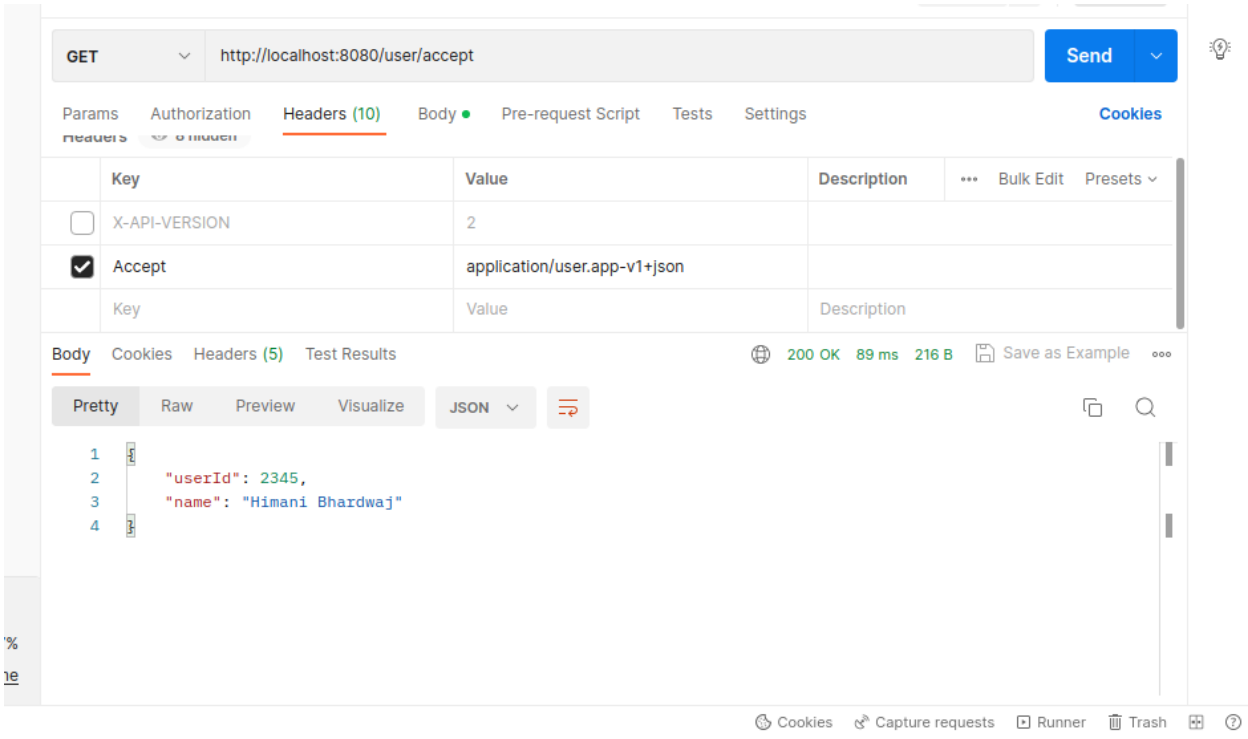


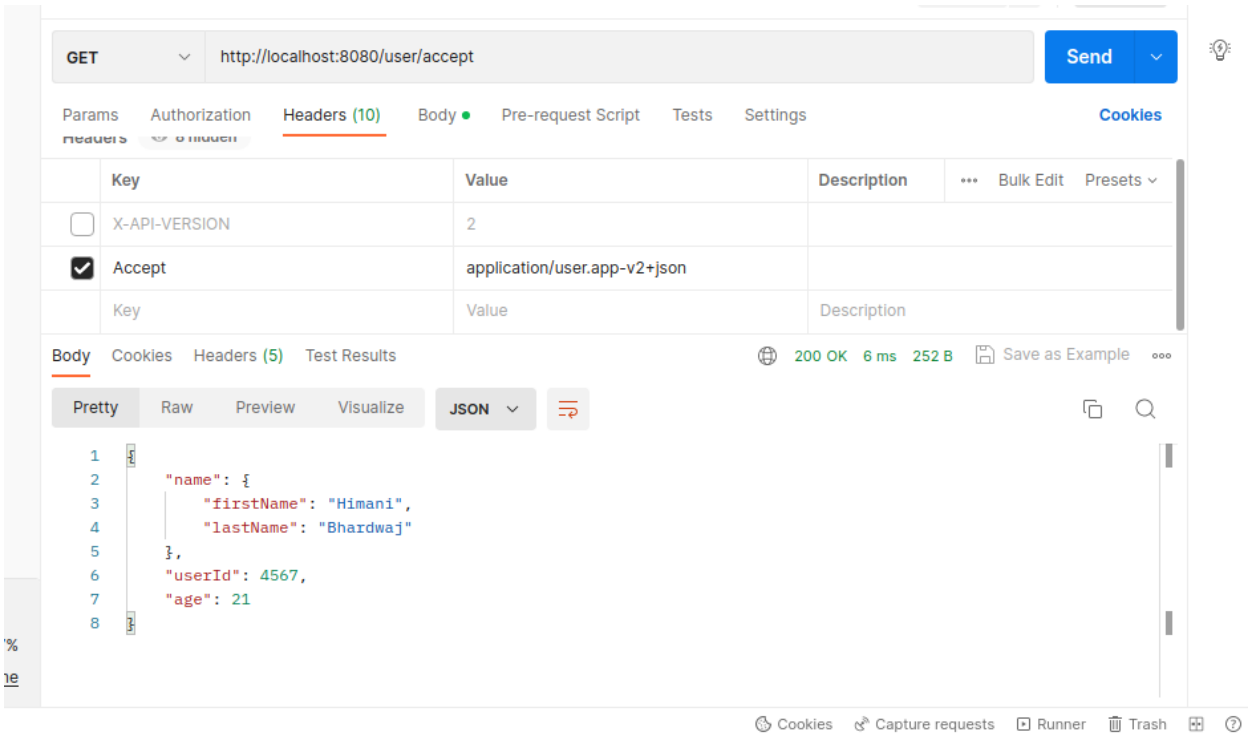
*Versioning Restful APIs

9. 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,

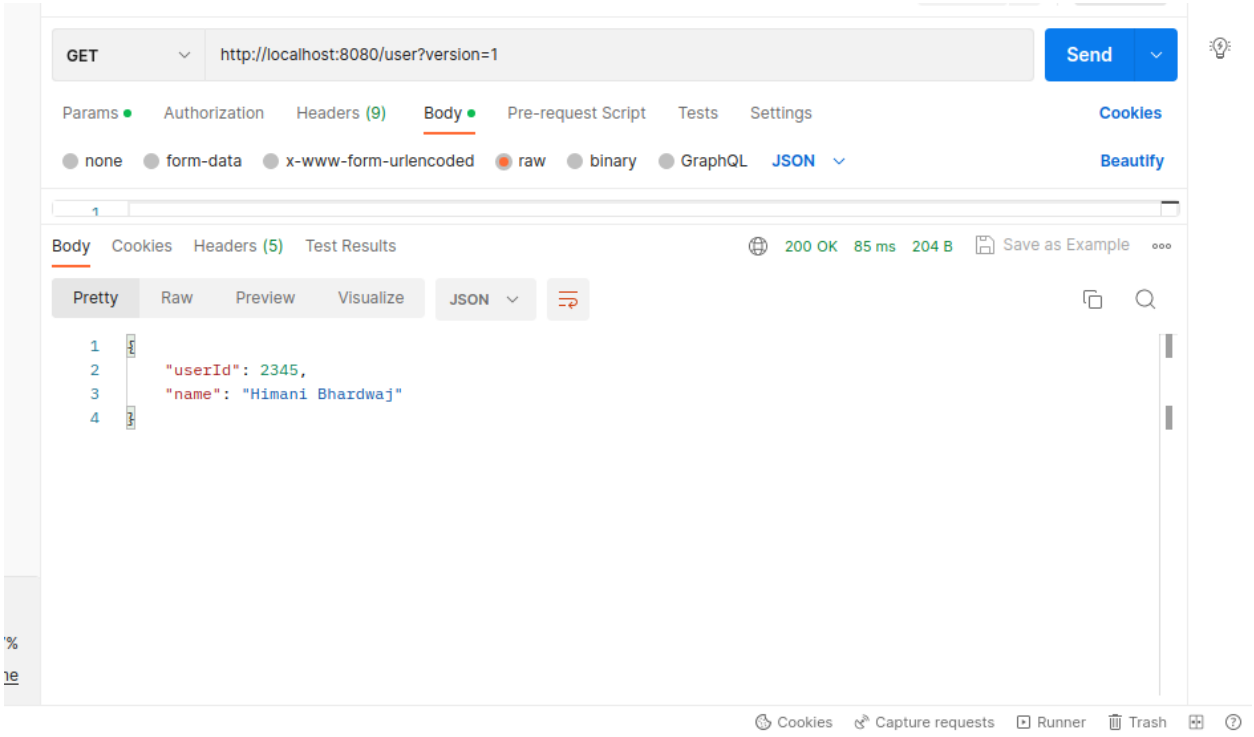
Now apply versioning using the following methods:

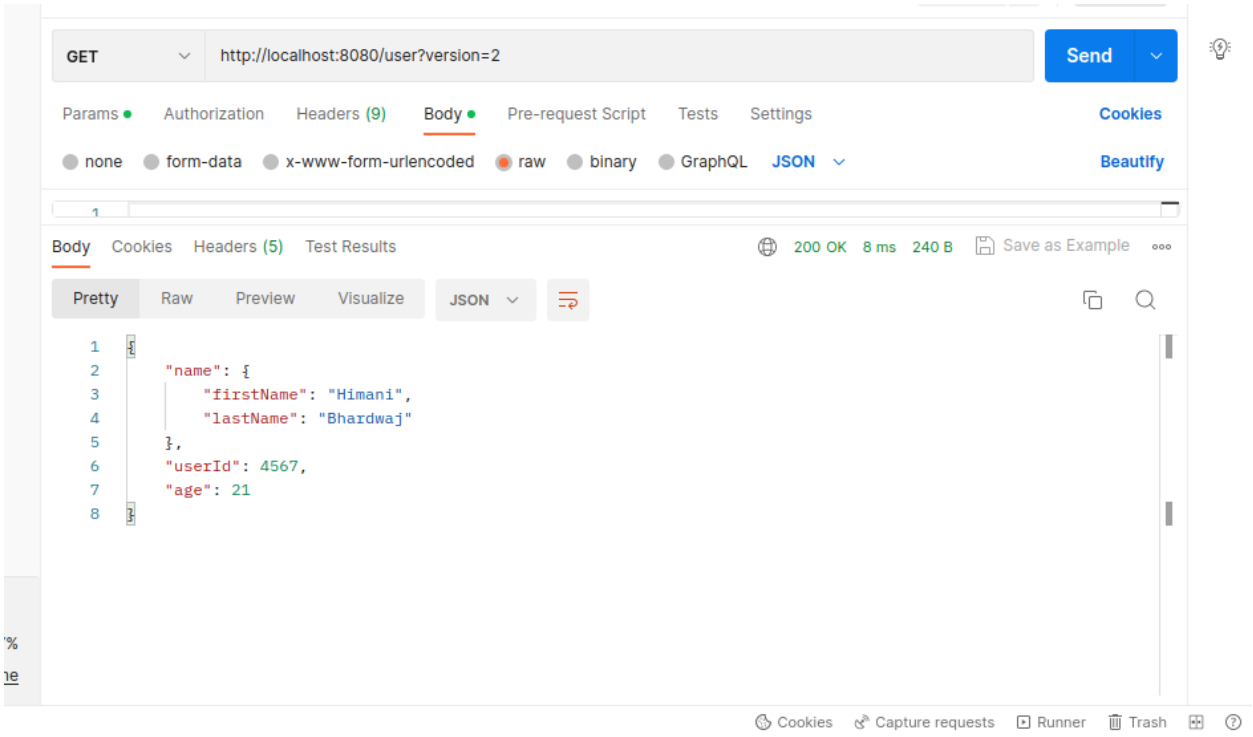
- Media Type Versioning


GET
http://localhost:8080/user/accept
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
Key
Value
Description
Bulk Edit
Presets
X-API-VERSION
2
Accept
application/user.app-v1+json
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
89 ms
216 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"userId": 2345,
"name": "Himani Bhardwaj"
}
Cookies
Capture requests
Runner
Trash

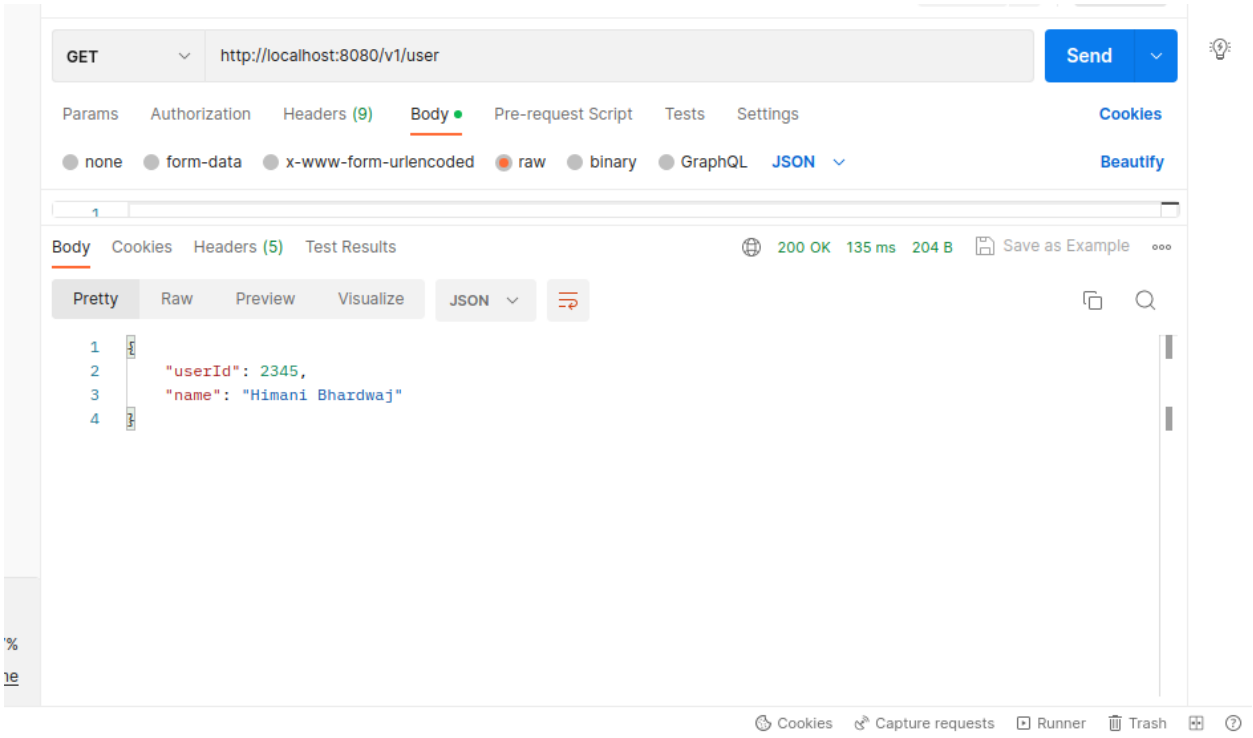

GET
http://localhost:8080/user/accept
Send
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Cookies
Key
Value
Description
Bulk Edit
Presets
X-API-VERSION
2
Accept
application/user.app-v2+json
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
6 ms
252 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"name": {
"firstName": "Himani",
"lastName": "Bhardwaj"
},
"userId": 4567,
"age": 21
}
Cookies
Capture requests
Runner
Trash

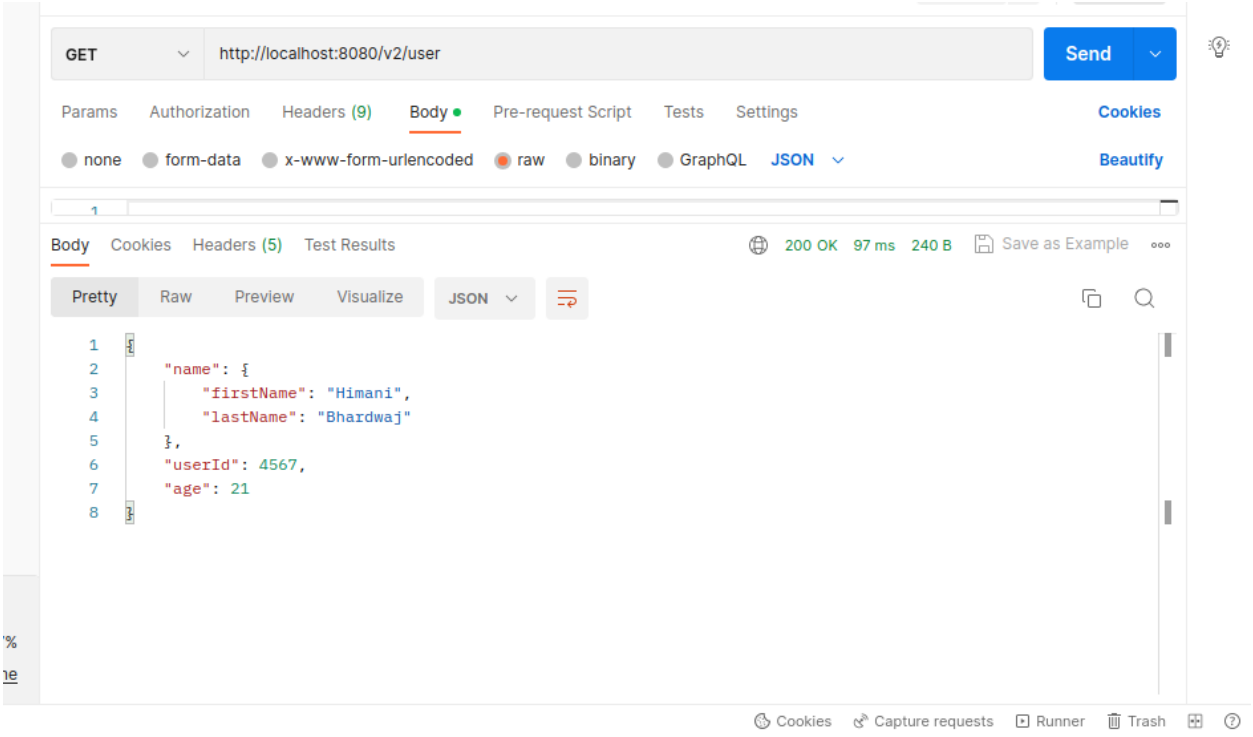
- Request Parameter versioning



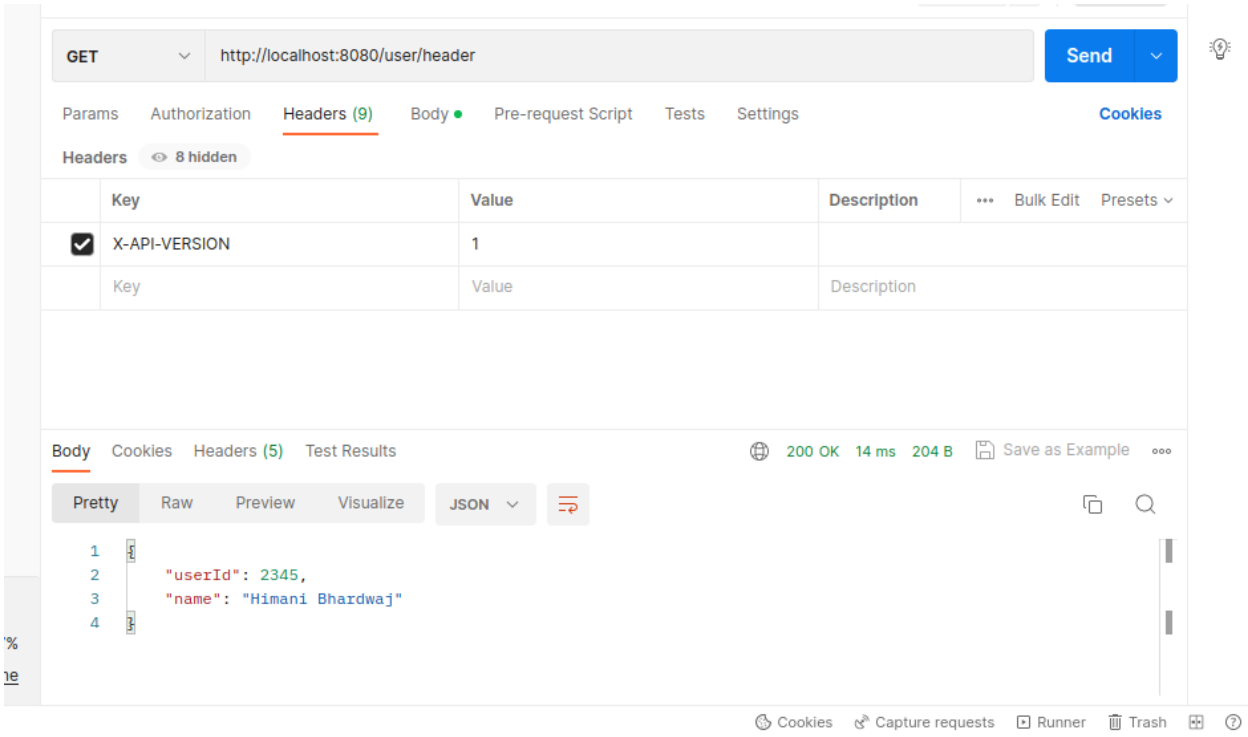


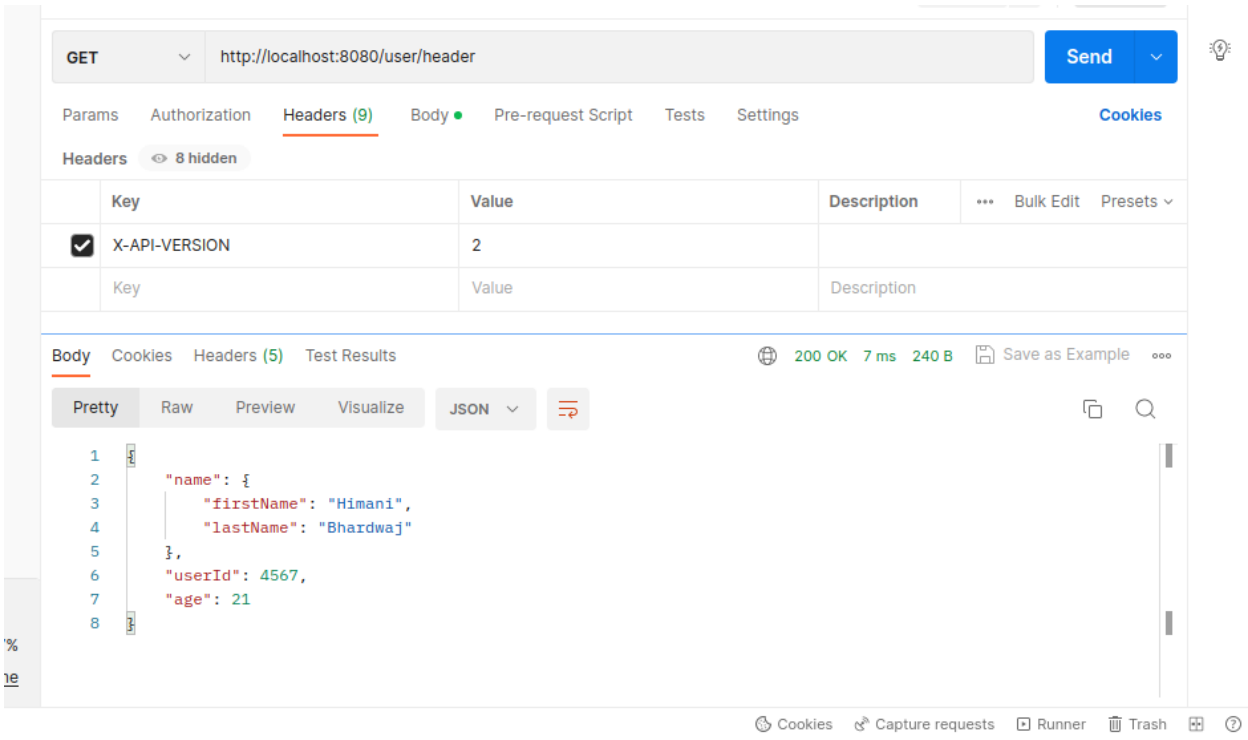
- URI versioning





- Custom Header Versioning


GET
http://localhost:8080/user/header
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
8 hidden
Key
Value
Description
Bulk Edit
Presets
X-API-VERSION
1
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
14 ms
204 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"userId": 2345,
"name": "Himani Bhardwaj"
}
Cookies
Capture requests
Runner
Trash


GET
http://localhost:8080/user/header
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
Headers
8 hidden
Key
Value
Description
Bulk Edit
Presets
X-API-VERSION
2
Key
Value
Description
Body
Cookies
Headers (5)
Test Results
200 OK
7 ms
240 B
Save as Example
Pretty
Raw
Preview
Visualize
JSON
{
"name": {
"firstName": "Himani",
"lastName": "Bhardwaj"
},
"userId": 4567,
"age": 21
}
Cookies
Capture requests
Runner
Trash

*HATEOAS

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