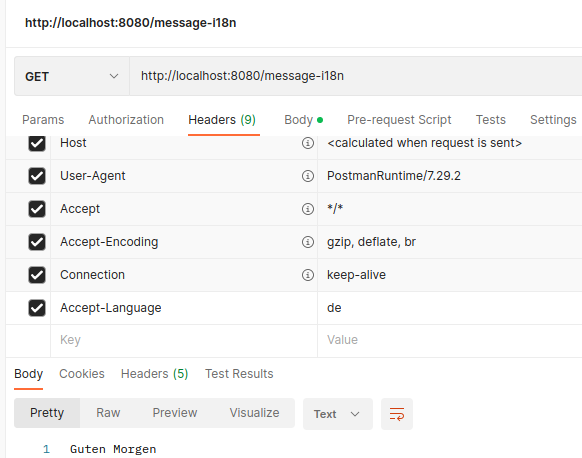
Exercise: RestFul Web Service Part 2

\*Internationalization

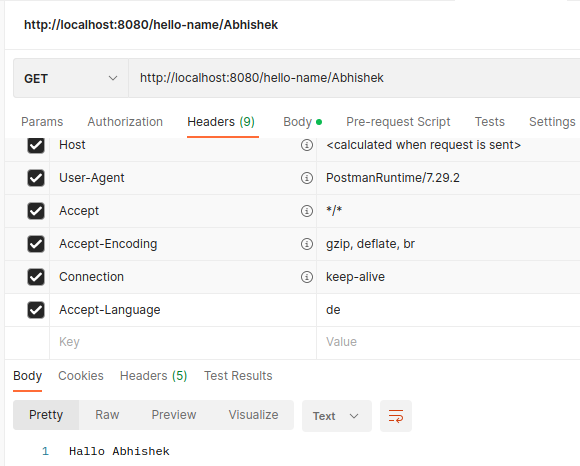
1. Add support for Internationalization in your application allowing messages to be shown in English, German and Swedish, keeping English as default.

Answer:



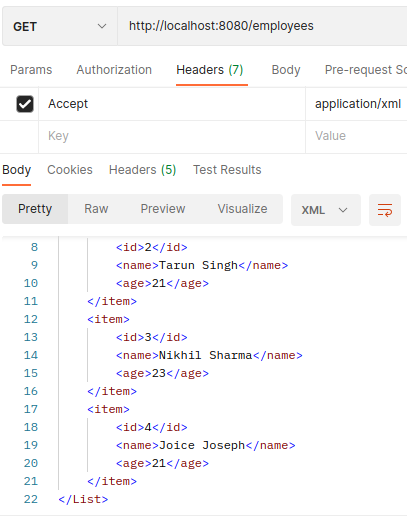
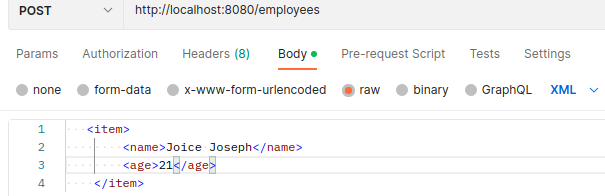
1. Create a GET request which takes "username" as param and shows a localized message "Hello Username". (Use parameters in message properties)

Answer:



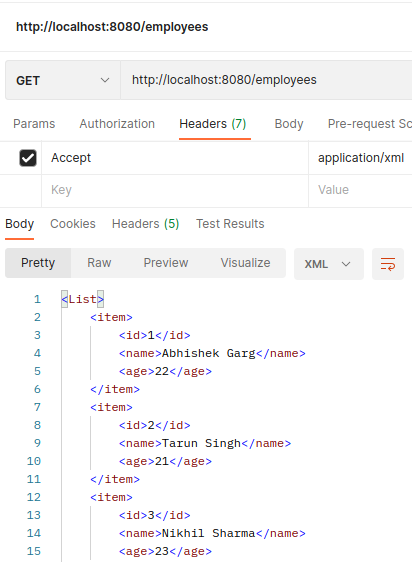
\*Content Negotiation

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

Answer: Added Jackson-datafromat-xml dependency in pom.xml

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

Answer: Added Jackson-datafromat-xml dependency in pom.xml



\*Swagger

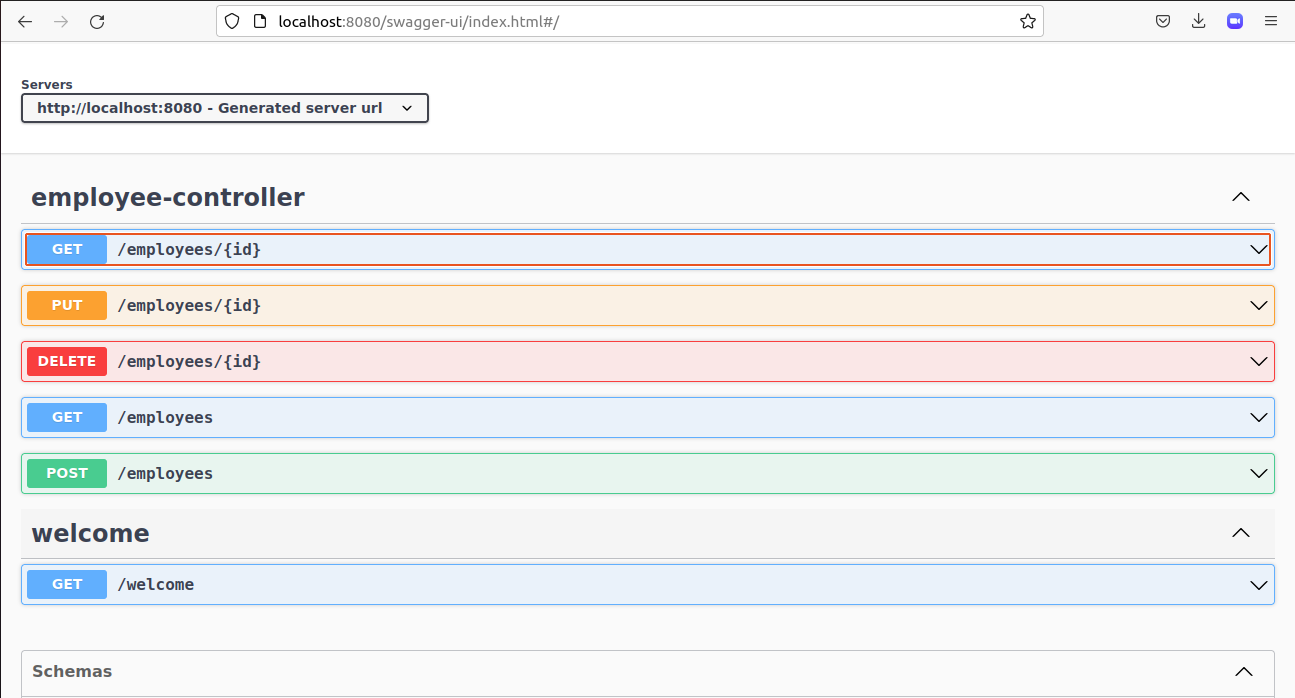
5. Configure swagger plugin and create document of following methods:

Get details of User using GET request.

Save details of the user using POST request.

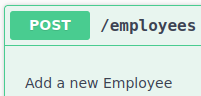
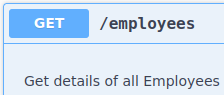
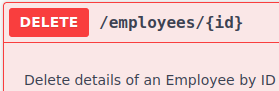
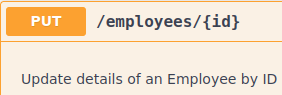
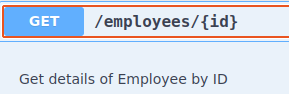
Delete a user using DELETE request.

Answer: Added springdoc-openapi-ui dependency in pom.xml



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.

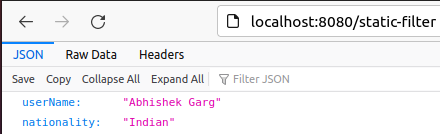
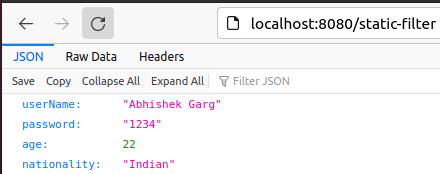
Answer: Added @Operation annotation in Controller



\*Static and Dynamic filtering

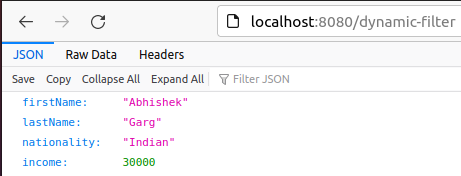
8. Create API which saves details of User (along with the password) but on successfully saving returns only non-critical data. (Use static filtering)

Answer:



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

Answer:



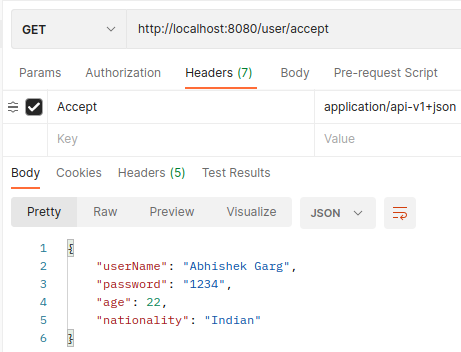
\*Versioning Restful APIs

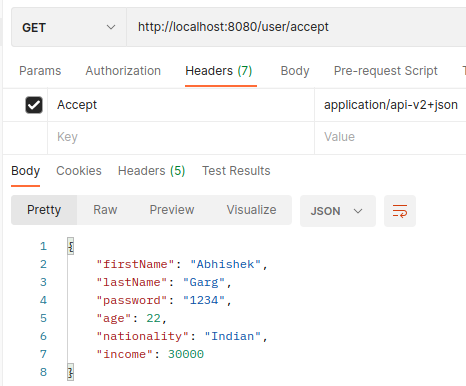
10. Create 2 API for showing user details. The first api should return only basic details of the user and the other API should return more/enhanced details of the user,

Now apply versioning using the following methods:

* MimeType Versioning

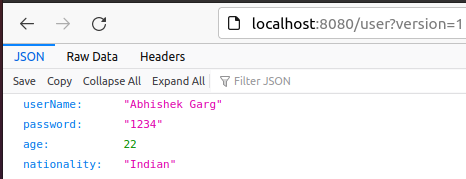
Answer:

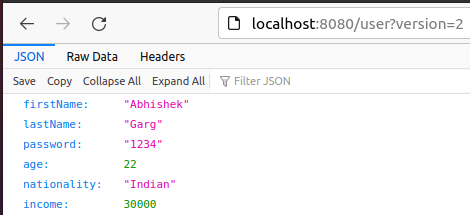




* Request Parameter versioning

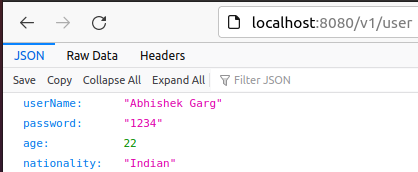
Answer:

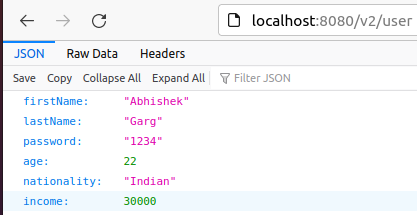




* URI versioning

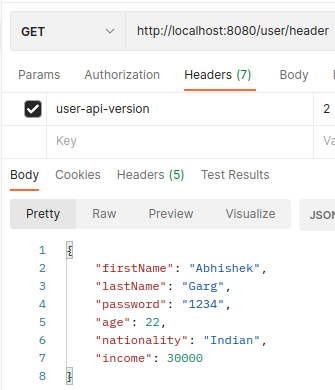
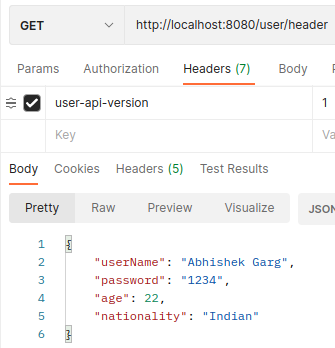
Answer:





* Custom Header Versioning

Answer:



\*HATEOAS

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

Answer:

