**REST WITH SPRING - 2 ASSIGNMENT**

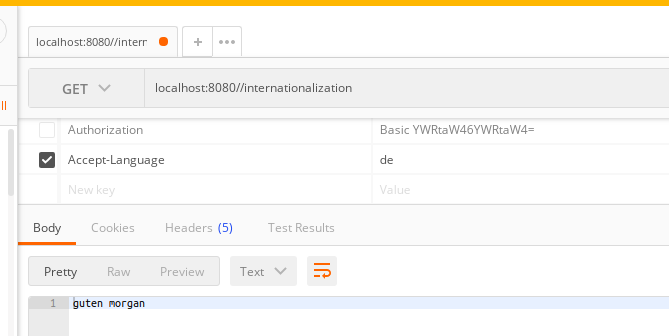
**RAGHAV GUPTA**

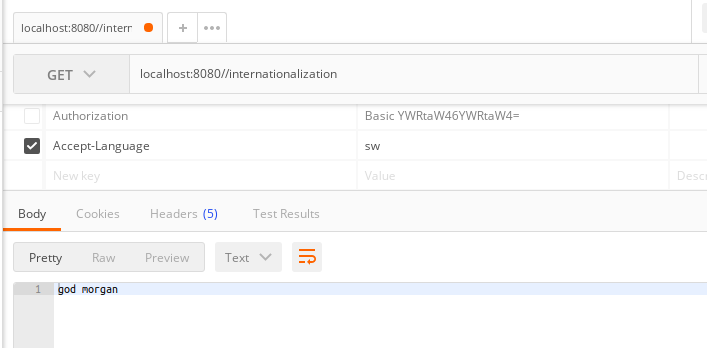
**======================================**

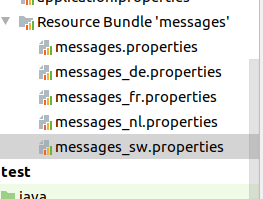
**\*Internationalization**

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

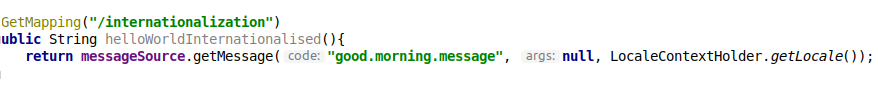
**SEE PACKAGE USERS**

****

****

****

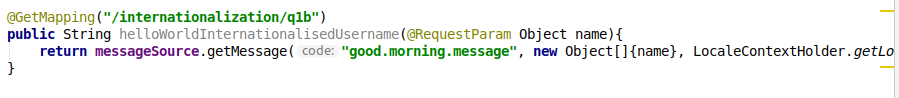
****

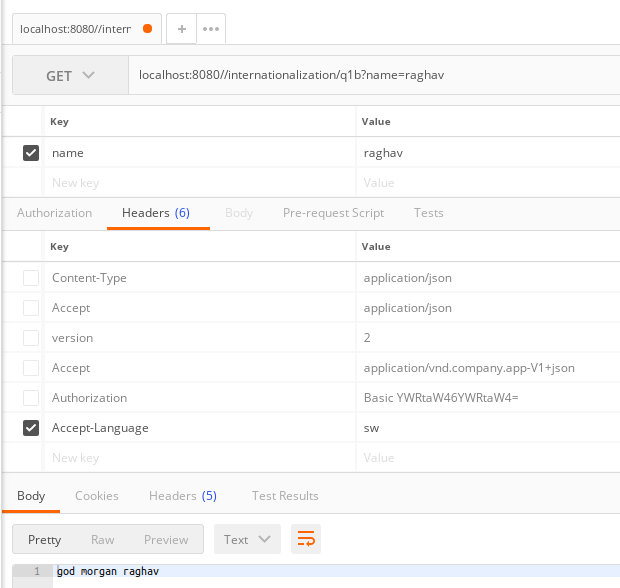
****

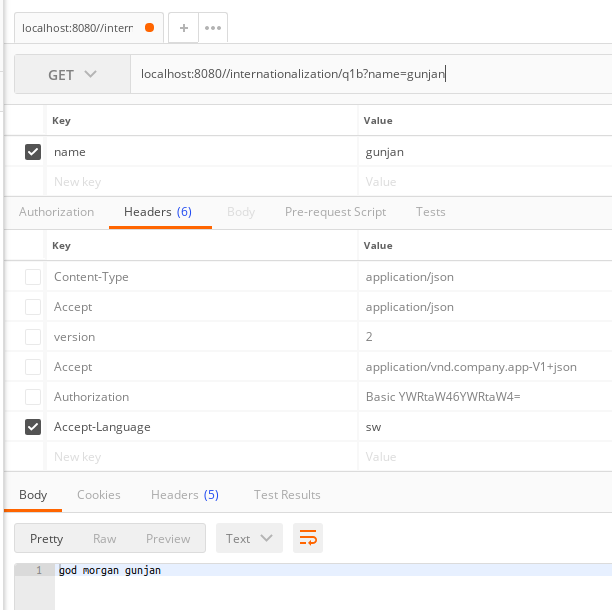
See properly in the project please. Here i have shown the main things that i need to make multiple message.properties files and the code. Output is also shown here.

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

**SEE PACKAGE ‘USERS’**

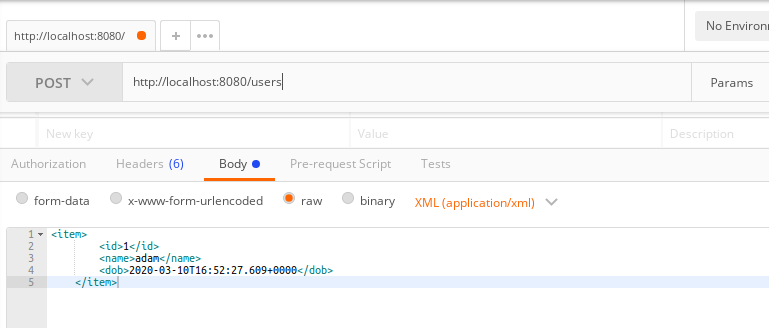
****

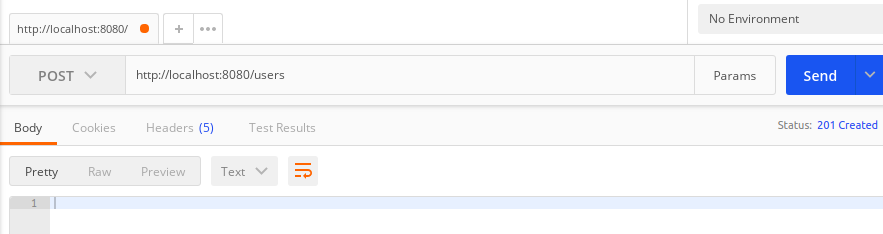
****

****

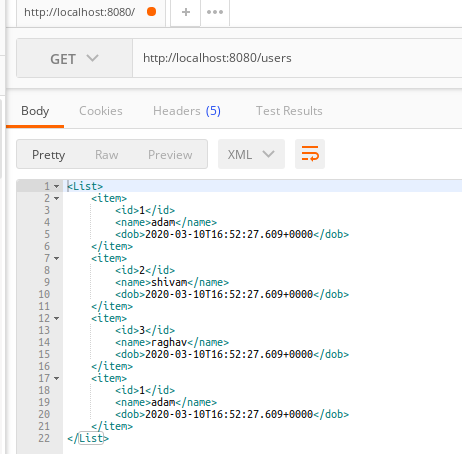
**\*Content Negotiation**

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

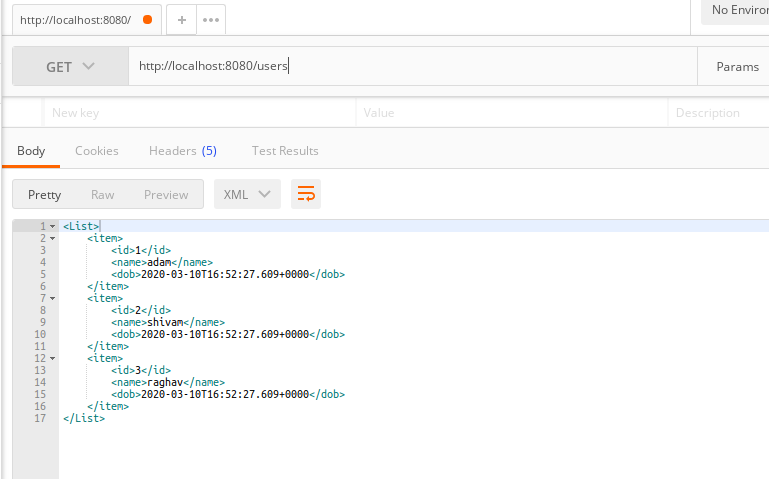
****

****

**Now we can see that the resource has been created.**

****

**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.**

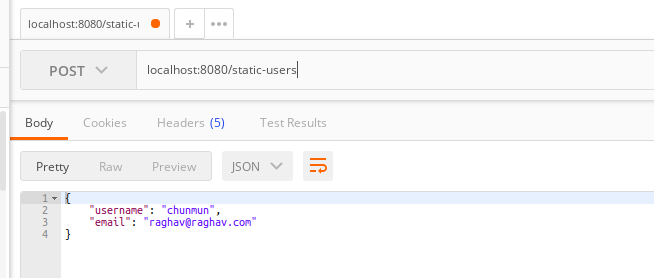
**7. In swagger documentation, add the description of each class and URI so that in swagger UI the purpose of class and URI is clear.**

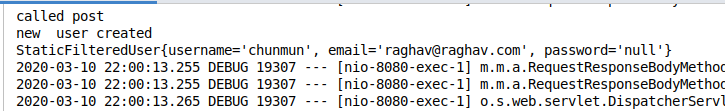
**For q5 and q7, see SWAGGERCONFIG file.**

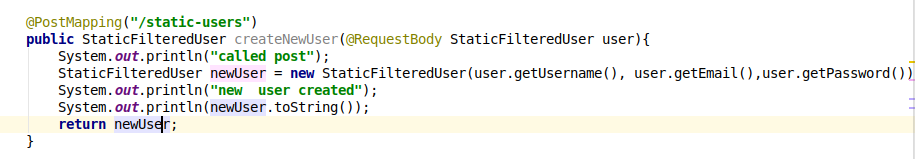
**\*Static and Dynamic filtering**

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

**SEE ‘FILTERINGDEMO’ PACKAGE**

****

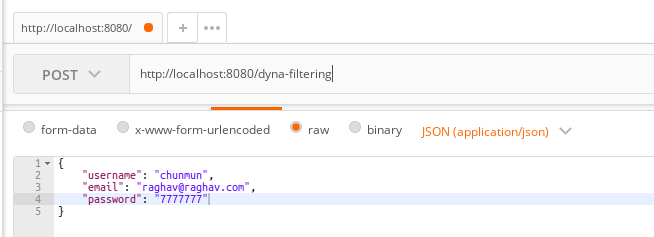
****

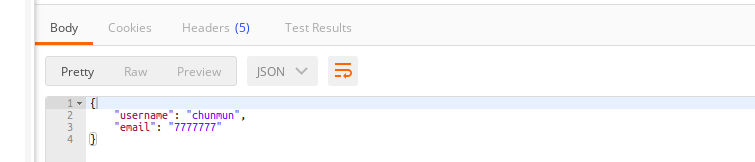
****

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

**SEE ‘DYNAMICFILTERING’ PACKAGE**

****

****

****

**\*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**

**FOR THIS QUESTION ---------------------- REFER TO VERSIONING PACKAGE IN THE PROJECT**

**\*HATEOAS**

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

Iplementation is in the project, i will show that time.