Social Network Analysis Tool

Prerquisite for project:

1. JDK 21
2. Any IDE.

How to run the project:

1. Navigate to https://github.com/ashishmacherla/socialnetwork and clone the project from remote repository to your local by using the command
"**git clone https://github.com/ashishmacherla/socialnetwork.git**"
2. Import it as a Spring or Java project in your desired IDE.
3. Run as a spring or java application.
Note: Depends on the IDE (like eclipse), you need to build the artifact before running the project. Command to build the project "**gradle clean build**" .

User Management

This section explains how to perform actions on adding, removing, updating, getting all users details and also to get user details for each individual userId.

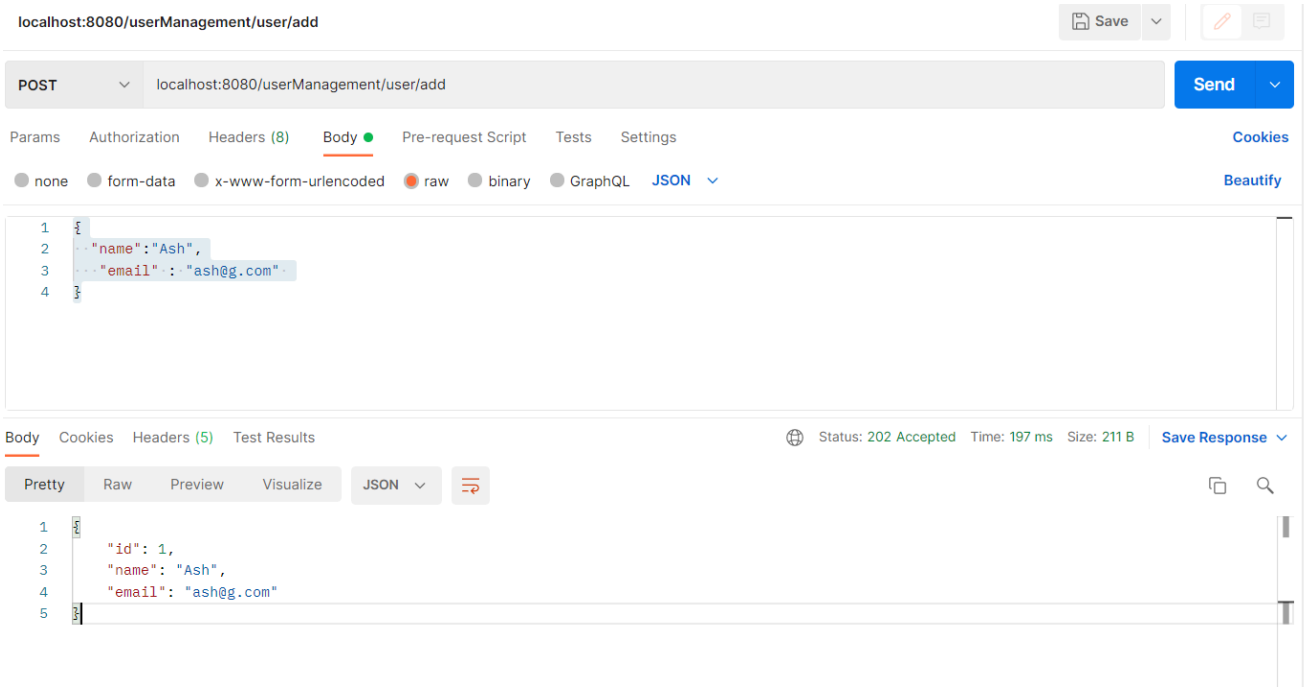
1. Add User:

HTTP ACTION: POST
URI: localhost:8080/userManagement/user/add
RequestBody:

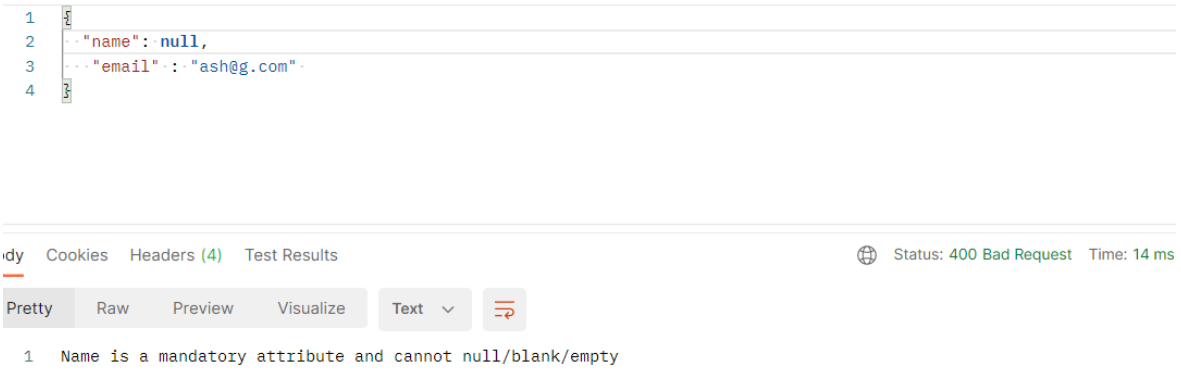
```
{
  "name":"Ash",
   "email" : "ash@g.com"
}
```

Response Code: 202 Accepted

Sample Request:



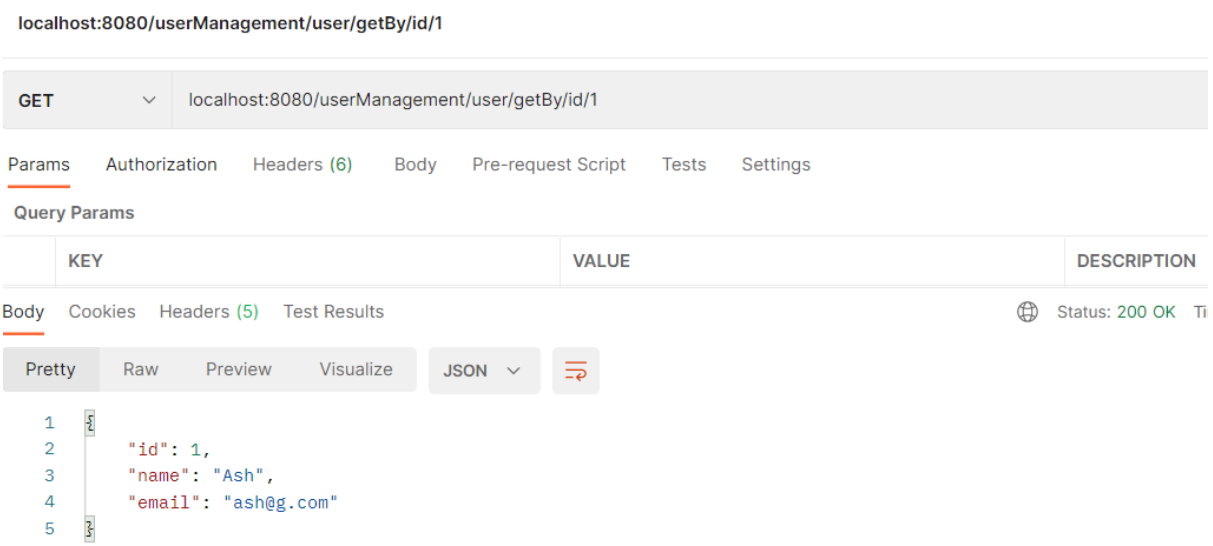
Note: Name and Email addresses are mandatory attributes and can't be null. For email the email format should be in the following a@a.com. Any incorrect/missing data will result in the 400 BadRequest
Sample error:



2. Get user by userID:

HTTP ACTION: GET
URI: localhost:8080/userManagement/user/getBy/id/{userId}
Response Code: 200 OK

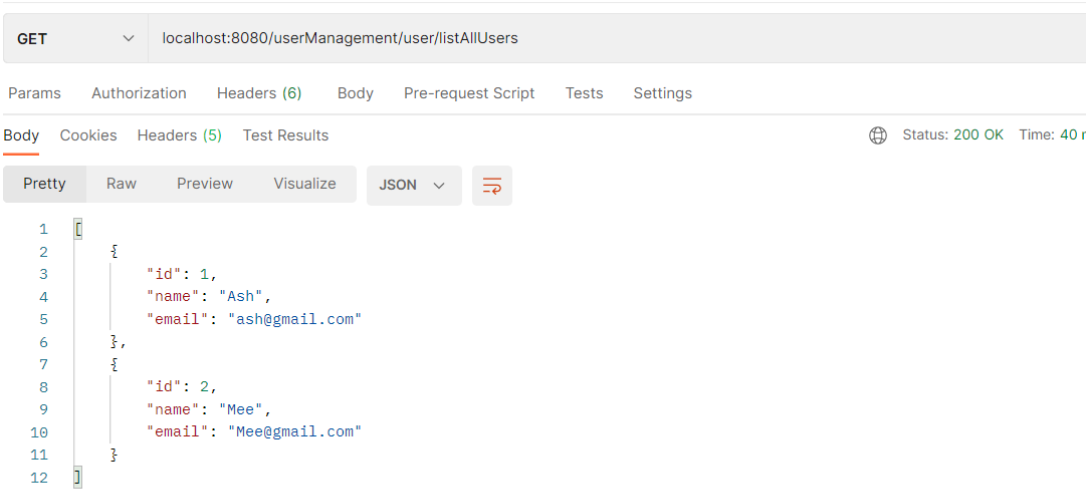
Sample Request:



Note: Id is the mandatory attribute. If missing will throw an Error.

3.GET ALL Users:

HTTP ACTION: GET
URI: localhost:8080/userManagement/user/listAllUsers
Response Code: 200 OK
Sample Request:



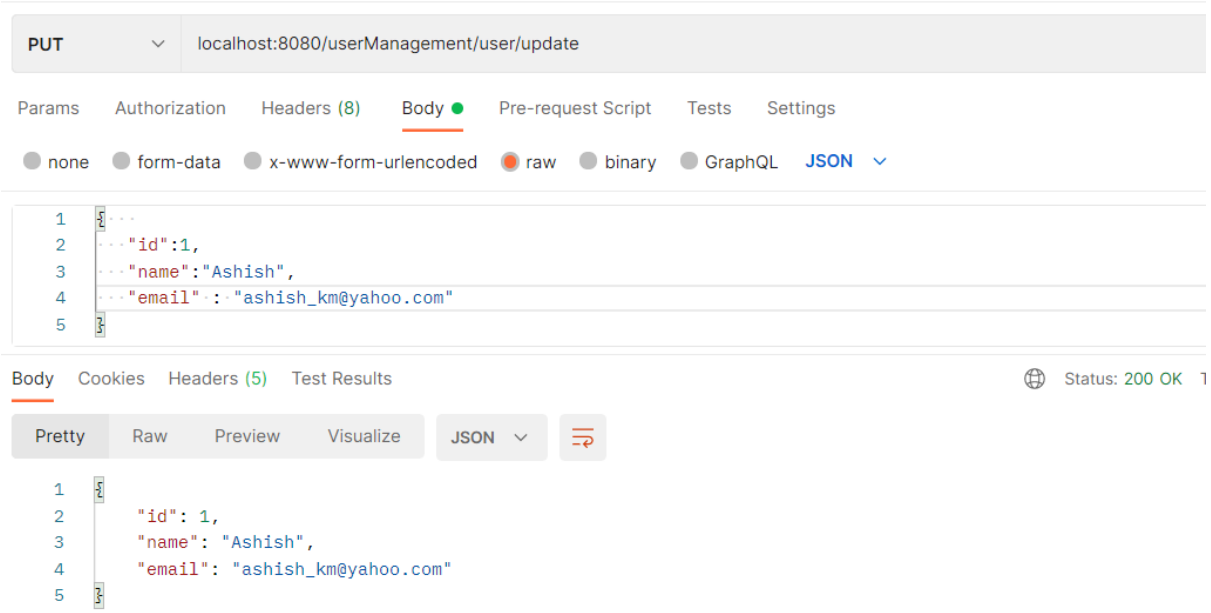
4.Update user:

HTTP ACTION: PUT
URI: localhost:8080/userManagement/user/update
RequestBody:

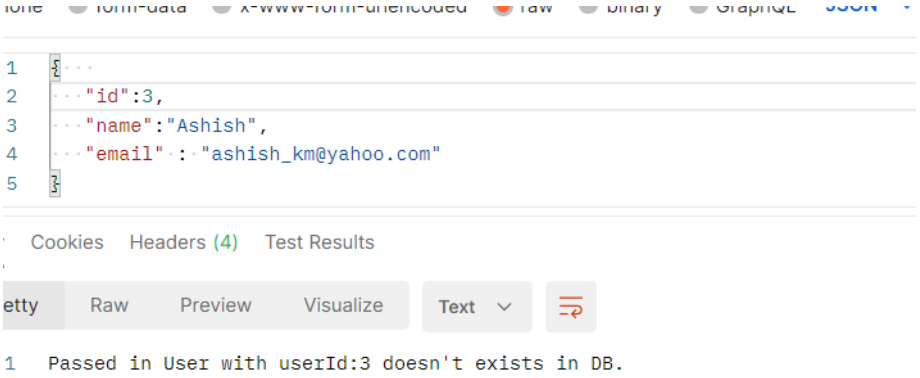
```
{
  "id":1,
  "name":"Ashish",
  "email" : "ashish_km@yahoo.com"
}
```

Response Code: 200 OK
Sample Request:

localhost:8080/userManagement/user/update

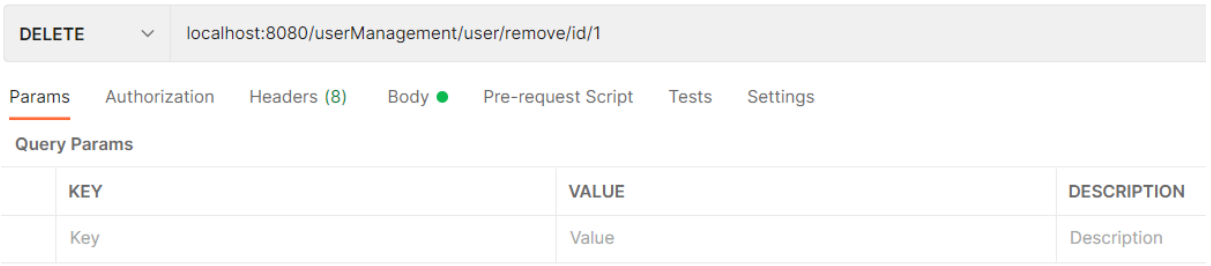


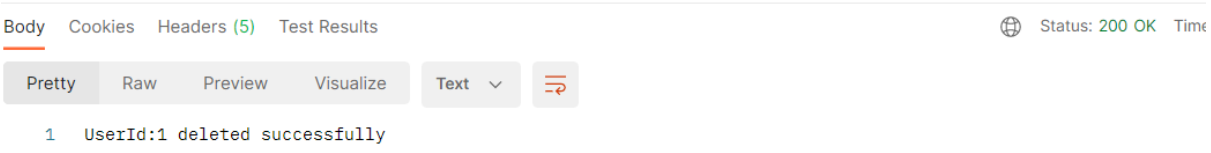
Note: Id is the mandatory attribute. If it's missing/passed in invalid id(aka userId) will result in the following error:



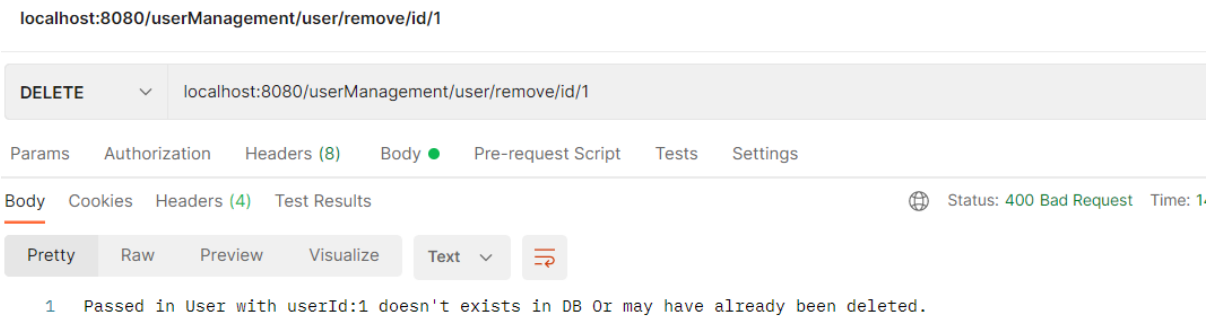
5. Delete/Remove user by userId:

HTTP ACTION: DELETE
URI: localhost:8080/userManagement/user/remove/id/{userId}
Response Code: 200 OK
Sample Request:





Note: Trying to delete an already deleted user will result in the following error:

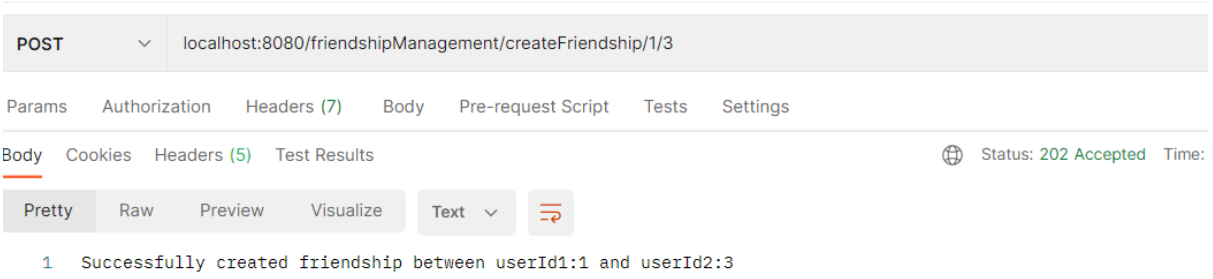


Friendship Management

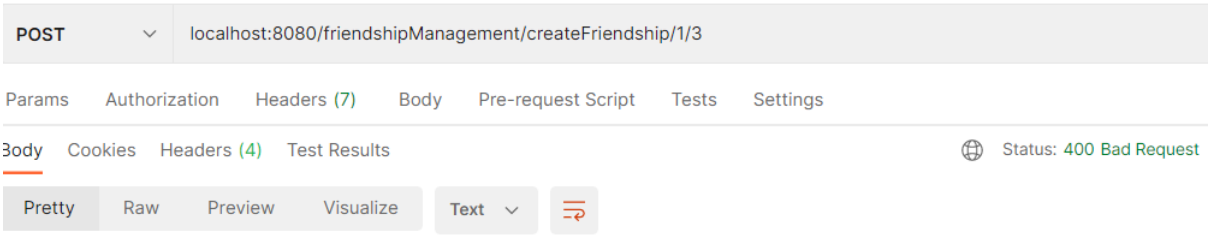
This section explains how to perform actions on creating and removing friendships among users and also to list all friends of a particular user.

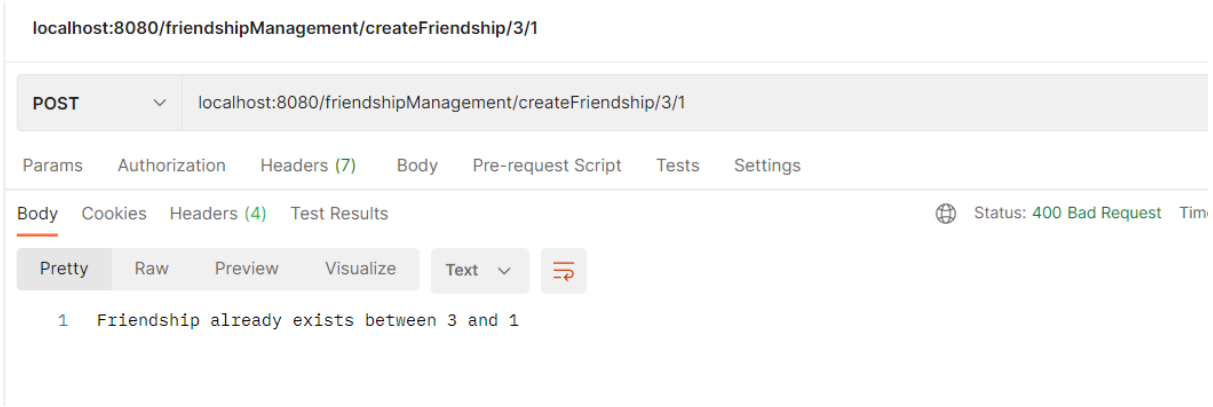
1.Creating friendship:

HTTP ACTION: POST
URI: localhost:8080/friendshipManagement/createFriendship/{userId1}/{userId2}
Response Code: 200 OK
Sample Request: localhost:8080/friendshipManagement/createFriendship/1/3



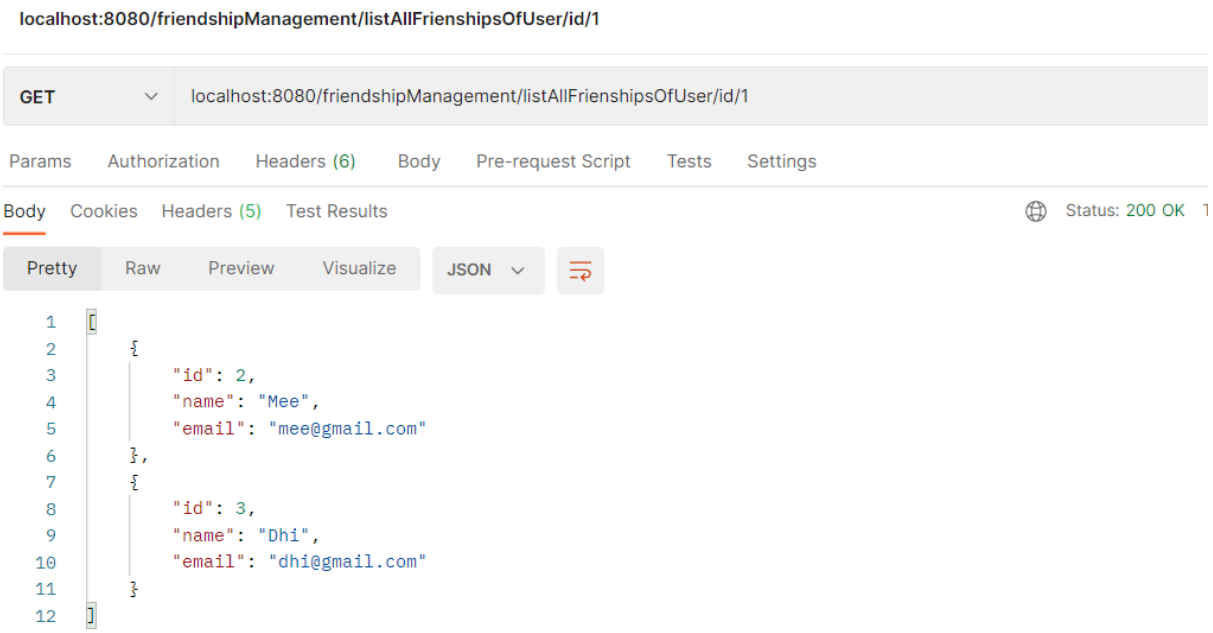
Note: If we are trying to again create friendship with userId: 1 and userId:3 OR vice versa (creating friendship with userId:3 and userId:1) we get the following error with
400 BadRequest.See the 2 screenshots below:





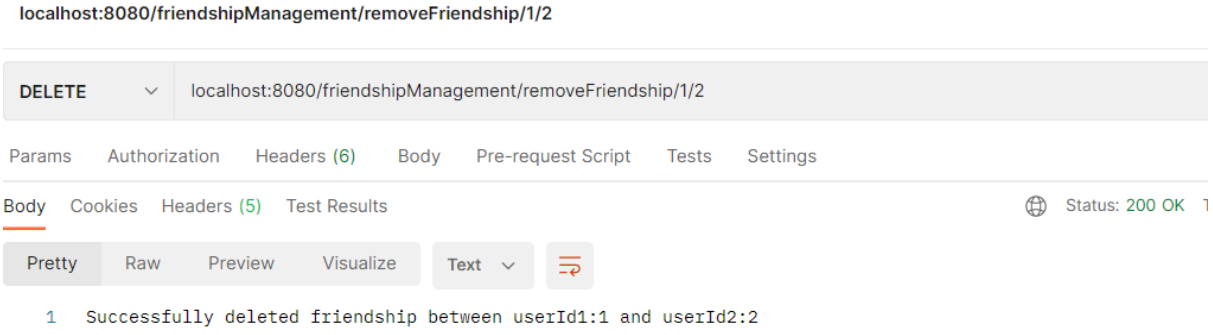
2. Listing friends of a User:

HTTP ACTION: GET
URI: localhost:8080/friendshipManagement/listAllFrienshipsOfUser/id/{userId}
Response Code: 200 OK
Sample Request: localhost:8080/friendshipManagement/listAllFrienshipsOfUser/id/1

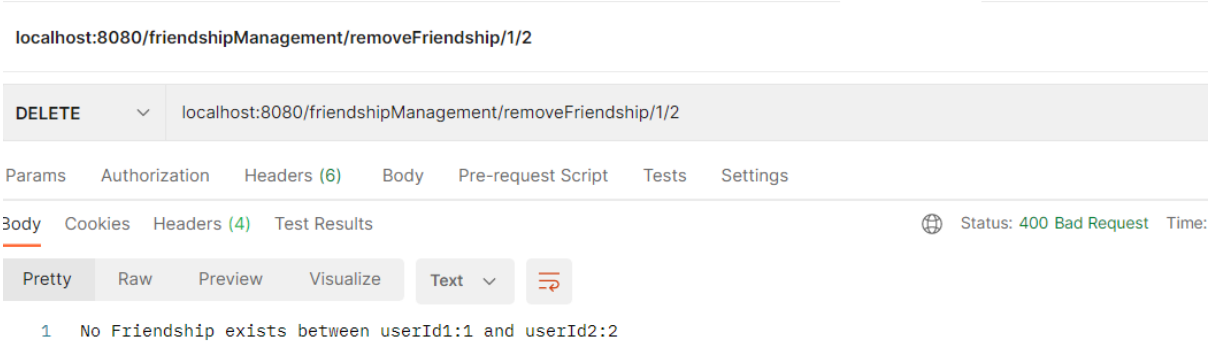


3. Delete/Removing friends between two users:

HTTP ACTION: DELETE
URI: localhost:8080/friendshipManagement/removeFriendship/{userId1}/{userId2}
Response Code: 200 OK
Sample Request: localhost:8080/friendshipManagement/removeFriendship/1/2



Note: If we are trying to delete a already deleted friendship(duplicate request) we get the following error:

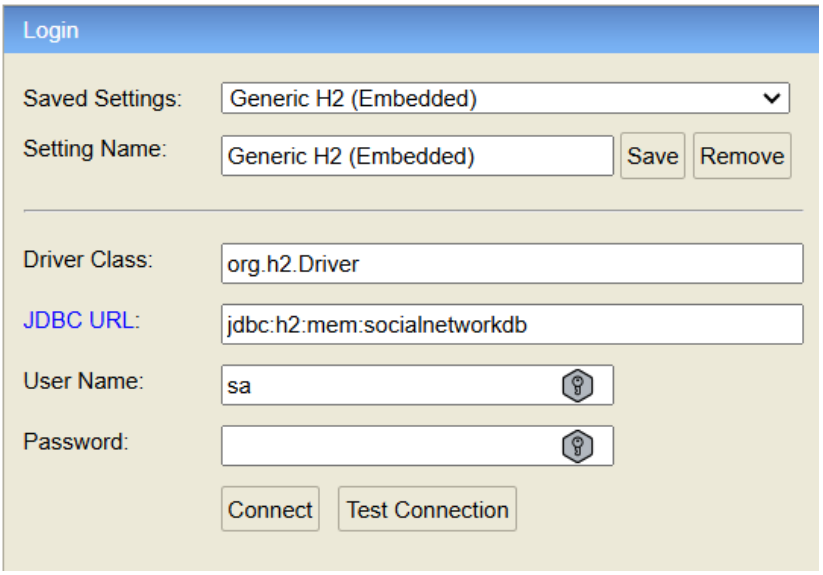


To validate the data in the database:

1. Login to the local H2 database with the following URL:

http://localhost:8080/h2-console/login.jsp

2. Click on **Connect** button.



3. Run the following queries:

3.1 Select * from users;

3.2 Select * from friendships;

3.1 Response:

```
select * from users;
```

select * from users;

ID	EMAIL	NAME
1	ashish_km@yahoo.com	Ashish
2	mee@gmail.com	Mee
3	dhi@gmail.com	Dhi

(3 rows, 1 ms)

3.2 Response:

```
select * from friendship;
```

select * from friendship;

ID	USERS1_ID	USERS2_ID
1	1	3
3	2	3

(2 rows, 2 ms)