

Social Network Analysis Tool

Prerequisite 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:

localhost:8080/userManagement/user/add

POST localhost:8080/userManagement/user/add

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

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "name": "Ash",
3   "email": "ash@g.com"
4 }
```

Body Cookies Headers (5) Test Results

Status: 202 Accepted Time: 197 ms Size: 211 B Save Response

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Ash",
4   "email": "ash@g.com"
5 }
```

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:

```
1 {
2   "name": null,
3   "email": "ash@g.com"
4 }
```

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time: 14 ms

Pretty Raw Preview Visualize Text

```
1 Name is a mandatory attribute and cannot null/blank/empty
```

2. Get user by userID:

HTTP ACTION: GET

URI: localhost:8080/userManagement/user/getBy/id/{userId}

Response Code: 200 OK

Sample Request:

localhost:8080/userManagement/user/getBy/id/1

GET localhost:8080/userManagement/user/getBy/id/1

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

Query Params

KEY	VALUE	DESCRIPTION
-----	-------	-------------

Body Cookies Headers (5) Test Results Status: 200 OK Ti

Pretty Raw Preview Visualize JSON

```
1 {
2   "id": 1,
3   "name": "Ash",
4   "email": "ash@g.com"
5 }
```

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:

GET localhost:8080/userManagement/user/listAllUsers

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

Body Cookies Headers (5) Test Results Status: 200 OK Time: 40 r

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 1,
4     "name": "Ash",
5     "email": "ash@gmail.com"
6   },
7   {
8     "id": 2,
9     "name": "Mee",
10    "email": "Mee@gmail.com"
11  }
12 ]
```

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

PUT

localhost:8080/userManagement/user/update

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettings

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

{

"id": 1,

"name": "Ashish",

"email": "ashish_km@yahoo.com"

}

BodyCookiesHeaders (5)Test Results

PrettyRawPreviewVisualizeJSON

1

2

3

4

5

{

"id": 1,

"name": "Ashish",

"email": "ashish_km@yahoo.com"

}

Status: 200 OK 1

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

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

1

2

3

4

5

{

"id": 3,

"name": "Ashish",

"email": "ashish_km@yahoo.com"

}

CookiesHeaders (4)Test Results

ettyRawPreviewVisualizeText

1 Passed in User with userId:3 doesn't exists in DB.

5. Delete/Remove user by userId:

HTTP ACTION: DELETE

URI: localhost:8080/userManagement/user/remove/id/{userId}

Response Code: 200 OK

Sample Request:

DELETE

localhost:8080/userManagement/user/remove/id/1

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body

Cookies

Headers (5)

Test Results

Status: 200 OK

Time

Pretty

Raw

Preview

Visualize

Text

1

UserId:1 deleted successfully

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

localhost:8080/userManagement/user/remove/id/1

DELETE

localhost:8080/userManagement/user/remove/id/1

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Body

Cookies

Headers (4)

Test Results

Status: 400 Bad Request

Time: 1.

Pretty

Raw

Preview

Visualize

Text

1

Passed in User with userId:1 doesn't exists in DB Or may have already been deleted.

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

POST localhost:8080/friendshipManagement/createFriendship/1/3

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

Body Cookies Headers (5) Test Results Status: 202 Accepted Time:

Pretty Raw Preview Visualize Text

```
1 Successfully created friendship between userId1:1 and userId2:3
```

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:

POST localhost:8080/friendshipManagement/createFriendship/1/3

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

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time:

Pretty Raw Preview Visualize Text

```
1 Friendship already exists between 1 and 3
```

localhost:8080/friendshipManagement/createFriendship/3/1

POST localhost:8080/friendshipManagement/createFriendship/3/1

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

Body Cookies Headers (4) Test Results Status: 400 Bad Request Time:

Pretty Raw Preview Visualize Text

```
1 Friendship already exists between 3 and 1
```

2. Listing friends of a User:

HTTP ACTION: GET

URI: localhost:8080/friendshipManagement/listAllFriendshipsOfUser/id/{userId}

Response Code: 200 OK

Sample Request: localhost:8080/friendshipManagement/listAllFriendshipsOfUser/id/1

localhost:8080/friendshipManagement/listAllFriendshipsOfUser/id/1

The screenshot shows a REST client interface with a GET request to `localhost:8080/friendshipManagement/listAllFriendshipsOfUser/id/1`. The response is a JSON array of two objects, each representing a friend. The status is 200 OK.

```
GET localhost:8080/friendshipManagement/listAllFriendshipsOfUser/id/1
```

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

Body Cookies Headers (5) Test Results Status: 200 OK 1

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "id": 2,
4     "name": "Mee",
5     "email": "mee@gmail.com"
6   },
7   {
8     "id": 3,
9     "name": "Dhi",
10    "email": "dhi@gmail.com"
11  }
12 ]
```

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

localhost:8080/friendshipManagement/removeFriendship/1/2

The screenshot shows a REST client interface with a DELETE request to `localhost:8080/friendshipManagement/removeFriendship/1/2`. The response is a text message indicating the friendship was successfully deleted. The status is 200 OK.

```
DELETE localhost:8080/friendshipManagement/removeFriendship/1/2
```

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

Body Cookies Headers (5) Test Results Status: 200 OK 1

Pretty Raw Preview Visualize Text

```
1 Successfully deleted friendship between userId1:1 and userId2:2
```

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

localhost:8080/friendshipManagement/removeFriendship/1/2

DELETE localhost:8080/friendshipManagement/removeFriendship/1/2

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

Body Cookies Headers (4) Test Results

Status: 400 Bad Request Time:

Pretty Raw Preview Visualize Text

```
1 No Friendship exists between userId1:1 and userId2:2
```

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.

Login

Saved Settings: Generic H2 (Embedded) ▼

Setting Name: Generic H2 (Embedded) Save Remove

Driver Class: org.h2.Driver

JDBC URL: jdbc:h2:mem:socialnetworkdb

User Name: sa

Password:

Connect Test Connection

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)