1. **Structures**

* **ResponseObject**

{

String Status; // take 2 values only: “OK” and “ERROR”

String Message;

}

1. **ViewModels**

* **AccountSummViewModel**

**public** String **\_id**;  
**public** String **name**;  
**public** String **email**;  
**public** Date **dob**;  
**public boolean gender**;

* **AccountViewModel**

**public** String **\_id**;  
**public** String **name**;  
**public** String **email**;  
**public** Date **dob**;  
**public boolean gender**;  
**public** List<AccountSummViewModel> **friend\_list**;  
**public** List<PostSummViewModel> **posted\_posts**;  
**public** List<PostSummViewModel> **followed\_posts**;  
**public** List<PostSummViewModel> **author\_of\_posts**;

* **CommentViewModel**

**public** String **\_id**;  
**public** Date **dateCreated**;  
**public** String **content**;

* **FileViewModel**

**public** String **fileName**;  
**public long length**;

* **PostSummViewModel**

**public** String **\_id**;  
**public** Date **dateCreated**;  
**public** String **title**;  
**public** String **description**;  
**public** Date **publicationDate**;

* **PostViewModel**

**public** String **\_id**;  
**public** AccountSummViewModel **posted\_by**;  
**public** Date **dateCreated**;  
**public** String **title**;  
**public** String **description**;  
**public** Date **publicationDate**;  
**public** List<AccountSummViewModel> **authors**;  
**public** List<AccountSummViewModel> **liked\_by**;  
**public** List<AccountSummViewModel> **followed\_by**;

1. **APIs**

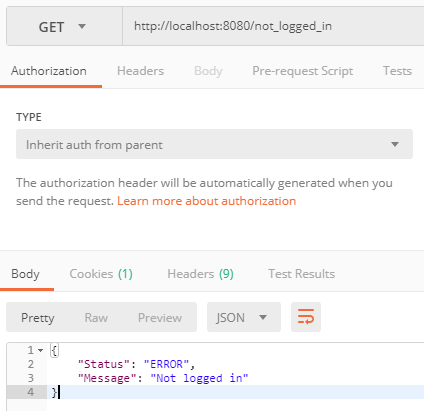
* **/not\_logged\_in**

**Description**: Inform that user is not logged in

**Method**: GET

**RequestParams**: None

**Return**: ResponseObject



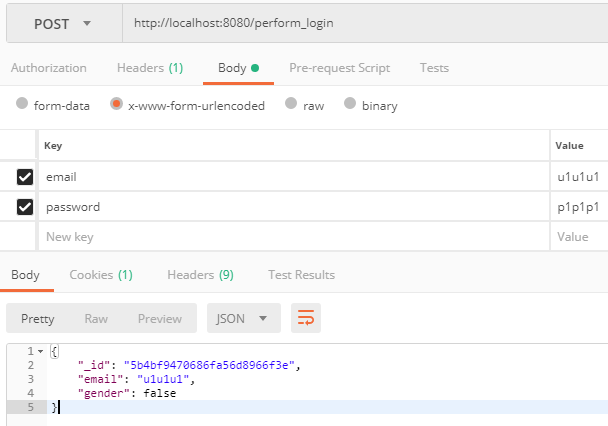
* **/perform\_login**

**Description**: Used to login

**Method**: POST

**RequestParams**:

**Return**: Redirect to /account\_info if logged in or to /not\_logged\_in if could not logged in

****

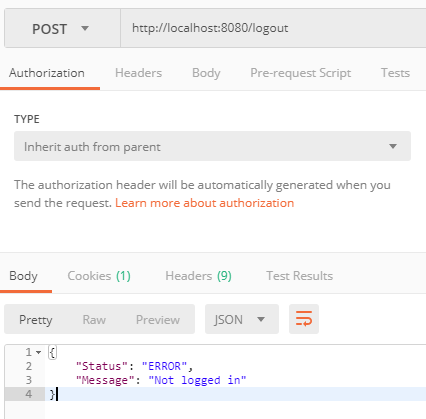
* **/logout**

**Description**: Used to logout

**Method**: POST

**RequestParams**: None

**Return**: Redirect to /not\_logged\_in

****

* **/account\_info**

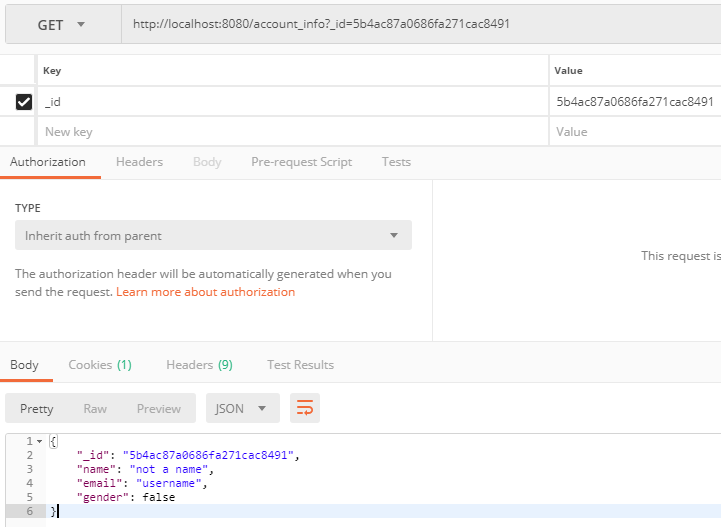
**Description**: Get account information

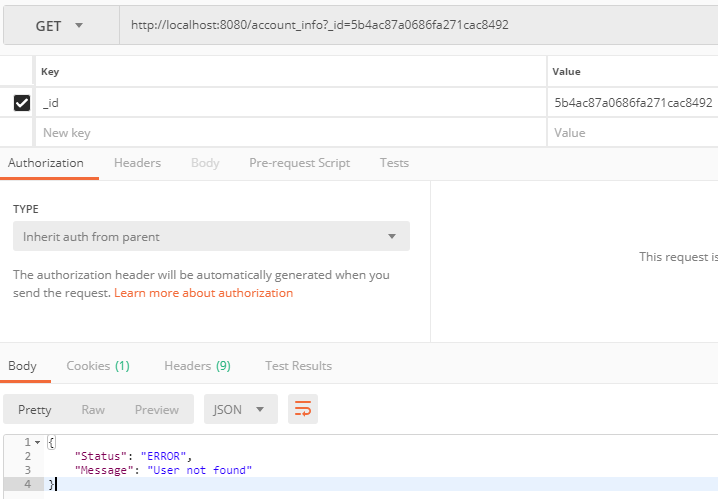
**Method**: GET

**RequestParams**:

\_id(not required): User’s ID

**Return**: AccountSummaryViewModel or ResponseObject if user not found





* **/changepass**

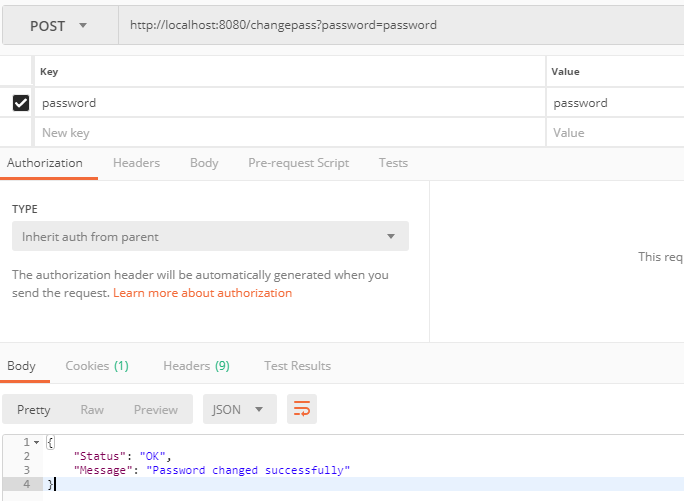
**Description**: Used to changed current user’s password

**Method**: POST

**RequestParams**:

password: new password

**Return**: ResponseObject



* **/register (FOR TESTING ONLY)(will be removed in the final version)**

**Description**: Used to register new account

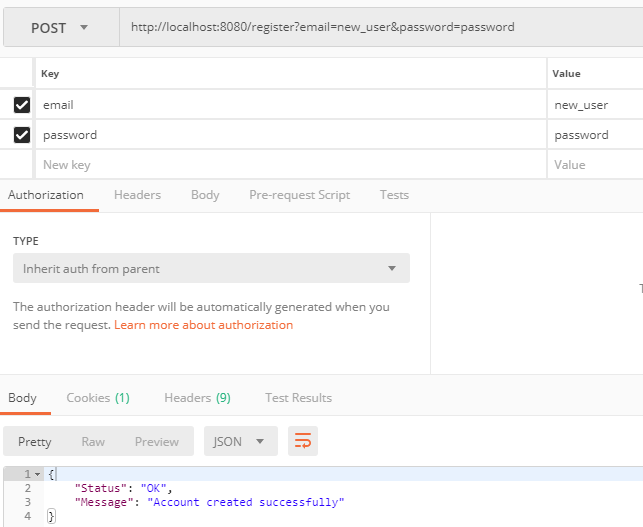
**Method**: POST

**RequestParams**:

email: username

password: password of new account

**Return**: ResponseObject



* **/list/comments**

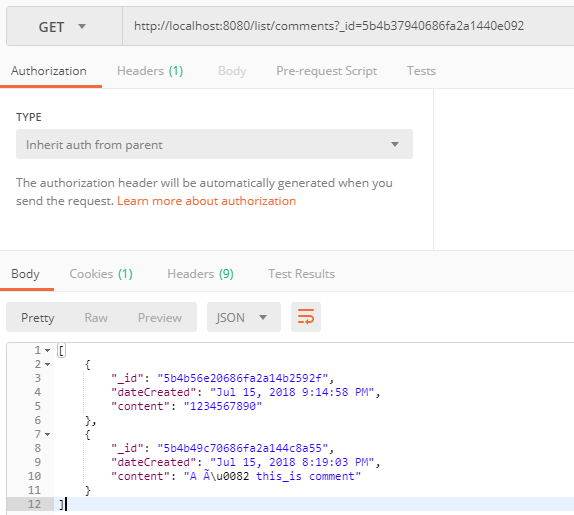
**Description**: List comments of a post

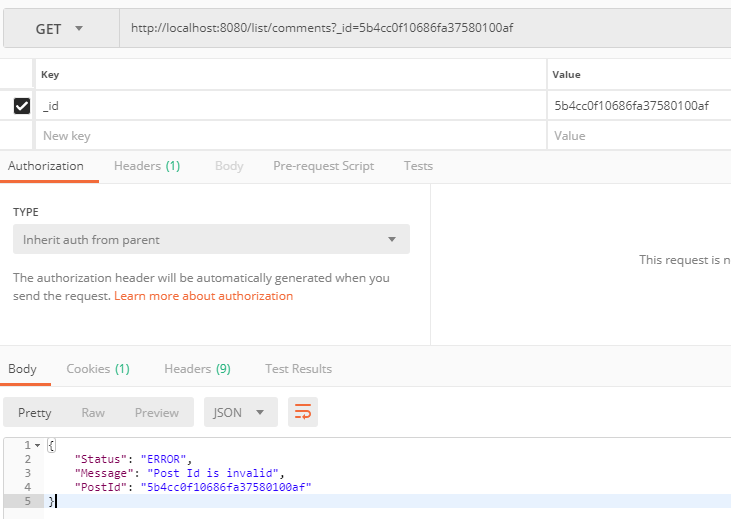
**Method**: GET

**RequestParams**:

\_id: Post’s ID

**Return**: Array of CommentViewModel or ResponseObject with additional field “PostId” if error

****

****

* **/comment**

**Description**: Let current user comment in a post

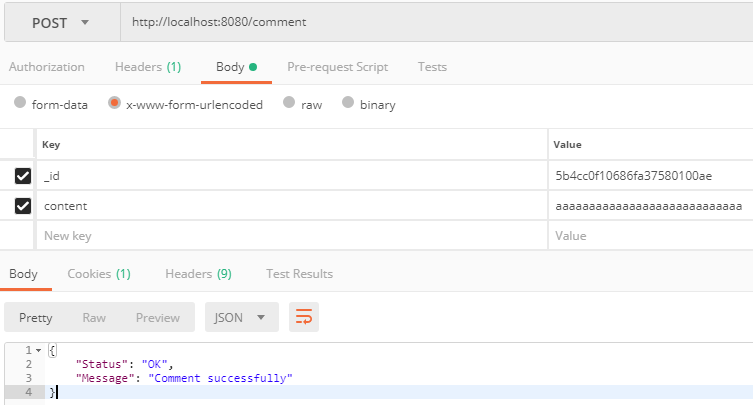
**Method**: POST

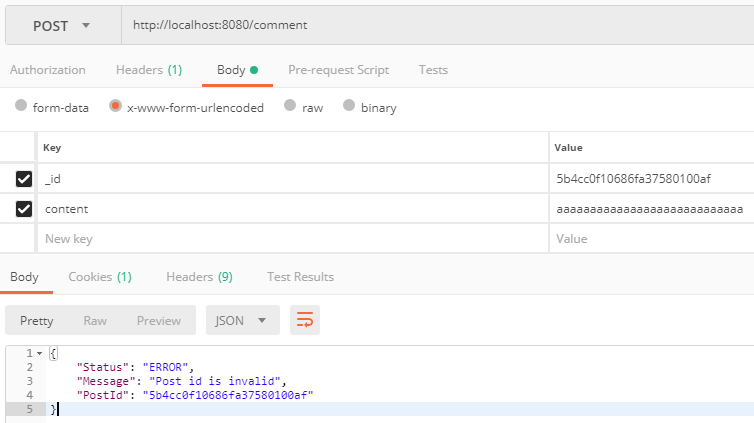
**RequestParams**:

\_id: Post’s ID

content: Content of comment

**Return**: normal ResponseObject or ResponseObject with additional field “PostId” if error





* **/post/{post\_id}**

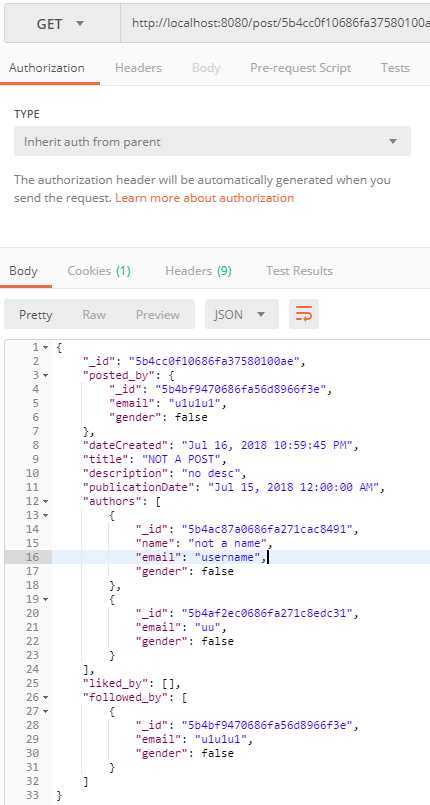
**Description**: Get post information

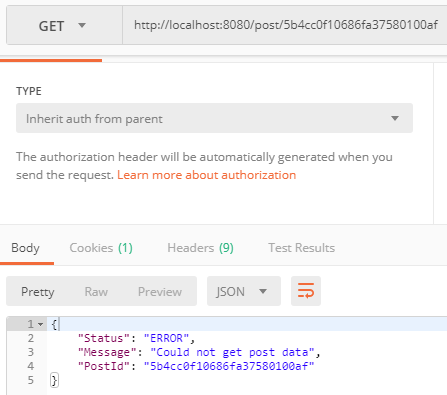
**Method**: GET

**RequestParams**:

post\_id(path variable): Post’s ID

**Return**: PostViewModel or ResponseObject with additional field “PostId” if error





* **/post/react**

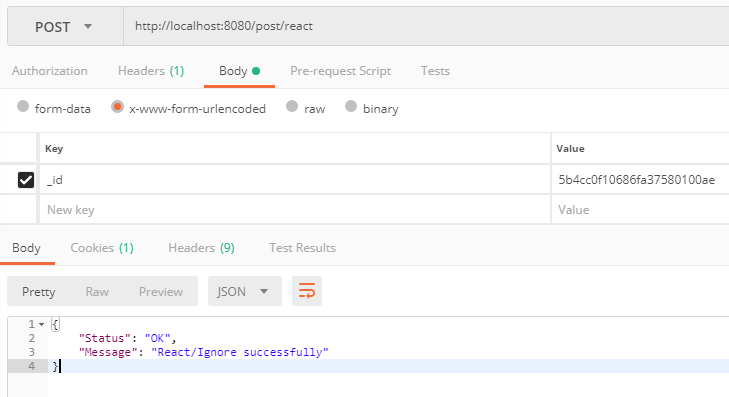
**Description**: Toggle post like/unlike

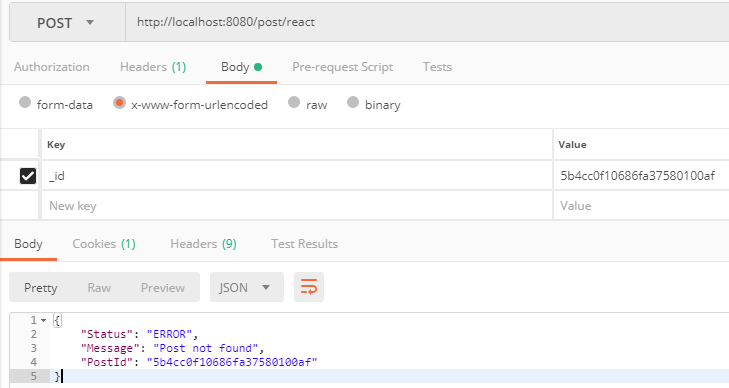
**Method**: POST

**RequestParams**:

\_id: Post’s ID

**Return**: ResponseObject or ResponseObject with additional field “PostId” if error





* **/post/follow**

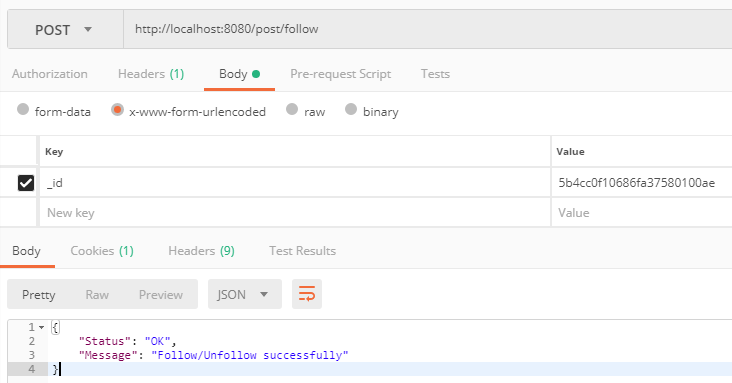
**Description**: Toggle post follow/unfollow

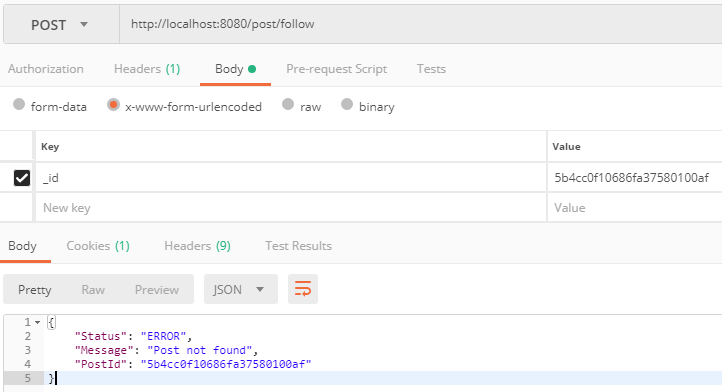
**Method**: POST

**RequestParams**:

\_id: Post’s ID

**Return**: ResponseObject or ResponseObject with additional field “PostId” if error





* **/post/is\_reacted**

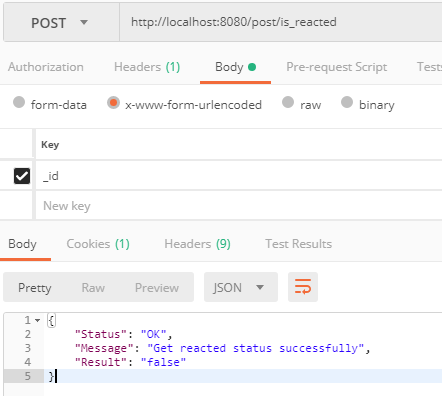
**Description**: Check if current user like post or not

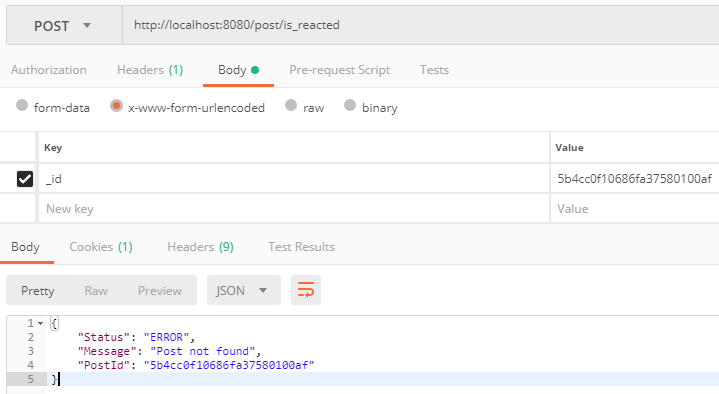
**Method**: POST

**RequestParams**:

\_id: Post’s ID

**Return**: ResponseObject with additional field “Result” or ResponseObject additional field “PostId” if error





* **/post/is\_followed**

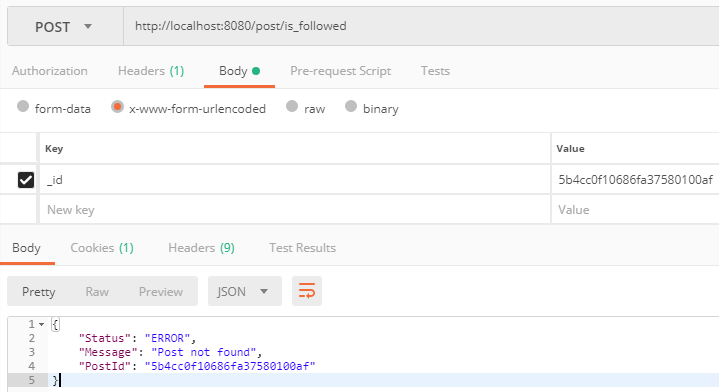
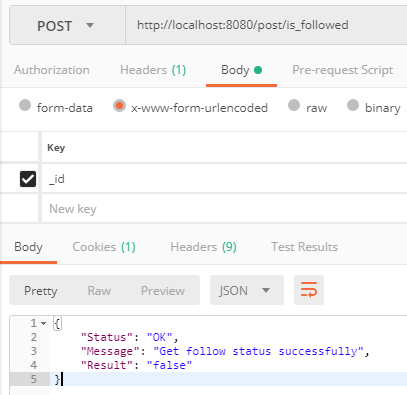
**Description**: Check if current user follow post or not

**Method**: POST

**RequestParams**:

\_id: Post’s ID

**Return**: ResponseObject with additional field “Result” or ResponseObject additional field “PostId” if error



* **/search/documents**

**Description**: Used to search documents

**Method**: GET

**RequestParams**:

sort (not required): type of search, take 1 of 3 values: “date\_created”, “publication\_date” and “most\_followed”. If not specified, default to “date\_created”

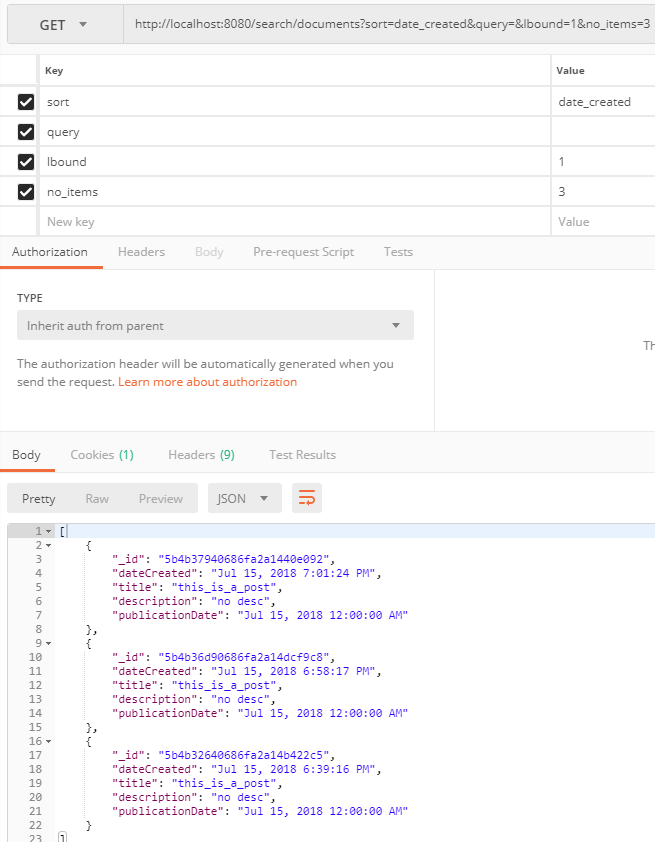
query (not required): the query string to search

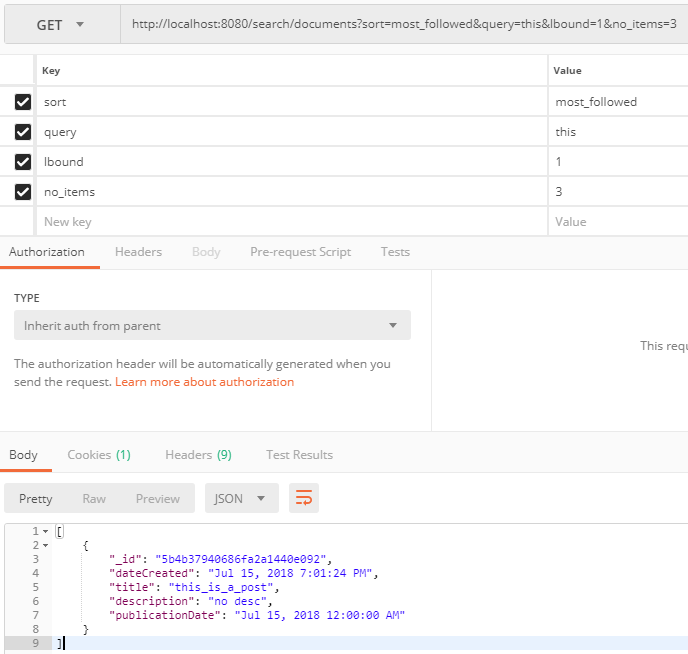
lbound (not required): lower bound of searching result, not work if no\_item not exist

no\_items (not required): number of items in the result, not work if lbound not exist

**Return**: Array of PostSummViewModel

**Note**: Due to limitation of Neo4J, “most\_followed” searching mode can only return Posts at least 1 follower





* **/search/users**

**Description**: Used to search users

**Method**: GET

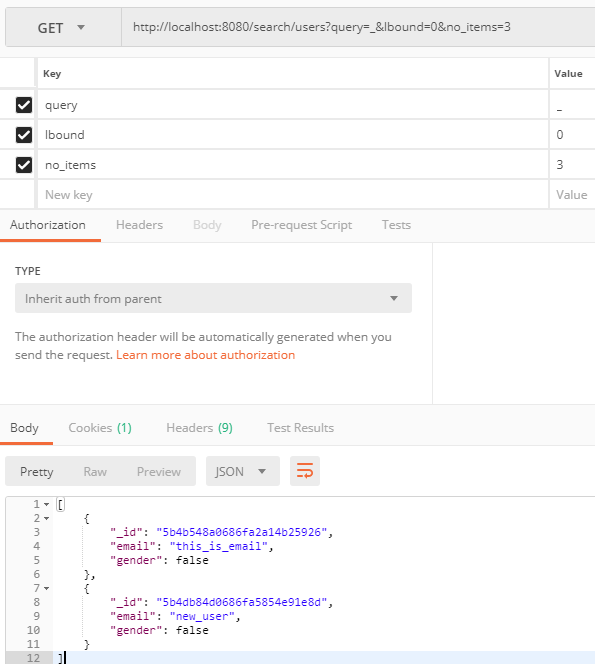
**RequestParams**:

query (not required): the query string to search

lbound (not required): lower bound of searching result, not work if no\_item not exist

no\_items (not required): number of items in the result, not work if lbound not exist

**Return**: Array of AccountSummViewModel



* **/upload**

**Description**: Used to post a post

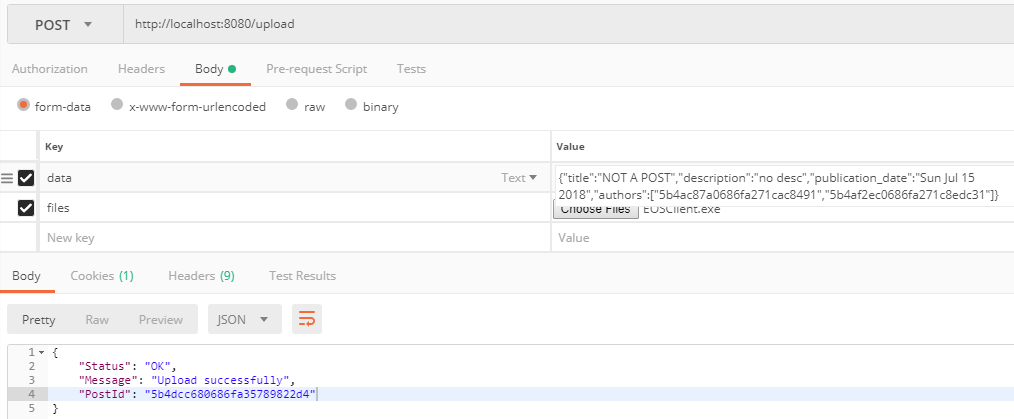
**Method**: POST

**RequestParams**:

data: string of json object (image bellow)

files: enctype='multipart/form-data’ file

**Return**: ResponseObject with additional field “PostId”



* **/download/{post\_id}**

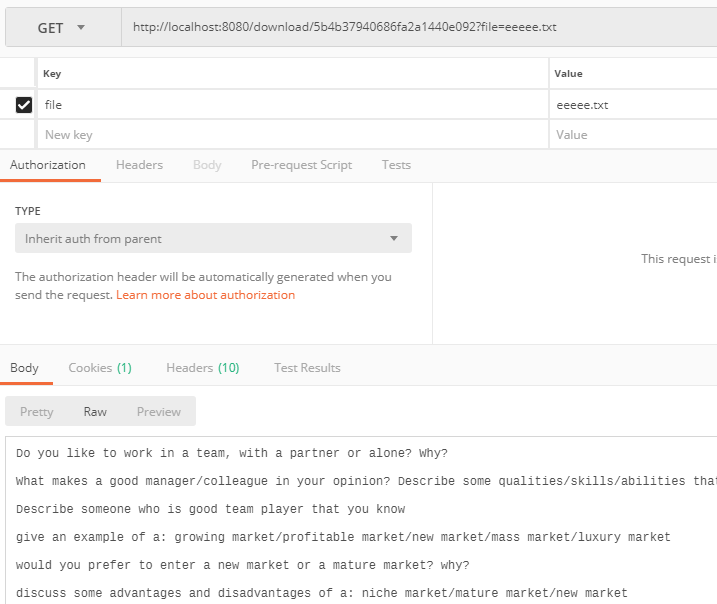
**Description**: Used to download file from post

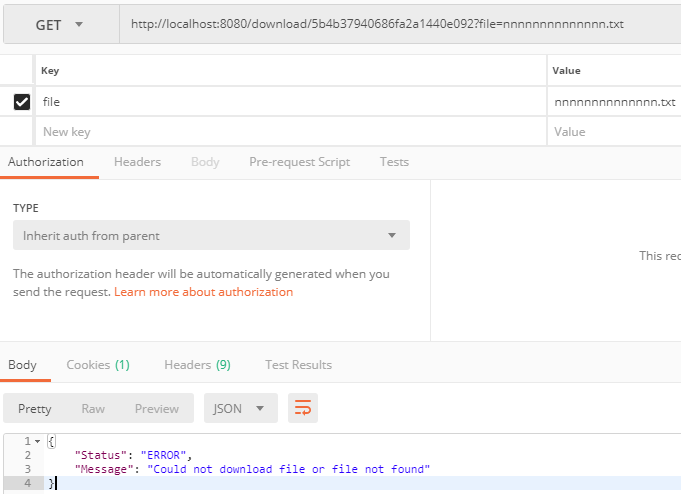
**Method**: GET

**RequestParams**:

file: filename

**Return**: File content in bytes or ResponseObject if error





* **/list/files/{post\_id}**

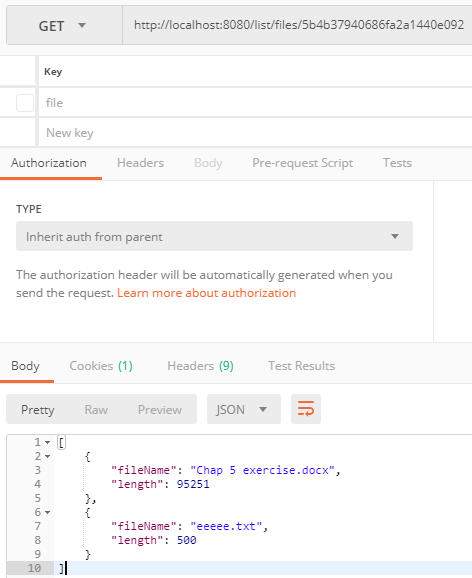
**Description**: Get list of files in the specified path that store

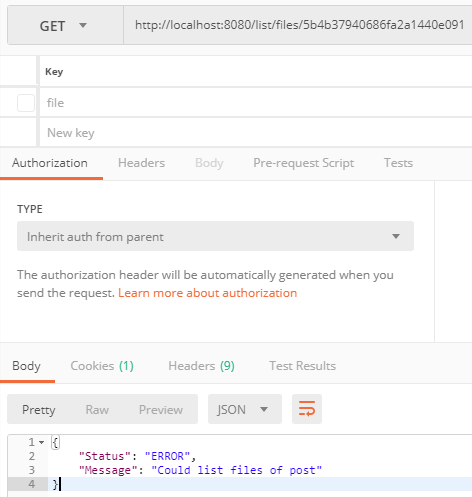
**Method**: GET

**RequestParams**:

post\_id(path variable): Post’s ID

**Return**: Array of FileViewModel or ResponseObject if post is not exists





* **/user/is\_friend**

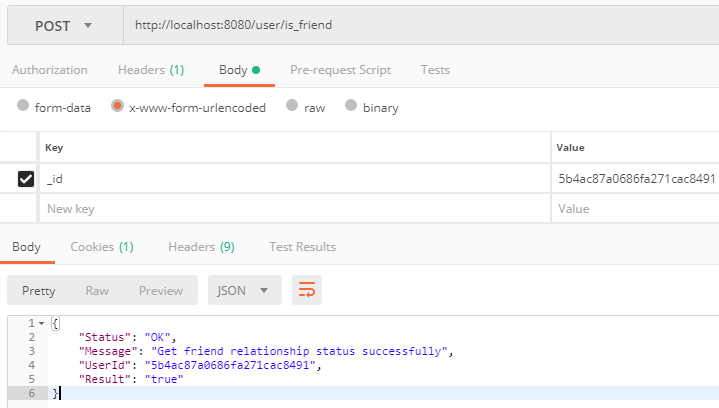
**Description**: Check if specified user is current user’s friend

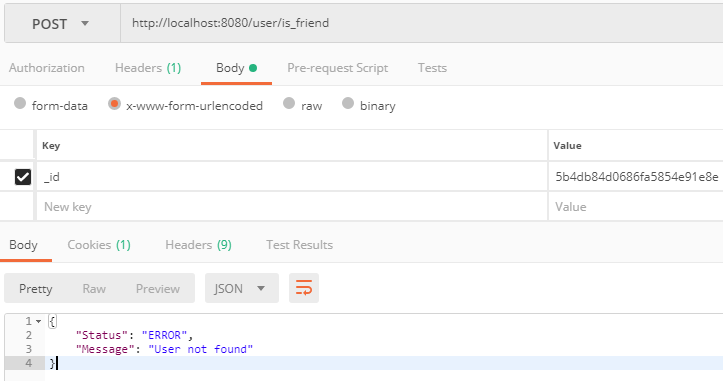
**Method**: POST

**RequestParams**:

\_id: User’s ID

**Return**: ResponseObject with additional field “UserId” and “Result” or ResponseObject if error





* **/user/is\_friend\_request\_sent**

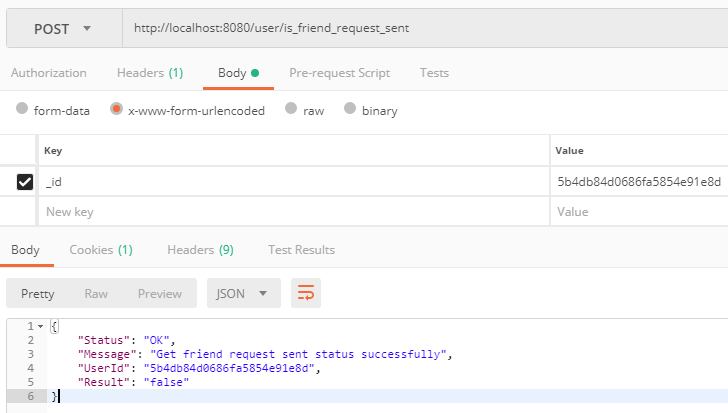
**Description**: Check if specified user has received current user’s friend request

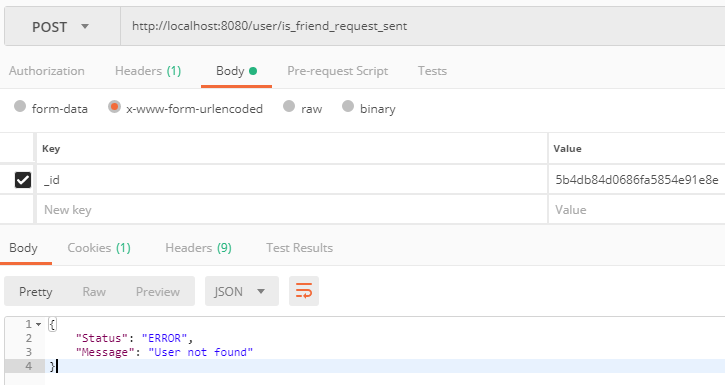
**Method**: POST

**RequestParams**:

\_id: User’s ID

**Return**: ResponseObject with additional field “UserId” and “Result” or ResponseObject if error





* **/user/is\_friend\_request\_received**

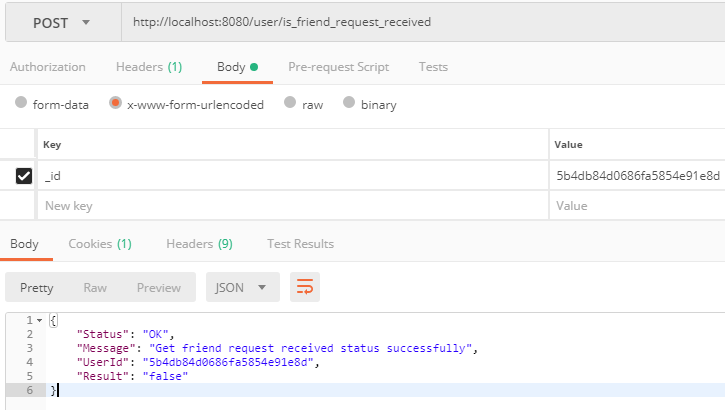
**Description**: Check if current user has received specified user’s friend request

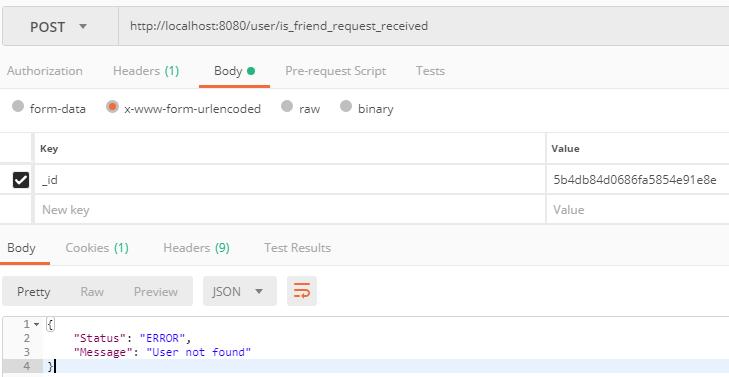
**Method**: POST

**RequestParams**:

\_id: User’s ID

**Return**: ResponseObject with additional field “UserId” and “Result” or ResponseObject if error





* **/user/friend\_request\_process**

**Description**: Used to process friend request

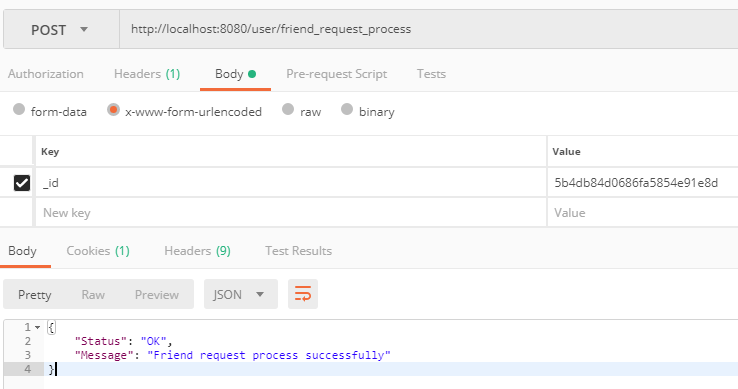
**Method**: POST

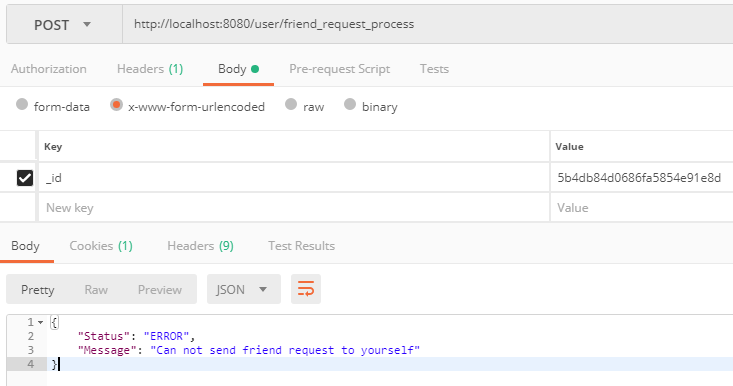
**RequestParams**:

\_id: User’s ID. If friend request sent, then cancel it. If friend request received

accept (not required): used to accept or refuse friend request from another user, accecpt=true, refuse=false

**Return**: ResponseObject





* **/user/{user\_id}**

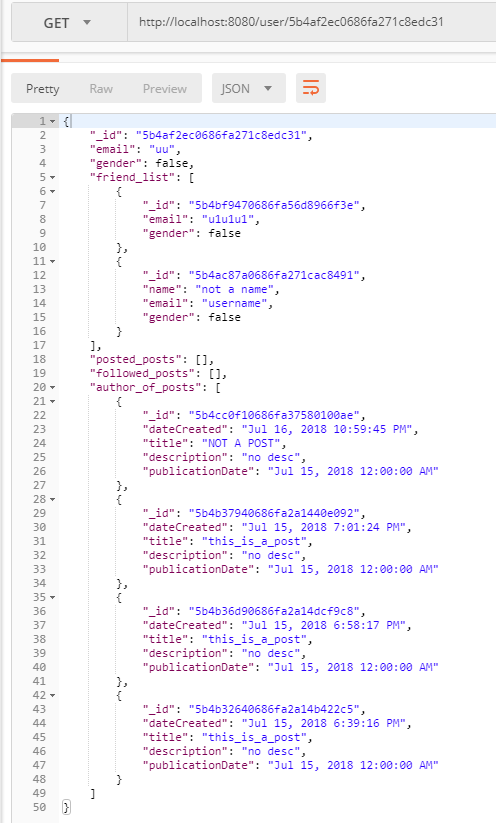
**Description**: Get user information

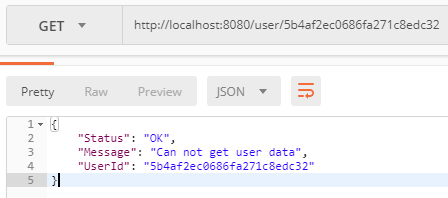
**Method**: GET

**RequestParams**:

User\_id: User’s ID

**Return**: AccountViewModel or ResponseObject with additional filed “UserId” if error





* Added /delete/post, delete/comment, added list of friend requested by in AccountViewModel