**Social media app rest api**

**Author: Nikos Alveniotis**

Posts rest api:

|  |  |  |  |
| --- | --- | --- | --- |
| Description | Url | RequestBody | `Request Params |
| Sign up user | POST /api/auth/signup | com.nikos.posts.payload.request.LoginRequest |  |
| User Login | POST /api/auth/signin | com.nikos.posts.payload.request.SignupRequest |  |
| Get Refresh Token | POST /api/auth/refreshtoken | com.nikos.posts.payload.request.TokenRefreshRequest |  |
| Logout | POST /api/auth/logout | com.nikos.posts.payload.request.LogoutRequest |  |
| Get Posts with pagination | GET /api/posts |  | pageNumber default=0  pageSize default=25 |
| Get Posts with pagination sorted by updateTime descending | GET /api/posts/sorted | N/A | pageNumber default=0  pageSize default=25 |
| Get a user posts by its email with pagination | GET /api/posts/users/{username} | N/A | pageNumber default=0  pageSize default=25 |
| Get my posts with pagination | GET /api/posts/mine | N/A | pageNumber default=0  pageSize default=25 |
| Get comments by post id with pagination | GET /api/posts/{id}/comments | N/A | pageNumber default=0  pageSize default=25 |
| Save a new Post | POST /api/posts | com.nikos.posts.dto.PostDto |  |
| Update a post | PUT /api/posts/{id} | com.nikos.posts.dto.PostDto |  |
| Delete a post | DELETE /api/posts/{id} | N/A |  |
| Get all Comments with pagination | GET /api/comments | N/A | pageNumber default=0  pageSize default=25 |
| Get all comments sorted by time updated descending with pagination | GET /api/comments/sorted | N/A | pageNumber default=0  pageSize default=25 |
| Save a new comment in the database | POST /api/comments | Com.nikos.posts.dto.CommentDto |  |
| Update a comment | PUT /api/comments/{id} | Com.nikos.posts.dto.CommentDto |  |
| Delete a comment | DELETE /api/comments{id} | N/A |  |

Database used: MySQL.

Cache used: Spring boot default implementation with annotations.

First run the app :

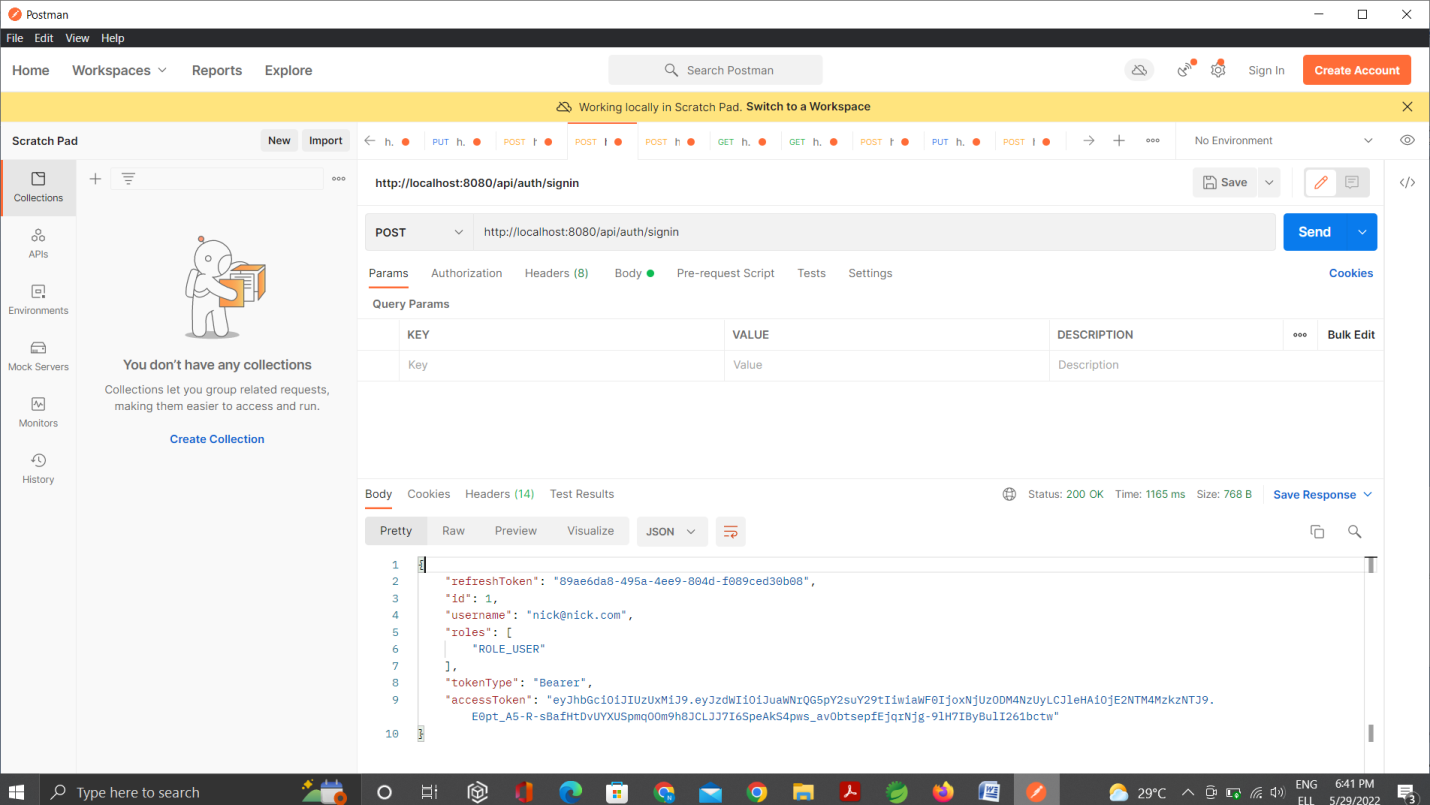
mvn spring-boot:run

Run the above sql statements:

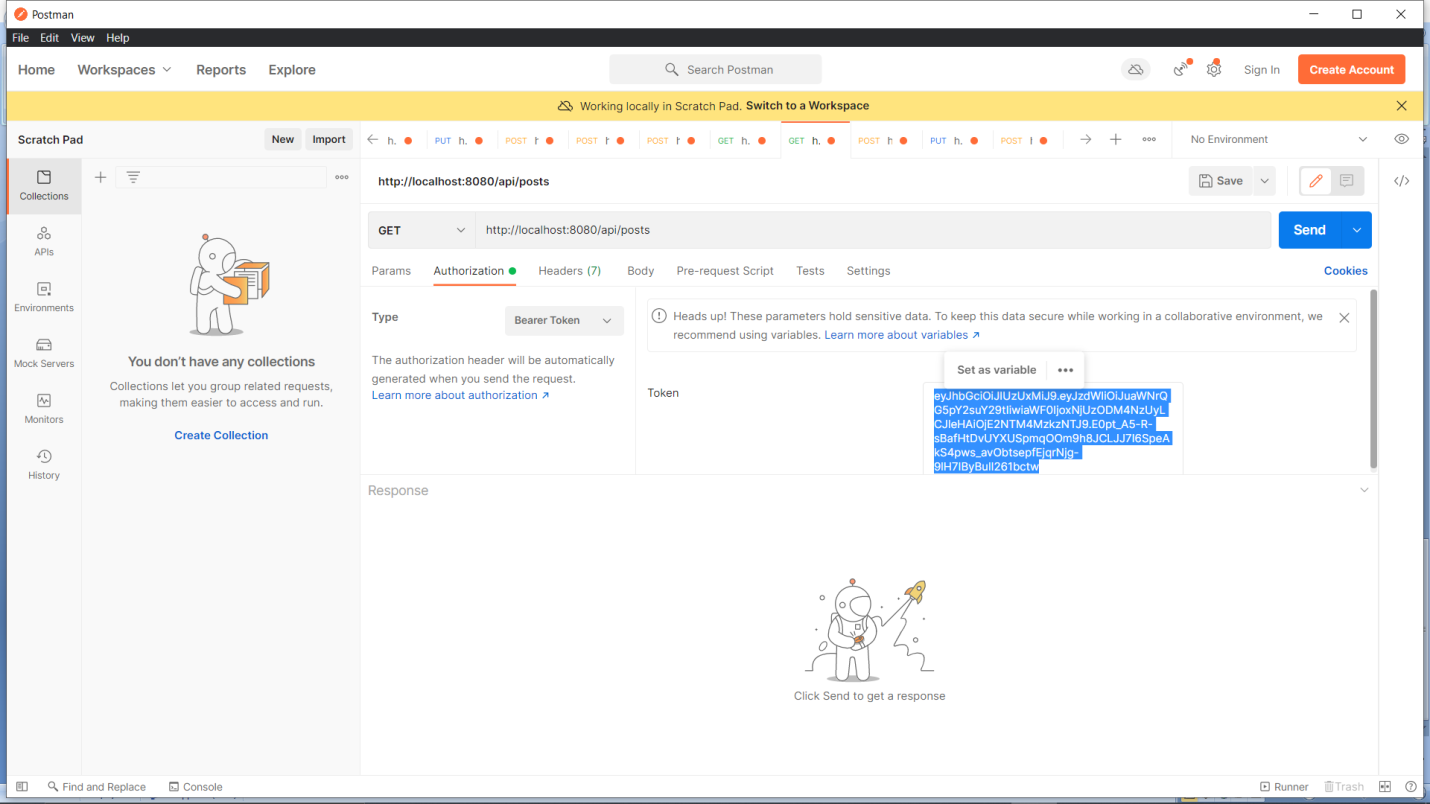
INSERT INTO roles(name) VALUES('ROLE\_USER');

INSERT INTO roles(name) VALUES('ROLE\_ADMIN');

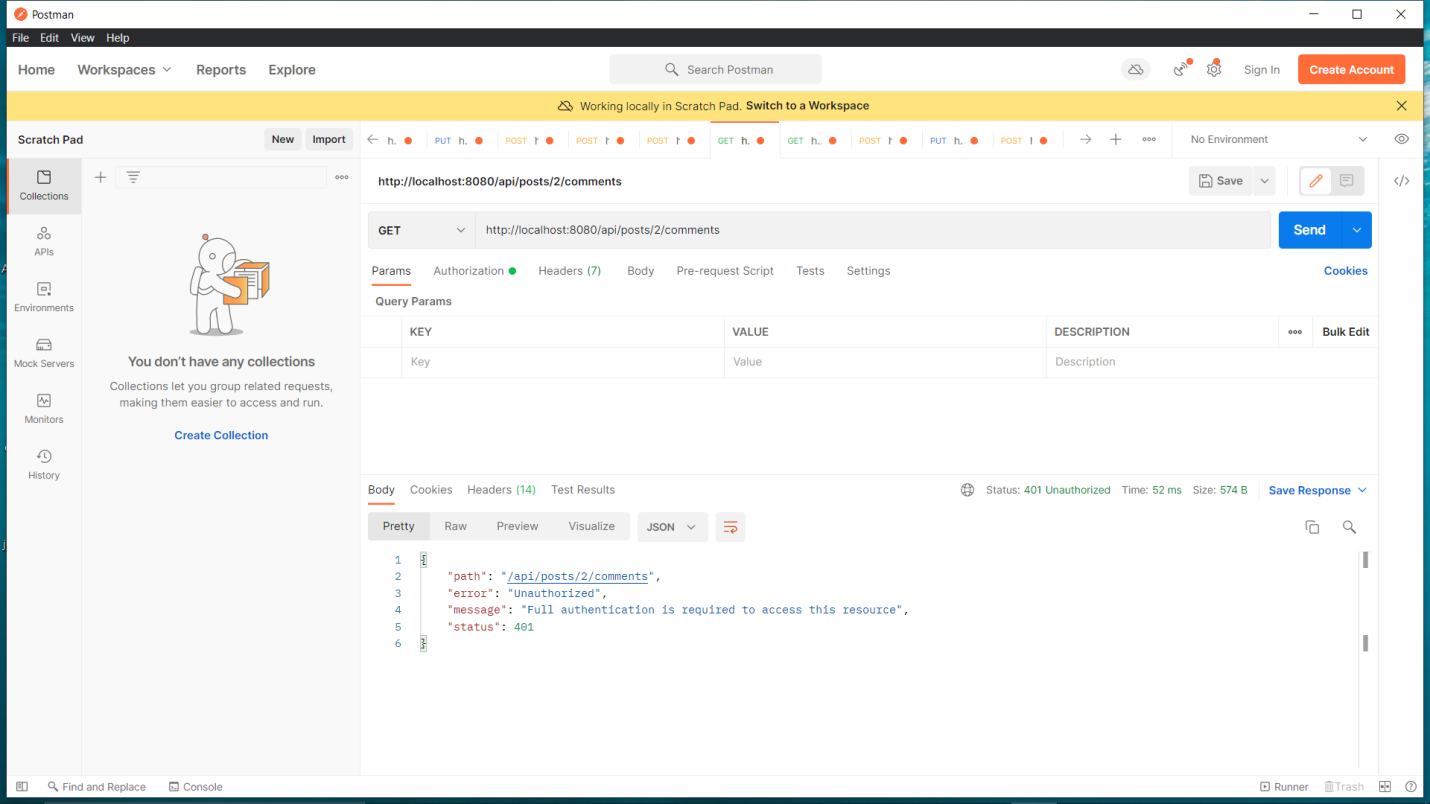
Signin User



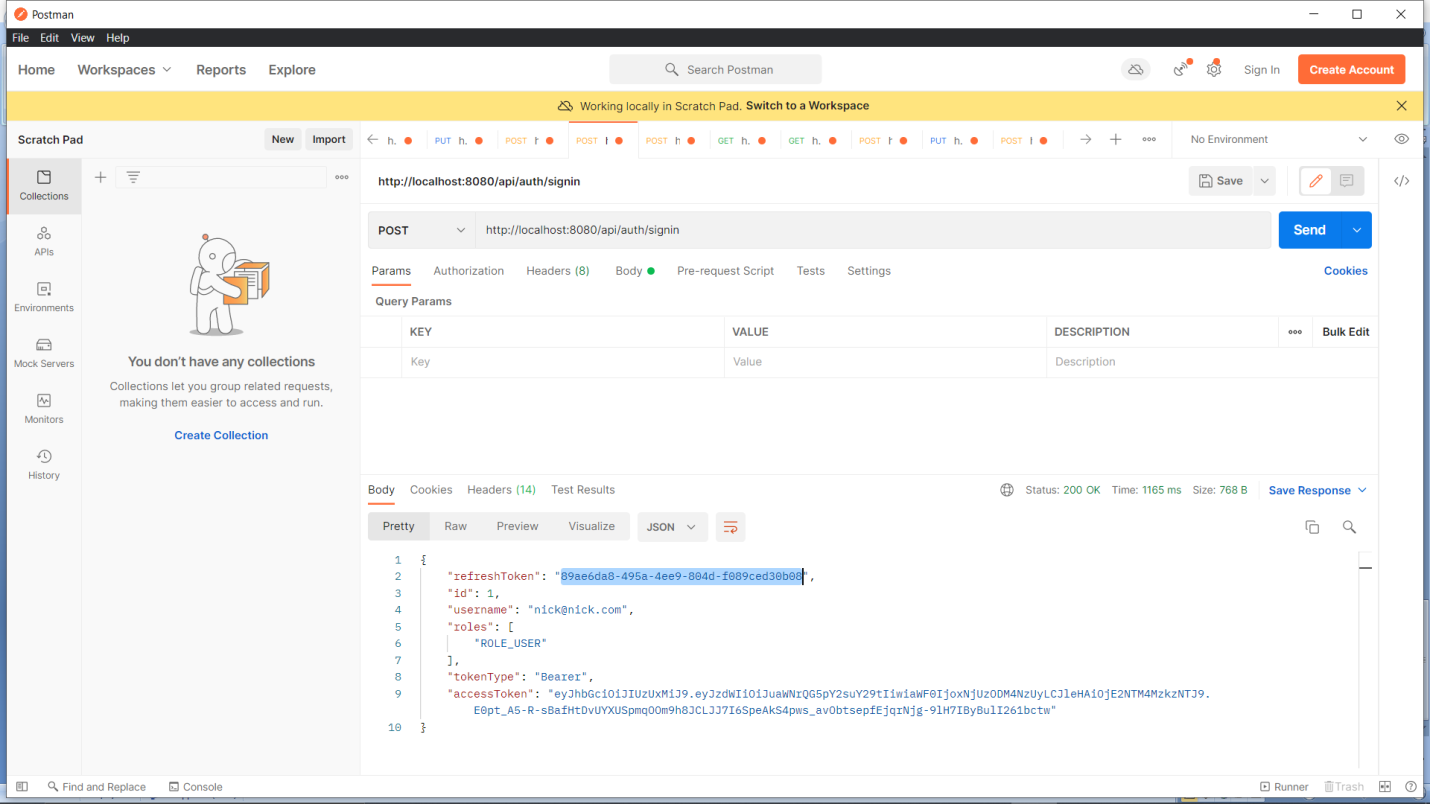
After you get the above screen copy the accessToken value and then in every subsequent request in a new tab select the Authorization tab and select Bearer Token from the dropdown and paste it in the Token field.



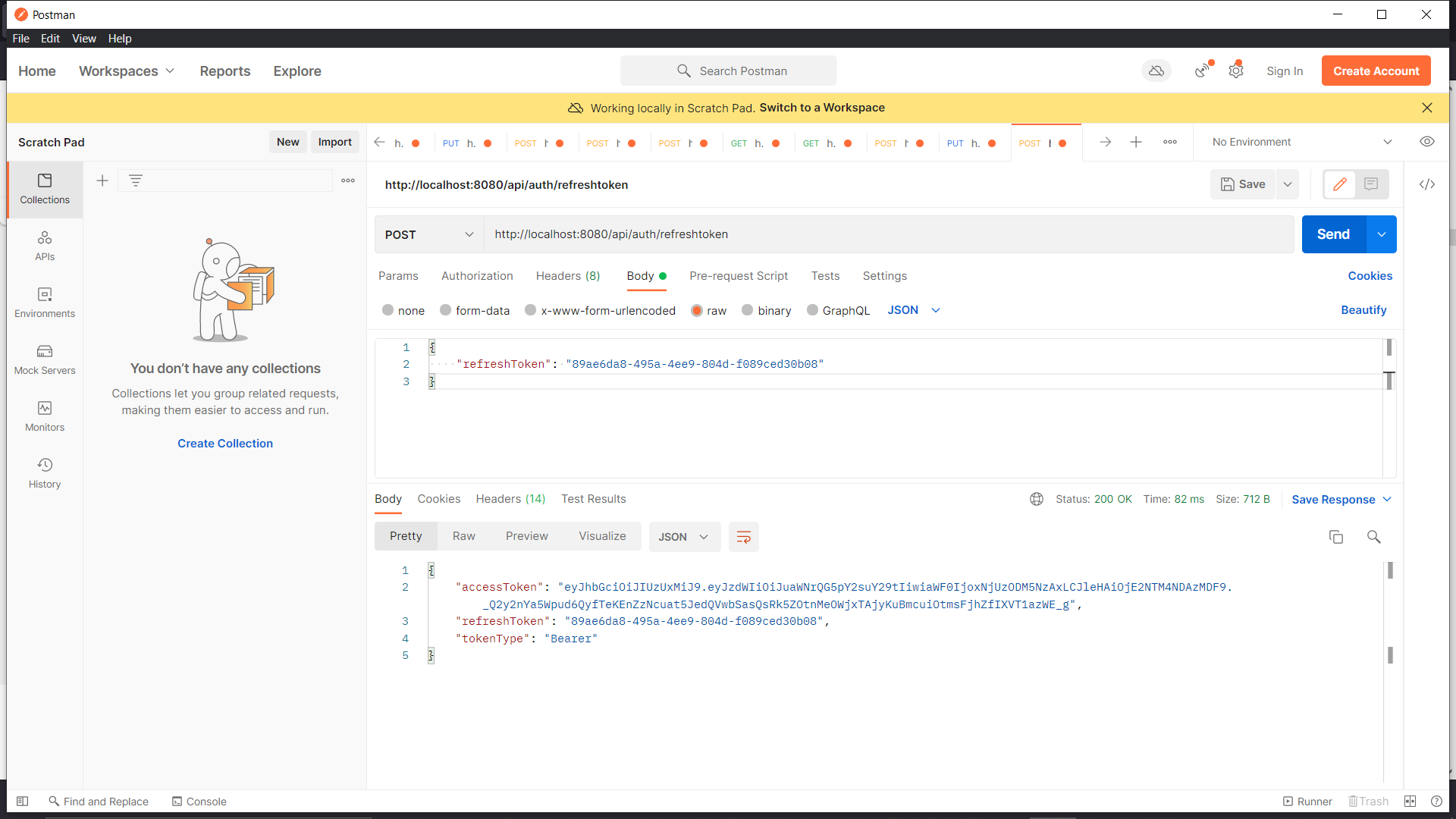
Then you will get the response normally. When the token expires you will get the following response in postman



So then you get the refreshToken we got after the first successful signin



And use it as request body in the POST /api/auth/refreshtoken request:



Then we get the the accessToken and use it in every subsequent request as described above.

Sample request for POST /api/auth/signup

{

    "username": "nick@nick.com",

    "password": "nick12345"

}

Sample request for Post /api/auth/signin

{

    "username": "nick@nick.com",

    "password": "nick12345"

}

Sample request for POST /api/auth/refreshtoken

{

    "refreshToken": "89ae6da8-495a-4ee9-804d-f089ced30b08"

}

Here the refreshToken value is whatever you get from the refreshToken field of the POST auth/api/signin request

Sample request for the logout

{

    "userId": 1

}

Endpoint for fetching posts with pagination GET /api/posts?pageNumber=0&pageSize=5

Endpoint for fetching posts with pagination sorted by the creation or if updated the update time:

GET /api/posts/sorted?pageNumber=0&pageSize=5

Endpoint for fetching posts by username with pagination sorted by the creation or if updated the update time :

GET /api/posts/users/{username}?pageNumber=0&pageSize=5

Endpoint for fetching logged in user’s posts with pagination sorted by the creation or if updated the update time:

GET /api/posts/mine?pageNumber=0&pageSize=5

Sample request for endpoint for saving posts. No username needed because logged in user’s username is used:

POST /api/posts

{

    "text": "A post!",

}

Sample request for updating posts endpoint. No username needed because logged in user’s username is used

PUT /api/posts/{id}

{

    "text": "A post!",

}

Delete posts endpoint:

DELETE /api/posts{id}

Comments

Endpoint for fetching comments with pagination:

GET /api/comments?pageNumber=0&pageSize=5

Endpoint for fetching comments with pagination sorted by the creation or if updated the update time:

GET /api/comments/sorted?pageNumber=0&pageSize=5

Endpoint for fetching comments by post with pagination sorted by the creation or if updated the update time. Here id is the post id :

GET /api/posts/{id}/comments?pageNumber=0&pageSize=5

Sample request for endpoint for saving comments. No username needed because logged in user’s username is used:

POST /api/comments

{

    "text": "Ole!!!",

    "postId": 6

}

Sample request for updating posts endpoint. No username needed because logged in user’s username is used

PUT /api/comments/{id}

{

    "text": "Ole!!!",

    "postId": 6

}

Delete comments endpoint:

DELETE /api/comments{id}