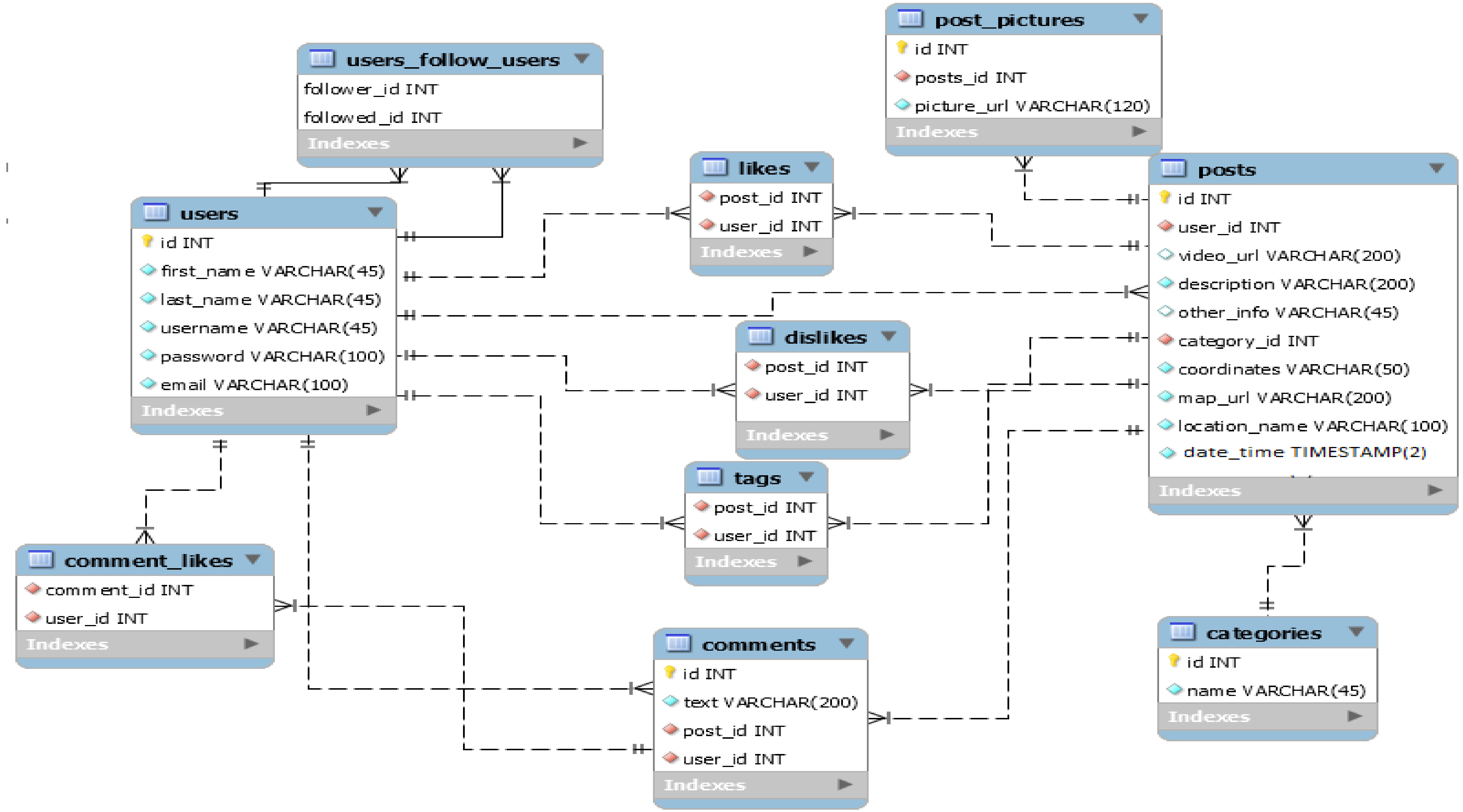
Traveller Online

Who are we?

What did we do?

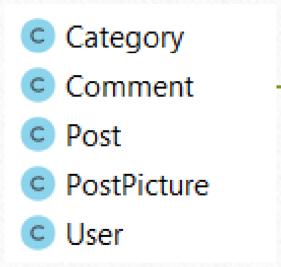
What did we use?

- Java
- Hibernate API
- Spring framework and Lombok
- MySQL
- Argon framework
- Postman
- MVC
- Git
- Intelij


users_follow_users
follower_id INT
followed_id INT
Indexes
post_pictures
id INT
posts_id INT
picture_url VARCHAR(120)
Indexes
likes
post_id INT
user_id INT
Indexes
posts
id INT
user_id INT
video_url VARCHAR(200)
description VARCHAR(200)
other_info VARCHAR(45)
category_id INT
coordinates VARCHAR(50)
map_url VARCHAR(200)
location_name VARCHAR(100)
date_time TIMESTAMP(2)
Indexes
users
id INT
first_name VARCHAR(45)
last_name VARCHAR(45)
username VARCHAR(45)
password VARCHAR(100)
email VARCHAR(100)
Indexes
dislikes
post_id INT
user_id INT
Indexes
tags
post_id INT
user_id INT
Indexes
comment_likes
comment_id INT
user_id INT
Indexes
comments
id INT
text VARCHAR(200)
post_id INT
user_id INT
Indexes
categories
id INT
name VARCHAR(45)
Indexes

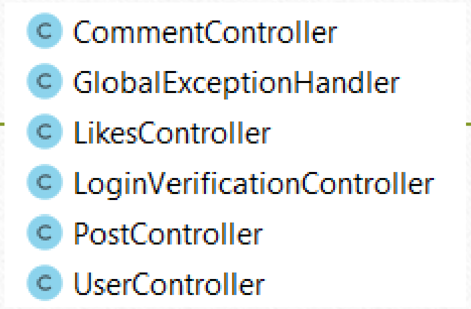
Class Diagrams

POJOs



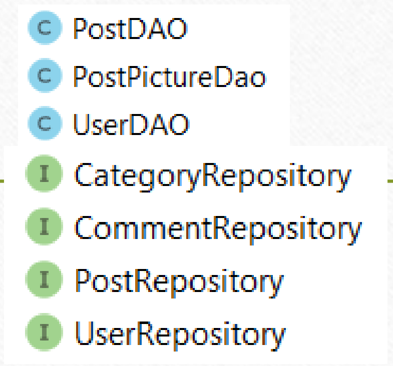
- Category
- Comment
- Post
- PostPicture
- User

Controllers



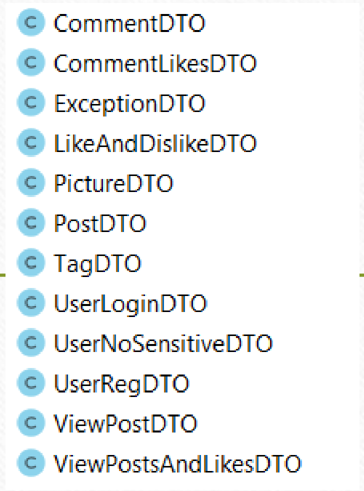
- CommentController
- GlobalExceptionHandler
- LikesController
- LoginVerificationController
- PostController
- UserController

DAOs & Repositories



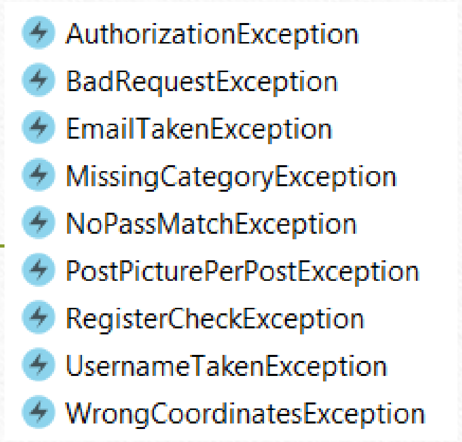
- PostDAO
- PostPictureDao
- UserDAO
- CategoryRepository
- CommentRepository
- PostRepository
- UserRepository

DTOs



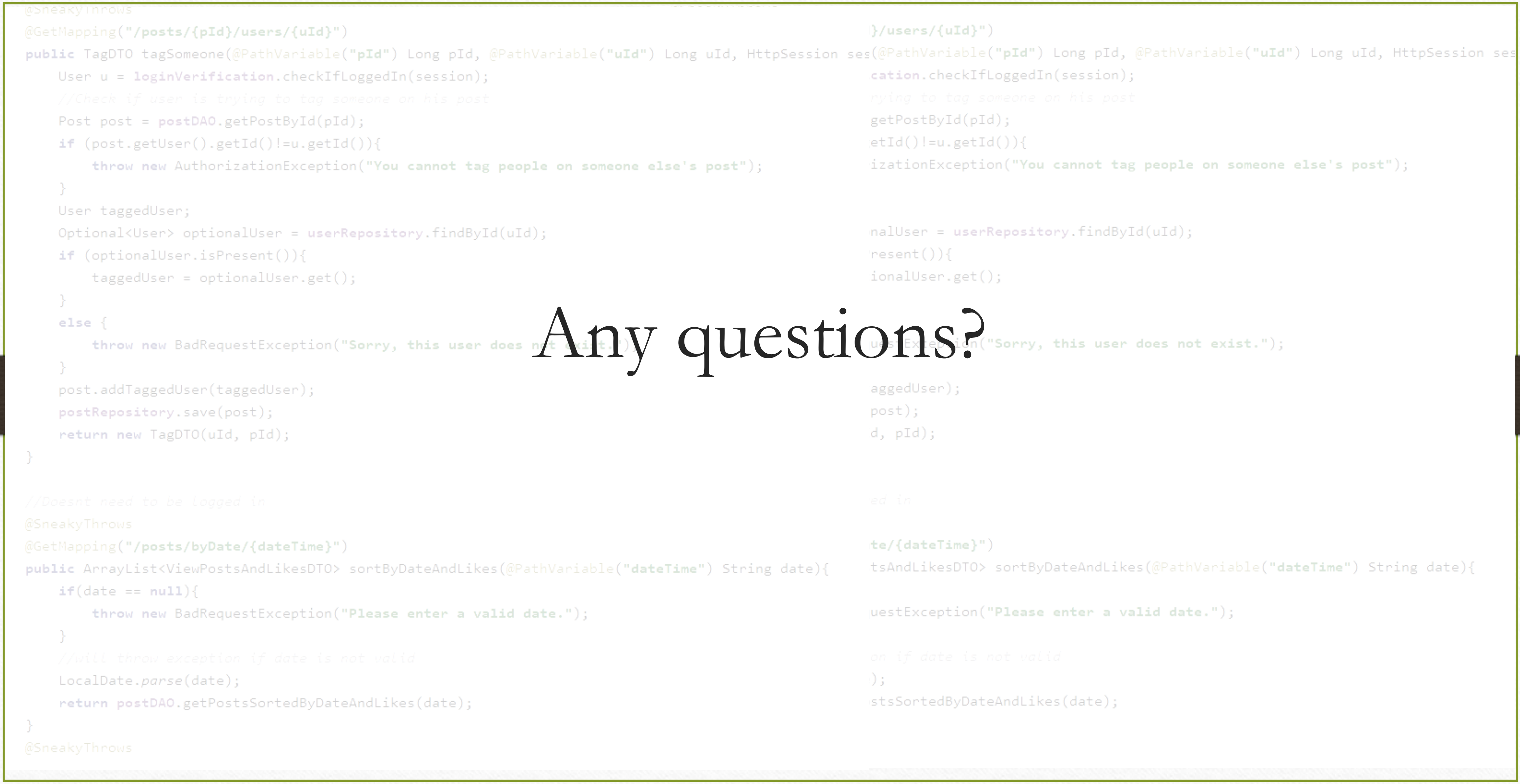
- CommentDTO
- CommentLikesDTO
- ExceptionDTO
- LikeAndDislikeDTO
- PictureDTO
- PostDTO
- TagDTO
- UserLoginDTO
- UserNoSensitiveDTO
- UserRegDTO
- ViewPostDTO
- ViewPostsAndLikesDTO

Exceptions



- AuthorizationException
- BadRequestException
- EmailTakenException
- MissingCategoryException
- NoPassMatchException
- PostPicturePerPostException
- RegisterCheckException
- UsernameTakenException
- WrongCoordinatesException

Let’s get started!



```
@SneakyThrows
@GetMapping("/posts/{pId}/users/{uId}")
public TagDTO tagSomeone(@PathVariable("pId") Long pId, @PathVariable("uId") Long uId, HttpSession ses
    User u = loginVerification.checkIfLoggedIn(session);
    //Check if user is trying to tag someone on his post
    Post post = postDAO.getPostById(pId);
    if (post.getUser().getId()!=u.getId()){
        throw new AuthorizationException("You cannot tag people on someone else's post");
    }
    User taggedUser;
    Optional<User> optionalUser = userRepository.findById(uId);
    if (optionalUser.isPresent()){
        taggedUser = optionalUser.get();
    }
    else {
        throw new BadRequestException("Sorry, this user does not exist.");
    }
    post.addTaggedUser(taggedUser);
    postRepository.save(post);
    return new TagDTO(uId, pId);
}

//Doesnt need to be logged in
@SneakyThrows
@GetMapping("/posts/byDate/{dateTime}")
public ArrayList<ViewPostsAndLikesDTO> sortByDateAndLikes(@PathVariable("dateTime") String date){
    if(date == null){
        throw new BadRequestException("Please enter a valid date.");
    }
    //will throw exception if date is not valid
    LocalDate.parse(date);
    return postDAO.getPostsSortedByDateAndLikes(date);
}
@SneakyThrows
```

```
}/users/{uId}")
(@PathVariable("pId") Long pId, @PathVariable("uId") Long uId, HttpSession ses
cation.checkIfLoggedIn(session);
rying to tag someone on his post
getPostById(pId);
etId()!=u.getId()){
izationException("You cannot tag people on someone else's post");

nalUser = userRepository.findById(uId);
resent()){
ionalUser.get();

uestException("Sorry, this user does not exist.");

aggedUser);
post);
d, pId);

ed in

te/{dateTime}")
tsAndLikesDTO> sortByDateAndLikes(@PathVariable("dateTime") String date){

uestException("Please enter a valid date.");

on if date is not valid
);
stsSortedByDateAndLikes(date);
```

Any questions?