
Youniversity Documentation

Release 1.0

The Youniversity Team

December 18, 2015

1	Getting started	1
1.1	Starting the web server	1
1.2	Configure Facebook login	1
2	API Documentation	3
2.1	auth	3
2.2	group	5
3	Modules Index	7
3.1	User package	7
3.2	authapi package	9
3.3	group package	10
3.4	newsfeed package	12
3.5	notification package	13
4	Indices and tables	17
	HTTP Routing Table	19
	Python Module Index	21

Getting started

1. Setup virtual environment:

```
virtualenv venv  
source venv/bin/activate
```

2. Setup requirements:

```
pip install -r requirements.txt
```

3. Sync database:

```
python manage.py migrate
```

4. Add user:

```
python manage.py createsuperuser
```

5. Run webserver:

```
python manage.py runserver
```

1.1 Starting the web server

1. Run webserver:

```
python manage.py runserver
```

2. Enter the front page at <http://localhost:8000/static/index.html>

1.2 Configure Facebook login

1. Enter Django administration page at <http://localhost:8000/admin/>
2. Click *Sites* then fix the first site name from `example.com` to `localhost:8000`
3. Go back to front page and click *Social applications*
4. Add new social application. Enter the following information:
 - Name: Facebook
 - Client ID, Secret Key: As given by Facebook ([Register a new Facebook application](#))

- Sites: Add `localhost:8000` to the list

API Documentation

The API root is at `/api`

2.1 auth

This endpoint is handled by `authapi.views`

GET /api/auth/check

Check whether the current user is logged in and retrieve information about the user.

This endpoint is handled by `authapi.views.UserViewSet.get()`

Example request:

```
GET /api/auth/check HTTP/1.1
Host: social.whs.in.th
Accept: application/json, text/javascript
Cookie: sessionid=.....
```

Example response when logged in:

```
HTTP/1.1 200 OK
Content-Type: application/json
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, OPTIONS

{"whs"}
```

Example response when not logged in:

```
HTTP/1.1 403 FORBIDDEN
Content-Type: application/json
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, OPTIONS

""
```

Response JSON Object

- **username** (*string*) – Username of current user

Status Codes

- **403 Forbidden** – User is not logged in

POST /api/auth/login

Authenticate user in by username/password combination. For Youniversity, it is usually used to authenticate against KU database via IMAP.

This endpoint is handled by `authapi.views.LoginViewSet.post()`

Example request:

```
POST /api/auth/login HTTP/1.1
Host: social.whs.in.th
Accept: application/json, text/plain, */*
Content-Type: application/json; charset=UTF-8

{"username": "example", "password": "example"}
```

Example of success response:

```
HTTP/1.0 200 OK
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Allow: POST, OPTIONS
Set-Cookie: csrftoken=Q0oxVmaGJUkyIV9tWuaLjl5yySa4HMcE; expires=Sun, 02-Oct-2016 09:37:27 GMT; M
Set-Cookie: sessionid=2cceti4ju0x6t3l8wl62awpdt16sp2p; expires=Sun, 18-Oct-2015 09:37:27 GMT; h

"whs"
```

Example of failed response:

```
HTTP/1.0 403 FORBIDDEN
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Allow: POST, OPTIONS

{"detail": "Cannot log you in"}
```

Request JSON Object

- **username** (*string*) – Username
- **password** (*string*) – Password

Response JSON Object

- **username** (*string*) – Username of current user

Status Codes

- **403 Forbidden** – Cannot log user in with given credentials. The credentials may be invalid, the server may not be able to connect to authentication database, or the user may be disabled.

POST /api/auth/logout

This endpoint is handled by `authapi.views.LogoutViewSet.post()`

Example request:

```
POST /api/auth/logout HTTP/1.1
Host: social.whs.in.th
Accept: application/json, text/javascript
Cookie: sessionid=.....
```

****Example response*:**


```
HTTP/1.1 200 OK
Content-Type: application/json
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, OPTIONS
```

2.2 group

This endpoint is handled by `group.views`

GET /api/group/

This endpoint is handled by `group.views.GroupList()`

Example request:

```
GET /api/group HTTP/1.1
Host: social.whs.in.th
Accept: application/json, text/javascript
Cookie: sessionid=.....
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, OPTIONS

[{"name": "test", "description": "Description text", "short_description": "Requirement to join group"}
```

Response JSON Array of Objects

- **name** (*string*) – Group name
- **description** (*string*) – What's this group about
- **short_description** (*string*) – Requirement to join group
- **type** (*int*) – 0 = normal, 1 = classroom

Status Codes

- 401 **Unauthorized** – Not authenticated

GET /api/group/ (int: id)

View basic information about a group

This endpoint is handled by `group.views.GroupViewDetail.get()`

Example request:

```
GET /api/group/1 HTTP/1.1
Host: social.whs.in.th
Accept: application/json, text/javascript
Cookie: sessionid=.....
```

Example response:

```
HTTP/1.1 200 OK
Content-Type: application/json
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, OPTIONS
```

```
{ "name": "test", "description": "Description text", "short_description": "Requirement to join group",
```

Response JSON Object

- **name** (*string*) – Group name
- **description** (*string*) – What's this group about
- **short_description** (*string*) – Requirement to join group
- **type** (*int*) – 0 = normal, 1 = classroom
- **member_status** (*int*) – -1 = not a member, 0 = request pending, 1 = member, 2 = administrator

Status Codes

- **404 Not Found** – No such group

Modules Index

3.1 User package

3.1.1 Submodules

3.1.2 User.models module

```
class User.models.UserProfile(id, user, firstname, lastname, birthday, gender, faculty, major, types,
                             country, city, created, picture, cover)
    Bases: django.db.models.base.Model

    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist

    exception UserProfile.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned

    UserProfile.get_gender_display(*moreargs, **morekwargs)

    UserProfile.objects = <django.db.models.manager.Manager object>

    UserProfile.user

User.models.create_profile(sender, instance, created, **kwargs)
    Create a matching profile whenever a user object is created.

User.models.user_cover_directory_path(instance, filename)

User.models.user_picture_directory_path(instance, filename)
```

3.1.3 User.serializers module

```
class User.serializers.FirstUserProfileSerializer(instance=None, data=<class
                                                  rest_framework.fields.empty>,
                                                  **kwargs)
    Bases: rest_framework.serializers.ModelSerializer

    class Meta

        fields = ('user', 'faculty', 'country', 'created')

    model
        alias of UserProfile
```

```
class User.serializers.FriendShipSerializer (instance=None, data=<class
rest_framework.fields.empty>, **kwargs)
Bases: rest_framework.serializers.ModelSerializer
```

```
class Meta
```

```
fields = ('user', 'created')
```

```
model
```

```
alias of Friend
```

```
class User.serializers.PictureField (read_only=False, write_only=False, required=None, de-
fault=<class rest_framework.fields.empty>, initial=<class
rest_framework.fields.empty>, source=None, label=None,
help_text=None, style=None, error_messages=None, val-
idators=None, allow_null=False)
```

```
Bases: rest_framework.fields.Field
```

```
to_representation (obj)
```

```
class User.serializers.UserCoverSerializer (instance=None, data=<class
rest_framework.fields.empty>, **kwargs)
Bases: rest_framework.serializers.ModelSerializer
```

```
class Meta
```

```
fields = ('cover',)
```

```
model
```

```
alias of UserProfile
```

```
class User.serializers.UserProfilePictureSerializer (instance=None, data=<class
rest_framework.fields.empty>,
**kwargs)
Bases: rest_framework.serializers.ModelSerializer
```

```
class Meta
```

```
fields = ('picture',)
```

```
model
```

```
alias of UserProfile
```

```
class User.serializers.UserProfileSerializer (instance=None, data=<class
rest_framework.fields.empty>, **kwargs)
Bases: rest_framework.serializers.ModelSerializer
```

```
class Meta
```

```
fields = ('user', 'firstname', 'lastname', 'birthday', 'gender', 'faculty', 'major', 'types', 'country', 'city', 'picture')
```

```
model
```

```
alias of UserProfile
```

```
read_only_fields = ('picture', 'cover')
```

```
class User.serializers.UserSerializer (instance=None, data=<class
rest_framework.fields.empty>, **kwargs)
Bases: rest_framework.serializers.ModelSerializer
```

```
class Meta
```

```
    fields = ('id', 'username', 'first_name', 'last_name', 'profile')
```

```
model
```

```
    alias of User
```

3.1.4 User.views module

3.2 authapi package

This module contains API related to retrieving current user information.

It should be merged into user module later once that is implemented.

3.2.1 Submodules

3.2.2 authapi.views module

```
class authapi.views.LoginViewSet (**kwargs)
```

```
    Bases: rest_framework.views.APIView
```

Log a user in with username/password combination

It could be accessed at *POST* `/api/auth/login`

```
post (request)
```

Log a user in

Parameters `request` – Django Rest Framework request object

Returns Username or 403

```
serializer_class
```

alias of *UserSerializer*

```
class authapi.views.LogoutViewSet (**kwargs)
```

```
    Bases: rest_framework.views.APIView
```

Logout of current session

It could be accessed at *POST* `/api/auth/logout`

```
post (request)
```

Destroy current session

Parameters `request` – Django Rest Framework request object

Returns 200

```
class authapi.views.UserViewSet (**kwargs)
```

```
    Bases: rest_framework.views.APIView
```

Validate current user session

Check whether the current user is logged in and retrieve information about the user.

It could be accessed at *GET* `/api/auth/check`

get (*request*)
Get logged user information

Parameters **request** – Django Rest Framework request object

Returns Current logged username as string, or 403

serializer_class
alias of UserSerializer

3.3 group package

3.3.1 Submodules

3.3.2 group.models module

class group.models.**Group** (*id, name, gtype, type, category, description, short_description, activities, per-*
mission, date, parent, cover)
Bases: django.db.models.base.Model

exception DoesNotExist
Bases: django.core.exceptions.ObjectDoesNotExist

exception Group.MultipleObjectsReturned
Bases: django.core.exceptions.MultipleObjectsReturned

Group.**category**

Group.**children**

Group.**get_next_by_date** (*moreargs, **morekwargs)

Group.**get_previous_by_date** (*moreargs, **morekwargs)

Group.**groupmember_set**

Group.**objects** = <django.db.models.manager.Manager object>

Group.**parent**

class group.models.**GroupCategory** (*id, name*)
Bases: django.db.models.base.Model

exception DoesNotExist
Bases: django.core.exceptions.ObjectDoesNotExist

exception GroupCategory.MultipleObjectsReturned
Bases: django.core.exceptions.MultipleObjectsReturned

GroupCategory.**group_set**

GroupCategory.**objects** = <django.db.models.manager.Manager object>

class group.models.**GroupMember** (*id, group, user, role*)
Bases: django.db.models.base.Model

exception DoesNotExist
Bases: django.core.exceptions.ObjectDoesNotExist

exception GroupMember.MultipleObjectsReturned
Bases: django.core.exceptions.MultipleObjectsReturned

```

GroupMember.create(new_group, new_user)

GroupMember.group

GroupMember.objects = <django.db.models.manager.Manager object>

GroupMember.user

group.models.group_cover_directory_path(instance, filename)

```

3.3.3 group.serializers module

```

class group.serializers.GroupCategorySerializer(instance=None, data=<class
rest_framework.fields.empty>, **kwargs)

```

```

Bases: rest_framework.serializers.ModelSerializer

```

```

class Meta

```

```

    app_label = 'group_category'

```

```

    fields = ('id', 'name')

```

```

    model

```

```

        alias of GroupCategory

```

```

class group.serializers.GroupCoverSerializer(instance=None, data=<class
rest_framework.fields.empty>, **kwargs)

```

```

Bases: rest_framework.serializers.ModelSerializer

```

```

class Meta

```

```

    app_label = 'social_group'

```

```

    fields = ('cover',)

```

```

    model

```

```

        alias of Group

```

```

class group.serializers.GroupMemberSerializer(instance=None, data=<class
rest_framework.fields.empty>, **kwargs)

```

```

Bases: rest_framework.serializers.ModelSerializer

```

```

class Meta

```

```

    fields = ('user', 'role')

```

```

    model

```

```

        alias of GroupMember

```

```

class group.serializers.GroupSerializer(instance=None, data=<class
rest_framework.fields.empty>, **kwargs)

```

```

Bases: rest_framework.serializers.ModelSerializer

```

```

class Meta

```

```

    app_label = 'social_group'

```

```

    extra_kwargs = {'category': {'required': False}}

```

```

    fields = ('id', 'name', 'description', 'short_description', 'activities', 'type', 'category', 'member_status', 'date', 'n

```

```
    model
        alias of Group
    read_only_fields = ('cover',)
class group.serializers.SubGroupSerializer (instance=None, data=<class
                                         rest_framework.fields.empty>, **kwargs)
    Bases: rest_framework.serializers.ModelSerializer
class Meta

    app_label = 'subgroup'
    fields = ('id', 'name')
    model
        alias of Group
class group.serializers.UserSerializer (instance=None, data=<class
                                         rest_framework.fields.empty>, **kwargs)
    Bases: rest_framework.serializers.ModelSerializer
class Meta

    fields = ('id', 'username', 'first_name', 'last_name', 'profile')
    model
        alias of User
```

3.3.4 group.views module

3.4 newfeed package

3.4.1 Submodules

3.4.2 newfeed.models module

```
class newfeed.models.Comment (id, post, user, text, datetime, file)
    Bases: django.db.models.base.Model
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    exception Comment.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned
    Comment.get_next_by_datetime (*moreargs, **morekwargs)
    Comment.get_previous_by_datetime (*moreargs, **morekwargs)
    Comment.objects = <django.db.models.manager.Manager object>
    Comment.post
    Comment.user
class newfeed.models.Post (id, user, text, datetime, target_type, target_id, target_name, al-
                           low_submission, pinned)
    Bases: django.db.models.base.Model
```


exception DoesNotExistBases: `django.core.exceptions.ObjectDoesNotExist``Post.FORMAT()`**exception Post.MultipleObjectsReturned**Bases: `django.core.exceptions.MultipleObjectsReturned``Post.comment_set``Post.get_next_by_datetime(*moreargs, **morekwargs)``Post.get_previous_by_datetime(*moreargs, **morekwargs)``Post.objects = <django.db.models.manager.Manager object>``Post.target_object`

Provides a generic relation to any object through content-type/object-id fields.

`Post.target_type``Post.user``newsfeed.models.comment_file_name(instance, filename)`

3.4.3 newsfeed.serializer module

3.4.4 newsfeed.views module

3.5 notification package

3.5.1 Submodules

3.5.2 notification.models module

class `notification.models.Notification` (*id, user, datetime, text, target_type, target_id, link_type, link_item, reference_detail*)Bases: `django.db.models.base.Model`**exception DoesNotExist**Bases: `django.core.exceptions.ObjectDoesNotExist`**exception Notification.MultipleObjectsReturned**Bases: `django.core.exceptions.MultipleObjectsReturned``Notification.get_next_by_datetime(*moreargs, **morekwargs)``Notification.get_previous_by_datetime(*moreargs, **morekwargs)``Notification.link_type``Notification.objects = <django.db.models.manager.Manager object>``Notification.readed``Notification.receiver``Notification.target_object`

Provides a generic relation to any object through content-type/object-id fields.

`Notification.target_type`

```
Notification.user
Notification.usernotification_set
class notification.models.UserNotification(id, notification, receiver)
    Bases: django.db.models.base.Model
    exception DoesNotExist
        Bases: django.core.exceptions.ObjectDoesNotExist
    exception UserNotification.MultipleObjectsReturned
        Bases: django.core.exceptions.MultipleObjectsReturned
    UserNotification.notification
    UserNotification.objects = <django.db.models.manager.Manager object>
    UserNotification.receiver
```

3.5.3 notification.serializer module

```
class notification.serializer.GetNotificationSerializer(instance=None, data=<class
    rest_framework.fields.empty>,
    **kwargs)
    Bases: rest_framework.serializers.ModelSerializer
    class Meta
        fields = ('id', 'user', 'datetime', 'text', 'target_type', 'target_id', 'link_type', 'link_item', 'reference_detail', 'receiver')
        model = Notification
            alias of Notification
class notification.serializer.NotificationSerializer(instance=None, data=<class
    rest_framework.fields.empty>,
    **kwargs)
    Bases: rest_framework.serializers.ModelSerializer
    class Meta
        fields = ('id', 'user', 'datetime', 'text', 'target_type', 'target_id', 'link_type', 'link_item', 'reference_detail', 'receiver')
        model = Notification
            alias of Notification
class notification.serializer.TypeSerializer(instance=None, data=<class
    rest_framework.fields.empty>, **kwargs)
    Bases: rest_framework.serializers.ModelSerializer
    class Meta
        fields = ('id', 'model')
        model = ContentType
            alias of ContentType
```

```

class notification.serializer.UpdateNotificationSerializer (instance=None,
                                                         data=<class
                                                         rest_framework.fields.empty>,
                                                         **kwargs)

Bases: rest_framework.serializers.ModelSerializer

class Meta

    fields = ('id', 'user', 'readed')

    model
        alias of Notification

class notification.serializer.UserNotificationSerializer (instance=None,
                                                         data=<class
                                                         rest_framework.fields.empty>,
                                                         **kwargs)

Bases: rest_framework.serializers.ModelSerializer

class Meta

    fields = ('id', 'username', 'first_name', 'last_name')

    model
        alias of User

UserNotificationSerializer.auto_created = True

class notification.serializer.UserProfileSerializer (instance=None, data=<class
                                                         rest_framework.fields.empty>,
                                                         **kwargs)

Bases: rest_framework.serializers.ModelSerializer

class Meta

    fields = ('picture',)

    model
        alias of UserProfile

class notification.serializer.UserSerializer (instance=None, data=<class
                                                         rest_framework.fields.empty>, **kwargs)

Bases: rest_framework.serializers.ModelSerializer

class Meta

    fields = ('id', 'username', 'first_name', 'last_name', 'profile')

    model
        alias of User

```

3.5.4 notification.views module

```

class notification.views.NotificationView (**kwargs)
    Bases: rest_framework.views.APIView

    This class is an API for get specific notification.

```

get (*request*)

Get notification of specific user. :param request: Django Rest Framework request object. :param format: pattern for Web APIs.

Returns list of notification of that user.

serializer_class

alias of GetNotificationSerializer

class notification.views.**NotificationViewList** (***kwargs*)

Bases: rest_framework.views.APIView

This class is an API for create and get all notification.

add (*user, data, receiver_set, type, link_item, reference_detail, format=None*)

Create and save notification to database. :param user: user who create notification. :param data: Json information of target of post :param receiver_set: set of receiver who will receive

this notification.

Parameters

- **type** – type of action that create this notification.
- **link_item** – information of action that create this notification.
- **reference_detail** – information of the post target object
- **format** – pattern for Web APIs.

Return:

get (*request, format=None*)

Get all of notification in database. :param request: Django Rest Framework request object. :param format: pattern for Web APIs.

Returns list of all notification in database.

serializer_class

alias of NotificationSerializer

class notification.views.**UpdateNotification** (***kwargs*)

Bases: rest_framework.views.APIView

This class is an API for update readed notification.

get (*request, noti_id, format=None*)

Get notification in database and mark as readed. :param request: Django Rest Framework request object. :param noti_id: id of notification. :param format: pattern for Web APIs.

Returns updated notificaton.

get_object (*noti_id*)

Get notification object by id. :param noti_id: id of notification.

Returns notification object.

serializer_class

alias of UpdateNotificationSerializer

update_read (*noti_id, format=None*)

Update notification to readed by id. :param noti_if: id of notification. :param format: pattern for Web APIs.

Return:

Indices and tables

- `genindex`
- `modindex`
- `search`
- HTTP Routing Table

/api

GET /api/auth/check,3
GET /api/group/,5
GET /api/group/(int:id),5
POST /api/auth/login,3
POST /api/auth/logout,4

a

`authapi`, 9
`authapi.views`, 9

g

`group`, 10
`group.models`, 10
`group.serializers`, 11

n

`newsfeed`, 12
`newsfeed.models`, 12
`notification`, 13
`notification.models`, 13
`notification.serializer`, 14
`notification.views`, 15

u

`User`, 7
`User.models`, 7
`User.serializers`, 7

A

add() (notification.views.NotificationViewList method), 16

app_label (group.serializers.GroupCategorySerializer.Meta attribute), 11

app_label (group.serializers.GroupCoverSerializer.Meta attribute), 11

app_label (group.serializers.GroupSerializer.Meta attribute), 11

app_label (group.serializers.SubGroupSerializer.Meta attribute), 12

authapi (module), 9

authapi.views (module), 9

auto_created (notification.serializer.UserNotificationSerializer attribute), 15

C

category (group.models.Group attribute), 10

children (group.models.Group attribute), 10

Comment (class in newsfeed.models), 12

Comment.DoesNotExist, 12

Comment.MultipleObjectsReturned, 12

comment_file_name() (in module newsfeed.models), 13

comment_set (newsfeed.models.Post attribute), 13

create() (group.models.GroupMember method), 10

create_profile() (in module User.models), 7

E

extra_kwargs (group.serializers.GroupSerializer.Meta attribute), 11

F

fields (group.serializers.GroupCategorySerializer.Meta attribute), 11

fields (group.serializers.GroupCoverSerializer.Meta attribute), 11

fields (group.serializers.GroupMemberSerializer.Meta attribute), 11

fields (group.serializers.GroupSerializer.Meta attribute), 11

fields (group.serializers.SubGroupSerializer.Meta attribute), 12

fields (group.serializers.UserSerializer.Meta attribute), 12

fields (notification.serializer.GetNotificationSerializer.Meta attribute), 14

fields (notification.serializer.NotificationSerializer.Meta attribute), 14

fields (notification.serializer.TypeSerializer.Meta attribute), 14

fields (notification.serializer.UpdateNotificationSerializer.Meta attribute), 15

fields (notification.serializer.UserNotificationSerializer.Meta attribute), 15

fields (notification.serializer.UserProfileSerializer.Meta attribute), 15

fields (notification.serializer.UserSerializer.Meta attribute), 15

fields (User.serializers.FirstUserProfileSerializer.Meta attribute), 7

fields (User.serializers.FriendShipSerializer.Meta attribute), 8

fields (User.serializers.UserCoverSerializer.Meta attribute), 8

fields (User.serializers.UserProfilePictureSerializer.Meta attribute), 8

fields (User.serializers.UserProfileSerializer.Meta attribute), 8

fields (User.serializers.UserSerializer.Meta attribute), 9

FirstUserProfileSerializer (class in User.serializers), 7

FirstUserProfileSerializer.Meta (class in User.serializers), 7

FORMAT() (newsfeed.models.Post method), 13

FriendShipSerializer (class in User.serializers), 7

FriendShipSerializer.Meta (class in User.serializers), 8

G

get() (authapi.views.UserViewSet method), 9

get() (notification.views.NotificationView method), 15

get() (notification.views.NotificationViewList method), 16

get() (notification.views.UpdateNotification method), 16

[get_gender_display\(\) \(User.models.UserProfile method\), 7](#)
[get_next_by_date\(\) \(group.models.Group method\), 10](#)
[get_next_by_datetime\(\) \(newsfeed.models.Comment method\), 12](#)
[get_next_by_datetime\(\) \(newsfeed.models.Post method\), 13](#)
[get_next_by_datetime\(\) \(notification.models.Notification method\), 13](#)
[get_object\(\) \(notification.views.UpdateNotification method\), 16](#)
[get_previous_by_date\(\) \(group.models.Group method\), 10](#)
[get_previous_by_datetime\(\) \(newsfeed.models.Comment method\), 12](#)
[get_previous_by_datetime\(\) \(newsfeed.models.Post method\), 13](#)
[get_previous_by_datetime\(\) \(notification.models.Notification method\), 13](#)
[GetNotificationSerializer \(class in notification.serializer\), 14](#)
[GetNotificationSerializer.Meta \(class in notification.serializer\), 14](#)
[Group \(class in group.models\), 10](#)
[group \(group.models.GroupMember attribute\), 11](#)
[group \(module\), 10](#)
[Group.DoesNotExist, 10](#)
[group.models \(module\), 10](#)
[Group.MultipleObjectsReturned, 10](#)
[group.serializers \(module\), 11](#)
[group_cover_directory_path\(\) \(in module group.models\), 11](#)
[group_set \(group.models.GroupCategory attribute\), 10](#)
[GroupCategory \(class in group.models\), 10](#)
[GroupCategory.DoesNotExist, 10](#)
[GroupCategory.MultipleObjectsReturned, 10](#)
[GroupCategorySerializer \(class in group.serializers\), 11](#)
[GroupCategorySerializer.Meta \(class in group.serializers\), 11](#)
[GroupCoverSerializer \(class in group.serializers\), 11](#)
[GroupCoverSerializer.Meta \(class in group.serializers\), 11](#)
[GroupMember \(class in group.models\), 10](#)
[GroupMember.DoesNotExist, 10](#)
[GroupMember.MultipleObjectsReturned, 10](#)
[groupmember_set \(group.models.Group attribute\), 10](#)
[GroupMemberSerializer \(class in group.serializers\), 11](#)
[GroupMemberSerializer.Meta \(class in group.serializers\), 11](#)
[GroupSerializer \(class in group.serializers\), 11](#)
[GroupSerializer.Meta \(class in group.serializers\), 11](#)

L

[link_type \(notification.models.Notification attribute\), 13](#)

[LoginViewSet \(class in authapi.views\), 9](#)
[LogoutViewSet \(class in authapi.views\), 9](#)

M

[model \(group.serializers.GroupCategorySerializer.Meta attribute\), 11](#)
[model \(group.serializers.GroupCoverSerializer.Meta attribute\), 11](#)
[model \(group.serializers.GroupMemberSerializer.Meta attribute\), 11](#)
[model \(group.serializers.GroupSerializer.Meta attribute\), 11](#)
[model \(group.serializers.SubGroupSerializer.Meta attribute\), 12](#)
[model \(group.serializers.UserSerializer.Meta attribute\), 12](#)
[model \(notification.serializer.GetNotificationSerializer.Meta attribute\), 14](#)
[model \(notification.serializer.NotificationSerializer.Meta attribute\), 14](#)
[model \(notification.serializer.TypeSerializer.Meta attribute\), 14](#)
[model \(notification.serializer.UpdateNotificationSerializer.Meta attribute\), 15](#)
[model \(notification.serializer.UserNotificationSerializer.Meta attribute\), 15](#)
[model \(notification.serializer.UserProfileSerializer.Meta attribute\), 15](#)
[model \(notification.serializer.UserSerializer.Meta attribute\), 15](#)
[model \(User.serializers.FirstUserProfileSerializer.Meta attribute\), 7](#)
[model \(User.serializers.FriendShipSerializer.Meta attribute\), 8](#)
[model \(User.serializers.UserCoverSerializer.Meta attribute\), 8](#)
[model \(User.serializers.UserProfilePictureSerializer.Meta attribute\), 8](#)
[model \(User.serializers.UserProfileSerializer.Meta attribute\), 8](#)
[model \(User.serializers.UserSerializer.Meta attribute\), 9](#)

N

[newsfeed \(module\), 12](#)
[newsfeed.models \(module\), 12](#)
[Notification \(class in notification.models\), 13](#)
[notification \(module\), 13](#)
[notification \(notification.models.UserNotification attribute\), 14](#)
[Notification.DoesNotExist, 13](#)
[notification.models \(module\), 13](#)
[Notification.MultipleObjectsReturned, 13](#)
[notification.serializer \(module\), 14](#)
[notification.views \(module\), 15](#)

NotificationSerializer (class in notification.serializer), 14
 NotificationSerializer.Meta (class in notification.serializer), 14
 NotificationView (class in notification.views), 15
 NotificationViewList (class in notification.views), 16

O

objects (group.models.Group attribute), 10
 objects (group.models.GroupCategory attribute), 10
 objects (group.models.GroupMember attribute), 11
 objects (newsfeed.models.Comment attribute), 12
 objects (newsfeed.models.Post attribute), 13
 objects (notification.models.Notification attribute), 13
 objects (notification.models.UserNotification attribute), 14
 objects (User.models.UserProfile attribute), 7

P

parent (group.models.Group attribute), 10
 PictureField (class in User.serializers), 8
 Post (class in newsfeed.models), 12
 post (newsfeed.models.Comment attribute), 12
 post() (authapi.views.LoginViewSet method), 9
 post() (authapi.views.LogoutViewSet method), 9
 Post.DoesNotExist, 12
 Post.MultipleObjectsReturned, 13

R

read_only_fields (group.serializers.GroupSerializer.Meta attribute), 12
 read_only_fields (User.serializers.UserProfileSerializer.Meta attribute), 8
 readed (notification.models.Notification attribute), 13
 receiver (notification.models.Notification attribute), 13
 receiver (notification.models.UserNotification attribute), 14

S

serializer_class (authapi.views.LoginViewSet attribute), 9
 serializer_class (authapi.views.UserViewSet attribute), 10
 serializer_class (notification.views.NotificationView attribute), 16
 serializer_class (notification.views.NotificationViewList attribute), 16
 serializer_class (notification.views.UpdateNotification attribute), 16
 SubGroupSerializer (class in group.serializers), 12
 SubGroupSerializer.Meta (class in group.serializers), 12

T

target_object (newsfeed.models.Post attribute), 13
 target_object (notification.models.Notification attribute), 13

target_type (newsfeed.models.Post attribute), 13
 target_type (notification.models.Notification attribute), 13

to_representation() (User.serializers.PictureField method), 8

TypeSerializer (class in notification.serializer), 14
 TypeSerializer.Meta (class in notification.serializer), 14

U

update_read() (notification.views.UpdateNotification method), 16
 UpdateNotification (class in notification.views), 16
 UpdateNotificationSerializer (class in notification.serializer), 14
 UpdateNotificationSerializer.Meta (class in notification.serializer), 15
 user (group.models.GroupMember attribute), 11
 User (module), 7
 user (newsfeed.models.Comment attribute), 12
 user (newsfeed.models.Post attribute), 13
 user (notification.models.Notification attribute), 13
 user (User.models.UserProfile attribute), 7
 User.models (module), 7
 User.serializers (module), 7
 user_cover_directory_path() (in module User.models), 7
 user_picture_directory_path() (in module User.models), 7
 UserCoverSerializer (class in User.serializers), 8
 UserCoverSerializer.Meta (class in User.serializers), 8
 UserNotification (class in notification.models), 14
 UserNotification.DoesNotExist, 14
 UserNotification.MultipleObjectsReturned, 14
 usernotification_set (notification.models.Notification attribute), 14
 UserNotificationSerializer (class in notification.serializer), 15
 UserNotificationSerializer.Meta (class in notification.serializer), 15
 UserProfile (class in User.models), 7
 UserProfile.DoesNotExist, 7
 UserProfile.MultipleObjectsReturned, 7
 UserProfilePictureSerializer (class in User.serializers), 8
 UserProfilePictureSerializer.Meta (class in User.serializers), 8
 UserProfileSerializer (class in notification.serializer), 15
 UserProfileSerializer (class in User.serializers), 8
 UserProfileSerializer.Meta (class in notification.serializer), 15
 UserProfileSerializer.Meta (class in User.serializers), 8
 UserSerializer (class in group.serializers), 12
 UserSerializer (class in notification.serializer), 15
 UserSerializer (class in User.serializers), 8
 UserSerializer.Meta (class in group.serializers), 12
 UserSerializer.Meta (class in notification.serializer), 15
 UserSerializer.Meta (class in User.serializers), 8

UserViewSet (class in authapi.views), [9](#)