**Spyne socialapp Documentation**

1. **API Documentation:**

**SPYNE API**

**﻿**

* POST

Create User

http://localhost:8000/api/users/create/

﻿Body

raw (json)

json

{

"username": "testuser",

"password": "password123",

"mobile\_no": "1234567890",

"email": "testuser@example.com"

}

* POST

Login User

http://localhost:8000/api/auth/login/

﻿Body

raw (json)

json

{

"username": "testuser",

"password": "password123"

}

* GET

Get Users

http://localhost:8000/api/users/

﻿

* PUT

Update User

http://localhost:8000/api/users/update/1/

Body

raw (json)

json

{

"username": "updateduser",

"email": "updateduser@example.com"

}

* DELETE

Delete User

http://localhost:8000/api/users/delete/1/

﻿

* POST

Create Discussion

http://localhost:8000/api/discussions/create/

Body

raw (json)

json

{

"user": 1,

"text": "This is a discussion post",

"hashtags": "#discussion"

}

* GET

Get Discussions

http://localhost:8000/api/discussions/

* PUT

Update Discussion

http://localhost:8000/api/discussions/update/1/

﻿

* DELETE

Delete Discussion

http://localhost:8000/api/discussions/delete/1/

﻿

* GET

Get Discussions by Tag

http://localhost:8000/api/discussions/tags/?tags=discussion

Query Params

tags

discussion

* GET

Get Discussions by Text

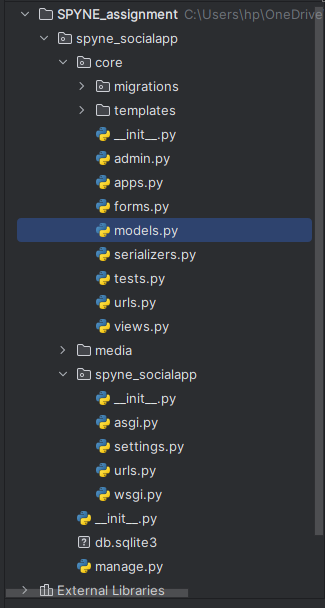
http://localhost:8000/api/discussions/search/?text=discussion

Query Params

text

discussion

1. **Code Summary:**

****

**Forms.py**

UserCreationForm

The UserCreationForm extends Django's UserCreationForm and adds additional fields for email and mobile\_no.

LoginForm

The LoginForm is a custom form for user authentication with fields for username and password. It includes a clean method to validate the credentials using Django's authenticate function.

DiscussionForm

The DiscussionForm is a model form for creating discussions. It includes fields for text, image, and hashtags, and is associated with the Discussion model via the Meta class.

**Models.py**

Model Summary

User Model

mobile\_no: A unique character field to store the user's mobile number (max length 15).

email: A unique email field.

followers: A many-to-many relationship to the User model itself, representing users that follow this user.

groups: Custom relationship to auth.Group with related name custom\_user\_set.

user\_permissions: Custom relationship to auth.Permission with related name custom\_user\_set.

Discussion Model

user: A foreign key to the User model, indicating the creator of the discussion.

text: A text field to store the content of the discussion.

image: An optional image field, storing the image in the images/ directory.

hashtags: A character field to store hashtags associated with the discussion (max length 255).

created\_on: A datetime field that automatically sets the timestamp when the discussion is created.

view\_count: An integer field to store the number of views the discussion has received.

Comment Model

discussion: A foreign key to the Discussion model, indicating which discussion the comment belongs to.

user: A foreign key to the User model, indicating the creator of the comment.

text: A text field to store the content of the comment.

created\_on: A datetime field that automatically sets the timestamp when the comment is created.

likes: An integer field to store the number of likes the comment has received.

replies: A many-to-many relationship to the Comment model itself, representing replies to this comment.

**Serializer Summary**

UserSerializer

The UserSerializer serializes the User model.

Fields:

id: The unique identifier for the user.

username: The username of the user.

mobile\_no: The user's mobile number.

email: The user's email address.

followers: A list of users who follow this user.

following: A list of users whom this user is following.

DiscussionSerializer

The DiscussionSerializer serializes the Discussion model.

Fields:

id: The unique identifier for the discussion.

user: The user who created the discussion.

text: The text content of the discussion.

image: An optional image associated with the discussion.

hashtags: Hashtags associated with the discussion.

created\_on: The timestamp when the discussion was created.

view\_count: The number of views the discussion has received.

CommentSerializer

The CommentSerializer serializes the Comment model.

Fields:

id: The unique identifier for the comment.

discussion: The discussion to which the comment belongs.

user: The user who created the comment.

text: The text content of the comment.

created\_on: The timestamp when the comment was created.

likes: The number of likes the comment has received.

replies: A list of replies to this comment.

**URL Configuration Summary**

This URL configuration sets up various endpoints for the Django application, including admin, user management, authentication, and discussions.

#### Endpoints

1. \*\*Admin Site\*\*

- \*\*URL:\*\* `admin/`

- \*\*View:\*\* `admin.site.urls`

- \*\*Description:\*\* Access the Django admin interface.

2. \*\*Home Page\*\*

- \*\*URL:\*\* ``

- \*\*View:\*\* `views.home`

- \*\*Name:\*\* `home`

- \*\*Description:\*\* Displays the home page.

3. \*\*User List\*\*

- \*\*URL:\*\* `users/`

- \*\*View:\*\* `views.user\_list`

- \*\*Name:\*\* `user-list`

- \*\*Description:\*\* Lists all users.

4. \*\*Create User\*\*

- \*\*URL:\*\* `users/create/`

- \*\*View:\*\* `views.user\_create`

- \*\*Name:\*\* `user-create`

- \*\*Description:\*\* Endpoint to create a new user.

5. \*\*User Login\*\*

- \*\*URL:\*\* `login/`

- \*\*View:\*\* `views.login`

- \*\*Name:\*\* `login`

- \*\*Description:\*\* Endpoint for user login.

6. \*\*Discussion List\*\*

- \*\*URL:\*\* `discussions/`

- \*\*View:\*\* `views.discussion\_list`

- \*\*Name:\*\* `discussion-list`

- \*\*Description:\*\* Lists all discussions.

7. \*\*Create Discussion (API)\*\*

- \*\*URL:\*\* `api/discussions/create/`

- \*\*View:\*\* `views.discussion\_create`

- \*\*Name:\*\* `discussion-create`

- \*\*Description:\*\* API endpoint to create a new discussion.

8. \*\*User Profile\*\*

- \*\*URL:\*\* `accounts/profile/`

- \*\*View:\*\* `views.profile`

- \*\*Name:\*\* `profile`

- \*\*Description:\*\* Displays the user's profile page.

9. \*\*API Authentication\*\*

- \*\*URL:\*\* `api/auth/`

- \*\*Include:\*\* `rest\_framework.urls`

- \*\*Description:\*\* Includes default authentication endpoints provided by Django REST framework.

10. \*\*API Registration\*\*

- \*\*URL:\*\* `api/auth/registration/`

- \*\*Include:\*\* `dj\_rest\_auth.registration.urls`

- \*\*Description:\*\* Includes registration endpoints provided by `dj-rest-auth`.

11. \*\*Create User (API)\*\*

- \*\*URL:\*\* `api/users/create/`

- \*\*View:\*\* `views.user\_create`

- \*\*Name:\*\* `api-user-create`

- \*\*Description:\*\* API endpoint to create a new user.

**Views Summary**

This summary describes the functionality of each view in the provided Django views file.

---

#### Home View

- \*\*Function:\*\* `home`

- \*\*URL Name:\*\* `home`

- \*\*Description:\*\* Renders the home page template.

- \*\*Template:\*\* `home.html`

#### User List View

- \*\*Function:\*\* `user\_list`

- \*\*URL Name:\*\* `user-list`

- \*\*Description:\*\* Retrieves all users from the database and renders the user list template.

- \*\*Template:\*\* `user\_list.html`

- \*\*Context:\*\*

- `users`: Queryset of all users.

#### User Create View

- \*\*Function:\*\* `user\_create`

- \*\*URL Name:\*\* `user-create`

- \*\*Description:\*\* Handles user creation. Displays a form for user input and processes form submissions to create a new user.

- \*\*Template:\*\* `user\_create.html`

- \*\*Context:\*\*

- `form`: Instance of `UserCreationForm`.

#### Login View

- \*\*Function:\*\* `login`

- \*\*URL Name:\*\* `login`

- \*\*Description:\*\* Handles user login. Displays a login form and authenticates users based on the provided credentials.

- \*\*Template:\*\* `login.html`

- \*\*Context:\*\*

- `form`: Instance of `LoginForm`.

#### Discussion List View

- \*\*Function:\*\* `discussion\_list`

- \*\*URL Name:\*\* `discussion-list`

- \*\*Description:\*\* Retrieves all discussions from the database and renders the discussion list template. Requires user to be logged in.

- \*\*Template:\*\* `discussion\_list.html`

- \*\*Context:\*\*

- `discussions`: Queryset of all discussions.

#### Discussion Create View

- \*\*Function:\*\* `discussion\_create`

- \*\*URL Name:\*\* `discussion-create`

- \*\*Description:\*\* Handles discussion creation. Displays a form for discussion input and processes form submissions to create a new discussion. Requires user to be logged in.

- \*\*Template:\*\* `discussion\_create.html`

- \*\*Context:\*\*

- `form`: Instance of `DiscussionForm`.

#### Profile View

- \*\*Function:\*\* `profile`

- \*\*URL Name:\*\* `profile`

- \*\*Description:\*\* Renders the user's profile page. Requires user to be logged in.

- \*\*Template:\*\* `profile.html`

- \*\*Context:\*\*

- `user`: The currently logged-in user.

---