



This documentation is not published yet, this is only a preview. Changes to the publishing settings will reload this page.

# Spring\_Acme

Red social Spring Acme, enfocada en expresar y transmitir tus pensamientos e ideas a través de publicaciones y comentar en las publicaciones de tus amigos, al mismo tiempo puedes conocer nuevas personas y seguir.

## Users

Sección User. En esta sección encontrarás la opción de crear un usuario, iniciar sesión, actualizar tu información y obtener tu información.

### POST Login



```
http://localhost:8080/auth/login
```

Iniciar sesión. Retorna el Token para iniciar sesión en la red Social.

### AUTHORIZATION Basic Auth

<b>Username</b>	<username>
<b>Password</b>	<password>

### HEADERS

Content-Type

application/json

Body raw (json)

---

json

```
{
  "username": "johndoe",
  "password": "securepasswo"
}
```

---

## PATCH Update



http://localhost:8080/api/users/johndoe

Actualizar la información del usuario. Para el desarrollo de este endpoint se requiere ya haber generado el token y estar en uso.

AUTHORIZATION Bearer Token

---

Token

<token>

## HEADERS

---

Content-Type

application/json

Body raw (json)

---

json

```
{
  "name": "John Doe",
  "username": "johndoe",
  "celphone": "1234567890",
  "email": "johndoe@example.com",
  "bibliography": "Hola, soy John",
  "profilePicture": "https://images.unsplash.com/photo-1511367461989-f85a21"
}
```

## POST Register



http://localhost:8080/auth/register

Registrar al usuario para tener acceso a la Red Social.

### AUTHORIZATION Basic Auth

**Username** <username>

**Password** <password>

### HEADERS

**Content-Type** application/json

### Body raw (json)

json

```
{
  "name": "Fernando Gutierrez",
  "username": "fer123",
  "celphone": "1234567000",
  "email": "fer@example.com",
  "dateBirth": "1992-04-01",
  "password": "fer12345"
}
```

---

## GET getUser



```
http://localhost:8080/api/users/johndoe
```

Retornar la información específica de un usuario. Requiere el username del usuario para poder acceder a los datos. Para el desarrollo de este endpoint se requiere ya haber generado el token y estar en uso.

### AUTHORIZATION Bearer Token

Token	<token>
-------	---------

---

## Follows

Sección Follow. En donde encontrarás el proceso de seguir a otro usuario, dejar de seguir, ver la lista de quienes te siguen y quienes sigues.

---

## GET Following



```
http://localhost:8080/api/follows/johndoe/following
```

Retorna una lista de los usuarios que sigues. Requiere el username del usuario para generar la lista de usuarios. Para el desarrollo de este endpoint se requiere ya haber generado el token y estar en uso.

#### AUTHORIZATION Bearer Token

---

Token <token>

### GET notFollowed



```
http://localhost:8080/api/follows/johndoe/not-followed
```

Retorna una lista de usuarios que no sigue el usuario. Requiere el username del usuario. Para el desarrollo de este endpoint se requiere ya haber generado el token y estar en uso.

#### AUTHORIZATION Bearer Token

---

Token <token>

### DELETE unFollow



```
http://localhost:8080/api/follows/johndoe/unfollow/spellax
```

Retorna una respuesta, un boolean donde informa si el usuario si ya dejo de seguir al otro usuario. Se requiere el username del usuario y del otro usuario a quién se desea seguir. En el primer apartado se inserta el username del usuario y el último consiste del username del usuario a quién desea seguir.

#### AUTHORIZATION Bearer Token

---

Token <token>

## POST Follow



```
http://localhost:8080/api/follows/johndoe/follow/fer123
```

Retorna una respuesta, un boolean donde informa si el usuario si inició siguiendo al otro usuario. Se requiere el username del usuario y del otro usuario a quién se desea seguir. En el primer apartado se inserta el username del usuario y el último consiste del username del usuario a quién desea seguir.

### AUTHORIZATION Bearer Token

Token	<token>
-------	---------

## Posts

Sección Post. Encontrarás la creación de Posts, obtener su información, obtener todos los posts de los usuarios que sigue, actualizar y eliminar un Post, obtener todas los Posts del usuario, ordenar los Posts de los otro usuarios de manera descendente y ascendente, agregar y eliminar Tags a un Post.

## POST addPost



```
http://localhost:8080/posts/fer123
```

Crea el Post del usuario, requiere el username del usuario. En esta ocasión, no es obligatorio la inserción de la imagen, si el usuario lo desea, lo hace; el dato importante es la descripción del Post como requerimiento para la creación del Post.

### AUTHORIZATION Bearer Token

Token <token>

## HEADERS

---

Content-Type application/json

Body raw (json)

---

json

```
{
  "imageUrl": "",
  "description": "Soy Fernando, un gusto."
}
```

**GET** getPost



http://localhost:8080/posts/2

Retorna la información de un Post. Requiere la ID del Post para recibir la información solicitada.

**AUTHORIZATION** Bearer Token

---

Token <token>

## HEADERS

---

Content-Type application/json

## GET getAllPosts



```
http://localhost:8080/posts/following/johndoe
```

Retorna lista de Posts que el usuario sigue. Requiere el username del Usuario para recibir la información solicitada.

### AUTHORIZATION Bearer Token

Token	<token>
-------	---------

## PUT updatePost



```
http://localhost:8080/posts/johndoe/1
```

Actualiza el Post. Requiere el username del Usuario y el ID del Post para el proceso como a su vez la información que desea que sea actualizada.

### AUTHORIZATION Bearer Token

Token	<token>
-------	---------

### HEADERS

Content-Type	application/json
--------------	------------------

Body	raw (json)
------	------------



json

```
{
  "imageUrl": "https://caminoincamachupicchu.org/cmingutd/wp-content/uploads/2018/07/07-07-2018-14-00-00.jpg",
  "description": "Hola, soy nuevo. Mi nombre es john Doe."
}
```

## DELETE deletePost



http://localhost:8080/posts/3

Elimina un Post. Requiere el ID del Post para poder eliminarlo.

### AUTHORIZATION Bearer Token

Token <token>

## GET getMyPosts



http://localhost:8080/posts/myPosts/johndoe

Retorna una lista de Post, en esta ocasión los Posts del usuario. Requiere el username del Usuario.

### AUTHORIZATION Bearer Token

Token <token>

## GET postsOrderByDesc



```
http://localhost:8080/posts/following/desc/johndoe
```

Retorna una lista de Posts de manera descendente. Requiere el username del usuario.

#### AUTHORIZATION Bearer Token

Token <token>

### POST addTagByPost



```
http://localhost:8080/posts/1/tags/BellaLuna
```

Agregar un Tag a un Post. Rquiere el ID del Post y el Tag.

#### AUTHORIZATION Bearer Token

Token <token>

### DELETE deleteTagByPost



```
http://localhost:8080/posts/1/tags/6
```

Eliminar un Tag de un Post. Rquiere el ID del Post y el ID Tag.

#### AUTHORIZATION Bearer Token

Token <token>

### GET postsOrderByAsc



```
http://localhost:8080/posts/following/asc/johndoe
```

Retorna una lista de Posts de manera ascendente. Requiere el username del usuario.

#### AUTHORIZATION Bearer Token

---

Token <token>

## Tags

Sección Tags. Encontrarás la posibilidad de crear y eliminar un Tag

#### POST addTag



```
http://localhost:8080/api/tag
```

Crear un Tag. Requiere un Tag insertado.

#### AUTHORIZATION Bearer Token

---

Token <token>

#### HEADERS

---

Content-Type application/json

Body raw (json)

---

json

```
{"tag": "HappyBirthDay"}
```

## DELETE deleteTag



```
http://localhost:8080/api/tag/4
```

Elimina un Tag. Requiere el ID Tag.

**AUTHORIZATION** Bearer Token

Token

<token>

## Likes

Sección Likes. Encontrarás la opción de agregar y eliminar un Like, y retornar al cantidad de likes existentes.

## POST addLike



```
http://localhost:8080/api/likes/post/2/user/johndoe
```

Crear y agregar un Like a un Post. Requiere el ID del Post y el username Usuario.

**AUTHORIZATION** Bearer Token

Token

<token>

## DELETE removeLike



```
http://localhost:8080/api/likes/post/2/user/johndoe
```

Eliminar un Like a un Post. Requiere el ID del Post y el username Usuario.

## AUTHORIZATION Bearer Token

Token

<token>

## GET countLikes



```
http://localhost:8080/api/likes/count/1
```

Retorna la cantidad de Likes que tiene un Post. Requiere ID del Post.

## AUTHORIZATION Bearer Token

Token

<token>

# Comments

Sección Comments. Encontrarás la opción agregar, eliminar, actualizar y eliminar un comentario, obtener la información de un comentario, agregar un comentario a otro comentario, retornar la cantidad de comentarios de un Post.

## POST addComment



```
http://localhost:8080/api/comments/post/1/user/johndoe?content=Soy tu tercero comentario
```

Crear un Comentario a un Post. Requiere la información del Comentario y el ID del Post.

#### AUTHORIZATION Bearer Token

---

Token <token>

#### HEADERS

---

Content-Type application/json

#### PARAMS

---

content Soy tu tercero comentario

---

### GET getComment



```
http://localhost:8080/api/comments/4
```

Retorna la información del Comentario. Requiere el ID del Comentario.

#### AUTHORIZATION Bearer Token

---

Token <token>

---

### PUT updateComment



```
http://localhost:8080/api/comments/3?content=Soy tu tercero comentario
```

Actualiza un comentario. Requiere la información actualizada y el ID del Comentario.

#### AUTHORIZATION Bearer Token

---

Token <token>

#### PARAMS

---

content Soy tu tercero comentario

---

#### DELETE deleteComment



```
http://localhost:8080/api/comments/11
```

Elimina un comentario. Requiere el ID del Comentario.

#### AUTHORIZATION Bearer Token

---

Token <token>

---

#### POST addReply



```
http://localhost:8080/api/comments/10/reply?username=johndoe&content=Soy la respu  
esta de la respuesta nueva
```

Crea y agrega un comentario a otro comentario. Requiere el ID del Comentario Padre, el Username del usuario quién redacta y el contenido del comentario.

## AUTHORIZATION Bearer Token

---

Token <token>

## PARAMS

---

username johndoe

content Soy la respuesta de la respuesta nueva

## GET countCommentsByPost



```
http://localhost:8080/api/comments/post/1/count
```

Retorna la cantidad de Comentarios de un Post. Requiere el ID del Post.

## AUTHORIZATION Bearer Token

---

Token <token>

# Notifications