Documentación de la API - BeerSP

Esta documentación describe los endpoints para gestionar Usuarios y Amistades en la aplicación BeerSP.

URL Base de la API: /usuarios

1. Endpoints de Usuarios

Gestión de la creación, lectura, actualización y eliminación de usuarios.

POST /usuarios/

- **Resumen**: Registrar nuevo usuario
- **Método**: POST
- **Descripción**: Crea un nuevo usuario en la base de datos.
- **Cuerpo de la Petición (Request Body)**:
    - UsuarioCreate (JSON):
    - {
    - "username": "nuevo_usuario",
    - "email": "usuario@ejemplo.com",
    - "password": "una_contraseña_segura",
    - "birth_date": "YYYY-MM-DD"
    - }
- **Respuesta Exitosa (201 CREATED)**:
    - Usuario (JSON): El objeto del usuario creado (sin la contraseña).
    - {
    - "id": 1,
    - "username": "nuevo_usuario",
    - "email": "usuario@ejemplo.com",
    - "birth_date": "YYYY-MM-DD",
    - "friends": []

- }

- **Respuestas de Error**:
  - 409 CONFLICT: Si el correo electrónico o el nombre de usuario ya están registrados.
  - 500 INTERNAL SERVER ERROR: Si ocurre un error inesperado en la base de datos.

GET /usuarios/

- **Resumen**: Obtener lista de todos los usuarios
- **Método**: GET
- **Descripción**: Devuelve una lista de todos los usuarios registrados en el sistema.
- **Cuerpo de la Petición**: Ninguno.
- **Respuesta Exitosa (200 OK)**:
  - List[Usuario] (JSON): Un array de objetos de usuario.
  - [
  - {
  - "id": 1,
  - "username": "usuario1",
  - "email": "usuario1@ejemplo.com",
  - "birth_date": "YYYY-MM-DD",
  - "friends": [2]
  - },
  - {
  - "id": 2,
  - "username": "usuario2",
  - "email": "usuario2@ejemplo.com",
  - "birth_date": "YYYY-MM-DD",
  - "friends": [1]

    - }
    -]

GET /usuarios/{user_id}

- **Resumen**: Obtener usuario por ID
- **Método**: GET
- **Descripción**: Busca y devuelve un único usuario basado en su ID numérico.
- **Parámetros de Ruta (Path Parameters)**:
    - user_id (integer, **Requerido**): El ID del usuario a buscar.
- **Respuesta Exitosa (200 OK)**:
    - Usuario (JSON): El objeto del usuario encontrado.
- **Respuestas de Error**:
    - 404 NOT FOUND: Si no se encuentra un usuario con ese ID.

PUT /usuarios/{user_id}

- **Resumen**: Actualizar información del usuario
- **Método**: PUT
- **Descripción**: Actualiza la información de un usuario existente.
- **Parámetros de Ruta**:
    - user_id (integer, **Requerido**): El ID del usuario a actualizar.
- **Cuerpo de la Petición**:
    - Usuario (JSON): Un objeto Usuario con los campos a actualizar.
- **Respuesta Exitosa (200 OK)**:
    - Usuario (JSON): El objeto del usuario con la información actualizada.
- **Respuestas de Error**:
    - 404 NOT FOUND: Si el usuario con user_id no existe.

DELETE /usuarios/{user_id}

- **Resumen**: Eliminar usuario por ID

- **Método**: DELETE
- **Descripción**: Elimina permanentemente a un usuario de la base de datos.
- **Parámetros de Ruta**:
    - user_id (integer, **Requerido**): El ID del usuario a eliminar.
- **Respuesta Exitosa (200 OK)**:
- {
- "message": "Usuario con ID 1 eliminado correctamente."
- }
- **Respuestas de Error**:
    - 404 NOT FOUND: Si el usuario con user_id no existe.

2. Endpoints de Amistad

Gestión de las relaciones de amistad entre usuarios.

POST /usuarios/{user_id}/amigos/

- **Resumen**: Agregar amigo a usuario
- **Método**: POST
- **Descripción**: Crea una relación de amistad. (Nota: El endpoint ignora el {user_id} en la ruta y usa los IDs del cuerpo de la petición).
- **Cuerpo de la Petición**:
    - FriendRequest (JSON):
    - {
    - "user_id": 1,
    - "friend_id": 2
    - }
- **Respuesta Exitosa (201 CREATED)**:
    - FriendRelationship (JSON):
    - {

- "success": true,
- "message": "Amigo agregado correctamente."
- }

- **Respuestas de Error**:
  - 201 CREATED (con success: false): Si los usuarios no existen, o si el user_id y friend_id son iguales.

GET /usuarios/{user_id}/amigos/

- **Resumen**: Obtener lista de amigos del usuario
- **Método**: GET
- **Descripción**: Obtiene una lista de todos los perfiles de usuario que son amigos del user_id especificado.
- **Parámetros de Ruta**:
  - user_id (integer, **Requerido**): El ID del usuario del cual se quieren ver los amigos.
- **Respuesta Exitosa (200 OK)**:
  - List[Usuario] (JSON): Una lista de objetos de usuario (solo los amigos).
- **Respuestas de Error**:
  - 500 INTERNAL SERVER ERROR: Si ocurre un error al consultar la base de datos.

DELETE /usuarios/{user_id}/amigos/{friend_id}

- **Resumen**: Eliminar amigo de usuario
- **Método**: DELETE
- **Descripción**: Elimina la relación de amistad entre dos usuarios.
- **Parámetros de Ruta**:
  - user_id (integer, **Requerido**): El ID del usuario principal.
  - friend_id (integer, **Requerido**): El ID del amigo a eliminar.
- **Respuesta Exitosa (200 OK)**:

    - FriendRelationship (JSON):
    - {
    - "success": true,
    - "message": "Amigo eliminado correctamente."
    - }
- **Respuestas de Error**:
    - 200 OK (con success: false): Si la amistad no existía.