# DOCUMENTACIÓN API

Este proyecto consiste en el desarrollo de una **API REST en Python con Flask y SQLAlchemy** que gestiona información de jugadores de fútbol. La aplicación permite realizar operaciones CRUD (crear, leer, actualizar y eliminar) sobre los registros de jugadores, almacenando sus datos (nombre, equipo y número) en una base de datos SQLite. Gracias a la integración con SQLAlchemy, los datos permanecen guardados incluso después de reiniciar el servidor, brindando persistencia y facilidad de consulta. El proyecto está estructurado para ejecutarse en un entorno de desarrollo como GitHub Codespaces, facilitando pruebas mediante solicitudes HTTP con curl.

**Estructura del proyecto:**

- **src/app.py:** Punto de entrada de la aplicación Flask. Define los endpoints de la API y gestiona la lógica principal.
- **requirements.txt:** Lista de dependencias necesarias para ejecutar el proyecto.
- **curl_examples.sh**: Ejemplos de comandos curl para probar manualmente la API.
- **.gitignore**: Archivos y carpetas que no deben incluirse en el repositorio.
- **README.md:** Documentación general del proyecto.

**Endpoints disponibles:**

- **GET /players:** Obtiene la lista de todos los jugadores.
- **GET /players/<id>:** Obtiene la información de un jugador por su ID.
- **POST /players:** Crea un nuevo jugador. Requiere un JSON con nombre, equipo y número.
- **PUT /players/<id>:** Actualiza la información de un jugador existente.
- **DELETE /players/<id>:** Elimina un jugador por su ID.

**Ejemplos de uso:**

- Obtener todos los jugadores existentes

```
# 🔹 Obtener todos los jugadores
curl http://localhost:5000/players
```

- Obtener un jugador por ID

```
# 🔹 Obtener un jugador por ID (ejemplo: 1)
curl http://localhost:5000/players/1
```

- Crear un nuevo jugador

```
# 🔹 Crear un jugador
curl -i -X POST http://localhost:5000/players \
-H "Content-Type: application/json" \
-d '{"name": "Lionel Messi", "team": "Inter Miami", "number": 10}'
```

- Actualizar un jugador existente

```
# 🔹 Actualizar un jugador (ejemplo: cambiar equipo del ID 1)
curl -i -X PUT http://localhost:5000/players/1 \
-H "Content-Type: application/json" \
-d '{"team": "FC Barcelona"}'
```

- Eliminar un jugador

```
# 🔹 Eliminar un jugador (ejemplo: ID 2)
curl -i -X DELETE http://localhost:5000/players/2
```

**Requisitos del proyecto:**

- **Lenguaje de programación:** Python 3.10 o superior.
- **Framework web:** Flask (para la creación de la API REST).
- **ORM:** SQLAlchemy (para la conexión y gestión de la base de datos).
- **Base de datos:** SQLite (base ligera, embebida y persistente).
- **Entorno de desarrollo:** GitHub Codespaces o cualquier IDE compatible con Python.
- **Cliente para pruebas:** curl o herramientas similares.
- **Gestión de dependencias:** pip con archivo requirements.txt.
- **Control de versiones:** Git y repositorio en GitHub.

**Instalación y ejecución:**

1. Clonar el repositorio desde GitHub:

```
git clone <URL_DEL_REPOSITORIO>
cd <NOMBRE_DEL_PROYECTO>
```

2. Instalar las dependencias necesarias para el proyecto:

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

3. Ejecutar la aplicación:

```
python app.py
```

4. Acceder a la API:
    - La aplicación se ejecutará en la dirección:

      http://127.0.0.1:5000

    - Los endpoints podrán probarse con curl

**Notas:**

- Los datos se almacenan de forma persistente en el archivo de base de datos players.db, ubicado en src/instance/. Esto garantiza que los registros permanezcan disponibles incluso después de reiniciar la aplicación.

- Todos los endpoints devuelven respuestas en formato JSON y utilizan los códigos de estado HTTP apropiados para indicar el resultado de cada operación.

- La estructura de la base de datos y el código pueden extenderse fácilmente para agregar nuevas entidades o funcionalidades según las necesidades del proyecto.