

Manual del Backend

En este manual se explica el funcionamiento del backend de la aplicación.

Tecnologías

El backend está construido con:

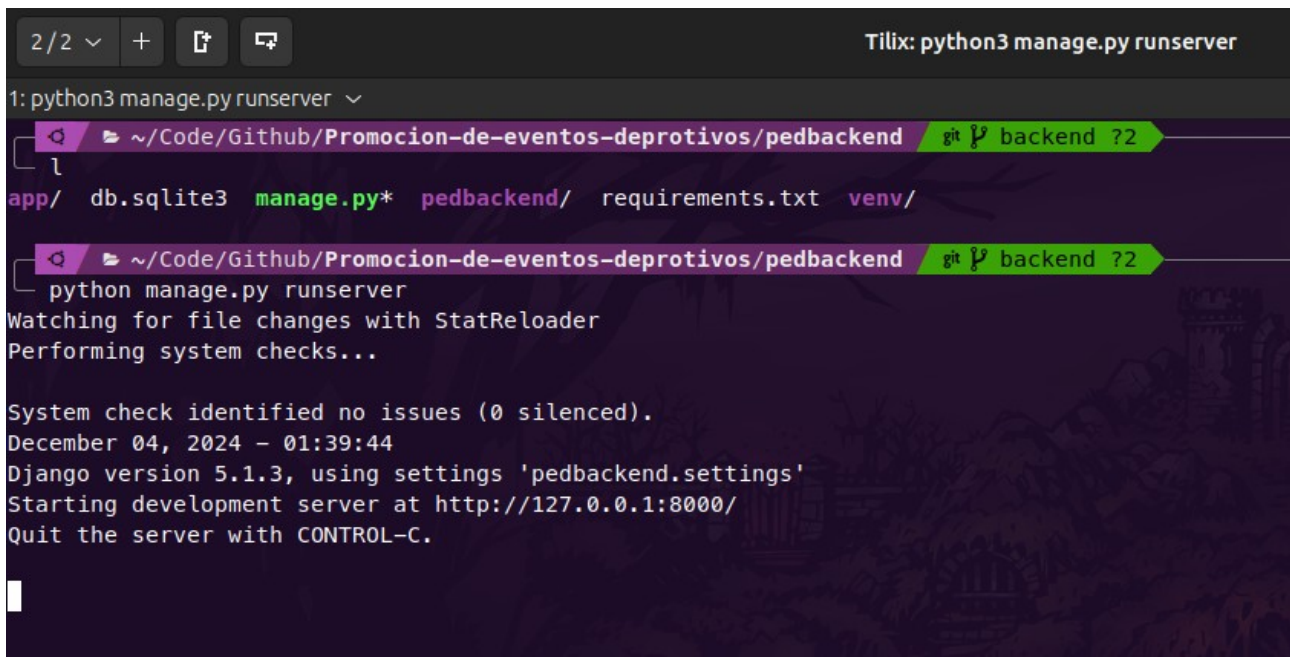
- Python 3.12.3
- Django 5.1.3
- SQLite 3

Ejecución

Para ejecutar el backend de la aplicación se debe tener en cuenta que se tienen las librerías de python requeridas. Las librerías necesarias se pueden instalar usando pip ejecutando el comando: *pip install -r requirements.txt*.

Una vez están instaladas las librerías necesarias, dentro del directorio llamado *pedbackend/* se encontrará un archivo llamado *manage.py*. Se debe situar allí la terminal y ejecutar el comando *python manage.py runserver*. Este comando ejecutará el servidor backend en el localhost (**127.0.0.1**) en el puerto **8000**, y quedará atento a solicitudes.

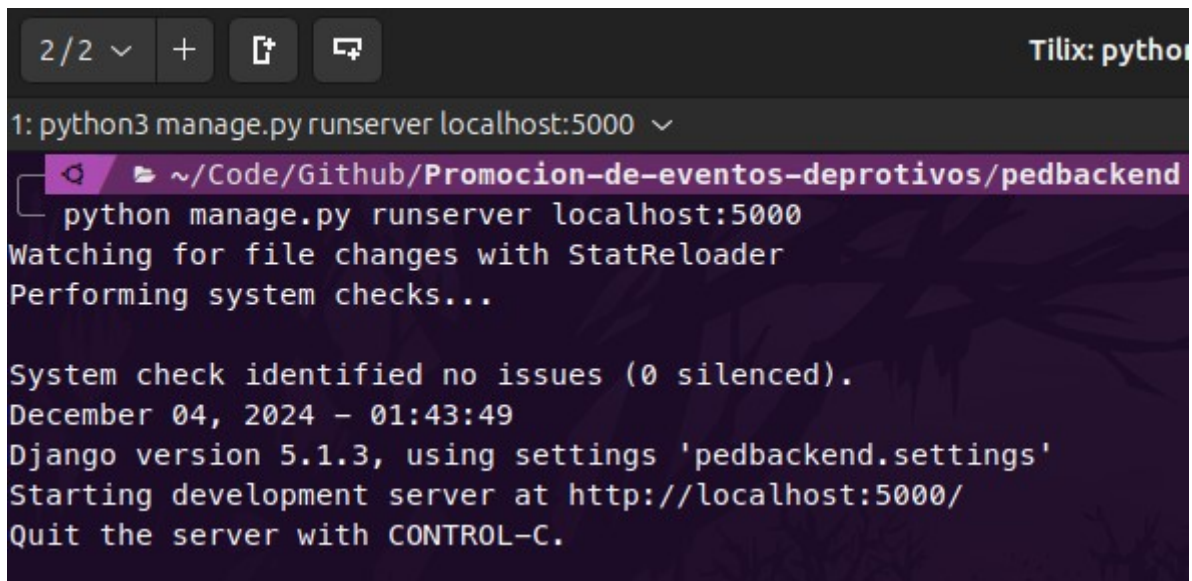
Se verá de esta forma:



```
Tilix: python3 manage.py runserver
1: python3 manage.py runserver
~/Code/Github/Promocion-de-eventos-deprotivos/pedbackend
app/ db.sqlite3 manage.py* pedbackend/ requirements.txt venv/
~/Code/Github/Promocion-de-eventos-deprotivos/pedbackend
python manage.py runserver
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
December 04, 2024 - 01:39:44
Django version 5.1.3, using settings 'pedbackend.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
```

El puerto puede ser cambiado a voluntad añadiendo como argumento al comando la IP y puerto deseado. Se vería de la siguiente forma:

A terminal window titled 'Tilix: python' showing the command 'python3 manage.py runserver localhost:5000' being executed. The output indicates the server is running on localhost:5000, watching for file changes with StatReloader, and performing system checks. The system check passed with no issues. The timestamp is December 04, 2024 - 01:43:49. The Django version is 5.1.3, using settings 'pedbackend.settings'. The development server is starting at http://localhost:5000/. The prompt to quit the server with CTRL-C is shown.

```
2/2  +  [ ]  [ ]  Tilix: python
1: python3 manage.py runserver localhost:5000  v
~/Code/Github/Promocion-de-eventos-deprotivos/pedbackend
python manage.py runserver localhost:5000
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
December 04, 2024 - 01:43:49
Django version 5.1.3, using settings 'pedbackend.settings'
Starting development server at http://localhost:5000/
Quit the server with CONTROL-C.
```

URLs

El backend responde ante 2 URLs, las cuales son:

- **/json_events/**: Solo solicitudes GET, no requiere argumentos y devuelve una respuesta en json con todos los datos registrados en la tabla *Events* de la base de datos. Cada elemento en la tabla será devuelto de la siguiente forma:

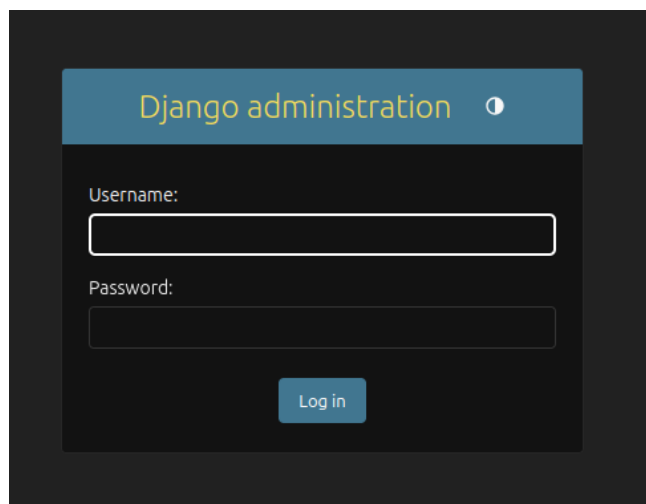
```
{
  "name": "Carrera 5K",
  "date": "2023-09-10",
  "location": "Parque Simón Bolívar",
  "requirements": "Inscripción previa",
  "sport": "Correr",
  "limit_date": "2025-01-01",
  "min_age": 18
},
```

- **/admin/**: Portal de administración de Django.

/admin/

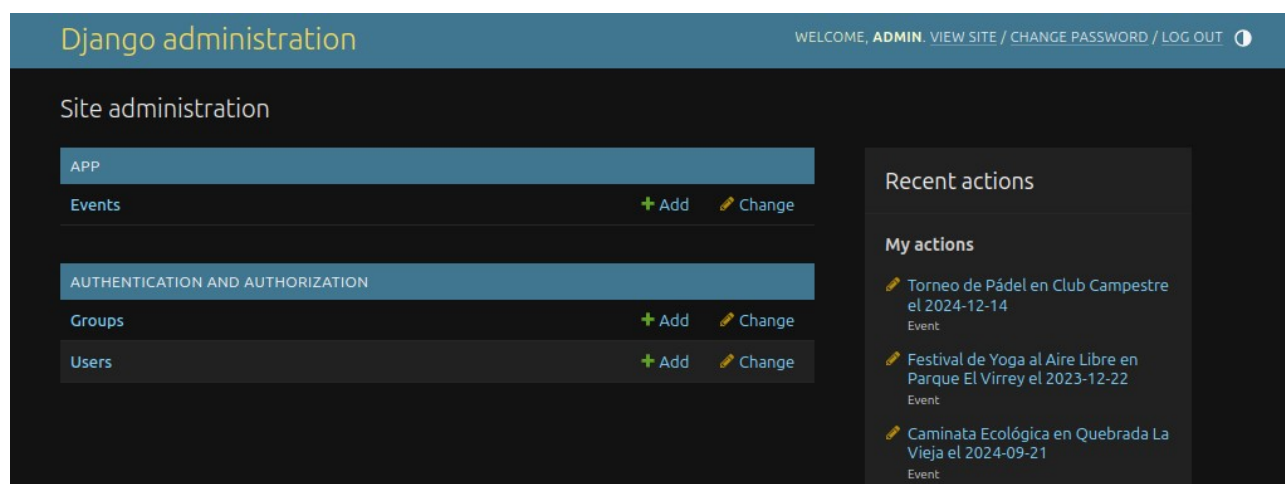
A través de la interfaz de administración de django se pueden manejar los registros de forma más segura e intuitiva, sin tener que modificar directamente la base de datos.

Al acceder a la ruta se mostrará la siguiente ventana de login:



Las credenciales para el inicio de sesión como administrador son **admin** como usuario y contraseña. Estas pueden ser cambiadas dentro de la interfaz de administración.

Una vez se ingresa a la interfaz, se verá de la siguiente forma:



A la izquierda se encuentra una opción llamada “Events” dentro de la aplicación “APP”. Al dar click izquierdo sobre “Events”, la interfaz mostrará todos los registros de la tabla “Events” de la base de datos. Se verá de la siguiente forma:

The screenshot shows the Django administration interface for the 'Events' app. The header includes the Django logo, the title 'Django administration', and user information: 'WELCOME, ADMIN. VIEW SITE / CHANGE PASSWORD / LOG OUT'. The breadcrumb trail is 'Home > App > Events'. The main content area is titled 'Select event to change' and features a search bar, an 'ADD EVENT +' button, and a table of events. A sidebar on the right contains a 'FILTER' section with options to 'Show counts' and filter by 'date' or 'location'.

<input type="checkbox"/>	NAME	DATE	1 ▲	LIMIT DATE	LOCATION	2 ▲	SPORT	3 ▲	MIN AGE
<input type="checkbox"/>	Carrera 5K	Sept. 10, 2023		Jan. 1, 2025	Parque Simón Bolívar		Correr		18
<input type="checkbox"/>	Ciclopaseo Nocturno	Nov. 15, 2023		Jan. 1, 2025	Ciclovia de la 7ma		Ciclismo		18
<input type="checkbox"/>	Festival de Yoga al Aire Libre	Dec. 22, 2023		Jan. 1, 2025	Parque El Virrey		Yoga		18
<input type="checkbox"/>	Torneo de Fútbol Amateur	Jan. 10, 2024		Jan. 1, 2025	Estadio Nacional		Fútbol		18
<input type="checkbox"/>	Competencia de Skateboarding	Feb. 14, 2024		Jan. 1, 2025	Skatepark de Usaquén		Skateboarding		18
<input type="checkbox"/>	Torneo de Tenis de	March 20, 2024		Jan. 1, 2025	Coliseo El Campín		Tenis de Mesa		18

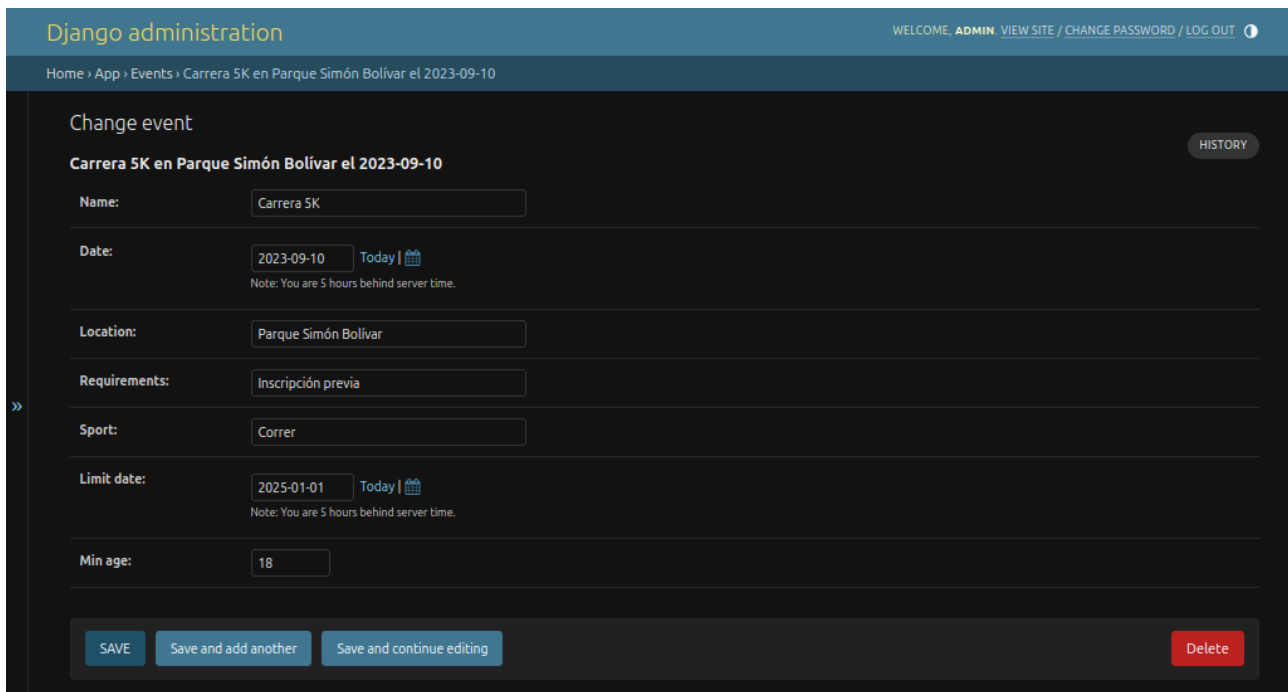
Crear un Nuevo Evento

Sí se desea ingresar un nuevo evento, se debe dar click sobre el botón “ADD EVENT +” que se encuentra en la parte superior derecha. Al hacerlo, se mostrará un formato que debe ser completado para crear un nuevo evento. Una vez se complete el formato, se debe dar click sobre el botón “SAVE”.

The screenshot shows the 'Add event' form in the Django administration interface. The header is the same as the previous screenshot. The breadcrumb trail is 'Home > App > Events > Add event'. The form is titled 'Add event' and contains several input fields: 'Name:', 'Date:', 'Location:', 'Requirements:', 'Sport:', 'Limit date:', and 'Min age:'. The 'Date' and 'Limit date' fields include a calendar icon and a note: 'Note: You are 5 hours behind server time.' At the bottom of the form are three buttons: 'SAVE', 'Save and add another', and 'Save and continue editing'.

Modificar y Borrar un Evento

En la página de eventos, los eventos aparecen con el nombre del evento en azul. Estos son enlaces clickeables que llevarán a la interfaz de modificación del evento seleccionado. Se verá de la siguiente forma:



The screenshot shows the Django administration interface for modifying an event. The page title is 'Change event' and the breadcrumb trail is 'Home > App > Events > Carrera 5K en Parque Simón Bolívar el 2023-09-10'. The event name is 'Carrera 5K en Parque Simón Bolívar el 2023-09-10'. The form fields are: Name (Carrera 5K), Date (2023-09-10, Today | 📅), Location (Parque Simón Bolívar), Requirements (Inscripción previa), Sport (Correr), Limit date (2025-01-01, Today | 📅), and Min age (18). The bottom of the form has buttons for 'SAVE', 'Save and add another', 'Save and continue editing', and 'Delete'.

Una vez aquí, los datos pueden ser reemplazados, sin embargo se debe tener en cuenta los tipos de datos que acepta la tabla en la base de datos.

Esta tabla fue creada usando el siguiente modelo, lo cual puede ayudar a aclarar que tipo de dato acepta cada campo:

```
File: models.py
1  from django.db import models
2
3  # Create your models here.
4
5  class Event(models.Model):
6      id = models.AutoField(primary_key=True)
7      name = models.CharField(max_length=255) # Nombre del evento
8      date = models.DateField() # Fecha del evento
9      location = models.CharField(max_length=255) # Lugar del evento
10     requirements = models.CharField(max_length=255) # Requisitos del evento
11     sport = models.CharField(max_length=255) # Deporte del evento
12     limit_date = models.DateField() # Fecha fin de inscripcion
13     min_age = models.SmallIntegerField() # Edad mínima para participar
14
```

Es importante tener en cuenta que, debido a lo anterior, no se pueden guardar valores de tipo "texto" en los campos **date**, **limit_date** y **min_age**. Para **date** y **limit_date** solo aceptarán valores que concuerden con el formato de fechas establecido y **min_age** solo aceptará números enteros desde el -32768 hasta el 32767.