**ENTREGA 3: Desarrollo Servidor básico con Django**

En este ejercicio he decidido implementar una tienda de videojuegos ya que tenía bastante similitud con los ejercicios vistos en clase. A continuación, explicaré aquellas partes de código que me parecen más importantes y que diferencian este ejercicio con los ejercicios anteriores:

**plataforma/models.py**

Texto

Descripción generada automáticamente

Modelo Plataforma en el que observamos dos campos:

* Nombre: nombre de la plataforma.
* Precio: precio en mercado.

**videojuego/models.py**

Texto

Descripción generada automáticamente

Modelo Videojuego en el que observamos dos campos:

* Nombre: nombre del videojuego.
* Año: año de lanzamiento en mercado.
* Plataforma: Nombre de la plataforma a la que pertenece el juego.

**plataforma/urls.py**

Texto

Descripción generada automáticamente

Para el modelo PLATAFORMA he creado 4 vistas genéricas. En la captura se observan las urls que se utilizan.

**videojuego/urls.py**

**Captura de pantalla con letras

Descripción generada automáticamente**

Para el modelo VIDEOJUEGO he creado 4 vistas genéricas y una api\_view propia. En la captura se observan las urls que se utilizan.

**videojuego/views.py**

Texto

Descripción generada automáticamente

Las vistas creadas en el modelo VIDEOJUEGO son las siguientes:

* Api\_view propia en la que el usuario añade el nombre de la plataforma en la url y se devuelve todos aquellos videojuegos propios a la plataforma introducida. Por ejemplo, si hacemos una petición GET y en la url se indica “PS5” la respuesta será un json con todos los videojuegos que existen en la api que sean de PS5.
* Vista genérica que mediante petición GET devuelve el videojuego con el ID indicado en la url.
* Vista genérica en la que si indicamos petición GET devuelve todos los videojuegos de la api pero si indicamos petición POST podemos añadir nuevos videojuegos a la api.
* Vista genérica que mediante petición DELETE eliminamos el videojuego con el ID indicado en la url.
* Vista genérica que mediante petición PUT modificamos el videojuego con el ID indicado en la url.

**plataforma/views.py**

**Texto

Descripción generada automáticamente**

Las vistas creadas en el modelo PLATAFORMA son las mismas vistas genéricas que encontramos en el modelo VIDEOJUEGO.

**Observaciones**

Ha sido una práctica muy interesante para empezar a ver como se utiliza Python en el ámbito de Backend y como aplicar los ejercicios vistos en clase de una manera más realista como por ejemplo la creación de la api\_view en la que se combinan modelos distintos. En este caso, he decidido hacer el ejercicio simulando una tienda de videojuegos ya que era un ejemplo bastante parecido al visto en clase con los libros y que se adaptaba muy bien a la idea de api\_view que pretendía hacer.

**Conclusiones**

En la primera clase de Python recuerdo tener bastantes problemas ya que no entendía cómo funcionaba la herramienta y eso hacía que no pudiese seguir correctamente el código que se exponía en clase. Tras esto, decidí revisar las clases por mi cuenta poco a poco y empecé a comprender el funcionamiento y todo empezó a ser mucho más fácil hasta el punto de que no he tenido dificultades para poder completar este ejercicio.

**Url del repositorio**: https://github.com/bielrc1906/Modulo\_6.git

Biel Roca Coll