Presentación Demo Observaciones

miércoles, 20 de abril de 2022 16:49

Considerar Unit Test de las partes más críticas de la aplicación parte más importantes.

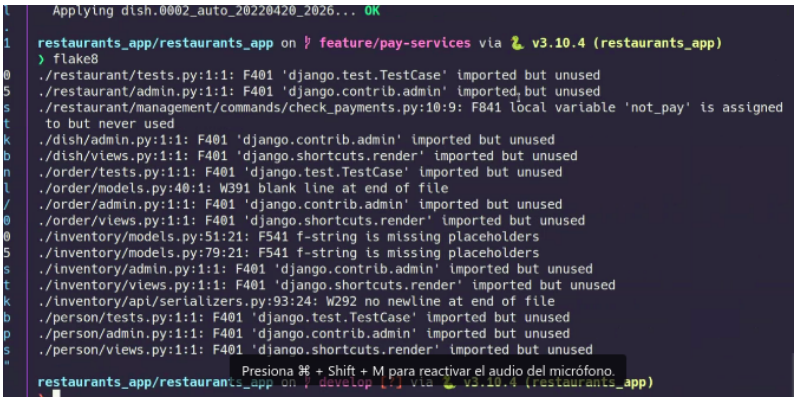
Ponernos de acuerdo (entre los desarrolladores) para usar la misma estructura común.

Ponernos de acuerdo para implementar las views de las mismas estructura común.

Link de todos los métodos de Django.

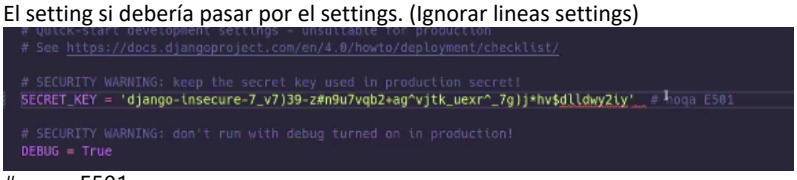
https://www.cdrf.co/

Recordar todo lo relacionado con Flake8.



- Los pecados capitales que no podemos cometer en la presentación.

- No errores de flake8.
Excluir las cosas que no deberian pasar por el item.
El setting si debería pasar por el settings. (Ignorar lineas settings)



#noqa: E501

Implementar las variables de entornos en settings.

Los tiempos de expiración de jwt configurables por variables de entorno.

No debe de haber ningún guardado de la base de datos adentro de los serializadores.

Los serializadores solo sirven para recibir data del usuario. Y validar la entrada del usuario. No deben de tener lógica de negocio dentro de vistas, las vistas son archivos cortos. (Hay que crear servicios aparte para la lógica de negocio para eso).

Las vistas más especiales, se puede generar su propia vista desde cero, a través de heredar de ApiView o GenericAPIView.

Priorizar las cosas más importantes, no se puede completar todo en dos semana es demasiado trabajo para dos desarrolladores, por esta razón es importante empezar a trabajar lo más importante.

Hay que terminar la parte de inventario y órdenes. Eso es lo más importante. Es de las cosas que se van a preguntar en la defensa.

Empezar a trabajar también la parte de autentificación.

Preguntas.

Sobre el menú, "El cual no es una tabla en especifica".

¿Como se debería implementar esa parte, ya que hay que sacar la información de varias tablas?

El menú verifica cuales platos están disponibles en cada branch en ese momento.

Consultar la disponibilidad del inventario.

¿Como mostrar esto?

Se definiría una serializador que no esté conectado a ningún modelo,
Poniéndole los campos que se consideren para presentarle al usuario, se podrian usar serializadores anidados.

Después se mapeoalos resultados de la base de datos a los serializadores que lo voy a mostrar al usuario.

Capa intermedia entre view y servicio que se puede llamar mapping.

El servicio se encarga de hacer las consultas en la base de datos.

Capa view.
Capa mappiing.
Capa servicio.

Las features que se van a revisar en la defensa son las siguientes:

Ordenes.
Inventario.
Autenticacion con google.

No se van a revisar tablas aisladas/pequeñas ni los CRUD de estas tablas, eso tiene menos importancia.

Hay que agregar docker al proyecto.