Realizar el proyecto usando Django 2.2 y postgres

1) Crear un app llamada bigbox y en el archivo **models.py** copiar el models.py que está en el correo.
2) Descomprimir fixtures.zip en la carpeta bigbox

Una vez creados los modelos, ejecutar los siguientes comandos en una terminal para que se haga la carga inicial:

python manage.py loaddata bigbox/fixtures/reason.json
python manage.py loaddata bigbox/fixtures/category.json
python manage.py loaddata bigbox/fixtures/activity.json
python manage.py loaddata bigbox/fixtures/box.json

Para verificar que todo está funcionando correctamente, ingresar a la shell de django y ejecutar los comandos para verificar los siguientes resultados:

Category.objects.all().count() == 25
Reason.objects.all().count() == 26
Activity.objects.all().count() == 148
Box.objects.all().count() == 1

3) Crear el siguiente archivo bigbox/**admin.py** y crear la vista del admin para los 4 modelos(Box, Activity, Category y Reason), Que me permita realizar un CRUD (Crear, Leer, Actualizar y Borrar) a cada uno de los modelos (Box, Activity, Category y Reason).

4) Crear las siguientes 4 Urls y Vistas para Box y Activity en bigbox/**view.py**
    a) base_url/box/
        i) Debe mostrar una lista de boxes
    b) base_url/box/{box_id}/
        i) Debe mostrar la información de la box
        ii) Debe mostrar las información primeras 5 activity (no importa el orden)
        iii) Debe haber un link que te redirija a base_url/box/{id}/activity/ donde se verán todas las actividades
    c) base_url/box/{box_id}/activity/
        i) Debe mostrar la lista de actividades dentro de el id de la box
        ii) Debe de mostrar las primeras 20 actividades y luego paginar de a 20 (usar el paginador de django)
    d) base_url/box/{box_id}/activity/{activity_id}/
        i) Debe mediante los parámetros recibidos de la url la relación entre actividad a través de la relación la box (ManyToManyField).
5) Modificar el modelo Box en **models.py,** agregar un nuevo campo de tipo CharField llamado slug con un max_length = 20,

    a) Generar la migración correspondiente
    b) Aplicar la migración correspondiente
    c) Crear una url y vista para buscar una box por el campo slug
        i) base_url/box/{slug}/ puede llamar al mismo template / archivo html del punto 4.b

6) Llenar el campo de slug con data (para probar que funcione)
7) Subir el repositorio a github
8) Subir el código una instancia de heroku

- Cosa a tener en cuenta :
    - Django Fixtures:
        - https://docs.djangoproject.com/en/2.2/howto/initial-data/
    - Django pagination:
        - https://docs.djangoproject.com/en/2.2/topics/pagination/
    - No es necesario aplicar css y js pero si quieres puedes usar:
        - https://getbootstrap.com/docs/4.4/examples/
        - https://getbootstrap.com/docs/4.4/examples/album/
    - Deploy Heroku
        - https://devcenter.heroku.com/articles/getting-started-with-python
        - https://devcenter.heroku.com/categories/working-with-django