Comandos Generales de Python

python test.py
streamlit run app.py

Comandos para Entornos Virtuales

rmdir /s venv
python -m venv venv
venv\Scripts\activate
Deactivate

Comandos para Gestión de Paquetes

python.exe -m pip install --upgrade pip
pip freeze > requirements.txt
pip freeze >> requirements.txt
pip install -r requirements.txt
pip uninstall -r requirements.txt -y

Comandos para Django

django-admin startproject miProyecto
cd miProyecto
python manage.py startapp clientes
python manage.py makemigrations
python manage.py migrate
python manage.py runserver
python manage.py changepassword

Orden Recomendado para un Proyecto Django

Crear el proyecto:

    django-admin startproject miProyecto
    cd miProyecto

Crear una aplicación:

    python manage.py startapp clientes

Configurar el entorno virtual:

    python -m venv venv
    venv\Scripts\activate

Instalar Django y otras dependencias:

    pip install django
    pip install -r requirements.txt

Configurar la base de datos:

python manage.py makemigrations

python manage.py migrate

Crear un superusuario:

python manage.py createsuperuser

Iniciar el servidor de desarrollo:

python manage.py runserver

Generar requirements.txt (al final del proyecto):

pip freeze > requirements.txt

Comandos Faltantes Aquí hay algunos comandos adicionales que podrían ser útiles:

python manage.py shell:
Descripción: Abre una consola interactiva de Django.

python manage.py createsuperuser:
Descripción: Crea un superusuario para acceder al panel de administración de Django.

python manage.py collectstatic:
Descripción: Recopila todos los archivos estáticos en una carpeta para producción.

python manage.py test:
Descripción: Ejecuta las pruebas unitarias del proyecto.

python manage.py shell
Ver los superusuarios existentes

Después Ejecuta
from django.contrib.auth.models import User
User.objects.filter(is_superuser=True)

Cambiar la contraseña de un superusuario existente
Si olvidaste la contraseña, puedes cambiarla con:
python manage.py changepassword nombre_usuario
from django.contrib.auth.models import User
user = User.objects.get(username='nombre_usuario')

```
user.set_password('nueva_contraseña')
user.save()
```

Registrar tus modelos en el Admin
Para que tus modelos Cliente y Factura aparezcan en el panel de administración, debes registrarlos en el archivo admin.py de tu aplicación:

```
from django.contrib import admin
from .models import Cliente, Factura

# Registrando los modelos básicos

admin.site.register(Cliente)
admin.site.register(Factura)

# O puedes personalizar la visualización:
# @admin.register(Cliente)
# class ClienteAdmin(admin.ModelAdmin):
#     list_display = ('nombre', 'apellidos', 'rfc', 'fecha_nacimiento', 'activo')
#     search_fields = ('nombre', 'apellidos', 'rfc')
#     list_filter = ('activo',)

# @admin.register(Factura)
# class FacturaAdmin(admin.ModelAdmin):
#     list_display = ('cliente', 'importe', 'pagada')
#     list_filter = ('pagada',)
```

Después de hacer estos cambios, deberías poder acceder al panel de administración con tu nuevo usuario y ver tus modelos registrados.

Resumen Orden recomendado:

Crear proyecto → Crear aplicación → Configurar entorno virtual → Instalar dependencias → Configurar base de datos → Iniciar servidor.

Comandos faltantes:

python manage.py shell,
python manage.py createsuperuser,
python manage.py collectstatic,
python manage.py test.