PASOS PARA CREAR PROYECTO DJANGO

- ✓ Crear carpeta del proyecto (código por consola 'mkdir' para crear carpeta)
  - Mkdir carpetaProyecto
- ✓ Entrar en carpetaProyecto y crear 2 carpetas dentro de carpetaProyecto.
  - Cd carpetaProyecto
    - Mkdir entorno
    - Mkdir repositorio
- ✓ Entrar en entorno y creamos el entorno virtual, hay que tener instalado virtualenv.
  - Cd entorno
    - Virtualenv nombreEntorno
- ✓ Entramos en nombreEntorno y luego en script y activamos el entorno.
  - Cd nombreEntorno
  - Cd script
    - Activate
- ✓ Nos tiene que aparecer el (nombreEntorno) al principio de la consola para saber que esta activado
  - **(entornoVirtual)**D:\silve\Escritorio\proyecto_django_final\entorno\entornoVirtual\Scripts>
- ✓ Volvemos hacia carpetaProyecto .
  - Cd..
  - Cd..
  - Cd..

- ✓ Entramos en repositorio y instalamos Django(este paso se puede hacer en cualquier carpeta mientra este activado el entornoVirtual) y creamos el proyecto Django.

    - Cd repositorio
        - Pip install django==version (instalar django puede ser en cualquier carpeta mientras este el entorno activado)
        - Django-admin startproject nombreProyecto (este si tiene que ser dentro de la carpeta repositorio)

- ✓ Entrar a nombreProyecto y verificar si corre

    - Cd nombreProyecto
        - Python manage.py runserver (este comando corre un servidor local para visualisar la pagina web)

CONFIGURACION DEL PROYECTO

- ✓ Estando dentro de nombreProyecto, a ese nivel de la carpeta creamos 3 carpetas:

    - Mkdir apps
    - Mkdir templates
    - Mkdir static

- ✓ Entramos en static y creamos 3 carpetas.

    - Cd static
        - Mkdir css
        - Mkdir js
        - Mkdir img

- ✓ Ahora volvemos a nombreProyecto y entramos a la carpeta llamada nombreProyecto que esta dentro de nombre proyecto(la carpeta se llama igual que la carpeta padre).

    - Cd..
    - Cd nombreProyecto

- ✓ Dentro de esta carpeta llamada nombreProyecto (Que esta dentro de la carpeta llamada igual) creamos otra carpeta llamada settings.

  - Mkdir settings

- ✓ Entramos en settings y creamos 3 archivos .py(en windows crear archivo por cmd es 'type nul > nombreArchivo.extends')

  - Type nul > local.py
  - Type nul > base.py
  - Type nul > production.py

- ✓ Los pasos siguientes se realizan directamente en el editor de texto

- ✓ En la carpeta nombreProyecto entramos en el archivo settings.py y copiamos todo y lo pegamos en el archivo base.py que esta dentro de la carpeta settings y borramos el archivo settings.py.

- ✓ Volvemos a entrar en base.py y cortamos estos codigo y pegamos en local.py y en local importamos .base.

Esto va en local.py:

- From .base import *
  - Linea 26
- DEBUG = True
  ALLOWED_HOSTS = []
  - Linea 76
- DATABASES = {
  'default': {
  'ENGINE': 'django.db.backends.sqlite3',
  'NAME': BASE_DIR / 'db.sqlite3',
  }
  }

- ✓ Luego entramos en el archivo manage.py y modificalos la linea 9

  - Antes
    - os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'blog.settings')
  - Ahora
    - os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'blog.settings.**local**')

- ✓ Entrar en archivo base.py y en la linea 54 en el diccionario TEMPLAPTES colocar y importa (os):

  - Import os

  - Antes
    - 'DIRS': [],
  - Despues
    - 'DIRS': [os.path.join(os.path.dirname(BASE_DIR),'templates')],

CREAR APPS

- ✓ Por consola entrar en la carpeta apps y crear una app.

  - Cd apps
    - Django-admin startapp nombreApps

- ✓ Ir a base.py y en el diccionario INSTALLED_APPS ingresar la apps creada.

  - antes
    - INSTALLED_APPS = [
      'django.contrib.admin',
      'django.contrib.auth',
      'django.contrib.contenttypes',
      'django.contrib.sessions',
      'django.contrib.messages',
      'django.contrib.staticfiles',
      ]

- ahora
  - INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'Apps.usuarios',

    ]

- ✓ Luego creamos una variable llamada AUTH_USER_MODELS

  - AUTH_USER_MODEL = 'usuarios.Usuario'

- ✓ Luego en la carpeta usuarios en el archivo apps.py modificamos la clase.

  - Antes
    - Class UsuariosConfig(Appconfig):
      Default_auto_field = 'django.db.models.BigAutoField'
      Name = 'usuarios'
  - Despues
    - Class UsuariosConfig(Appconfig):
      Default_auto_field = 'django.db.models.BigAutoField'
    - Name = '**apps.**usuarios'

- ✓ Por consola crear apps usuarios(este es el paso que se realizo en clases).

  - Cd apps
    - Django-admin startapp usuarios

Este genera una carpeta llamada usuarios dentro de la carpeta apps.

- ✓ Entramos en el archivo models.py que esta dentro de la carpeta usuarios y importamos AbstractUser y creamos la clase usuario

  - from django.contrib.auth.models import AbstractUser

- class Usuario(AbstractUser):
  pass

MIGRAR DATOS

- ✓ Comandos que se usan para migrar datos cuando se crean o modifican clases:
  - Makemigrations (para avisar que hay modificacion o creacion)
    - Uso por consola
      - Python manage.py make migrations
  - Migrate (para ingresar los datos modificados o creados en la base de datos)
    - Uso por consola
      - Python manage.py migrate

Siempre hay que estar parado en donde esta le archivo manage.py

CREAR VISTAS

- ✓ Crear un archivo llamado urls.py a cada apps que tengan(se puede hacer por consola o por el editor).
  - Consola
    - Estando dentro de la carpeta de la app crear
      - Type nul>urls.py
- ✓ Crear un archivo llamado views.py dentro de la carpeta nombreProyecto al mismo nivel que esta la carpeta settings.
  - Type nul>views.py
- ✓ Entramos en urls de la carpeta nombreProyecto y modificamos el urlpatterns=[] y importamos el views.py .

- from . import views

- Antes
  - urlpatterns = [
    path('admin/', admin.site.urls),
    ]
- Ahora
  - urlpatterns = [
    path('admin/', admin.site.urls),

    Path(‘’,views.Home, name = ‘home’),
    ]

✓ Luego en el archivo views.py de la carpeta nombreProyecto importamos render y creamos la funcion Home.

- from django.shortcuts import render

- def Home(request):
    return render(request,’home.html’)

✓ Luego ingresamos en la carpeta templates y creamos un archivo llamado igual que en el views home.html

- Cd templates
  - Type nul>home.html

CREANTO ARCHIVOS STATIC

✓ Entrar en base.py y chequear que en INSTALLED_APPS se encuentre:

- 'django.contrib.staticfiles',

✓ Tambien verificar que se ecuentre la variable:

- STATIC_URL = ‘/static/’

- ✓ Y crear la variable :

  - STATICFILES_DIRS=(
    os.path.join(os.path.dirname(BASE_DIR),"static"),
)

- ✓ Para referir los archivos static, ejemplo css se crea en el html el link y en el href se coloca :

  - <link rel="stylesheet" href="**{% static "css/style.css" %}**">
  - <script src="**{% static 'js/rain.js' %}**"></script>

SUPERUSUARIO

- ✓ Por consola y posicionado en la carpeta donde esta el manage.py ingresar el comando siguiente:

  - Python manage.py createsuperuser

- ✓ Va soliciart nombre de usuario, correo, y clave

CREAR LOGIN

- ✓ Creamos una carpeta en templates llamada usuario para colocar los template de login logout
- ✓ Creamos un html llamado login
- ✓ Ingresamos esta porcion de codigo:

  - {% extends 'base.html' %}
    {% load static %}

    {% block content %}
    <form method="POST">{%csrf_token%}
    {{form.errors}}
    <input type="text" name="username"
placeholder="USUARIO">

```
                        <input type="password" name="password"
placeholder="PASSWORD">

                        <input type="submit" name=""
value="ENVIAR">
                    </form>
{% endblock %}
```

- ✓ Luego ingresamos en el archivo base.py dentro de setting y introduciomos lo siguentes a demas importamos reverse_lazy :

  - LOGIN_REDIRECT_URL = reverse_lazy('home')
  - LOGOUT_REDIRECT_URL = reverse_lazy('home')
  - LOGIN_URL = reverse_lazy('login')
  - from django.urls import reverse_lazy

- ✓ Ingresamos al url principal y importamos

  - from django.contrib.auth import views as auth

- ✓ Luego en el diccionario urlpatterns colocamos

  - path('login/',auth.LoginView.as_view(template_name='usuarios/login.html'),name='login'),
  - path('logout/',auth.LogoutView.as_view(),name="logout"),

Crear una funcion de django en el html para saber si esta logeado o no

- ✓ Entramos al base.html o donde este el boton de login y modificamos :
  - {% if user.is_authenticated %}

```
    <strong>Hola {{ user.username }}</strong>
            <a href="{% url'logout' %}">LOGOUT</a>
{% else %}
    <a href="{% url 'login' %}">LOGIN</a>
```

Registrar
{% endif %}

CREAR REGISTRO

- ✓ Ahora en urls global ingresamos la direcicon de url para usar la apps de usuarios
  - path('Usuario/',include('apps.usuarios.urls'))
- ✓ Ahora en la urls de la apps usuarios escribimos el siguiente codigo
  - from django.urls import path

```
from . import views

    # LA URL EMPIEZA CON Noticias/ <este archivo>

app_name = 'usuarios'

urlpatterns = [

    path('registro/', views.Registro.as_view(), name = 'registro'),



]
```

- ✓ En views de la apps usuario ingresamos el siguiente codigo
  - from django.shortcuts import render
  - from django.views.generic import CreateView
  - from django.urls import reverse_lazy
  - from .forms import RegistroForm
  - class Registro(CreateView):
    #FORMULARIO DJANGO
    form_class = RegistroForm

```
    success_url = reverse_lazy('login')
    template_name = 'usuarios/registro.html'
```

- ✓ El la carpeta usuario creamos un archivo.py llamado forms.py y escribimos el siguiente codigo
  - from django import forms

  - from django.contrib.auth.forms import UserCreationForm
  - from .models import Usuario

  - class RegistroForm(UserCreationForm):

```
    email = forms.EmailField(label='Correo', required=True)
    first_name = forms.CharField(label='Nombre', required=True)
    last_name = forms.CharField(label='Apellido', required=True)
    password1 = forms.CharField(
        label='Contraseña', widget=forms.PasswordInput, required=True)
    password2 = forms.CharField(
        label='Confirmar Contraseña', widget=forms.PasswordInput,
required=True)

    class Meta:
        model = Usuario
        fields = [
            'first_name',
            'last_name',
            'username',
            'email',
            'password1',
            'password2'
        ]
```