en este video nos enseña a instalar python, crear un entorno virtual, descargar django en el entorno virtual e iniciar un nuevo proyecto desde django

para instalar python usamos "sudo apt install python3"

para hacer en entorno virtual he utilizado "python3 -m vemv nombre_del_entorno_virtual"

para descargar el django he utilizado "pip3 install django"

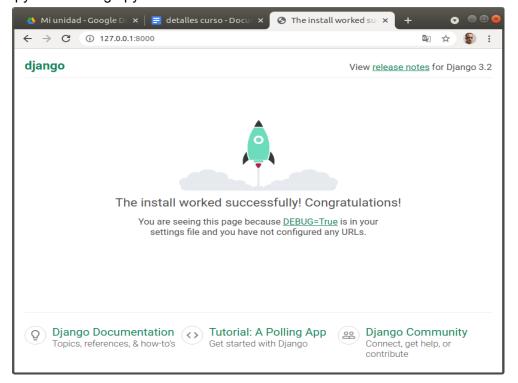
para crear el proyecto tenemos que poner lo siguiente "python3 localizacion_De_django_admin.py startproject nombre_del_projecto"

cabezazos: para encontrar "django-admin.py" tube qu ebuscar carpeta por carpeta hasta encontrar porque en el video esta en otra ubicación posiblemente por que lo este haciendo en windows, el archivo está en

"ven/lib64/python3.6/site-packages/django/bin/django-admin.py"

video 2

en este video aprendemos a iniciar el servidor del entorno de desarrollo, para ello tenemos que inciar el programa mange.py con la opcion de run server "python3 mange.py runserver"



vamos a crear un vínculo entre la base de datos y el proyecto para crear el vínculo usamos "python3 mange.py migrate"

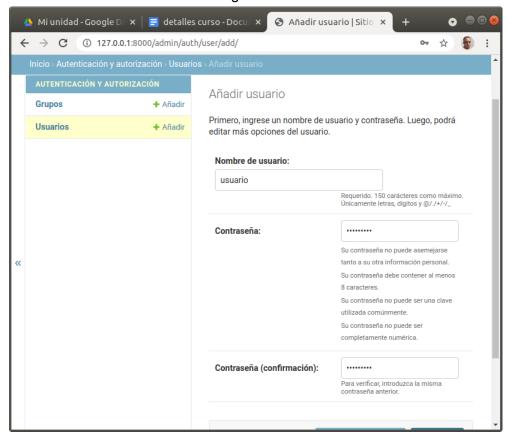
video 4

creamos un super usuario para el entorno de trabajo usando "python3 manage.py createsuperuser"

nos dirigimos a settings.py y cambimos "LANGUAGE_CODE" a "es" para cambiar el idioma a español



creamos un usuario desde el navegador



vamos a crear una aplicacion usando la terminal "python3 manage.py startapp boletin"

configuramos en settings.py para añadir la apliacion al entorno de trabajo para ello vamos a modificar "INSTALLED_APPS" y añadimos "boletín"

```
INSTALLED_APPS = []
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'boletin', You, seconds ago * Uncom
```

video 6

vamos a crear un modelo para registrar a los suaurios para ello tenemos que modificar models.py

para guardar todas las modificacione echa tenemos que utilizar "python3 mange.py makemigrations" y despues "python3 mange.py migrate"

video 7

vamos a crear objetos y guardarlos en la base de datos desde la shell para acceder "python3"

tenemos que hacer una importación del modelo que hemos creado usando "from boletin.models import Registrado"

después creamos una nueva variable del modelo

```
>>> gente = Registrado.objects.all()
```

ahora vamos a guardar objetos en la lista

```
Registrado.objects.create(nombre='sergio',email='sergio@hostname')
```

para poder objetos en la lista desde al interfaz debemos importar el modelo en admin.py y registrarlo en administracion para acceder al el

```
from .models import Registrado
admin.site.register(Registrado)
```

video 8

ahora vamos a personalizar el display del modelo modificando admin.py



video 9

vamos a hacer una vista, para hacerla debemos dirigirlo a views.py para crear la vista y urls.py para asignale una url

```
def inicio(request):
    return render[[request, "inicio.html", {}]]
```

```
from boletin import views
#from boletin.views inicio You, a minute ago

urlpatterns = [
    path('admin/', admin.site.urls),
    path('', views.inicio, name='inicio'),
]
```

configuramos la ubicación del directorio donde se tienen que buscar las plantillas, se configura en setting.py

y creamos en el directorio templates el archivo los archivos html que queramos



Video 11

vamos a crear formularios, para crear el formulario tenemos que crear un fichero forms.py en la apliacion y lo configuramos anusestre gusto

```
from django import forms

class RegForm(forms.Form):
    nombre = forms.CharField(max_length=100)
    edad = forms.IntegerField()
```

vamos a añadir el formulario a la vista

```
def inicio(request):
    form = RegForm()
    context = {
        "el_form" : form,
    }
    return render (request, "inicio.html", context)
```

después tenemos que añadir el formulario al html para que se puede rellenar



Video 13

vamos a configurar el formulario en el html para que guarde en la base de datos y hacer que no nos envíe a otra pagina



vamos ha poner "request.POST" para hacer refencia a lo que introducimos en el formulario y "or None" para que no nos alga mensaje de es un campo obligatorio

Hola mundo

Este campo es obligatorio.

Nombre:

Este campo es obligatorio.

Edad:

Registrama

después guardamos la información en del formulario en un formato limpio para poder usarla

video 15

vamos a hacer que el formulario almacene objetos nuevos usando nuestro modelo, primero tenemos que cambiar el formulario para que encaje con el modelo

```
class RegForm(forms.Form):
    nombre = forms.CharField(max_length=100)
    email = forms.EmailField() You, 25 minu
```

después tenemos que almacenar esos datos en el modelo, para ello tenemos que importar el modelo y crear una variable donde se guarden los datos que queremos utilizar y otra para insertar los datos

from django.shortcuts import render from .forms import RegForm from .models import Registrado			
# Create your views here.			
def	<pre>def inicio(request): form = RegForm(request.POST or None) if form.is_valid(): form_data = form.cleaned_data abc = form_data.get("email") abc2 = form_data.get("nombre") obj = Registrado.objects.create(email=abc, nombre=abc2) context = { "el_form" : form, } }</pre>		
	return render (re	equest, "inicio.html", context) You, 3 h	
	EMAIL	NOMBRE	
	paco@localhost	paco	
	sergio@hostname	sergio	

Hola mundo

Nombre: paco			
Email:	paco@localhost		
Registr	rama		

vamos a crear un modelo de formulario par sustituir el que nos introduce django por defecto, tenemos que crear el modelo en forms.py y asignale los campos que queramos que muestre

```
You, 3 minutes ago | 1 author (You)

class RegModelForm(forms.ModelForm):

You, 3 minutes ago | 1 author (You)

class Meta:

model = Registrado

fields =["nombre", "email"]
```

después nos vamos ha admin.py, importamos el modelo de formulario y lo sustituimos por el que tenemos por defecto

```
# Register your models here.
from .forms import RegModelForm
from .models import Registrado
You, seconds ago | 1 author (You)
class AdminRegistrado(admin.ModelAdmin):
    list_display = ["email", "nombre", "timestamp"]
    form = RegModelForm #modelo del formulario
    #list_display_links = ["nombre"] #campo que tienen el link haci
    list_filter = ["timestamp"]
    list_editable = ["nombre"] #campo que se permite modificar
    search_fiels = ["email", "nombre"] #campos que se pueden buscar
# class Meta:
# model = Registrado

admin.site.register(Registrado, AdminRegistrado)) You, 2 days a
```