# **Resumen Django para aprobar**

## Empezando proyecto

Nota: Si se te joden las tablas en phpmyadmin/localhost, te vas a mysql (en el lateral izquierdo), seleccionas todas, y eliges reparar tablas.

* **Creamos el entorno:** *python -m venv entornoNombre*
* **Entramos al proyecto:***Scripts\activate.bat*
* **Instalamos Django:** *pip install Django*
* **Creamos proyecto:** *django-admin startproject proyectoNombre*
* **Instalamos mysql:** *pip install mysqlclient*
* **Creamos la app:** *python manage.py startapp nombreApp*
* **Para poner en marcha el servidor:** *python manage.py runserver*
* **Crear base de datos en localhost/phpmyAdmin y cambiamos los settings.py (Linea 76):**

*DATABASES = {*

*'default': {*

*'ENGINE': 'django.db.backends.mysql',*

*'NAME': 'agendaEmpresas',*

*'USER': 'agendaEmpresas',*

*'PASSWORD': 'agendaEmpresas',*

*'HOST': 'localhost',*

*'PORT': '3306',*

*}*

*}*

**En la línea 33 añadimos nuestra app**

*INSTALLED\_APPS = [*

*'django.contrib.admin',*

*'django.contrib.auth',*

*'django.contrib.contenttypes',*

*'django.contrib.sessions',*

*'django.contrib.messages',*

*'django.contrib.staticfiles', #Esta fila es para que salga bonito admin*

*'principal.apps.PrincipalConfig'*

*]*

**nota: también cambiar en linea 111 el idioma a ‘es’**

* **Aunque no hayamos creado ningún modelo, hacemos las migraciones:** *python manage.py makemigrations*

*python manage.py migrate*

* **Modificamos los modelos: Lo mejor es fijarse en un proyecto ya creado, este es un ejemplo.**

**Y volvemos a hacer las migraciones.**

**nota: puede pedir install pillow:** *python -m pip install Pillow*

* **Añadimos para poder entrar en admin en admin.py**

*from django.contrib import admin*

*from .models import Empresas*

*# Register your models here.*

*admin.site.register(nombreApp)*

* **Creamos el admin:** *python manage.py createsuperuser*
* **Creamos el archivo urls.py en la carpeta de la aplicación.Se pondrá una cosa u otra si se usan clases o no.**
* **Creamos la carpeta templates, y dentro de la carpeta otra carpeta llamada nombreApp**
* **Modificamos el urls.py de proyectoNombre**

*from django.contrib import admin*

*from django.urls import path,include*

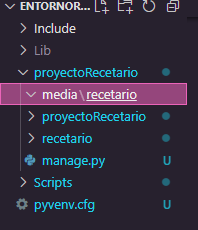
*urlpatterns = [*

*path('',include('nombreApp.urls')),*

*path('admin/', admin.site.urls),*

*]*

## Imágenes



Si quieres guardar imágenes, en proyectoNombre, tienes que crear la carpeta media y dentro de este la carpeta del nombre de la app.

Ejemplo con recetario.

**Hay que modificar el urls.py del proyecto**

*from django.conf import settings*

*if settings.DEBUG:*

*from django.conf.urls.static import static*

*urlpatterns += static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)*

**Settings.py añadirlo al final**

*import os*

*#MEDIA*

*MEDIA\_URL='/media/'*

*MEDIA\_ROOT=os.path.join(BASE\_DIR,'media')*

**nota: Si es con clases,** [**ir aquí**](#_kpatnpimeike)

## Django Sin Clases

* **Modificamos el archivo urls.py de la app**

*from django.urls import path*

*from . import views*

*urlpatterns = [*

*path('',views.listado, name='listado')]*

* **Creamos el template listado.html Lo mejor será buscar en otros proyectos para más ejemplos.**

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<title>Listado</title>*

*</head>*

*<body>*

*{% for e in empresas %}*

*<ul>*

*<li>{{e.nombre}}</li>*

*<li>{{e.tipo}}</li>*

*</ul>*

*{% endfor %}*

*</body>*

*</html>*

* **Modificamos el views.**

*from http.client import HTTPResponse*

*from django.shortcuts import render, get\_list\_or\_404*

*from django.template import loader*

*from django.http import HttpResponse*

*from .models import Empresas*

*# Create your views here.*

*def listado (request):*

*#modelo*

*empresas = get\_list\_or\_404(Empresas)*

*#contexto*

*ctx = {*

*'empresas': empresas,*

*}*

*#template*

*listadoHtml = loader.get\_template('principal/listado.html')*

*#retirn template*

*return HttpResponse(listadoHtml.render(ctx,request))*

## Django Con Clases

* **Modificamos urls.py de la app**

*from django.urls import path*

*# from . import views*

*from .views import EmpresaListView*

*urlpatterns = [*

*path('',EmpresaListView.as\_view(), name='listado')*

*]*

* **Modificamos views.py**

*from django.views.generic.list import ListView*

*from .models import Empresas*

*# Create your views here.*

*class EmpresaListView(ListView):*

*model = Empresas*

* Creamos el html, suele ser nombre\_listado.html, pero don't worry, si lo pones mal, te dice cómo se debe llamar. Usare de ejemplo, empresas\_list.html Aqui igual, buscar en proyectos ya creados.

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<title>Listado</title>*

*</head>*

*<body>*

*<h1>Listado Empresas</h1>*

*<!-- se tiene que llamar a la clase que hemos creado en las views, se tiene que llamar como modelo\_list -->*

*{% for e in empresas\_list %}*

*<ul>*

*<li>{{e.nombre}}</li>*

*<li>{{e.tipo}}</li>*

*</ul>*

*<hr>*

*{% endfor %}*

*</body>*

*</html>*

* De aquí en adelante es ir añadiendo views, sus respectivas urls y sus respectivos htmls. Hemos visto:
  + *DetailView*
  + *CreateView*
  + *UpdateView*
  + *DeleteView*

## Reutilizar HTML

Te creas dentro de templates/nombreApp, otra carpeta que se llame includes.

Ponemos el ejemplo de modificar o borrar una receta, y verifica que se está logueado.

*{% if request.user.is\_authenticated%}*

*<a href="{% url 'borrar' receta.id %}">Borrar</a><br>*

*<a href="{% url 'modificar' receta.id %}">Modificar</a>*

*{% endif %}*

Para poder utilizarlo, simplemente lo importamos en el html que queramos, de la siguiente manera.

***{% include ‘app/includes/nombre\_crear.html' %}***

*{% include 'recetario/includes/receta\_crear.html' %}*

## Formularios

Creamos el archivo **forms.py** dentro de la carpeta de la aplicación.

Usaremos de ejemplo, recetario, pero siempre puedes mirar otras tareas.

*from django import forms*

*from .models import Receta*

*class RecetaForm(forms.ModelForm):*

*class Meta:*

*model = Receta*

*fields = ['nombre','subnombre','ingredientes','imagen','receta','author','categorias']*

*widgets = {*

*'nombre': forms.TextInput(attrs={'style':'background-color:red'}),*

*'subnombre': forms.TextInput(attrs={'style':'background-color:grey','placeholder':'Escribe el subnombre'}),*

*}*

*labels = {*

*'subnombre': 'Sub-nombre'*

*}*

**Y modificamos las vistas:**

*from .forms import RecetaForm*

*class RecetarioCreateView(CreateView):*

*model = Receta*

*form\_class=RecetaForm*

*success\_url = reverse\_lazy('recetas')*

*class ventaCamionesUpdateView(UpdateView):*

*model = Camion*

*form\_class = CamionForm*

*template\_name\_suffix = '\_update\_form' #Lo que hace es buscar otro form*

*success\_url = reverse\_lazy('listado')*

**nota: recordar que en el formulario si tiene imagen tienes que poner enctype="multipart/form-data"**

## Log In y Log out

* **Creamos los registros:** *python manage.py startapp registration*
* **Modificamos urls.py del proyecto:**

*from django.contrib import admin*

*from django.urls import path,include*

*from django.conf import settings*

*urlpatterns = [*

*path('',include('ventaCamiones.urls')),*

*path('admin/', admin.site.urls),*

*#rutas de autenticacion*

*path('cuentas/',include('django.contrib.auth.urls')),*

*]*

*if settings.DEBUG:*

*from django.conf.urls.static import static*

*urlpatterns += static(settings.MEDIA\_URL,document\_root=settings.MEDIA\_ROOT)*

* **modificamos settings.py linea 34,**

*INSTALLED\_APPS = [*

*'registration',*

*'django.contrib.admin',*

*'django.contrib.auth',*

*'django.contrib.contenttypes',*

*'django.contrib.sessions',*

*'django.contrib.messages',*

*'django.contrib.staticfiles',*

*'ventaCamiones.apps.VentacamionesConfig'*

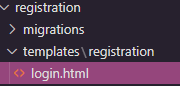
*]*

**Al final del todo añadimos**

*#Configuracion de autenticacion*

*LOGIN\_REDIRECT\_URL='listado'*

*LOGOUT\_REDIRECT\_URL='listado'*

* **En la carpeta registration que se creó, creamos la carpeta templates, dentro registration, y dentro el archivo login.html**

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<!-- Latest compiled and minified CSS -->*

*<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel=" stylesheet ">*

*<!-- Latest compiled JavaScript -->*

*<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js "></script>*

*<title>Login</title>*

*</head>*

*<body>*

*<div class="container">*

*<div class="row">*

*<div class="col-3 text-center"></div>*

*<div class="col-6 text-center">*

*<h1 class="display-4">Log in</h1>*

*<form method="post" enctype="multipart/form-data">*

*{% csrf\_token %}*

*{{ form.as\_p}}*

*<input type="submit" value="Login" class="btn btn-success">*

*</form>*

*</div>*

*</div>*

*</div>*

*</body>*

*</html>*

***Nota: También puedes inspeccionar la página, copiar lo que crea form.as\_p y pegarlo, y ya darle bootstrap***

* **Se añaden los botones en el html que se desee, de la siguiente manera**

**Revisar proyecto camiones para más info.**

*{% if not request.user.is\_authenticated%}*

*<a href="{%url 'login' %}" class="btn btn-primary">Login</a>*

*{% else %}*

*<a href="{%url 'logout' %}" class="btn btn-primary">Logout</a>*

*{% endif %}*

* **Evitar que puedan crear, modificar y borrar, tienen que estar logueados.En views.py de la aplicación.**

*from django.shortcuts import render*

*from django.views.generic.list import ListView*

*from django.views.generic.detail import DetailView*

*from django.views.generic.edit import CreateView, UpdateView, DeleteView*

*from django.urls import reverse\_lazy*

*from .forms import CamionForm*

*from .models import Camion*

***from django.contrib.auth.mixins import LoginRequiredMixin***

*# Create your views here.*

*class ventaCamionesListView(ListView):*

*model = Camion*

*class ventaCamionesDetailView(DetailView):*

*model = Camion*

*class ventaCamionesCreateView(LoginRequiredMixin,CreateView):*

***login\_url = '/cuentas/login/'***

*model = Camion*

*form\_class = CamionForm*

*success\_url = reverse\_lazy('listado')*

*class ventaCamionesUpdateView(LoginRequiredMixin,UpdateView):*

***login\_url = '/cuentas/login/'***

*model = Camion*

*form\_class = CamionForm*

*template\_name\_suffix = '\_update\_form' #Lo que hace es buscar otro form*

*success\_url = reverse\_lazy('listado')*

*class ventaCamionesDeleteView(LoginRequiredMixin,DeleteView):*

***login\_url = '/cuentas/login/'***

*model = Camion*

*success\_url = reverse\_lazy('listado')*

## Registro

* **Añadimos en urls.py del proyecto:**

*path('cuentas/',include('registration.urls')),*

* **En la carpeta de registration, creamos urls.py.**

*from django.urls import path*

*from .views import RegistroView*

*urlpatterns = [*

*path('registro/',RegistroView.as\_view(),name = 'registro')*

*]*

* **En views.py de registration:**

*from django.contrib.auth.forms import UserCreationForm*

*from django.views.generic import CreateView*

*from django.urls import reverse\_lazy*

*# Create your views here.*

*class RegistroView(CreateView):*

*form\_class = UserCreationForm*

*success\_url = reverse\_lazy('login')*

*template\_name = 'registration/registro.html'*

* **Creamos el archivo registro.html en los templates de registrations**

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<!-- Latest compiled and minified CSS -->*

*<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel=" stylesheet ">*

*<!-- Latest compiled JavaScript -->*

*<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js "></script>*

*<title>Registro</title>*

*</head>*

*<body>*

*<div class="container">*

*<div class="row">*

*<div class="col-3 text-center"></div>*

*<div class="col-6 text-center">*

*<h1 class="display-4">Registro</h1>*

*<form method="post" enctype="multipart/form-data">*

*{% csrf\_token %}*

*{{ form.as\_p}}*

*<input type="submit" value="Registrar" class="btn btn-success">*

*</form>*

*</div>*

*</div>*

*</div>*

*</body>*

*</html>*

* **Modificamos el views.py para que la url que devuelve sea distinta. Lo que conseguimos es que la url cambie de** [**http://127.0.0.1:8000/cuentas/login/**](http://127.0.0.1:8000/cuentas/login/) **a** [**http://127.0.0.1:8000/cuentas/login/?registrado**](http://127.0.0.1:8000/cuentas/login/?registrado) **y que se utilizara en login.html**

*from django.contrib.auth.forms import UserCreationForm*

*from django.views.generic import CreateView*

*from django.urls import reverse\_lazy*

*# Create your views here.*

*class RegistroView(CreateView):*

*form\_class = UserCreationForm*

***#success\_url = reverse\_lazy('login') +'?registrado'***

*template\_name = 'registration/registro.html'*

***def get\_success\_url(self) -> str:***

***return reverse\_lazy('login') +'?registrado'***

* **Modificamos login.html**

*<!DOCTYPE html>*

*<html lang="en">*

*<head>*

*<meta charset="UTF-8">*

*<meta http-equiv="X-UA-Compatible" content="IE=edge">*

*<meta name="viewport" content="width=device-width, initial-scale=1.0">*

*<!-- Latest compiled and minified CSS -->*

*<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel=" stylesheet ">*

*<!-- Latest compiled JavaScript -->*

*<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js "></script>*

*<title>Login</title>*

*</head>*

*<body>*

*<div class="container">*

*<div class="row">*

*<div class="col-3 text-center"></div>*

*<div class="col-6 text-center">*

*<h1 class="display-4">Log in</h1>*

***{% if 'registrado' in request.GET%}***

***<p class="text-success">Te has registrado correctamente</p>***

***{% endif %}***

*<form method="post" enctype="multipart/form-data">*

*{% csrf\_token %}*

*{{ form.as\_p}}*

*<input type="submit" value="Login" class="btn btn-success">*

*</form>*

*</div>*

*</div>*

*</div>*

*</body>*

*</html>*

* **Para añadir el email en el formulario de registro, tenemos que crear forms.py en la carpeta registration.**

*from django import forms*

*from django.contrib.auth.forms import UserCreationForm*

*from django.contrib.auth.models import User*

*#Esta clase hereda de UserCreationForm*

*class UsarioCrearFormConEmail(UserCreationForm):*

*email = forms.EmailField(required=True,help\_text="Obligatorio. 254 caracteres y un email valido")*

*#Para indicar y modificar atributos que necesita la clase en la que esta*

*#Se tiene que crear si o si dentro de la clase que la va a usar.*

*class Meta:*

*model = User*

*fields = ("username","email","password1","password2")*

* **Ahora vamos a views.py de registration e importamos la nueva clase creada en forms, y añadimos el email.**

*from .forms import UsarioCrearFormConEmail*

*#from django.contrib.auth.forms import UserCreationForm*

*from django.views.generic import CreateView*

*from django.urls import reverse\_lazy*

*from django import forms*

*# Create your views here.*

*class RegistroView(CreateView):*

*#form\_class = UserCreationForm*

*form\_class = UsarioCrearFormConEmail*

*#success\_url = reverse\_lazy('login') +'?registrado'*

*template\_name = 'registration/registro.html'*

*def get\_success\_url(self) -> str:*

*return reverse\_lazy('login') +'?registrado'*

*#Esto es para darle forma cuando en el html se use {{ form.as\_p}}*

*def get\_form(self, form\_class=None):*

*form = super(RegistroView,self).get\_form()*

*form.fields['username'].widget = forms.TextInput(attrs={'placeholder':'Nombre'})*

*form.fields['email'].widget = forms.EmailInput(attrs={'placeholder':'Email'})*

*form.fields['password1'].widget = forms.PasswordInput(attrs={'placeholder':'Contraseña'})*

*form.fields['password2'].widget = forms.PasswordInput(attrs={'placeholder':'Repita contraseña'})*

*return form*

## 

## 

## Recuperar contraseña

Nota: Parece ser que esto no entra.

## **En settings.py al final del todo añadimos**

*#Configuracion de correo*

*if DEBUG:*

*EMAIL\_BACKEND = "django.core.mail.backends.filebased.EmailBackend"*

*EMAIL\_FILE\_PATH = os.path.join(BASE\_DIR,"correoEnviados")*

*else:*

*#Configuracion de correo de verdad*

*pass*

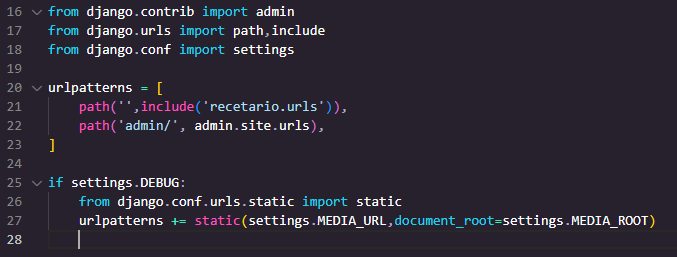
## Copiar una app de otro proyecto

Se tiene que seguir los pasos de [aquí](#_jr0s2gd9ba2m) excepto en el que creas la app.

Nota: no seas un kk he intentes copiar el settings, no funciona, sigue los pasos

Descargas el fichero .zip si la app ya la creó otra persona y descomprimirlo, o si es tuya, tienes que copiar la app, eliminar las migrations antes.

Modificar urls.py global:



Toca hacer las migraciones (Hay que estar dentro del proyecto) y crear superusuario.

***python manage.py makemigrations***

***python manage.py migrate***

***python manage.py createsuperuser***