



unl

Universidad
Nacional
de Loja

POSGRADO

Desarrollo de Software en Ambientes Cloud

Maestría en Ingeniería en Software - UNL



Carrera de Ingeniería en
Sistemas / Computación



Universidad
Nacional
de Loja





Bienvenidos!

Ing. Roberth Figueroa Díaz M.Sc.

roberth.figueroa@unl.edu.ec

Profesor



Universidad
Nacional
de Loja

1859



Carrera de Ingeniería en
Sistemas / Computación

Agenda

1. Tecnologías del lado del Servidor o Backend
2. Instalando Framework Django
 - Instalar Django
 - Identificar comandos claves
 - Realizar un ejemplo con Django
 - Conclusiones y recomendaciones en Django

INSTALACIÓN DJANGO

3.2.3

Virtual Env

virtualenv → Entorno Python Aislado

Virtualenv

[Virtualenv](#) is a tool that lets you create an isolated Python environment for your project. It creates an environment that has its own installation directories, that doesn't share dependencies with other `virtualenv` environments (and optionally doesn't access the globally installed dependencies either). You can even configure what version of Python you want to use for each individual environment. It's very much recommended to use `virtualenv` when dealing with Python applications.

Installation

To install `virtualenv` run:

```
pip install virtualenv
```

Virtual Env → entorno virtual

entorno principal
administrado por Sistema operativo

entorno virtual llamado Borrrar

```
[macrob:bin roberth$ activate
-bash: activate: command not found
[macrob:bin roberth$ source activate
(Borrrar) macrob:bin roberth$ deactivate
[macrob:bin roberth$ pip freeze
appdirs==1.4.4
asgiref==3.3.4
distlib==0.3.2
Django==3.2.3
filelock==3.0.12
pytz==2021.1
six==1.16.0
sqlparse==0.4.1
virtualenv==20.4.7
[macrob:bin roberth$ pwd
/Users/roberth/Developer/Borrrar/bin
[macrob:bin roberth$ source activate
(Borrrar) macrob:bin roberth$ pip freeze
asgiref==3.5.2
Django==4.0.5
sqlparse==0.4.2
(Borrrar) macrob:bin roberth$ date
Tue Jun 14 01:47:26 -05 2022
(Borrrar) macrob:bin roberth$ ]
```

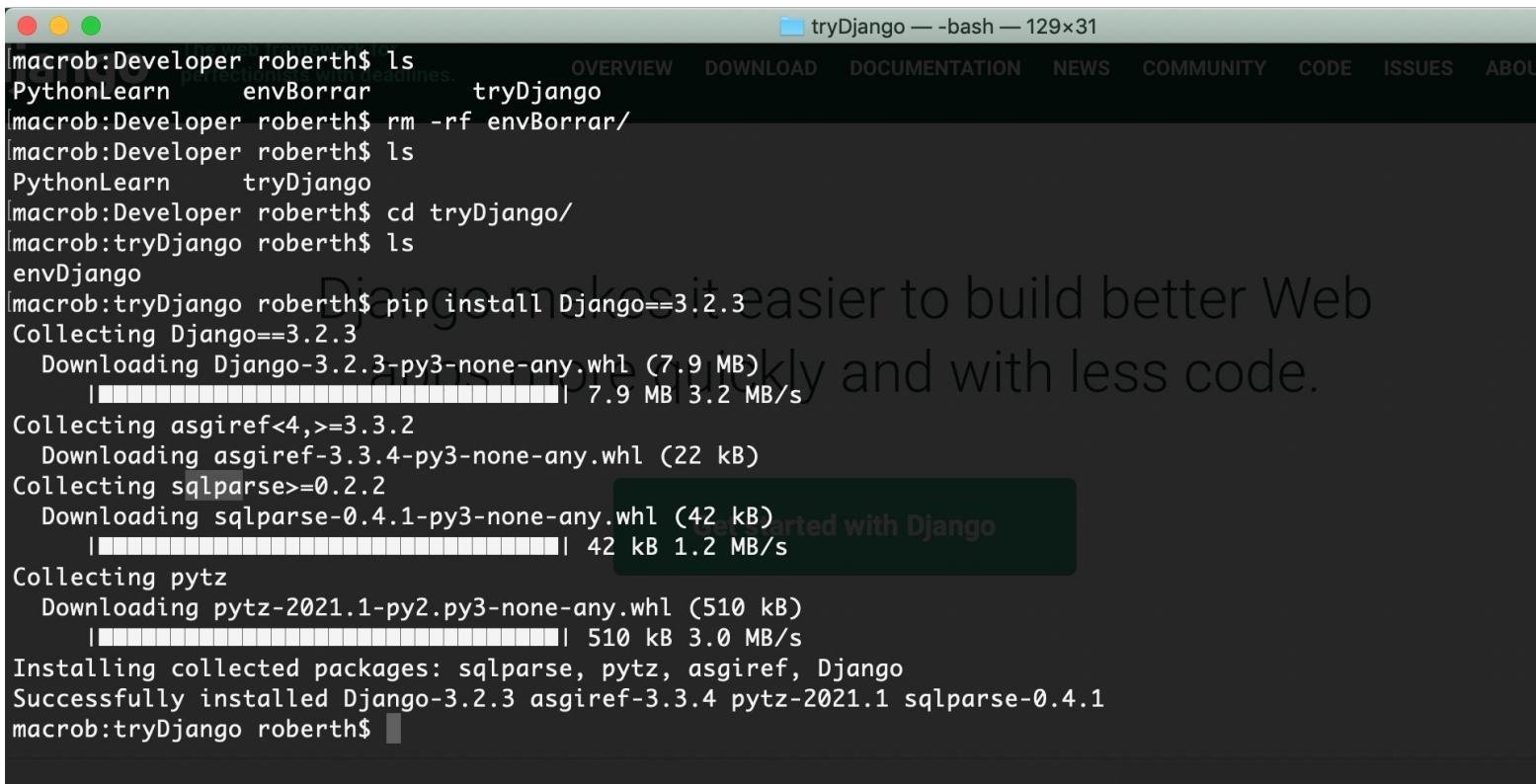
Pasos a seguir

AMBIENTE DESARROLLO

1. Instalando PIP en Python
 - Usar comando → **sudo easy_install pip**
2. Instalando virtual environment
 - Usar comando → **sudo pip install virtualenv**
3. *Crear el entorno virtual deseado*
 - Usar comando → **virtualenv envDjango**
 - O definir la versión de python al entorno → **virtualenv -p python3** .
 - Se crean carpetas **bin, lib** con archivos Python necesarios
4. *Instalar Django*
 - Comando para activar virtualenv → **source bin/activate**
 - Comando para instalar → **pip install Django==3.2.3**
 - Comando para actualizar la versión django → **pip install django -- upgrade**
 - Comando para ver la versión de django instalada → **pip freeze**
5. Crear la aplicación **mi_proyecto** dentro del environment
 - Comando → **django-admin startproject mi_proyecto**
6. Se genera la base de datos (SQLite):
 - Comando → **python manage.py migrate**
7. Se ejecuta el servidor de aplicaciones web:
 - Comando → **python manage.py runserver**

Proceso

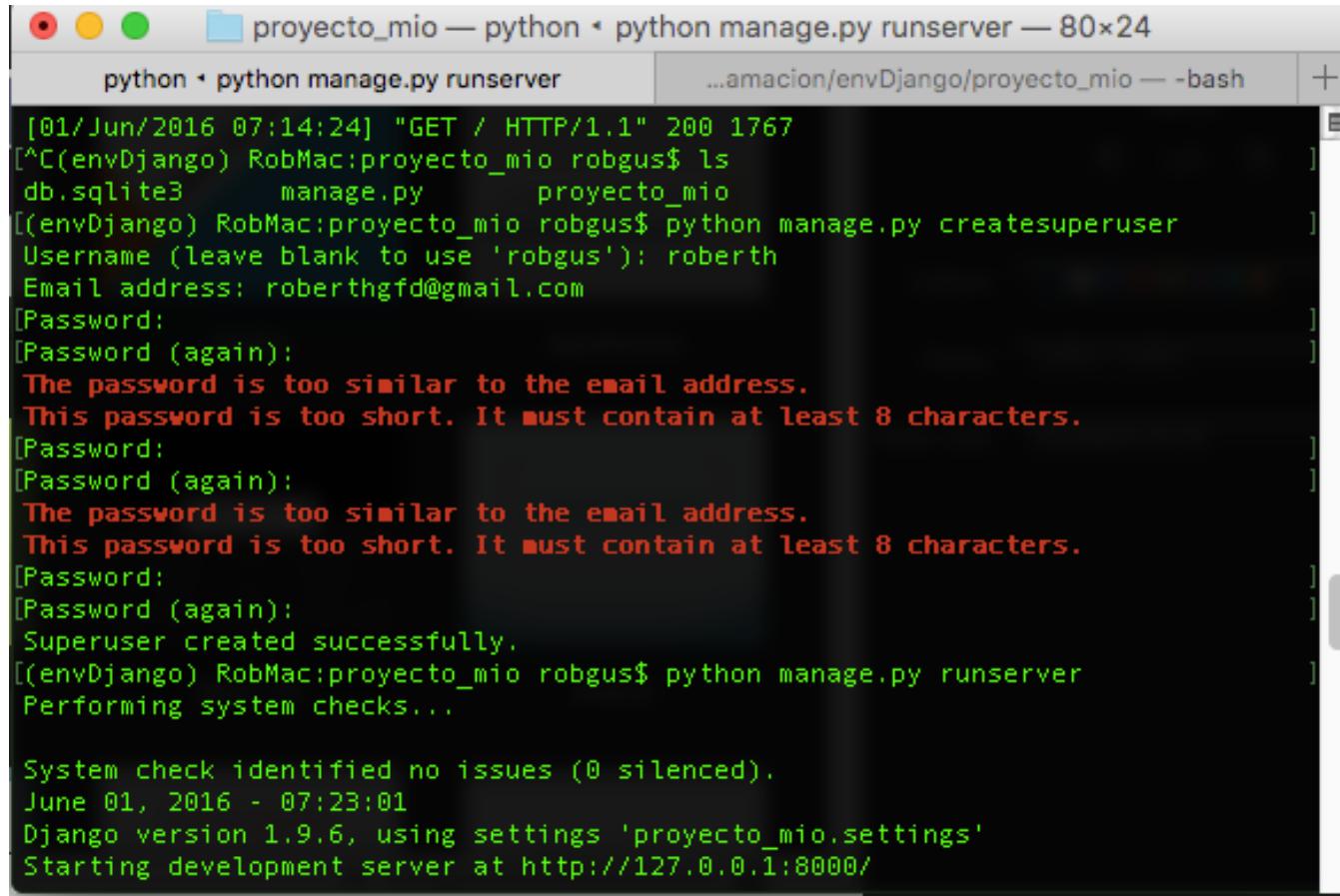
Instalación de Django 3.2.3



```
macrob:Developer roberth$ ls
PythonLearn  envBorrar  tryDjango
macrob:Developer roberth$ rm -rf envBorrar/
macrob:Developer roberth$ ls
PythonLearn  tryDjango
macrob:Developer roberth$ cd tryDjango/
macrob:tryDjango roberth$ ls
envDjango
macrob:tryDjango roberth$ pip install Django==3.2.3
Collecting Django==3.2.3
  Downloading Django-3.2.3-py3-none-any.whl (7.9 MB)
    |████████████████████████████████| 7.9 MB 3.2 MB/s
Collecting asgiref<4,>=3.3.2
  Downloading asgiref-3.3.4-py3-none-any.whl (22 kB)
Collecting sqlparse>=0.2.2
  Downloading sqlparse-0.4.1-py3-none-any.whl (42 kB)
Collecting pytz
  Downloading pytz-2021.1-py2.py3-none-any.whl (510 kB)
  Installing collected packages: sqlparse, pytz, asgiref, Django
    Successfully installed Django-3.2.3 asgiref-3.3.4 pytz-2021.1 sqlparse-0.4.1
macrob:tryDjango roberth$
```

Crear el administrador Django

- Crear el usuario
 - Usar comando → **python manage.py createsuperuser**



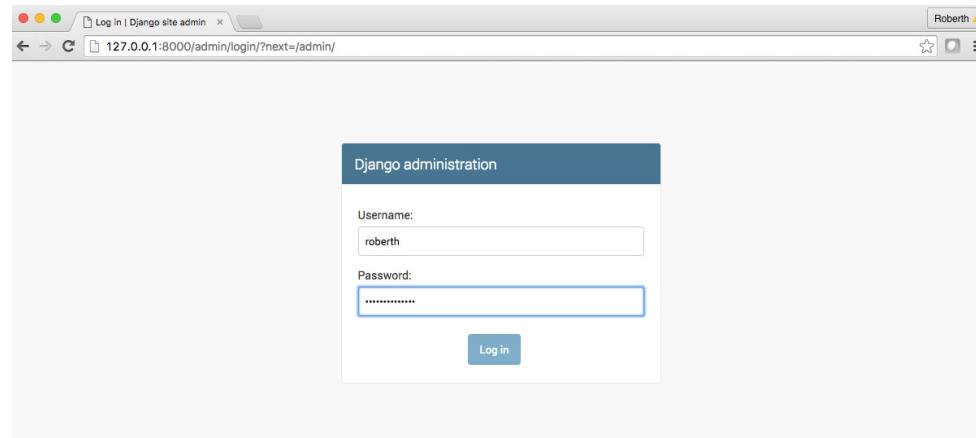
A terminal window titled "python manage.py runserver" is shown. The user runs "python manage.py createsuperuser". They enter a username ("roberth") and email address ("roberthgfd@gmail.com"). When prompted for a password, they enter a password that is too similar to the email address and too short. The terminal then successfully creates the superuser and starts a development server at "http://127.0.0.1:8000".

```
[01/Jun/2016 07:14:24] "GET / HTTP/1.1" 200 1767
[envDjango] RobMac: proyecto_mio robgus$ ls
db.sqlite3      manage.py      proyecto_mio
[envDjango] RobMac: proyecto_mio robgus$ python manage.py createsuperuser
Username (leave blank to use 'robgus'): roberth
Email address: roberthgfd@gmail.com
>Password:
>Password (again):
The password is too similar to the email address.
This password is too short. It must contain at least 8 characters.
>Password:
>Password (again):
The password is too similar to the email address.
This password is too short. It must contain at least 8 characters.
>Password:
>Password (again):
Superuser created successfully.
[envDjango] RobMac: proyecto_mio robgus$ python manage.py runserver
Performing system checks...

System check identified no issues (0 silenced).
June 01, 2016 - 07:23:01
Django version 1.9.6, using settings 'proyecto_mio.settings'
Starting development server at http://127.0.0.1:8000/
```

Crear el administrador Django

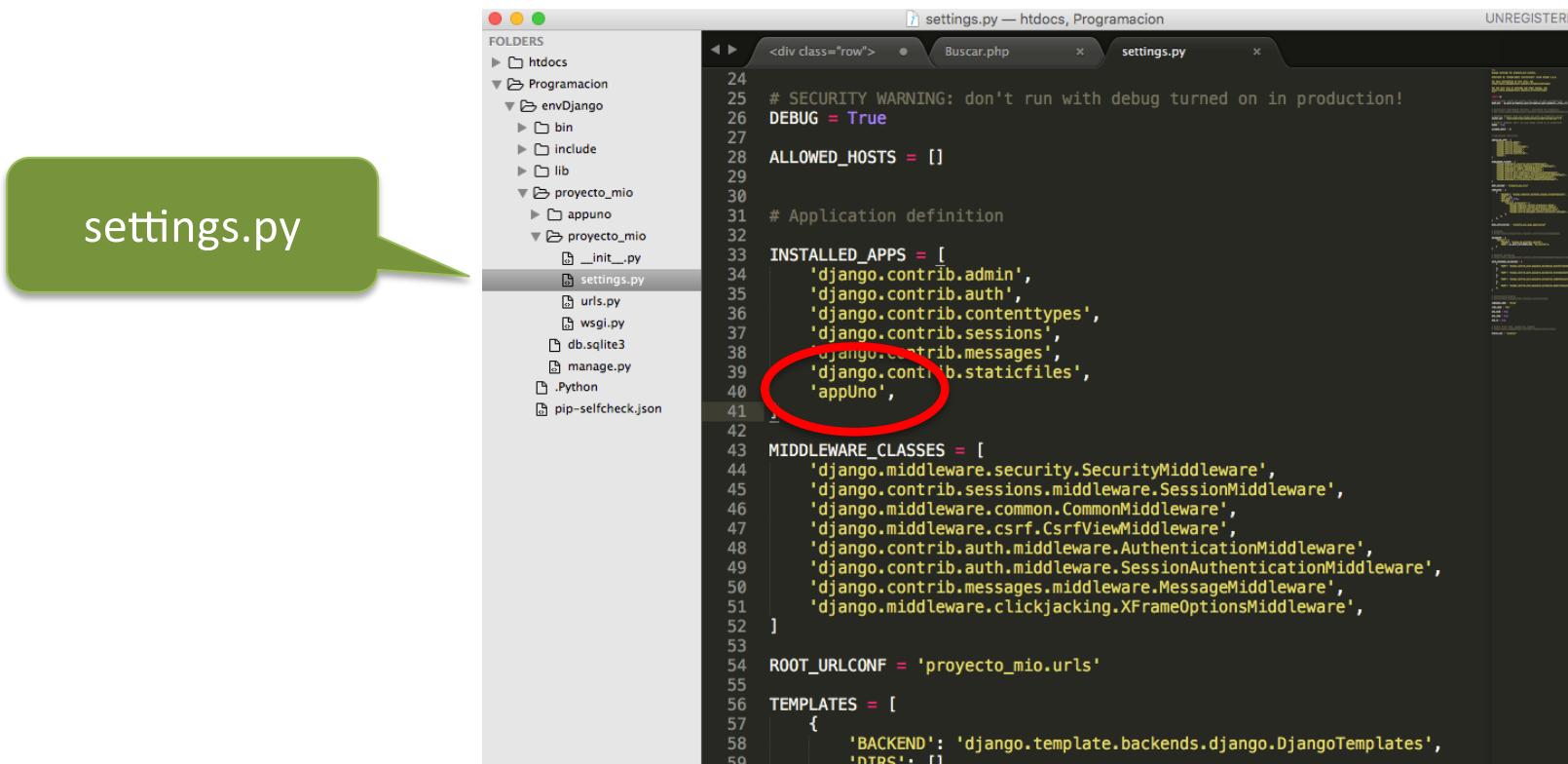
- Se ejecuta el servidor de aplicaciones :
 - Comando → **python manage.py runserver**
- Ir al navegador a:
 - **http://127.0.0.1:8000/admin/**



A screenshot of the Django admin dashboard. The URL in the address bar is "127.0.0.1:8000/admin/". The top navigation bar includes links for "WELCOME, ROBERTH. VIEW SITE / CHANGE PASSWORD / LOG OUT". The main content area is titled "Site administration" and shows two sections: "AUTHENTICATION AND AUTHORIZATION" (Groups and Users) and "Recent Actions" (My Actions, None available). A green speech bubble with the text "Usuario Admin Creado" points to the "WELCOME" message in the top right corner of the dashboard.

Creación de aplicaciones

- Crear el usuario
 - Usar comando → **python manage.py startapp appUno**
 - Agregar la appUno al settings.py → **sudo vim settings.py**



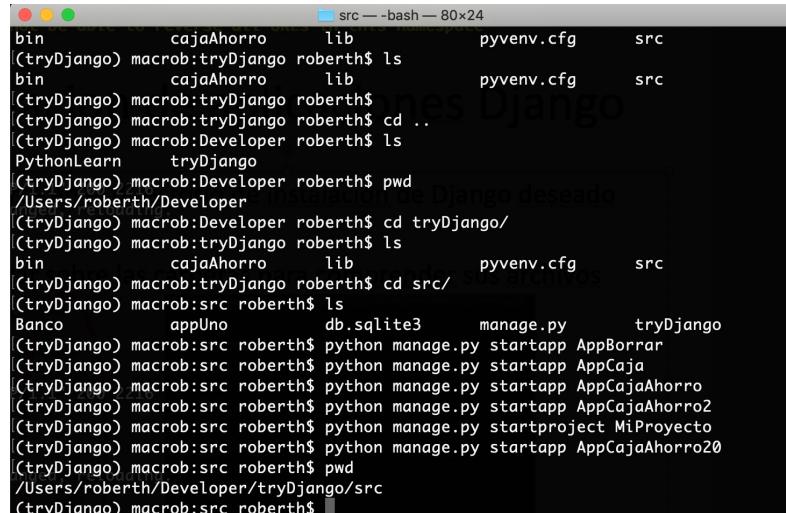
CREACIÓN DE MÁS APLICACIONES EN DJANGO

Creación de Aplicaciones Django

- Ubicarse en el directorio de instalación de Django

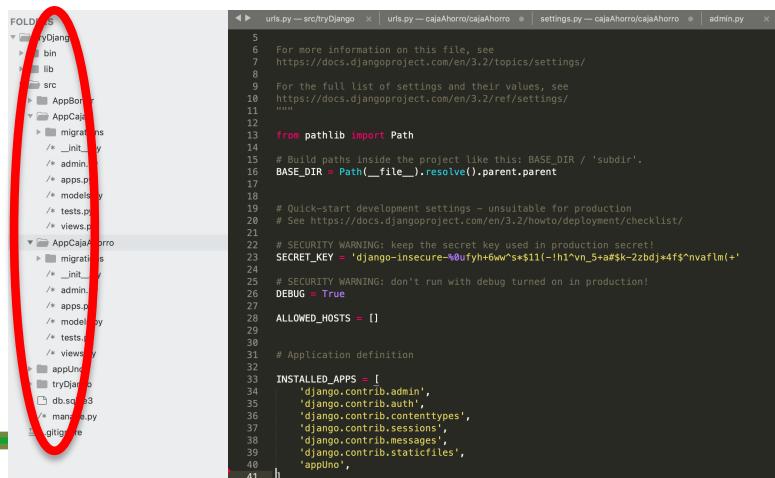
En nuestro caso:

→ /Users/roberth/Developer/tryDjango/src



```
src -- bash -- 80x24
bin      cajaAhorro    lib      pyvenv.cfg   src
(tryDjango) macrob:tryDjango roberth$ ls
bin      cajaAhorro    lib      pyvenv.cfg   src
(tryDjango) macrob:tryDjango roberth$ cd ..
(tryDjango) macrob:Developer roberth$ ls
PythonLearn tryDjango
(tryDjango) macrob:Developer roberth$ pwd
/Users/roberth/Developer
(tryDjango) macrob:Developer roberth$ cd tryDjango/
(tryDjango) macrob:tryDjango roberth$ ls
bin      cajaAhorro    lib      pyvenv.cfg   src
(tryDjango) macrob:tryDjango roberth$ cd src/
(tryDjango) macrob:src roberth$ ls
Banco    appUno        db.sqlite3  manage.py    tryDjango
(tryDjango) macrob:src roberth$ python manage.py startapp AppBorrar
(tryDjango) macrob:src roberth$ python manage.py startapp AppCaja
(tryDjango) macrob:src roberth$ python manage.py startapp AppCajaAhorro
(tryDjango) macrob:src roberth$ python manage.py startapp AppCajaAhorro2
(tryDjango) macrob:src roberth$ python manage.py startproject MiProyecto
(tryDjango) macrob:src roberth$ python manage.py startapp AppCajaAhorro20
(tryDjango) macrob:src roberth$ pwd
/Users/roberth/Developer/tryDjango/src
(tryDjango) macrob:src roberth$
```

- Identificar carpetas y archivos principales de cada aplicación



The screenshot shows a file browser with a red circle highlighting the 'src' folder under 'tryDjango'. To the right, a code editor displays the 'settings.py' file from the 'cajaAhorro' application. The code includes standard Django settings like 'ALLOWED_HOSTS', 'INSTALLED_APPS', and a complex 'SECRET_KEY' value.

```
5
6  For more information on this file, see
7  https://docs.djangoproject.com/en/3.2/topics/settings/
8
9  For the full list of settings and their values, see
10 https://docs.djangoproject.com/en/3.2/ref/settings/
11 """
12
13 from pathlib import Path
14
15 # Build paths inside the project like this: BASE_DIR / 'subdir'.
16 BASE_DIR = Path(__file__).resolve().parent.parent
17
18
19 # Quick-start development settings - unsuitable for production
20 # See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/
21
22 # SECURITY WARNING: keep the secret key used in production secret!
23 SECRET_KEY = 'django-insecure-#0ufyh+6w^s+$11(-lh1^vn_5+@#sk-2bdj*4f$^nvaflm(+'
24
25 # SECURITY WARNING: don't run with debug turned on in production!
26 DEBUG = True
27
28 ALLOWED_HOSTS = []
29
30
31 # Application definition
32
33 INSTALLED_APPS = [
34     'django.contrib.admin',
35     'django.contrib.auth',
36     'django.contrib.contenttypes',
37     'django.contrib.sessions',
38     'django.contrib.messages',
39     'django.contrib.staticfiles',
40     'appUno',
41 ]
```

Creación de aplicación con startapp

- Crear la aplicación CajaAhorro
 - Usar comando → `python manage.py startapp appCajaAhorro`
 - Agregar la `appCajaAhorro` al `settings.py` → `sudo vim settings.py`

settings.py

```
31
32 # Application definition
33
34 INSTALLED_APPS = [
35     'django.contrib.admin',
36     'django.contrib.auth',
37     'django.contrib.contenttypes',
38     'django.contrib.sessions',
39     'django.contrib.messages',
40     'django.contrib.staticfiles',
41     'appUno',
42     'CajaAhorro',
43 ]
44
45 MIDDLEWARE = [
46     'django.middleware.security.SecurityMiddleware',
47     'django.contrib.sessions.middleware.SessionMiddleware',
48     'django.middleware.common.CommonMiddleware',
49     'django.middleware.csrf.CsrfViewMiddleware',
50     'django.contrib.auth.middleware.AuthenticationMiddleware',
51     'django.contrib.messages.middleware.MessageMiddleware',
52     'django.middleware.clickjacking.XFrameOptionsMiddleware'
53 ]
54
55 ROOT_URLCONF = 'DemoDjango.urls'
56
57 TEMPLATES = [
58     {
59         'BACKEND': 'django.template.backends.django.DjangoTemplates',
60         'DIRS': [BASE_DIR, "templates"],
61         'APP_DIRS': True,
62         'OPTIONS': {
63             'context_processors': [
64                 'django.template.context_processors.debug',
65                 'django.template.context_processors.request',
66                 'django.contrib.auth.context_processors.auth',
67                 'django.contrib.messages.context_processors.messages',
68             ],
69             'libraries': {
70                 'staticfiles': 'staticfiles_tags',
71             }
72         }
73     }
74 ]
```

Comando makemigrations

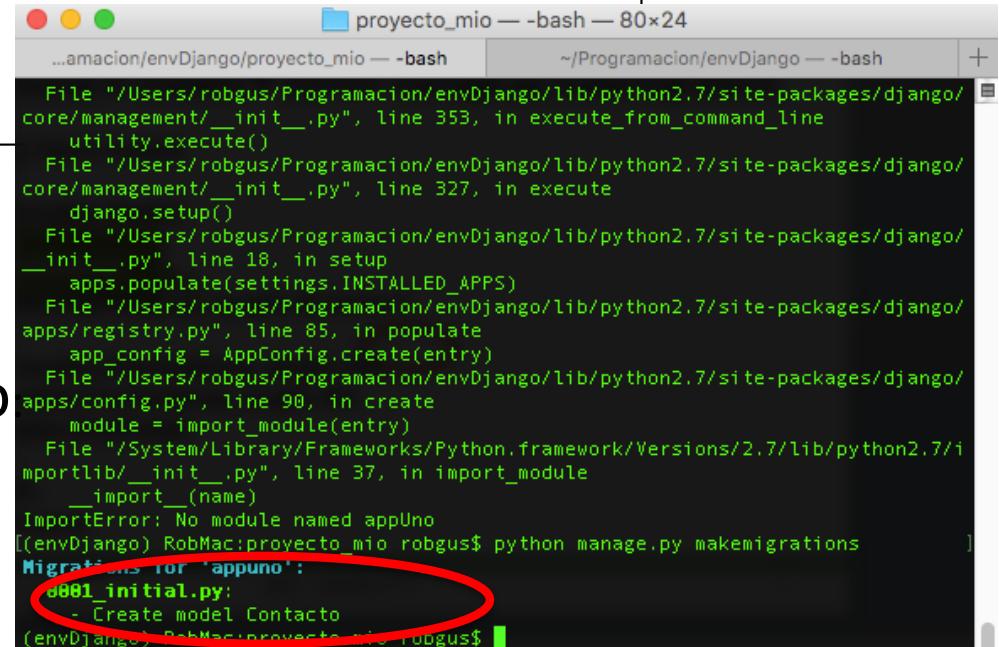
- Codificar el **models.py**, con la clase Contacto:

```
from django.db import models

class Contacto(models.Model):

    nombres=models.CharField(max_length=30,blank=True)
    apellidos=models.CharField(max_length=30)
    cedula=models.CharField(max_length=30)
    email=models.EmailField()

def __str__(self):
    return self.email
```

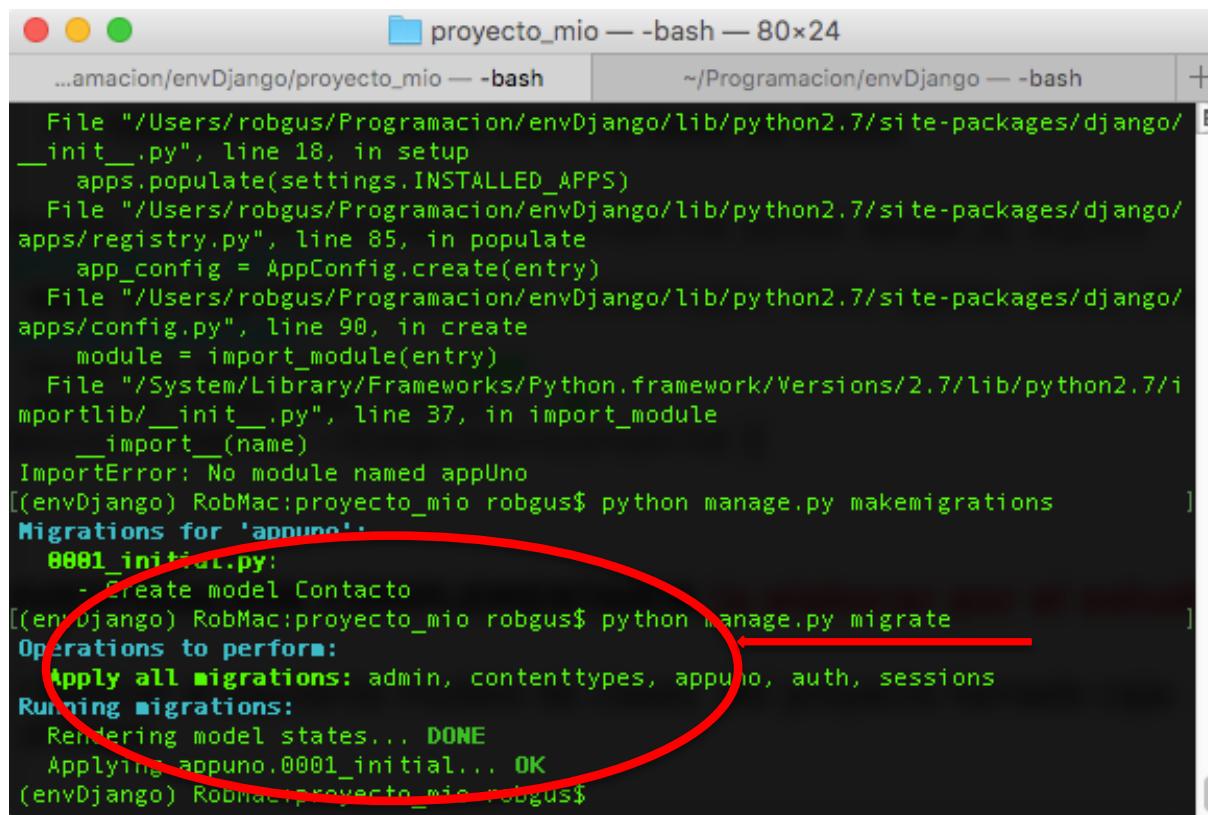


```
...amacion/envDjango/proyecto_mio -- bash ~/Programacion/envDjango -- bash +_
File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/
core/management/_init_.py", line 353, in execute_from_command_line
    utility.execute()
    File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/
core/management/_init_.py", line 327, in execute
    django.setup()
    File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/
__init__.py", line 18, in setup
    apps.populate(settings.INSTALLED_APPS)
    File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/
apps/registry.py", line 85, in populate
    app_config = AppConfig.create(entry)
    File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/
apps/config.py", line 90, in create
    module = import_module(entry)
    File "/System/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/i
mportlib/_init_.py", line 37, in import_module
    __import__(name)
ImportError: No module named appUno
[envDjango] RobMac:proyecto_mio robgus$ python manage.py makemigrations
Migrations for 'appuno':
  0001_initial.py:
    - Create model Contacto
[envDjango] RobMac:proyecto_mio robgus$
```

- Aplicar el modelo al proyecto
usar → **python manage.py makemigrations**

Comando migrate

- Mapear hacia la base de datos el modelo
 - usar → **python manage.py migrate**



```
...amacion/envDjango/proyecto_mio -- bash -- 80x24
...amacion/envDjango/proyecto_mio -- bash      ~/Programacion/envDjango -- bash + 
File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/__init__.py", line 18, in setup
    apps.populate(settings.INSTALLED_APPS)
  File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/apps/registry.py", line 85, in populate
    app_config = AppConfig.create(entry)
  File "/Users/robgus/Programacion/envDjango/lib/python2.7/site-packages/django/apps/config.py", line 90, in create
    module = import_module(entry)
  File "/System/Library/Frameworks/Python.framework/Versions/2.7/lib/python2.7/importlib/__init__.py", line 37, in import_module
    __import__(name)
ImportError: No module named appUno
[(envDjango) RobMac:proyecto_mio robgus$ python manage.py makemigrations
  Migrations for 'appUno':
    0001_initial.py:
      - Create model Contacto
[(envDjango) RobMac:proyecto_mio robgus$ python manage.py migrate
  Operations to perform:
    Apply all migrations: admin, contenttypes, appUno, auth, sessions
  Running migrations:
    Rendering model states... DONE
      Applying appUno.0001_initial... OK
[(envDjango) RobMac:proyecto_mio robgus$]
```

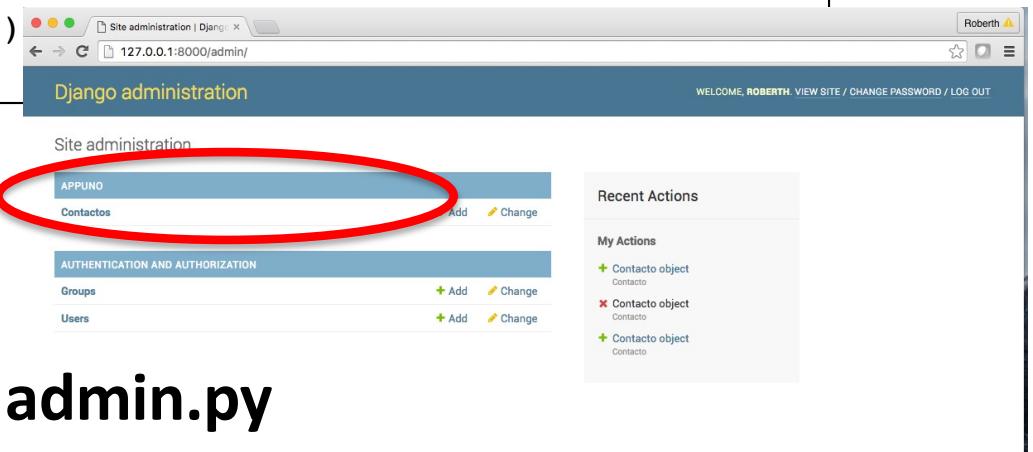
Anclar el modelo al admin.py

- Realizar cambios al **admin.py**

```
from django.contrib import admin
from .models import Contacto

class AdminContacto(admin.ModelAdmin):
    list_display = ["__str__","nombres","apellidos","email"]
    class Meta(object):
        model = Contacto

admin.site.register(Contacto,AdminContacto)
```



- Aplicar el anclaje al **admin.py**

usar → **python manage.py migrate**

Levantar el servidor de aplicaciones → **python manage.py runserver**

Resultado luego anclaje

- Pantalla modelo creado en código python.

The image consists of three screenshots of the Django admin interface:

- Screenshot 1: Site administration dashboard**
Shows the main admin dashboard with sections for APPUNO (Contactos, Groups, Users) and AUTHENTICATION AND AUTHORIZATION (Groups, Users). A red arrow points from this screen to the top of the second screenshot.
- Screenshot 2: Adding a new Contacto**
Shows the "Add contacto" form. The user has entered: Nombres: Roberto, Apellidos: Figueroa, Cedula: 1105034834, and Email: roberto.figueroa@uni.edu.ec. A red arrow points from the top of this screen to the top of the third screenshot.
- Screenshot 3: Listing the Contacto object**
Shows a list titled "Select contacto to change". It contains one item: "Contacto object" (Roberto, Figueroa, roberto.figueroa@uni.edu.ec). A red arrow points from the top of this screen to the bottom of the green callout box.

el `__str__` hará de índice en la tabla y el objeto

Uso de Views

Modificar View

- Realizar cambios al **view.py**

```
#from django.http import HttpResponseRedirect  
from django.shortcuts import render
```

```
# Create your views here.
```

```
def home_view(request, *args, **kwargs):  
    print (args, kwargs)  
    print (request)  
    print (request.user)  
    return render(request, "home.html", {})  
    #return HttpResponseRedirect("<h1>Hola Bienvenido al Sistema appUno</h1>")
```

```
def about_view(request):  
    return render(request, "about.html", {})
```

```
def contact_view(request):  
    return render(request, "contact.html", {})
```

Habilitar para enviar HTML directamente en el return

```
1  #from django.http import HttpResponseRedirect  
2  from django.shortcuts import render  
3  
4  # Create your views here.  
5  
6  def home_view(request, *args, **kwargs):  
7      print (args, kwargs)  
8      print (request)  
9      print (request.user)  
10     return render(request, "home.html", {})  
11     #return HttpResponseRedirect("<h1>Hola Bienvenido al Sistema appUno</h1>")  
12  
13  def about_view(request):  
14      return render(request, "about.html", {})  
15  
16  def contact_view(request):  
17      return render(request, "contact.html", {})  
18  
19  def demo_view(request):  
20      return render(request, "demo.html", {})
```

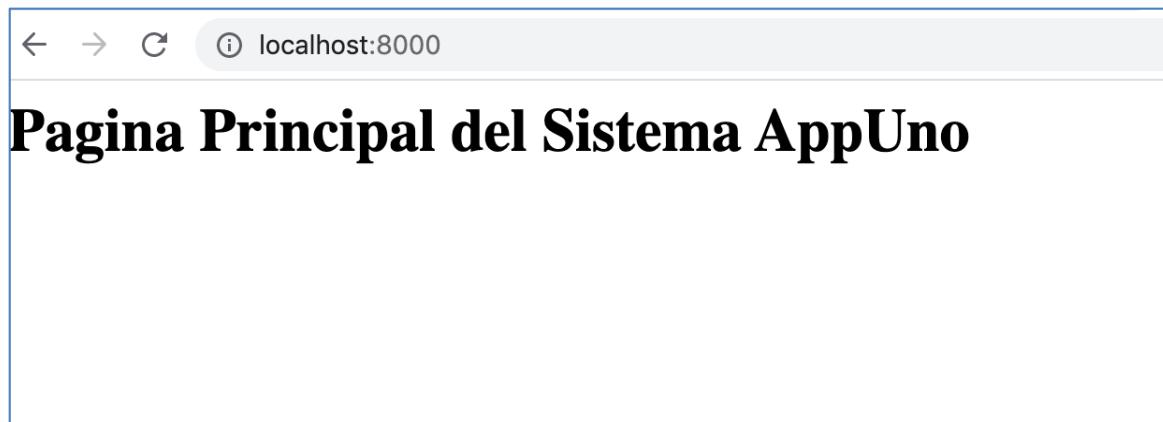
Agregar un elemento nuevo a la lista urlpatterns en urls.py

- Agregar la url deseada archivo **urls.py**
`path('about/', about_view),`
- **about_view** : es la vista donde se ubica la función en **views.py**

```
from django.contrib import admin
from django.urls import path
from appUno.views import home_view, contact_view, about_view

urlpatterns = [
    path('', home_view, name='home'),
    path('about/', about_view),
    path('contact/', contact_view),
    path('admin/', admin.site.urls),
]
```

Acceder a las nuevas url's



Uso de Templates

Configurar los templates en settings.py

- Directorio de templates:

modificar a la ubicación → 'DIRS': [],

- Quedando así: '**DIRS**'

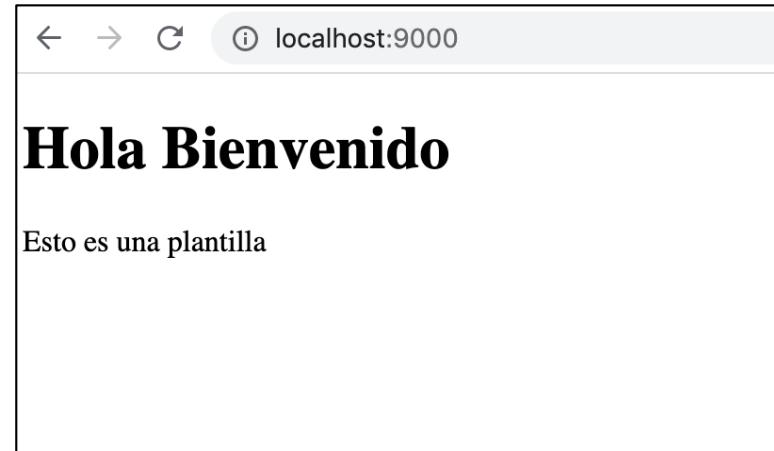
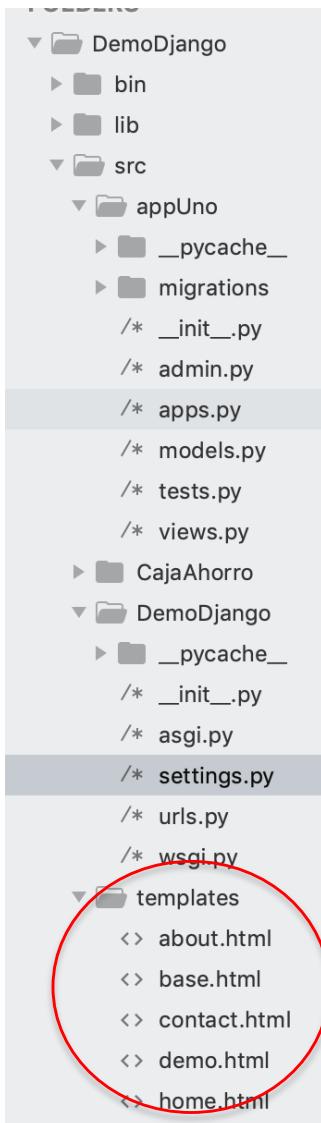
```
56
57     TEMPLATES = [
58         {
59             'BACKEND': 'django.template.backends.django.DjangoTemplates',
60             'DIRS': [BASE_DIR, "templates"],
61             'APP_DIRS': True,
62             'OPTIONS': {
63                 'context_processors': [
64                     'django.template.context_processors.debug',
65                     'django.template.context_processors.request',
66                     'django.contrib.auth.context_processors.auth',
67                     'django.contrib.messages.context_processors.messages',
68                 ],
69             },
70         },
71     ],
72 ]
```

```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS
```

Modifique las **views** para cada **templates**

```
1  from django.http import HttpResponse
2  from django.shortcuts import render
3
4
5  def home_view(request):
6
7      return HttpResponse("<h1>Hola Bienvenido Señor@s Estudiantes 5A</h1>")
8
9  def about_view(request):
10     return render(request, "about.html", {})
11
12 def contact_view(request):
13     return render(request, "contact.html", {})
14
15 def demo_view(request):
16     return render(request, "demo.html", {})
17
18
19
```

Jerarquía de carpetas para templates



```
11 |
12 |     from pathlib import Path
13 |
14 |
15 |     # Build paths inside the project like this: BASE_DIR / 'subdir'.
16 |     BASE_DIR = Path(__file__).resolve().parent.parent
17 |     #/Users/roberth/Developer/DemoDjango/src
18 |     print(BASE_DIR)
19 |
20 |     # Quick-start development settings - unsuitable for production
21 |     # See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/
22 |
23 |     # SECURITY WARNING: keep the secret key used in production secret!
24 |     SECRET_KEY = 'django-insecure-!w*p%em(4!*3q6zefyue0ke50&z3583ncweem(v7)3v1&^rgat*'
25 |
26 |     # SECURITY WARNING: don't run with debug turned on in production!
27 |     DEBUG = True
28 |
29 |     ALLOWED_HOSTS = []
```

Hay que recordar que la ruta de su proyecto de Django la puede ver mediante la variable **BASE_DIR**, que se ubica en el archivo **settings.py**

Creamos el archivo html contact.html

Se guarda el html en el directorio de **templates**:

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="utf-8">
5      <title>Conctacto pagina</title>
6  </head>
7  <body>
8  <h1>Pagina de contacto con plantilla DBP</h1>
9  </body>
10 </html>
```

Se crean más archivos html en el directorio de **templates**, según la necesidad

Agregar un elemento nuevo a la lista urlpatterns en url.py de ser necesario

```
"""DemoDjango URL Configuration

The `urlpatterns` list routes URLs to views. For more information please see:
    https://docs.djangoproject.com/en/3.2/topics/http/urls/
Examples:
Function views
    1. Add an import: from my_app import views
    2. Add a URL to urlpatterns: path('', views.home, name='home')
Class-based views
    1. Add an import: from other_app.views import Home
    2. Add a URL to urlpatterns: path('', Home.as_view(), name='home')
Including another URLconf
    1. Import the include() function: from django.urls import include, path
    2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
"""

from django.contrib import admin
from django.urls import path
from appUno.views import home_view, contact_view, about_view, demo_view

urlpatterns = [
    path('', home_view, name='home'),
    path('about/', about_view),
    path('demo/', demo_view),
    path('contact/', contact_view),
    path('admin/', admin.site.urls),
]

```

importamos las vistas

```
1 """DemoDjango URL Configuration
2
3 The `urlpatterns` list routes URLs to views. For more information please see:
4     https://docs.djangoproject.com/en/3.2/topics/http/urls/
5 Examples:
6 Function views
7     1. Add an import: from my_app import views
8         path('', views.home, name='home')
9 Class-based views
10    1. Add an import: from other_app.views import Home
11        path('', Home.as_view(), name='home')
12 Including another URLconf
13    1. Import the include() function: from django.urls import include, path
14        path('blog/', include('blog.urls'))
15
16 from django.contrib import admin
17 from django.urls import path
18 from appUno.views import home_view, contact_view, about_view, demo_view
19
20 urlpatterns = [
21     path('', home_view, name='home'),
22     path('about/', about_view),
23     path('demo/', demo_view),
24     path('contact/', contact_view),
25     path('admin/', admin.site.urls),
26 ]
27
```

Ver el resultado de uso de templates

The image displays three separate browser windows illustrating the use of Django templates:

- Top Window:** Shows a page titled "Hola Bienvenido a DemoDjango". It contains a message "Esta es una plantilla", a video player interface with controls (play, volume, etc.), and a table with columns "Nombre", "Producto", "Precio", and "Cantidad". The table data includes "Laptop HP" at 1000 and "Impresora HP" at 200.
- Middle Window:** Shows a page titled "About HTML con uso de plantilla".
- Bottom Window:** Shows a page titled "Pagina de Contacto para el Sistema AppUno".

Agregando código

Django versión 3.2.3

Agregar código python en templates

```
{% extends 'base.html' %}

{% block content %}

<h1>Hola</h1>
<p>Esto es una plantilla </p>

{% endblock %}

{% block content %}

{% endblock %}
```

Herencia de plantilla

Código para Verificar el usuario activo:

```
{{request.user.is_authenticated}}
```

Modificando contact.html

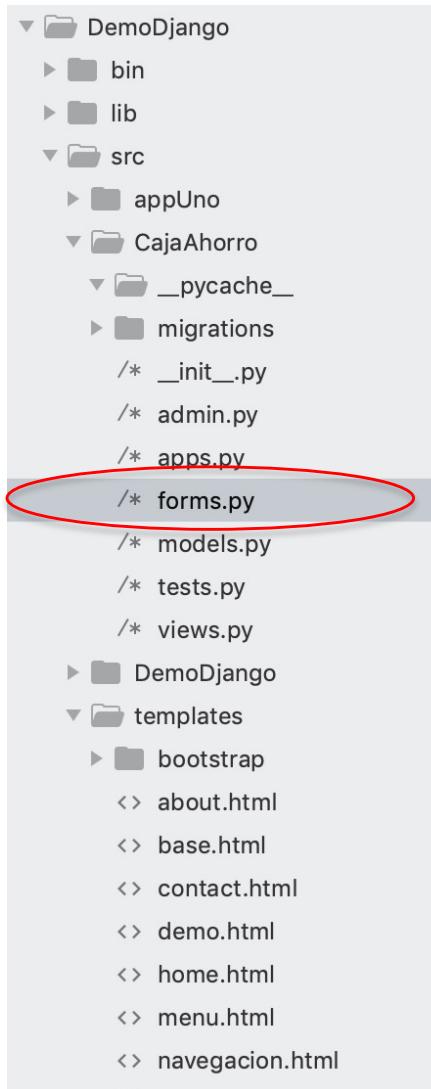
```
{% include "base.html" %}  
<!DOCTYPE html>  
<html>  
<head>  
    <meta charset="utf-8">  
    <title>Conctacto pagina</title>  
</head>  
<body>  
    <h1>Pagina de contacto con plantilla DBP</h1>  
  
    {{request.user.is_authenticated}}  
  
</body>  
</html>
```

Se importa base.html

Agregando formularios

Django versión 3.2.3

Creamos el archivo **forms.py** dentro de la app



Definir el formulario forms.py

```
from django import forms
from .models import Contacto

class Formulario(forms.ModelForm):
    class Meta:
        model = Contacto
        fields=["nombres", "apellidos", "cedula", "email"]
```

```
1 from django import forms
2 from .models import Contacto
3
4 class Formulario(forms.ModelForm):
5     class Meta:
6         model = Contacto
7         fields=["nombres", "apellidos", "cedula", "email"]|
```

Modificamos el archivo view.py

```
from django.shortcuts import render
from django.shortcuts import redirect
from .forms import Formulario
from .models import Contacto

def home_view(request):
    f = Formulario(request.POST or None)
    if request.method == 'POST':
        if f.is_valid():
            datos = f.cleaned_data
            c = Contacto()
            c.nombres = datos.get("nombres")
            c.apellidos = datos.get("apellidos")
            c.cedula = datos.get("cedula")
            c.email = datos.get("email")
            if c.save() != True:
                print('Imprimo en pantalla y guardo data en BD')
                print(f.cleaned_data)
                return redirect(home_view)
    context = {
        "form":f,
    }
    return render(request,"home.html",context)

def contact_view(request):
    return render(request,"contact.html",{})

def about_view(request):
    return render(request,"about.html",{})
```

f es un objeto de la Petición POST

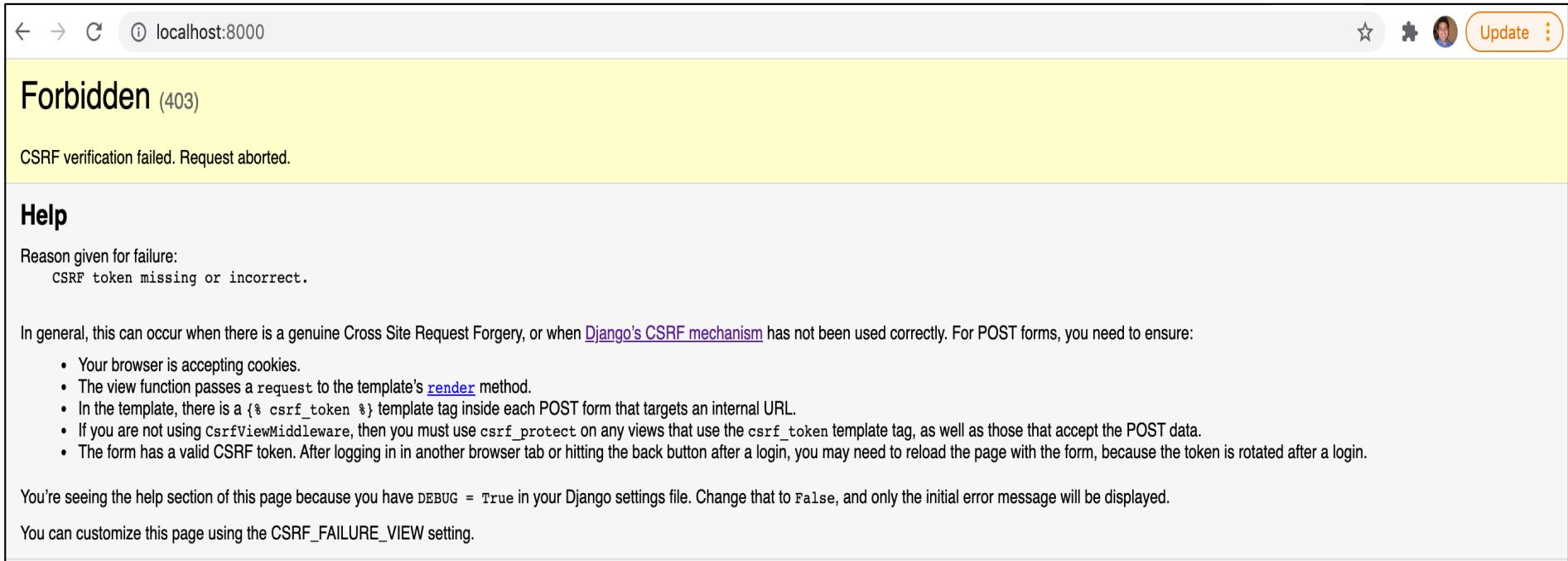
Validar los campos de entradas:

- Agregamos los requisitos de validación de POST
 - `request.POST`
- Agregamos `None` para que no se visualice los mensajes de
 - `request.POST or None`

El view.py modificado

```
1  from django.shortcuts import render
2  from django.shortcuts import redirect
3  from .forms import Formulario
4  from .models import Contacto
5
6  def home_view(request):
7      f = Formulario(request.POST or None)
8      if request.method == 'POST':
9          if f.is_valid():
10              datos = f.cleaned_data
11              c = Contacto()
12              c.nombres = datos.get("nombres")
13              c.apellidos = datos.get("apellidos")
14              c.cedula = datos.get("cedula")
15              c.email = datos.get("email")
16              if c.save() != True:
17                  print('Imprimo en pantalla y guardo data en BD')
18                  print(f.cleaned_data)
19                  return redirect(home_view)
20      context = {
21          "form":f,
22      }
23      return render(request,"home.html",context)
24
25  def contact_view(request):
26      return render(request,"contact.html",{})
27
28  def about_view(request):
29      return render(request,"about.html",{})
30
31  def demo_view(request):
32      return render(request,"demo.html",{})
```

Vulnerabilidad Cross Site Request Forgery **CSRF** en peticiones POST



The screenshot shows a web browser window with the URL `localhost:8000` in the address bar. The page title is "Forbidden (403)". The main content area displays the message "CSRF verification failed. Request aborted." Below this, under the heading "Help", it says "Reason given for failure: CSRF token missing or incorrect." It then provides general advice: "In general, this can occur when there is a genuine Cross Site Request Forgery, or when [Django's CSRF mechanism](#) has not been used correctly. For POST forms, you need to ensure:" followed by a bulleted list of six items. At the bottom, it states: "You're seeing the help section of this page because you have `DEBUG = True` in your Django settings file. Change that to `False`, and only the initial error message will be displayed." Finally, it says: "You can customize this page using the `CSRF_FAILURE_VIEW` setting."

← → ⌂ ⓘ localhost:8000 ☆ 🎫 📸 Update :

Forbidden (403)

CSRF verification failed. Request aborted.

Help

Reason given for failure:
CSRF token missing or incorrect.

In general, this can occur when there is a genuine Cross Site Request Forgery, or when [Django's CSRF mechanism](#) has not been used correctly. For POST forms, you need to ensure:

- Your browser is accepting cookies.
- The view function passes a `request` to the template's `render` method.
- In the template, there is a `{% csrf_token %}` template tag inside each POST form that targets an internal URL.
- If you are not using `CsrfViewMiddleware`, then you must use `csrf_protect` on any views that use the `csrf_token` template tag, as well as those that accept the POST data.
- The form has a valid CSRF token. After logging in in another browser tab or hitting the back button after a login, you may need to reload the page with the form, because the token is rotated after a login.

You're seeing the help section of this page because you have `DEBUG = True` in your Django settings file. Change that to `False`, and only the initial error message will be displayed.

You can customize this page using the `CSRF_FAILURE_VIEW` setting.

Modificamos la plantilla home.html

- `{{form}}` es la variable context enviada por view a la plantilla.
- `form.as_p` → muestra el formulario en párrafo en html usando la etiqueta `<p>` por defecto.

```
<h1>Pagina de Inicio</h1>
<form method="post" action=""> {% csrf_token %}
    {{form.as_p}}
    <input type="submit" value="Aceptar">
</form>
<h1>Despues del formulario</h1>
```

- Agregamos una seguridad web en nuestro formulario `{% csrf_token %}`

Nota: Un ataque CSRF obliga al navegador de una víctima autenticada a enviar una petición HTTP falsificado, incluyendo la sesión del usuario y cualquier otra información de autenticación incluida automáticamente, a una aplicación web vulnerable. Esto permite al atacante forzar al navegador de la víctima para generar pedidos que la aplicación vulnerable piensa son peticiones legítimas provenientes de la víctima.

El archivo home.html

```
1 <h1>Pagina de Inicio</h1>
2 <form method="post" action=""> {% csrf_token %}
3     {{form.as_p}}
4     <input type="submit" value="Aceptar">
5 </form>
6 <h1>Despues del formulario</h1>
```

Reiniciar el Servidor Web wsgi

python manage.py runserver

The screenshot shows a simple contact form with four input fields: 'Nombres' (Roberth), 'Apellidos' (Figueroa), 'Cedula' (1100434321), and 'Email' (roberth.figueroa@unl.edu). Below the form is a button labeled 'Aceptar'. The title of the page is 'Pagina de Inicio'.

Despues del formulario

The screenshot shows the Django Admin 'Contactos' list view. It displays a table with columns: CONTACTO, NOMBRES, APELLIDOS, CEDULA, and EMAIL. There are 22 selected contacts listed, including Roberth Figueroa D, Juan Pablo Martinez, and others. The left sidebar shows the 'APPUNTO' section with 'Contactos' selected.

CONTACTO	NOMBRES	APELLIDOS	CEDULA	EMAIL
Contacto object (22)	fsdfasf			
Contacto object (21)	Maria Cristina	Cris	227332	cris@cris.com
Contacto object (20)	Juanita	Julia	121312	asfas@dfdfa.com
Contacto object (19)	Juan	Martinez	748	juan@matinez.com
Contacto object (18)	Data	M	456789	data@d.com
Contacto object (17)	Roberth	Alberto	Husd	ads@adfas.ec
Contacto object (16)	Contacto	M	1232	mig@email.com
Contacto object (15)	Roberth	Figueroa D	121	ana@com
Contacto object (14)	Juan	Pablo	11	juan@gmail.com
Contacto object (13)	Demostracion	Demo	112567893	demo@demo.ec
Contacto object (12)	Rafael	Correa	43489	correa@correa.com
Contacto object (11)	Guillermo	Laso	112412	guillermo@laso.com
Contacto object (10)	inicio2	inicial2	122332	inicial@inicial.com

Verificar que los datos se guarden en la base de datos de la aplicación

Redireccionar una App específica modificando las urls

Django versión 3.2.3

Accediendo desde otro Host externo o computador a mi aplicación en Django

Django versión 3.2.3

Modificar el settings.py

```
8  For the full list of settings and their values, see
9  https://docs.djangoproject.com/en/3.2/ref/settings/
10 """
11
12     from pathlib import Path
13
14     # Build paths inside the project like this: BASE_DIR / 'subdir'.
15     BASE_DIR = Path(__file__).resolve().parent.parent
16     #/Users/roberth/Developer/DemoDjango/src
17     print(BASE_DIR)
18
19     # Quick-start development settings - unsuitable for production
20     # See https://docs.djangoproject.com/en/3.2/howto/deployment/checklist/
21
22     # SECURITY WARNING: keep the secret key used in production secret!
23     SECRET_KEY = 'django-insecure-!w*p%m(4!*3q6zefyue0k@50&z3583ncweem(v7)3v1&^rgat*'
24
25     # SECURITY WARNING: don't run with debug turned on in production!
26     DEBUG = True
27
28     #Para permitir acceder desde host externos '*'
29     ALLOWED_HOSTS = ['*']
30
31
32     # Application definition
33
34     INSTALLED_APPS = [
35         'django.contrib.admin',
36         'django.contrib.auth',
37         'django.contrib.contenttypes',
38         'django.contrib.sessions',
39         'django.contrib.messages',
40         'django.contrib.staticfiles',
41         'appUno',
42         'CajaAhorro',
43     ]
44
45
```

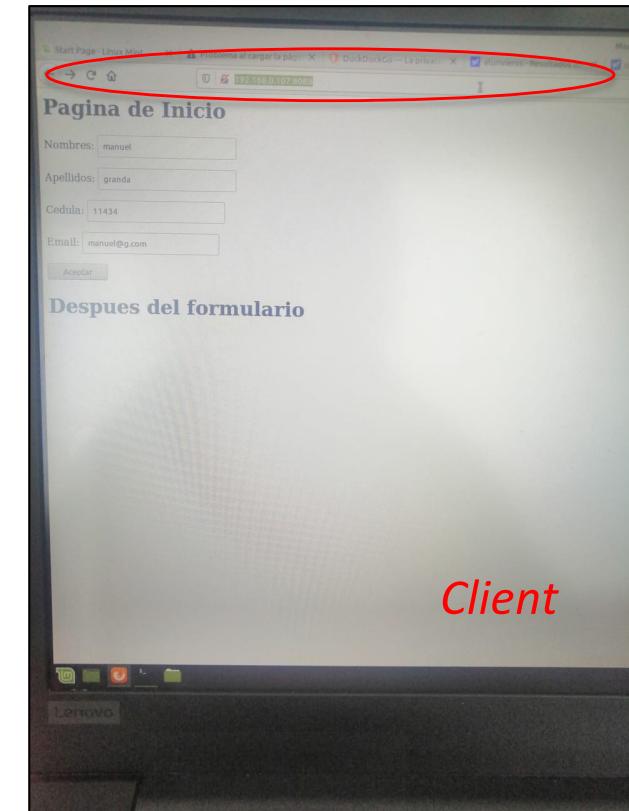
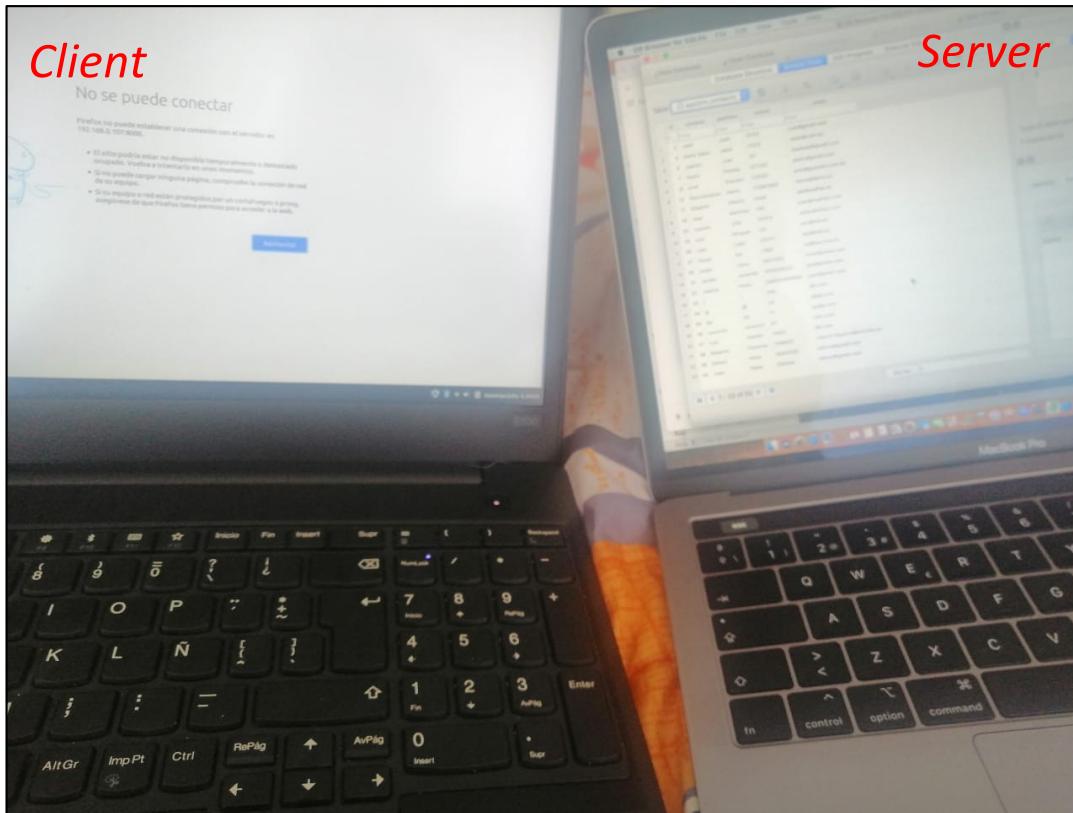
ALLOWED_HOST = ['*'] → Permite acceso a otros equipos

Reiniciar el Servidor Web WSGI 0:[puerto]

python manage.py runserver 0:8088

Invocar en el host remoto o equipo cliente

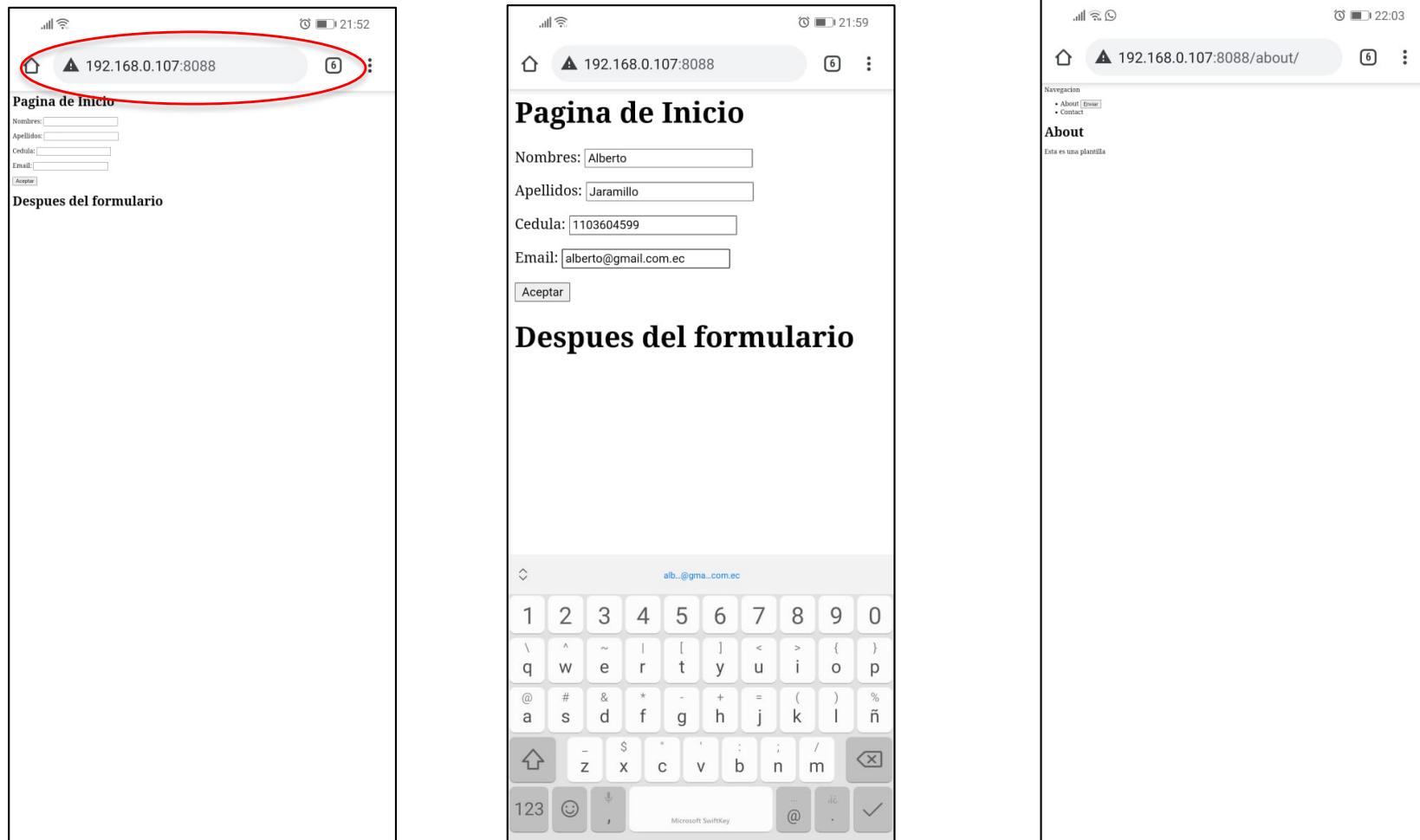
<http://192.168.0.111:8087/>



Usar en la LAN UNL → **python manage.py runserver 0:8081**
<http://10.20.137.253:8081/>

Ejecutar en el host remoto o celular

Colocar en el móvil → http://192.168.0.107:8088/



Ver en equipo servidor

```
Performing system checks... 3.067 ms 4.592 ms  
System check identified no issues (0 silenced).  
July 04, 2021 14:07:56  
Django version 3.2.3, using settings 'DemoDjango.settings'  
Starting development server at http://0:8088/ 6 ms  
Quit the server with CONTROL-C.  
[04/Jul/2021 14:08:09] "GET / HTTP/1.1" 200 746  
Not Found: /favicon.ico 620 ms  
[04/Jul/2021 14:08:09] "GET /favicon.ico HTTP/1.1" 404 2550  
Imprimo en pantalla y guardo data en BD  
{'nombres': 'manuel', 'apellidos': 'granda', 'cedula': '11434', 'email': 'manuel@g.com'}  
[04/Jul/2021 14:08:57] "POST / HTTP/1.1" 302 0  
[04/Jul/2021 14:08:57] "GET /about/ HTTP/1.1" 200 313
```

Ejecutar desde el otro Host y verificar en el servidor los nuevos datos almacenados

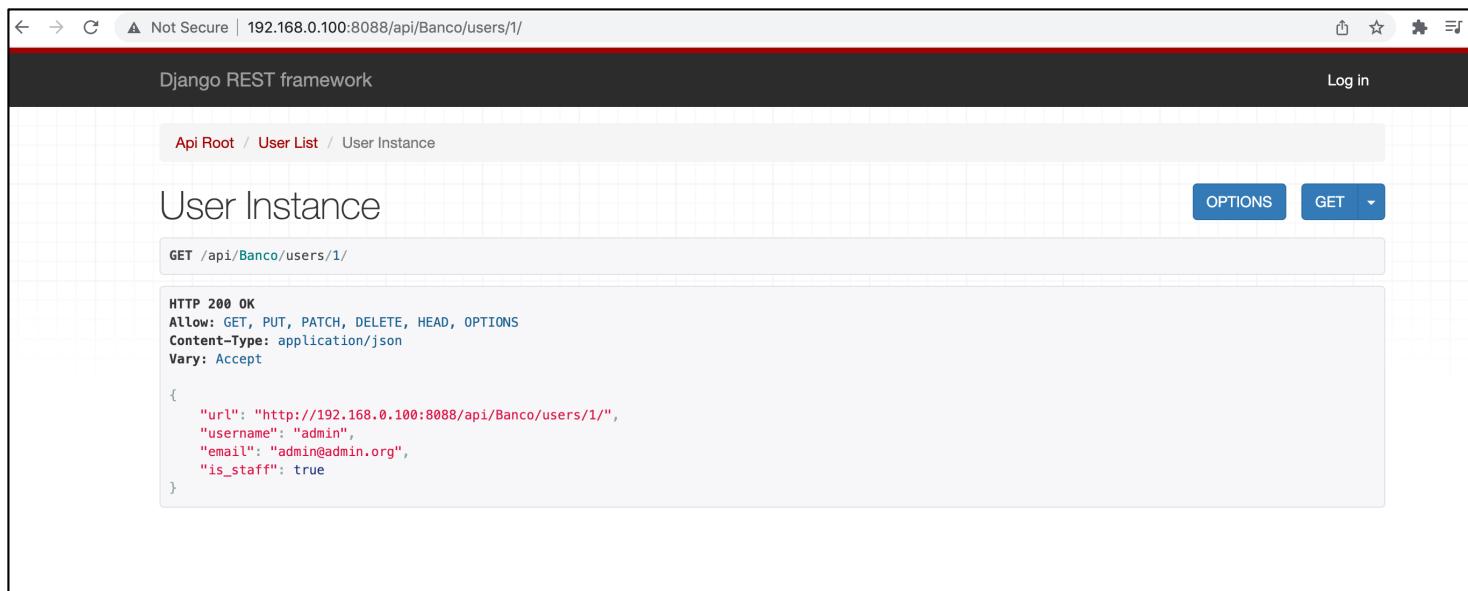
	id	nombres	apellidos	cedula	email
1	2	Juan	Juan	12132	juan@gmail.com
2	3	Maria Salas	salas	11223	salas@com.ec
3	4	Juanito	Juan	121	dsadsad@gmail.com
4	7	Pedro	Pareles	227332	pedro@gmail.com
5	8	Ariel	Graciani	123232	ariel@graciani.com.ec
6	13	Demonstracion	Demo	112567893	demo@demo.ec
7	17	Roberto	Alberto	120	ads@adsfas.ec
8	19	Juan	Martinez	748	juan@martinez.com
9	20	Juanita	Julia	121312	asfas@fdfsa.com
10	25	Luis	Miguel	120	ecuador@ecuador.ec
11	26	Loja	Lojan	232111	loja@loja.ec
12	27	Dbsql	Sql	7382	sql@sw.com.ec
13	30	Israel	Calva	5637923	israel@calva.com
14	31	Jenifer	Emilie	1978378232	jen@jenifer.com
15	32	Edison	Alava	5468326666666	edison@gmail.com
16	33	j	j	555	j@j.com
17	34	iii	iii	42	iii@iii.com
18	35	pp	pp	77	pp@pp.com
19	36	usuario02	usuario2	67	c@c.com
20	37	Luis	Castillo	74921	l@l.com
21	38	Roberto	Figueroa	1108923	roberto.figueroa@uni.edu.ec
22	39	Edison	Alava	19302332	edison@gmail.com
23	40	Juan	Pabio	234344	edison@gmail.com
24	41	manuel	granda	11434	manuel@g.com

Ver la DB en Browser for SQLite

CREANDO UNA API REST

Django versión 3.2.3

Creando una API REST



Envío de datos en formato Json por Http

Usando Django Rest Framework

<https://www.youtube.com/watch?v=F5mRW0jo-U4&t=400s>

<https://www.youtube.com/watch?v=TmsD8QExZ84>

<https://www.django-rest-framework.org/tutorial/quickstart/>

<https://docs.djangoproject.com/en/4.0/topics/db/queries/>

Ejemplo de Django con Vue: → <https://djackets.codewithstein.com/>

<https://www.youtube.com/watch?v=Yg5zkd9nm6w>

Usando Postman o navegador web

The screenshot shows the Postman web interface with the following details:

- Header Bar:** Shows multiple tabs including "rgfigueroa/POO", "node js express", "Curso de Node.", "Home - Django", "Quickstart - Dj", "http://postman-", "User Instance", "New Tab".
- Toolbar:** Home, Workspaces, API Network, Reports, Explore, Search Postman, Cloud icon, Invite button, Settings, Notifications, Global icon, Upgrade button.
- Left Sidebar:** My Workspace, Collections, APIs, Environments, Mock Servers, Monitors, Flows, History.
- Request Overview:** Title: PruebaTEST / http://postman-echo.com/get, Method: GET, URL: http://postman-echo.com/get, Status: 200 OK, Time: 868 ms, Size: 830 B, Save button.
- Request Details:** Headers tab selected, showing 6 headers:
 - x-forwarded-proto: http
 - x-forwarded-port: 80
 - host: postman-echo.com
 - x-amzn-trace-id: Root=1-61f82fe6-0a30d3ed63950735519d85c4
 - content-length: 0
 - user-agent: PostmanRuntime/7.29.0
- Body Tab:** Status: 200 OK, Body content:

```
1 "args": {},  
2 "headers": {  
3   "x-forwarded-proto": "http",  
4   "x-forwarded-port": "80",  
5   "host": "postman-echo.com",  
6   "x-amzn-trace-id": "Root=1-61f82fe6-0a30d3ed63950735519d85c4",  
7   "content-length": "0",  
8   "user-agent": "PostmanRuntime/7.29.0"  
9 }
```
- Bottom Navigation:** Console, response.json, Show all.

Contacto



Universidad
Nacional
de Loja



Ing. Roberth Figueroa Díaz M.Sc.
Carrera Computación
roberth.figueroa@unl.edu.ec
www.unl.edu.ec



UNL

Universidad
Nacional
de Loja

Gracias por su atención

REDES **UNL** OFICIALES



Universidad Nacional De Loja - UNL



@UnlOficial



UnlOficial



UnlOficial

Educamos para transformar