



# SISTEMA DE REGISTRO MEDICO PARA CLÍNICA DE MEDICINA INTEGRAL

## Manual de Instalación

Versión: 1.0

Fecha: 16/01/2018

<b>Proyecto</b>	Sistema de Registro Medico para Clínica de Medicina Integral		
<b>Entregable</b>	Manual de Instalación		
<b>Versión/Edición</b>	1.0	<b>Fecha Versión</b>	16/01/2018
<b>Aprobado por</b>	Carlos Cativo	<b>Fecha Aprobación</b>	18/01/2018
		<b>N.º Total de Páginas</b>	11

Nombre y Apellidos	
Cativo Argueta Carlos Antonio	CA14002
Granados Orellana Mauricio Ernesto	GO14002
Granados Cruz German Steven	GC14003
Navarro Alfaro Melanie Carolina	NA15004
Oliva Avelar Irving Vladimir	OA09020
Orellana Zepeda Dennys Miguel Ángel	OZ09001

# Índice

DESCRIPCIÓN DEL SISTEMA.....	3
Objetivo.....	3
Alcance .....	3
Funcionalidad .....	3
RECURSOS HARDWARE .....	3
Servidores.....	3
Estaciones cliente .....	4
Restricciones.....	4
RECURSOS SOFTWARE.....	4
Restricciones técnicas del sistema .....	4
INSTALACIÓN Y CONFIGURACIÓN DEL SISTEMA.....	5
VERIFICACIÓN DEL PROCESO DE INSTALACIÓN .....	13

# DESCRIPCIÓN DEL SISTEMA

## Objetivo

Con este proyecto se pretende poder apoyar a la institución *Clínica de medicina integral* en el registro medico de pacientes, así como a realización de consultas y escritura de recetas.

## Alcance

Poder mecanizar el proceso de llevar el control de pacientes, desde el paso de registrarlos a través de un expediente clínico hasta poder realizar consultas y pre-escribirle recetas médicas para continuar su tratamiento médico que puedan ser impresas y entregadas al paciente.

## Funcionalidad

Este sistema ofrece una facilidad para el registro de pacientes que acuden a una clínica a tomar una consulta de medicina general, pudiendo registrarlos y hacer operaciones en simples pasos, disminuyendo así el trabajo del médico que utiliza el sistema completamente, así como disminuyendo gastos indirectos como el gasto de papel pre fabricado para la elaboración de recetas y expedientes.

# RECURSOS HARDWARE

## Servidores

Se usará un servidor virtual en PythonAnywhere.com el cual es un entorno de desarrollo integrado en línea y un servicio de alojamiento web basado en el lenguaje de programación de Python. Proporciona acceso en el navegador a las interfaces de línea de comandos de Python y Bash basadas en servidor, junto con un editor de código con resaltado de sintaxis. Los archivos de programa se pueden transferir hacia y desde el servicio usando el navegador del usuario. Las aplicaciones web alojadas por el servicio se pueden escribir utilizando cualquier marco de aplicación basado en WSGI.

## Estaciones cliente

Dato		Valor recomendado
Procesador		Intel Pentium 4 o posterior compatible con SSE2
Memoria RAM		512 MB o más
Tamaño Almacenamiento		350 MB o más

## Restricciones

Restricción	Detalle
Uso mediante internet	Usar el producto requiere de un dispositivo a una red de internet.

## RECURSOS SOFTWARE

### Restricciones técnicas del sistema

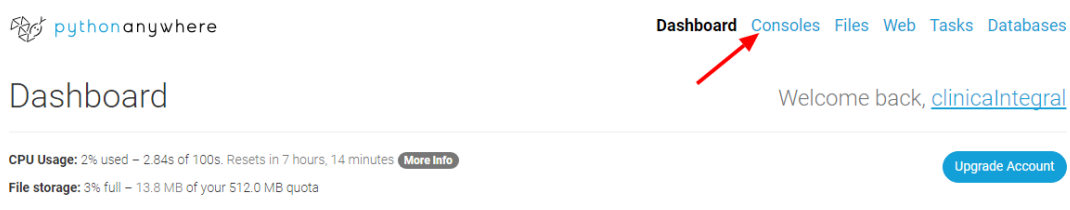
Elemento	Descripción
Sistema operativo	Windows, Linux, OSx, Android, iOS
Navegador	Chrome, Opera, Firefox, Internet Explorer, Microsoft Edge

# INSTALACIÓN Y CONFIGURACIÓN DEL SISTEMA

Para lograr que nuestro sistema funcione correctamente hay que seguir los siguientes pasos.

## 1. Bajar nuestro código en PythonAnywhere

Previamente hay que crear una cuenta en PythonAnywhere (PA), cuando ya se haya registrado será redirigido al panel de control o Dashboard, se selecciona el apartado "Consoles".



Ahora se selecciona la opción para iniciar una consola "Bash", que es la versión PA de una consola, como la que se encuentra en una PC.

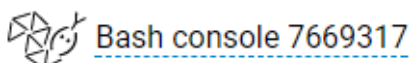


### Start a new console:

Python: [3.6](#) / [3.5](#) / [3.4](#) / [3.3](#) / [2.7](#) IPython: [3.6](#) / [3.5](#) / [3.4](#) / [3.3](#) / [2.7](#) PyPy: [2.7](#)  
Other: [Bash](#) | [MySQL](#)  
Custom: [+](#)

Se abrirá una consola, y ahora para descargar el proyecto en PythonAnywhere, mediante la clonación del repositorio, se ejecuta el comando:

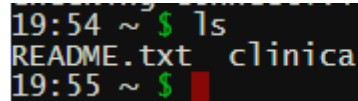
```
git clone https://github.com/cativo23/clinica.git
```



```
19:52 ~ $ git clone https://github.com/cativo23/clinica.git
Cloning into 'clinica'...
remote: Counting objects: 107, done.
remote: Compressing objects: 100% (89/89), done.
remote: Total 107 (delta 15), reused 105 (delta 13), pack-reused 0
Receiving objects: 100% (107/107), 770.35 KiB | 0 bytes/s, done.
Resolving deltas: 100% (15/15), done.
Checking connectivity... done.
19:54 ~ $
```

Para comprobar que se ha descargado la carpeta “clínica” del repositorio, se ejecuta el comando:

```
ls
```



```
19:54 ~ $ ls
README.txt  clinica
19:55 ~ $
```

## 2. Crear un entorno virtual en PythonAnywhere

Para que el proyecto funcione tenemos que crear un entorno virtual en PA, mediante el comando:

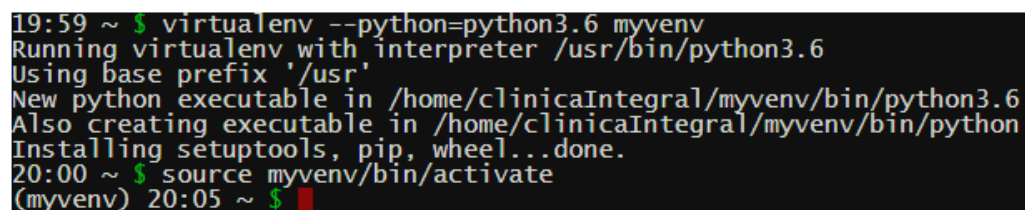
```
virtualenv --python=python3.6 myenv
```

y para activarlo se usa el comando:

```
source myenv/bin/activate
```



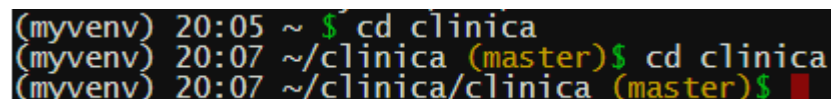
Bash console 7669317



```
19:59 ~ $ virtualenv --python=python3.6 myenv
Running virtualenv with interpreter /usr/bin/python3.6
Using base prefix '/usr'
New python executable in /home/clinicaIntegral/myenv/bin/python3.6
Also creating executable in /home/clinicaIntegral/myenv/bin/python
Installing setuptools, pip, wheel...done.
20:00 ~ $ source myenv/bin/activate
(myenv) 20:05 ~ $
```

Luego ejecutamos 2 veces el comando:

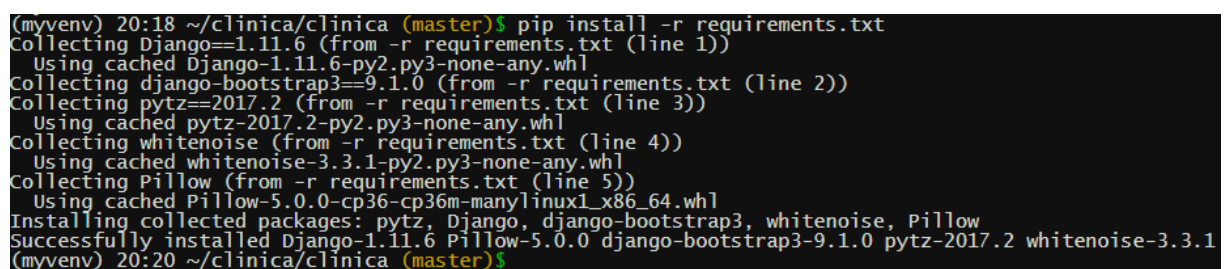
```
cd clinica
```



```
(myenv) 20:05 ~ $ cd clinica
(myenv) 20:07 ~/clinica (master)$ cd clinica
(myenv) 20:07 ~/clinica/clinica (master)$
```

Ya colocados en la posición del archivo *requirements.txt*, podemos instalar los requerimientos necesarios para que nuestra página funcione, para eso ejecutamos:

```
pip install -r requirements.txt
```



```
(myenv) 20:18 ~/clinica/clinica (master)$ pip install -r requirements.txt
Collecting Django==1.11.6 (from -r requirements.txt (line 1))
  Using cached Django-1.11.6-py2.py3-none-any.whl
Collecting django-bootstrap3==9.1.0 (from -r requirements.txt (line 2))
Collecting pytz==2017.2 (from -r requirements.txt (line 3))
  Using cached pytz-2017.2-py2.py3-none-any.whl
Collecting whitenoise (from -r requirements.txt (line 4))
  Using cached whitenoise-3.3.1-py2.py3-none-any.whl
Collecting Pillow (from -r requirements.txt (line 5))
  Using cached Pillow-5.0.0-cp36-cp36m-manylinux1_x86_64.whl
Installing collected packages: pytz, Django, django-bootstrap3, whitenoise, Pillow
Successfully installed Django-1.11.6 Pillow-5.0.0 django-bootstrap3-9.1.0 pytz-2017.2 whitenoise-3.3.1
(myenv) 20:20 ~/clinica/clinica (master)$
```

### 3. Recopilar ficheros estáticos

Para que los archivos estáticos (imágenes, hojas de estilo, etc.) se puedan usar se usa el comando:

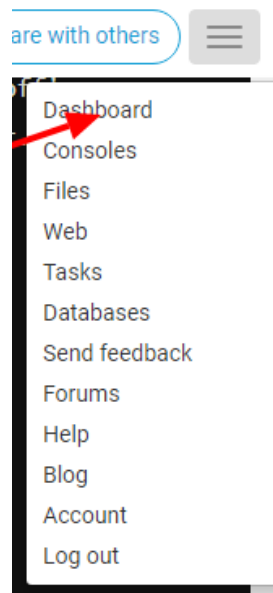
```
python manage.py collectstatic
```

```
(myvenv) 20:21 ~/clinica/clinica (master)$ python manage.py collectstatic
Copying '/home/clinicaIntegral/clinica/clinica/static/images/pic03.jpg'
Copying '/home/clinicaIntegral/clinica/clinica/static/images/pic07.jpg'
Copying '/home/clinicaIntegral/clinica/clinica/static/images/UES.png'
Copying '/home/clinicaIntegral/clinica/clinica/static/images/dark_tint.png'
Copying '/home/clinicaIntegral/clinica/clinica/static/images/pic08.jpg'
Copying '/home/clinicaIntegral/clinica/clinica/static/images/pic02.jpg'
```

```
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/css/jquery-ui.min.css'
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/css/jquery-ui.structure.min.css'
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/css/jquery-ui.theme.min.css'
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/js/scripts.js'
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/js/jquery-ui.min.js'
Copying '/home/clinicaIntegral/clinica/clinica/integral/static/js/jquery-3.2.1.min.js'
94 static files copied to '/home/clinicaIntegral/clinica/clinica/staticfiles'.
(myvenv) 20:21 ~/clinica/clinica (master)$
```

### 4. Crear base de datos en PA

Nos vamos al apartado “Databases” de PA y seleccionamos MySQL.



Luego en Create a Database ponemos el nombre “clínica” y si pide contraseña hay que recordarla.

#### Create a database

Your database names always start with your username + "\$". There's no need to type that prefix in below, though: PythonAnywhere will automatically add it.

Database name:

Create

En caso de que no haya pedido contraseña o se quiera cambiarla, en donde dice MySQL PASSWORD se puede cambiar por otra:

MySQL password:

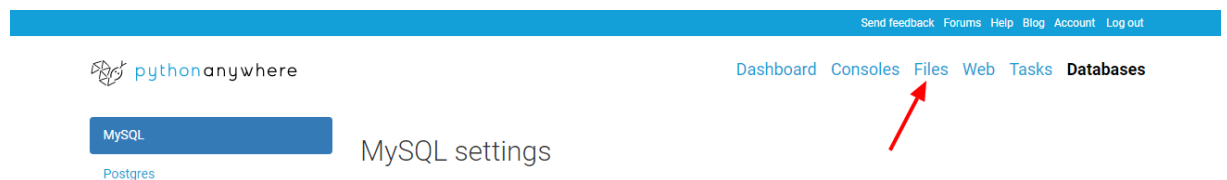
This should be different to your main PythonAnywhere password, because it is likely to appear in plain text in any web applications you write.

New password:

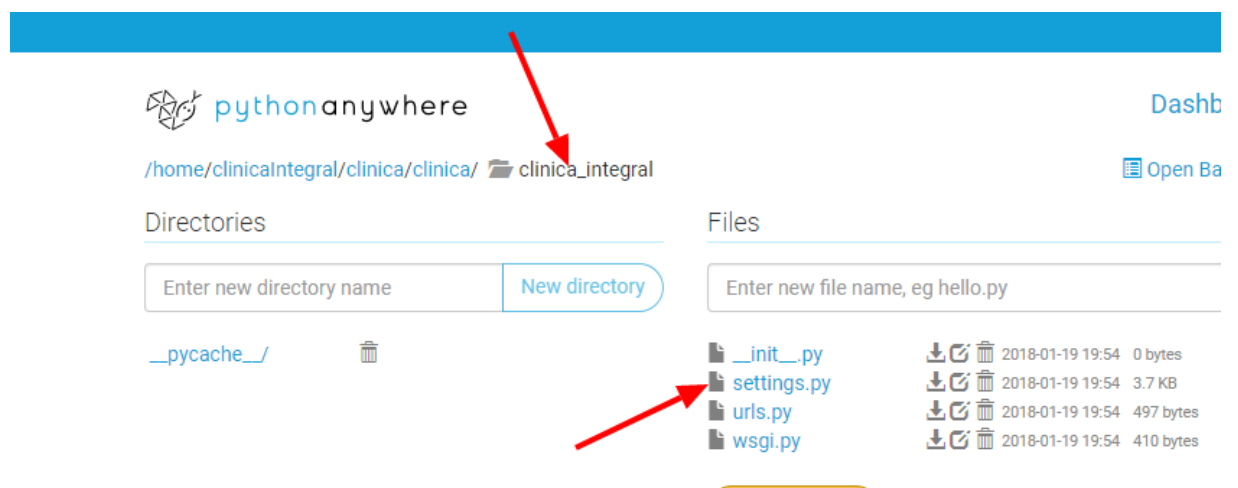
Confirm password:

Set MySQL password

Después nos vamos al apartado de Files:



Después navegamos hasta la carpeta clínica\_integral, dentro de ella está el archivo settings.py, abrimos el archivo:





Ahora en la parte DATABASE verificamos que esté así:

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.mysql',  
        'NAME': '<usuario>$clinica',  
        'USER': '<usuario>',  
        'PASSWORD': 'password',  
        'HOST': '<usuario>.mysql.pythonanywhere-services.com',  
        'PORT': '',  
    }  
}
```

```
DATABASES = {  
    'default': {  
        'ENGINE': 'django.db.backends.mysql',  
        'NAME': 'clinica1$clinica',  
        'USER': 'clinica1',  
        'PASSWORD': 'a',  
        'HOST': 'clinica1.mysql.pythonanywhere-services.com',  
        'PORT': '',  
    }  
}
```

Recuerda cambiar <usuario> y password con los que tu tengas, guarda el archivo antes de continuar.

De ahí creamos la base en la consola con:

```
python manage.py migrate
```

```

System check identified some issues:

WARNINGS:
?: (mysql.W002) MySQL Strict Mode is not set for database connection 'default'
  HINT: MySQL's Strict Mode fixes many data integrity problems in MySQL,
  nto errors. It is strongly recommended you activate it. See: https://docs.djan
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, integral, myauth, sessions
Running migrations:
  Applying myauth.0001_initial... OK
  Applying contenttypes.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0001_initial... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying integral.0001_initial... OK
  Applying integral.0002_auto_20171116_2110... OK
  Applying sessions.0001_initial... OK
(myvenv) 20:41 ~/clinica/clinica (master)$

```

y creamos un superusuario con:

```
python manage.py createsuperuser
```

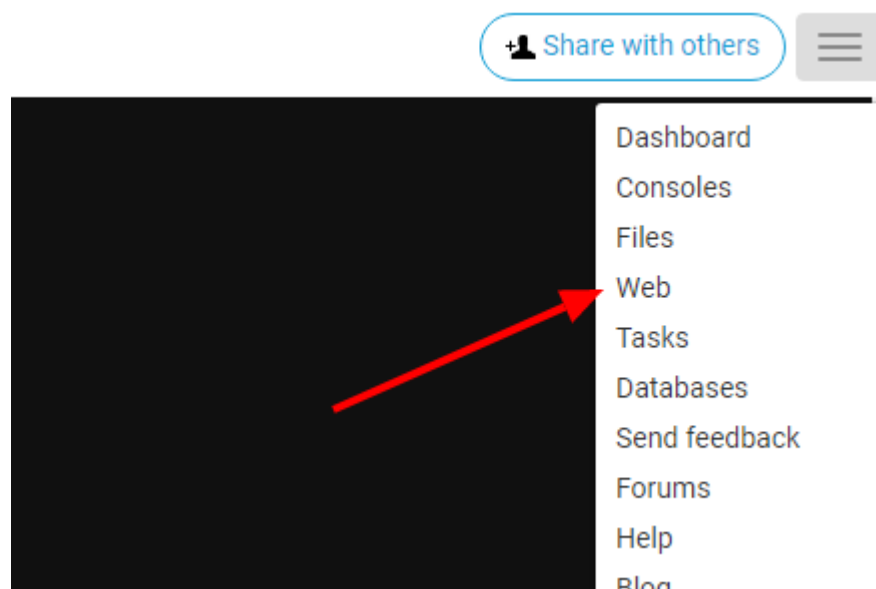
```

(myvenv) 20:44 ~/clinica/clinica (master)$ python manage.py createsuperuser
Username: root
Email address: cativo23.kt@gmail.com
Password:
Password (again):
Superuser created successfully.
(myvenv) 20:44 ~/clinica/clinica (master)$

```

## 5. Publicar nuestro blog como una aplicación web

Nos vamos al apartado de Web:



Luego damos click en Add a new web app:

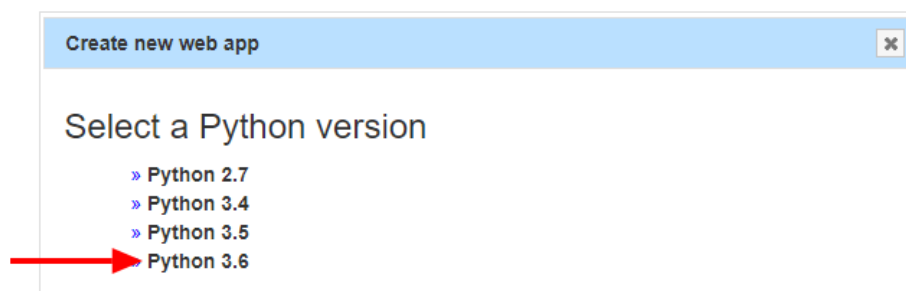
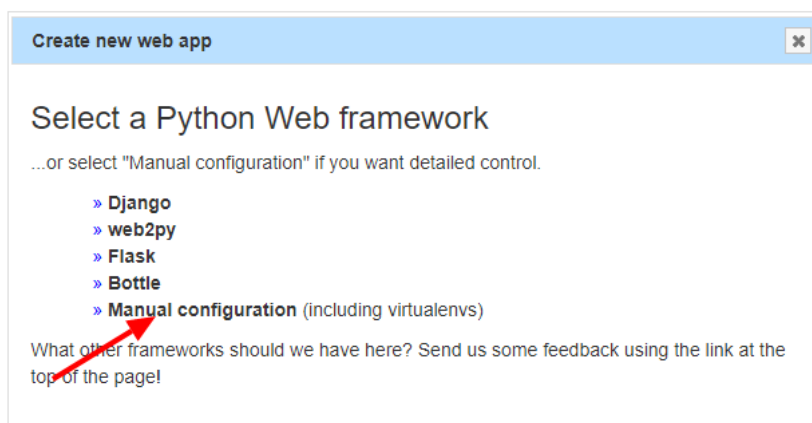


+ Add a new web app

You have no web apps

To create a PythonAnywhere-hosted web app, click the "Add a new web app" button to the left.

En la ventana de diálogo, después de confirmar el nombre de dominio, elije manual configuration (configuración manual) (la opción "Django" no). Luego, elije Python 3.6 y haz clic en "Next" para terminar con el asistente.

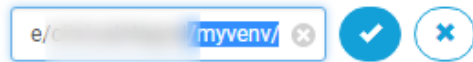


#### a. Configurar el virtualenv

En la sección "Virtualenv", haz clic en el texto rojo que dice "Enter the path to a virtualenv" (Introduce la ruta a un virtualenv) y escribe: `/home/<usuario>/myvenv/`

### Virtualenv:

Use a virtualenv to get different versions of flask, django etc from our default system ones. [More info here](#). You need to **Reload your web app** to activate it; NB - will do nothing if the virtualenv does not exist.



#### b. Configurar el fichero WSGI

Django funciona utilizando el "protocolo WSGI", un estándar para servir sitios web usando Python, que PythonAnywhere soporta. La forma de configurar PythonAnywhere para que reconozca nuestro blog Django es editar un fichero de configuración WSGI.

Haz clic en el enlace "WSGI configuration file" (en la sección "Code" en la parte de arriba de la página; se llamará algo parecido a `/var/www/<tu-usuario>_pythonanywhere_com_wsgi.py`) y te redirigirá al editor.

Elimina todo el contenido actual y reemplázalo con algo como esto

```
import os
```

```
import sys
```

```
path = '/home/<usuario>l/clinica/clinica'
```

```
if path not in sys.path:
```

```
    sys.path.append(path)
```

```
os.environ['DJANGO_SETTINGS_MODULE'] = 'clinica_integral.settings'
```

```
from django.core.wsgi import get_wsgi_application
```

```
from whitenoise.django import DjangoWhiteNoise
```

```
application = DjangoWhiteNoise(get_wsgi_application())
```



/var/www/███\_pythonanywhere\_com\_wsgi.py

```
1 import os
2 import sys
3
4 path = '/home/███/clinica/clinica'
5 if path not in sys.path:
6     sys.path.append(path)
7
8 os.environ['DJANGO_SETTINGS_MODULE'] = 'clinica_integral.settings'
9
10 from django.core.wsgi import get_wsgi_application
11 from whitenoise.django import DjangoWhiteNoise
12 application = DjangoWhiteNoise(get_wsgi_application())
```

## VERIFICACIÓN DEL PROCESO DE INSTALACIÓN

Para verificar si se ha instalado correctamente, primero ir a la url:

- ***http:// <usuario>.pythonanywhere.com***

Luego, intente acceder al sistema con el super usuario anteriormente creado.

Si pudo ingresar es que se ha instalado correctamente, caso contrario contacte el creado del sistema.