Despliegue de aplicación Django en PythonAnywhere

Configuraciones para instalar un proyecto Django 3.2 en PythonAnywhere.

Tabla de contenidos

Información inicial

Lo primero a tener en cuenta si estás usando una cuenta gratuita, es tener paciencia. Estas instrucciones sirven para la instalación de un blog creado en Django usando un diseño personalizado.

Se usaron las siguientes versiones base en local:

- **Lenguaje de programación:** Pyhton 3.6.9
- **Gestor de paquetes:** Pip 9.0.1
- **Framework web:** Django 3.2.16
- **Base de datos:** Sqlite3

Generar token en github para el servidor

- https://github.com/settings/tokens
- ghp_TOKEN_GENERADO_EN_GITHUB

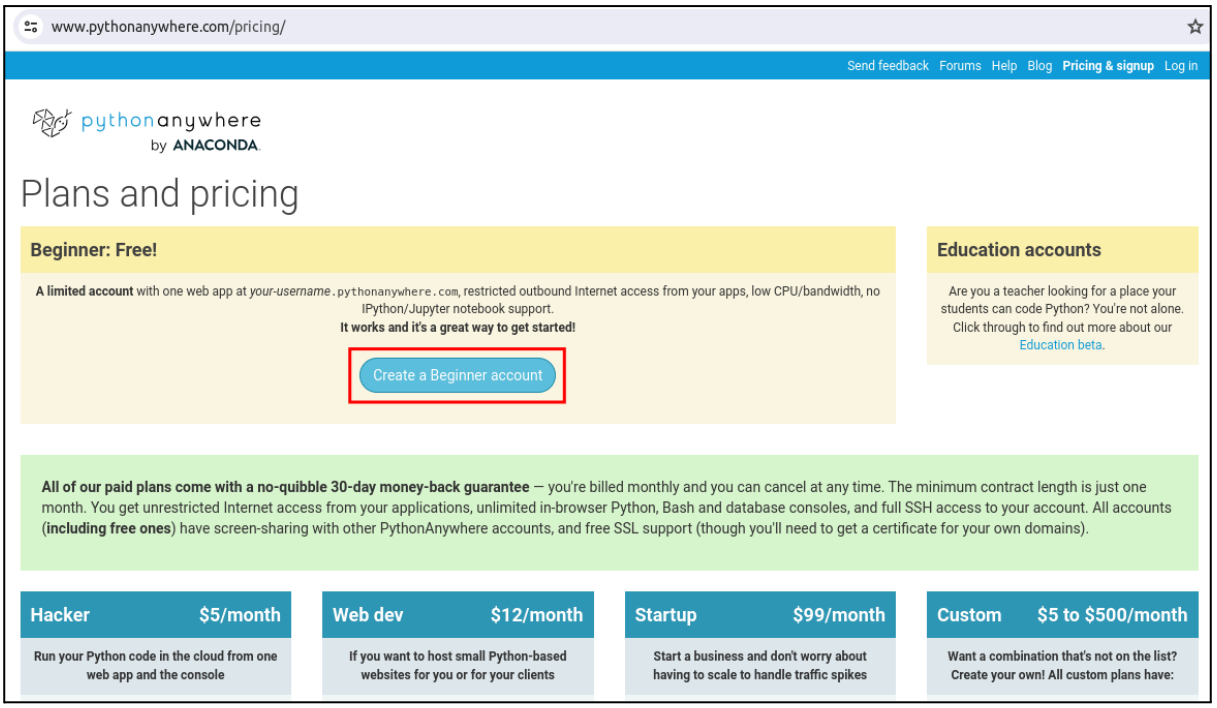
Ayuda

- **Deploying an existing Django project on PythonAnywhere:**
  - https://help.pythonanywhere.com/pages/DeployExistingDjangoProject/
- **Deploy a Django web app to Python Anywhere [FREE]:**
  - https://www.youtube.com/watch?v=xtnUwvjOThg
- **How to Deploy Django Web App to Pythonanywhere (with static files and media):**
  - https://www.youtube.com/watch?v=Gnwm7fQnt2c

Pasos para Despliegue 🚀

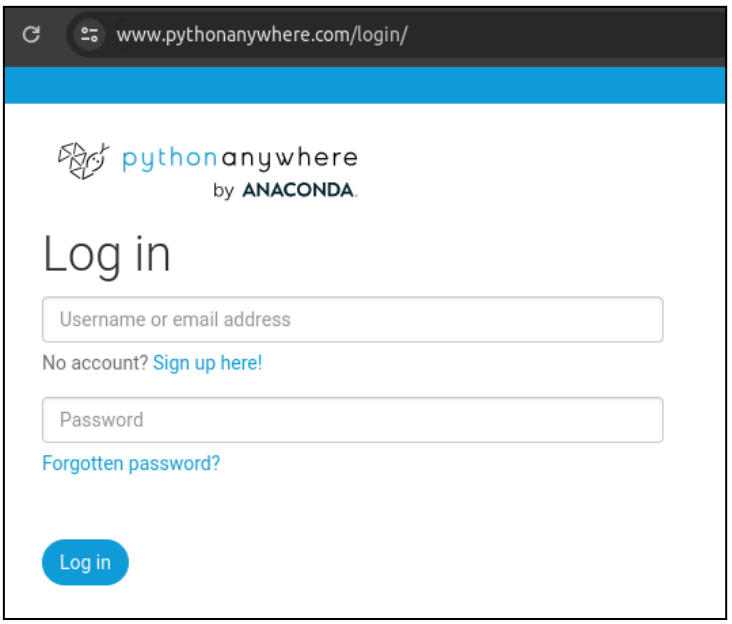
Se asume que existe previamente una cuenta beginner en pythonanywhere:

https://www.pythonanywhere.com/pricing/



Acceder a pythonanywhere

- https://www.pythonanywhere.com/user/xqazprog/
- Agregar usuario y password



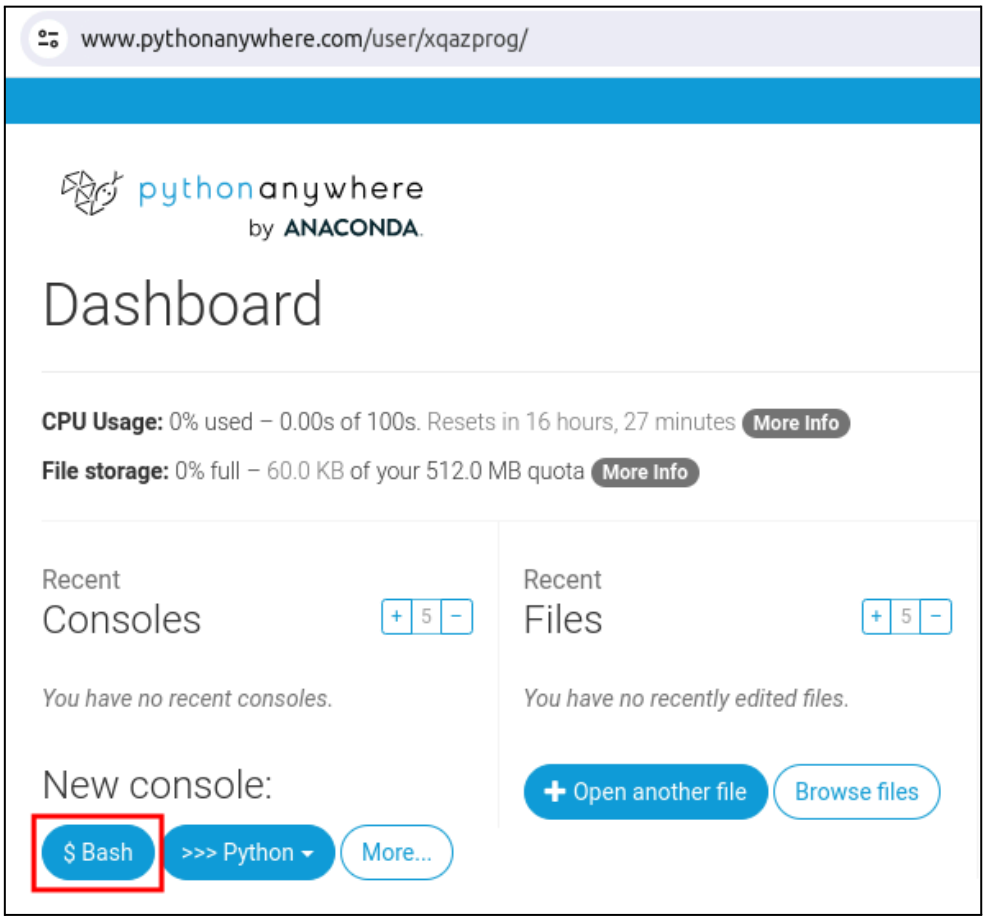
1. Crear y activar entorno virtual para python

- Versiones de python se ubican en `/usr/bin/`

Puede usar un virtualenv en una nueva aplicación web (creada usando la opción "Configuración manual") o en sus aplicaciones web existentes. Para utilizar un virtualenv en su aplicación web, haga lo siguiente.

Crear un entorno virtual

Vaya a la pestaña `Consoles` e inicie una consola `Bash`.

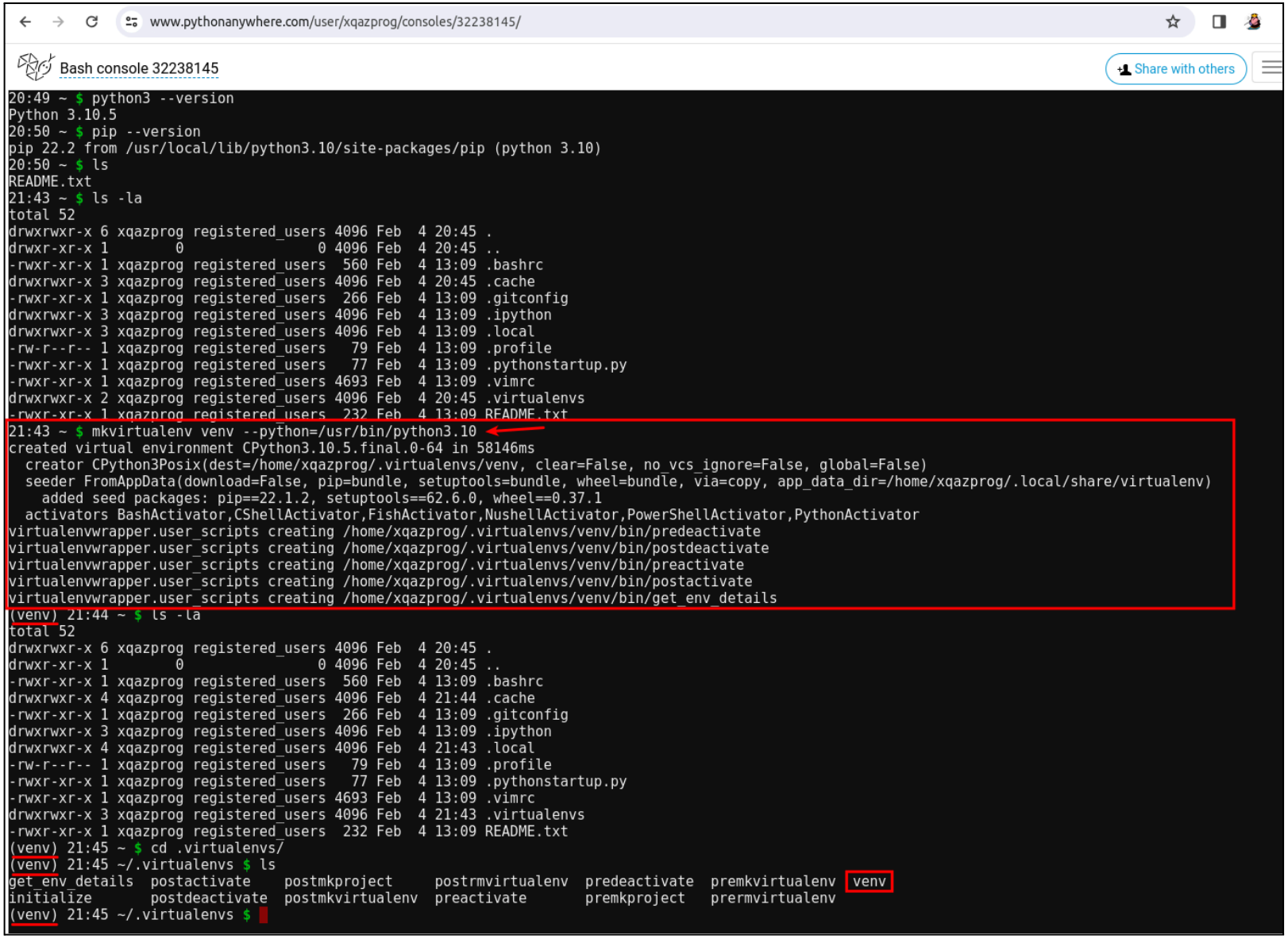


Recomendamos utilizar virtualenvwrapper, una práctica herramienta de línea de comandos, para crear su virtualenv.

Especifica qué versión de Python usar para tu virtualenv usando la opción `--python`, pero ten en cuenta que debe coincidir con la versión de Python que has elegido para tu aplicación web. Entonces, para crear un nuevo entorno virtual Python 3.10, ejecute este comando:

- `mkvirtualenv venv --python=/usr/bin/python3.6`
- `mkvirtualenv venv --python=/usr/bin/python3.10`

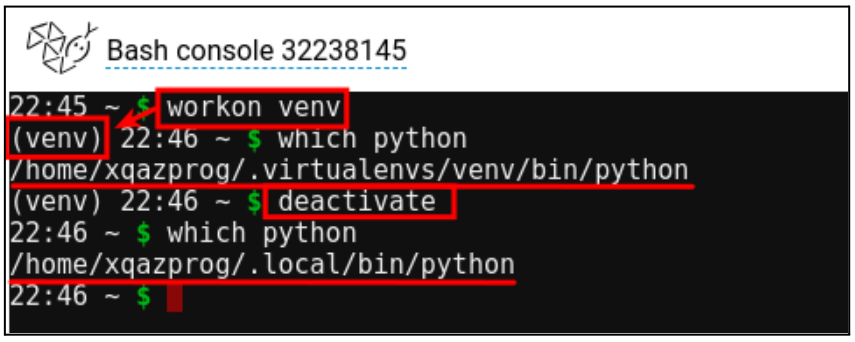
En la flecha se puede ver cómo se crea el entorno virtual.



Una vez que su virtualenv esté listo y activo, verá `(env) $` en su mensaje.

Desactivar y reactivar el virtualenv

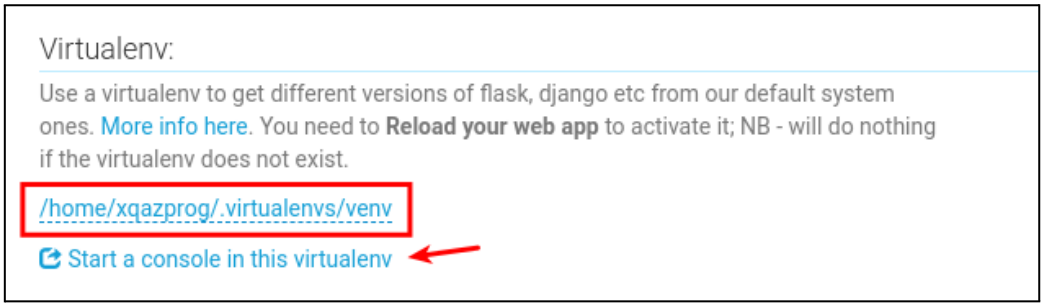
Una vez que cree su `virtualenv`, debe activarlo. Se activa automáticamente inmediatamente después de crearlo con `mkvirtualenv`. Se puede activar usando `workon` y para desactivar, utilice `deactivate`:

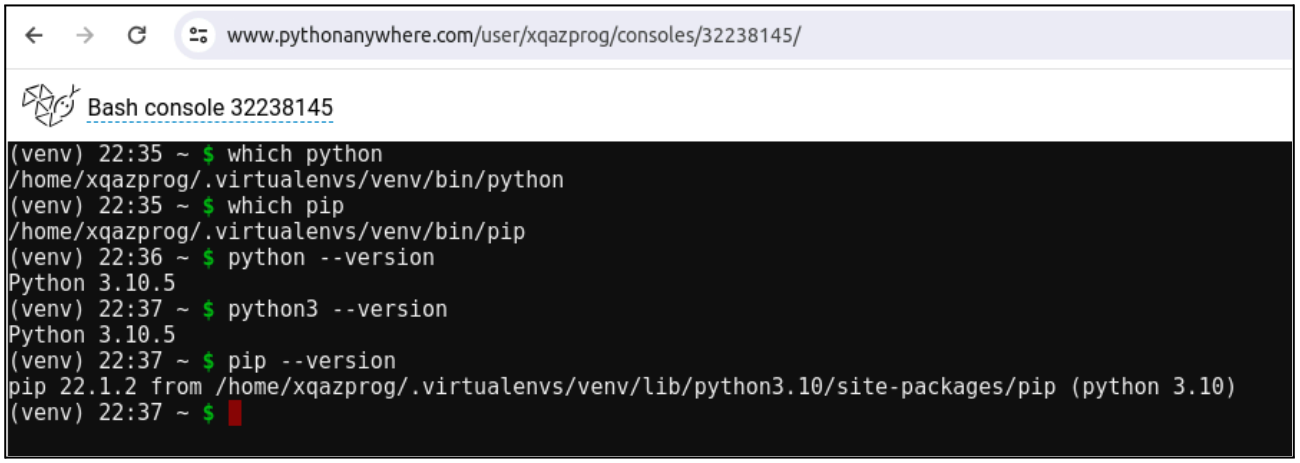


También se puede activar usando:

- `source /home/usuario/.virtualenvs/venv/bin/activate`

También se puede generar una nueva consola bash con el entorno virtual activado desde la pestaña web.





Instalar paquetes en su virtualenv

Para instalar paquetes en el virtualenv se puede utilizar pip

- `pip3 install django`
- `pip3 install whitenoise`

Para instalar una versión específica de Django

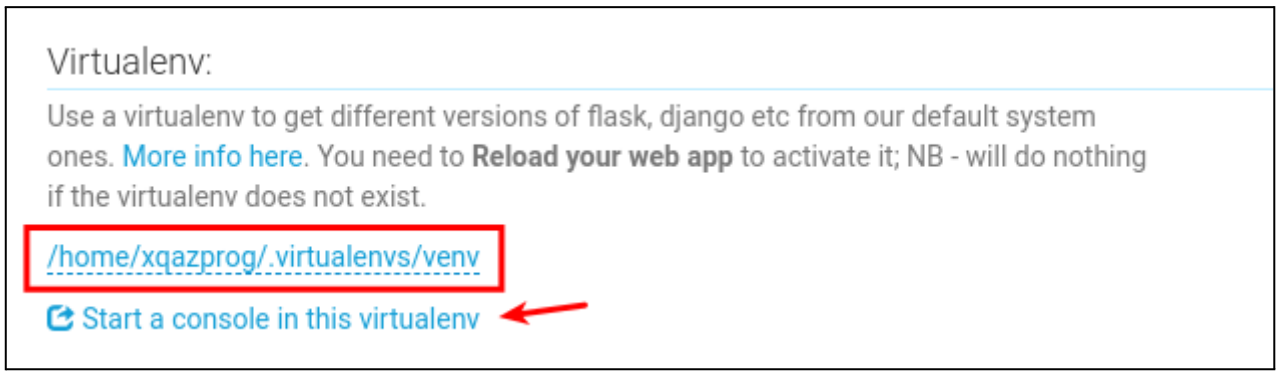
- `pip3 install django==3.2.6`

Configurar la aplicación para usar este entorno virtual

Ahora que tiene un entorno virtual y conoce su ruta, configure su aplicación para usar este entorno virtual.

Vaya a la pestaña Web y en la sección Virtualenv, ingrese la ruta:

- `/home/nombreusuario/.virtualenvs/myvirtualenv`
- `/home/xqazprog/.virtualenvs/venv`



SUGERENCIA

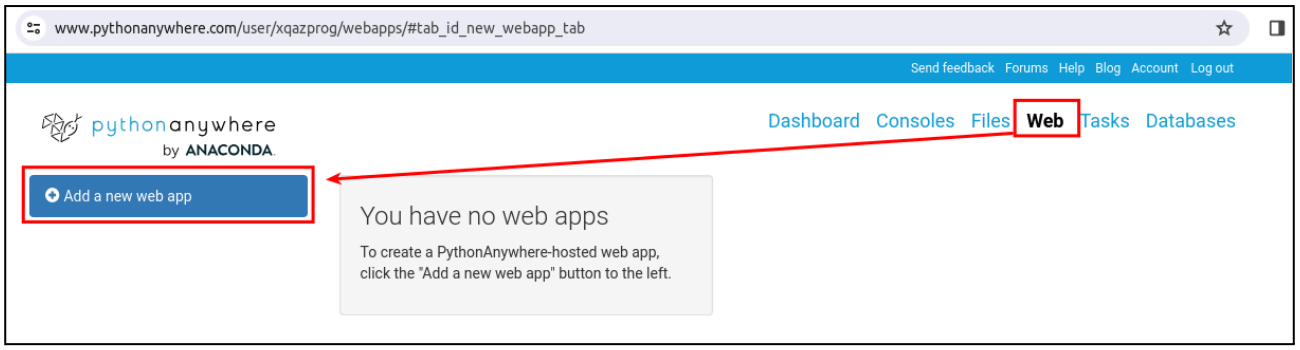
Si está utilizando virtualenvwrapper, puede simplemente ingresar el nombre del virtualenv, myvirtualenv, y el sistema adivinará automáticamente el resto de la ruta (/home/myusername/.virtualenvs, etc.) después de presionar ok.

Ahora, vuelva a cargar su aplicación web y debería encontrar que tiene acceso a todos los paquetes en su virtualenv, en lugar de tener acceos a los paquetes del sistema.

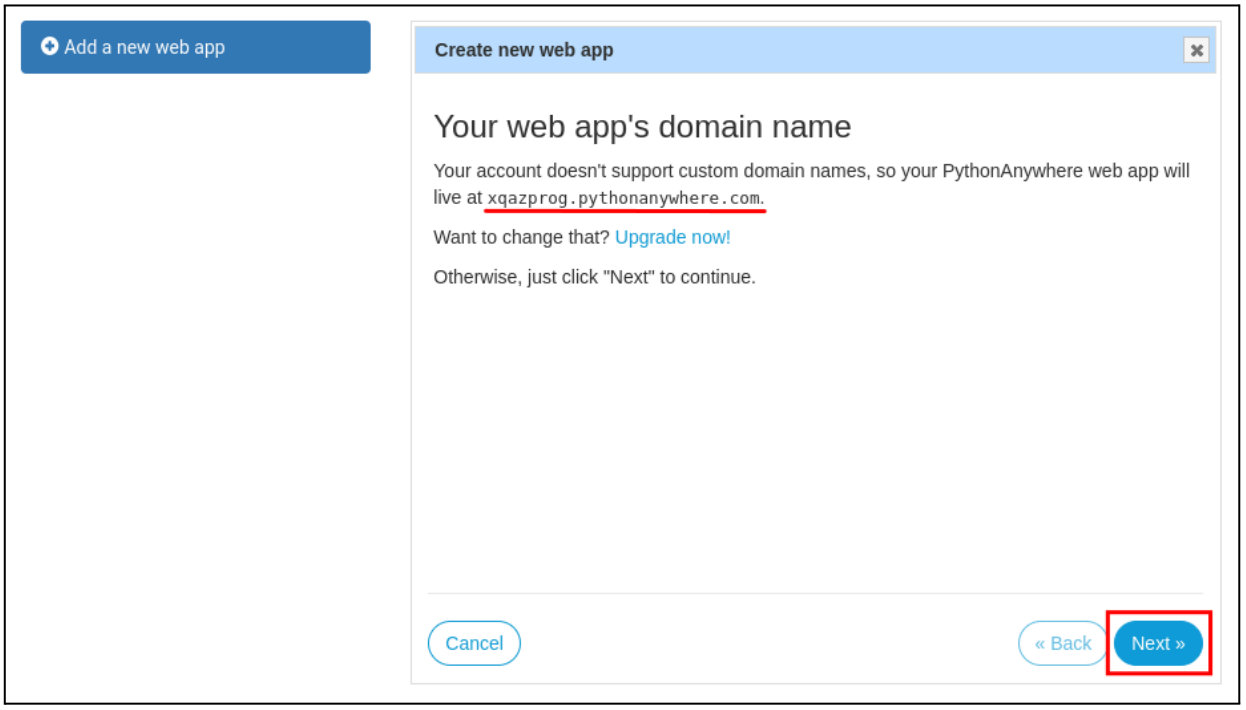
2. Crear nueva web app en pythonanywhere

Paso 1: Crear nueva web app

- https://www.pythonanywhere.com/user/xqazprog/webapps/
- **Nombre:** `xqazprog.pythonanywhere.com`

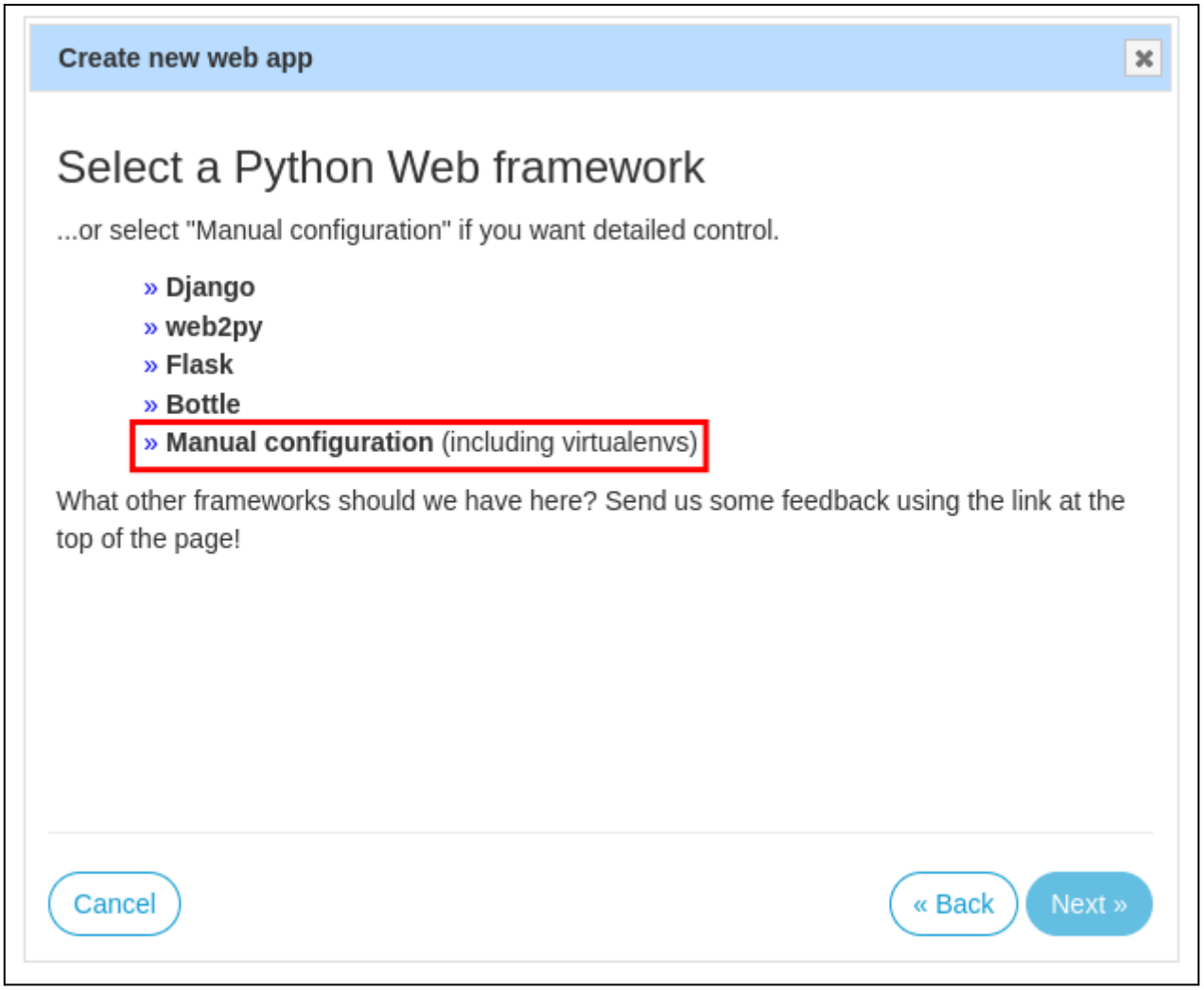


Luego de presionar el botón, se deben seguir las instrucciones para crear un nuevo nombre de dominio para la aplicación.



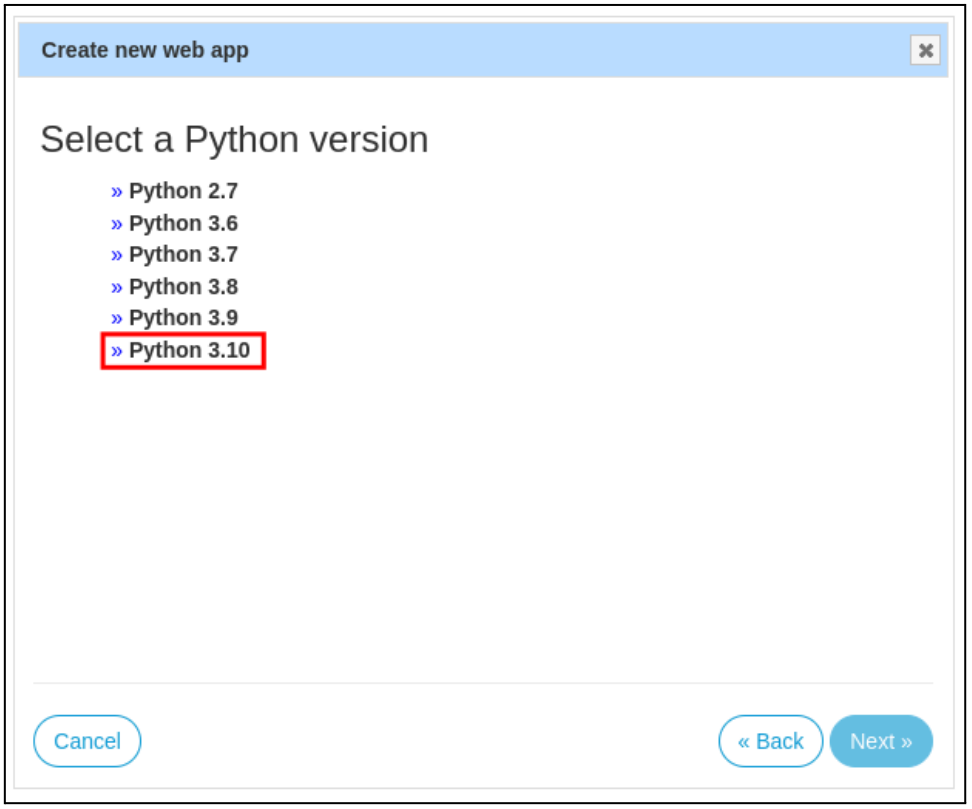
Paso 2: Crear nuevo framework de python

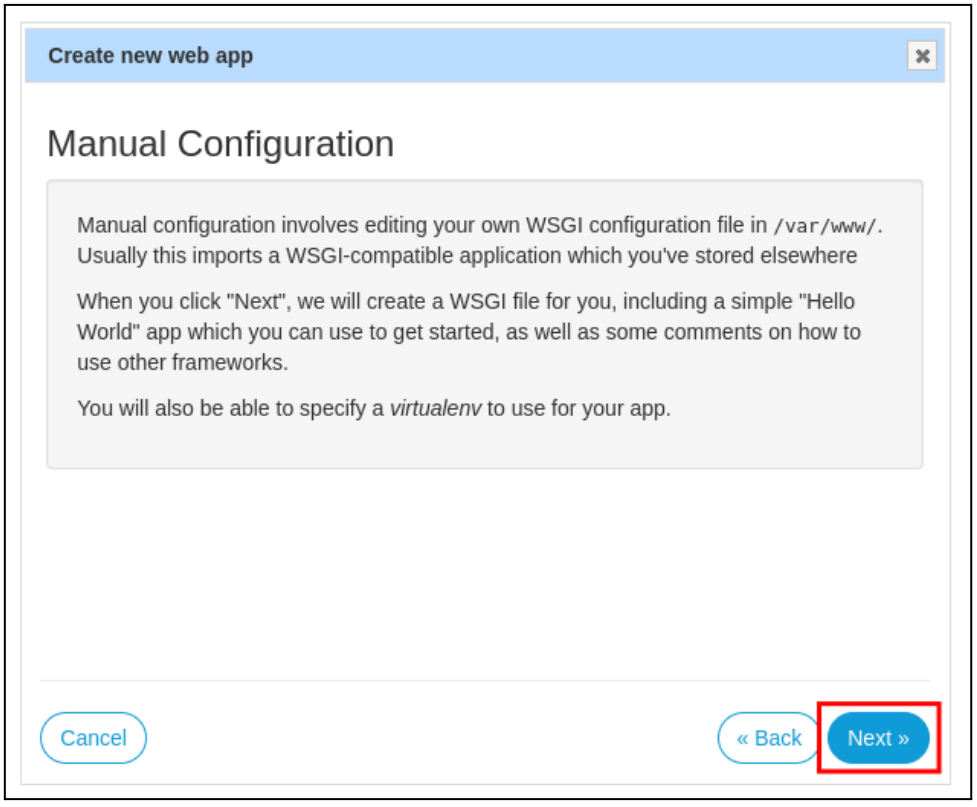
- Escoger configuración manual
    - La configuración manual implica editar su propio archivo de configuración WSGI en `/var/www/`. Por lo general, esto importa una aplicación compatible con WSGI que ha almacenado en otro lugar
    - Cuando haga clic en "Siguiente", crearemos un archivo WSGI para usted, incluida una aplicación simple "Hello World" que puede usar para comenzar, así como algunos comentarios sobre cómo usar otros marcos.
    - También podrá especificar un virtualenv para usar en su aplicación.



Paso 3: Seleccionar versión de python

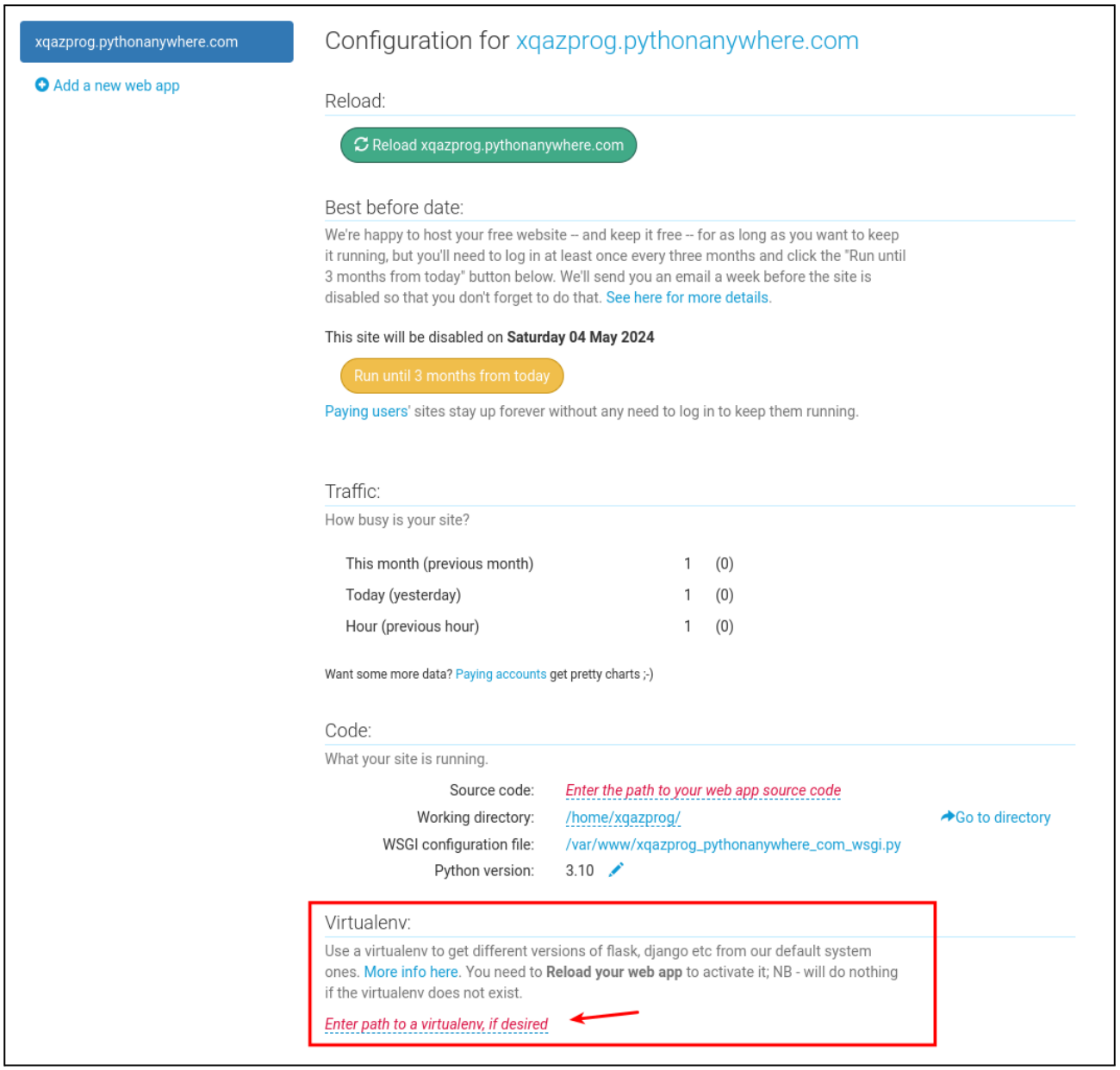
- Versiones desde 3.6 a 3.10
- Escoger Python 3.6

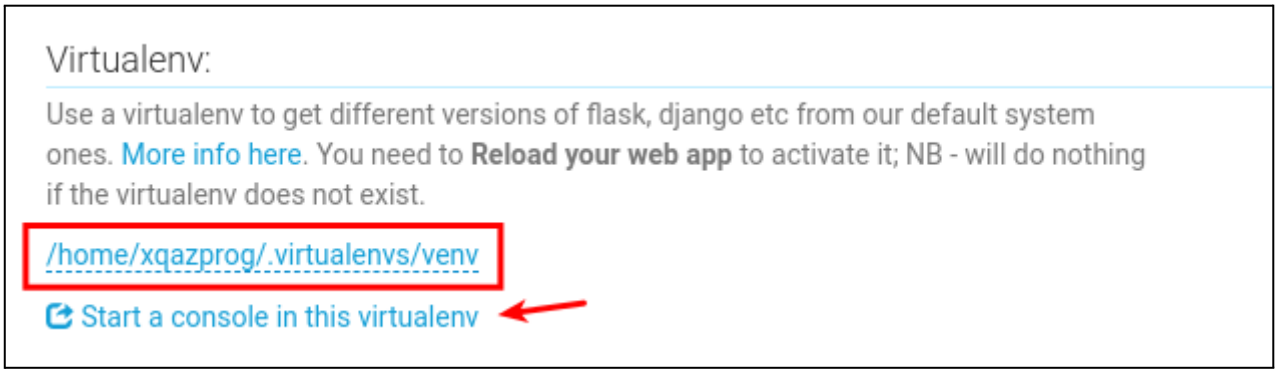




Paso 4: Seleccionar virtualenv

- Bajar hasta la sección `virtualenv`
- Agregar el virtualenv `venv` creado anteriormente
    - `/home/usuario/.virtualenvs/venv`

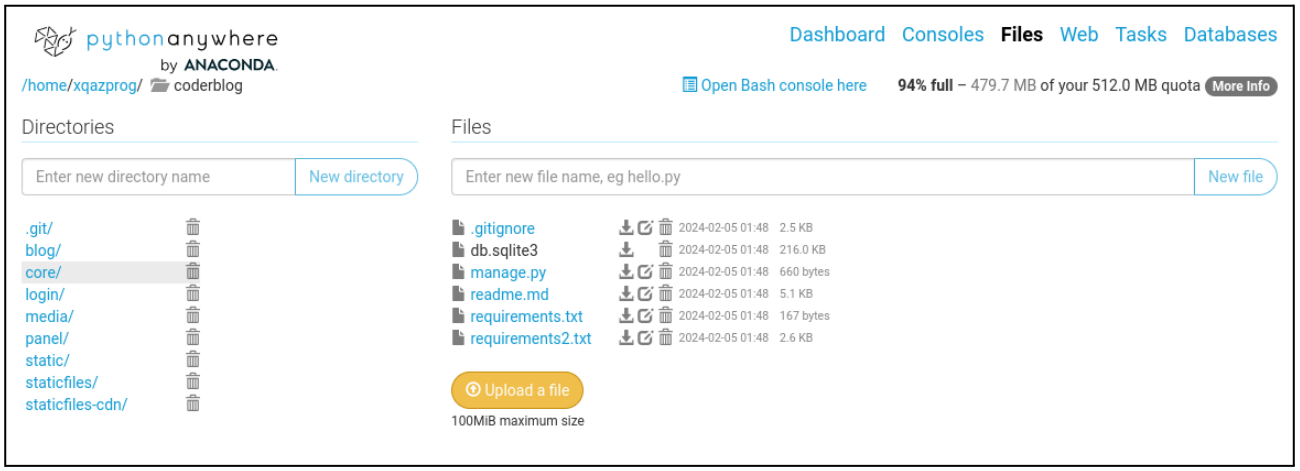




3. Configuración de webapp en servidor

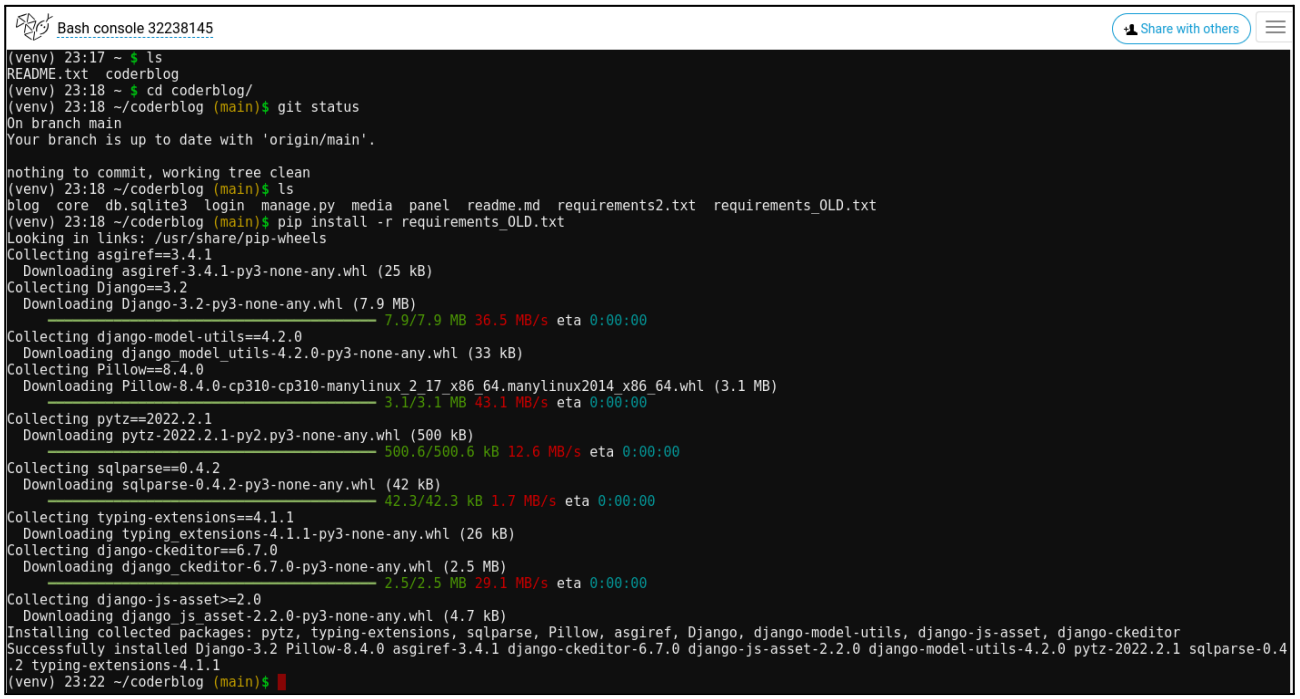
Paso 1: Clonar repositorio privado de github

- `git clone https://[username]:[token]@github.com/[username]/[repo]`
- `git clone https://Gabo-araya:ghp_TOKEN_GENERADO_EN_GITHUB@github.com/Gabo-araya/repositorio`



Paso 2: Instalar los requerimientos

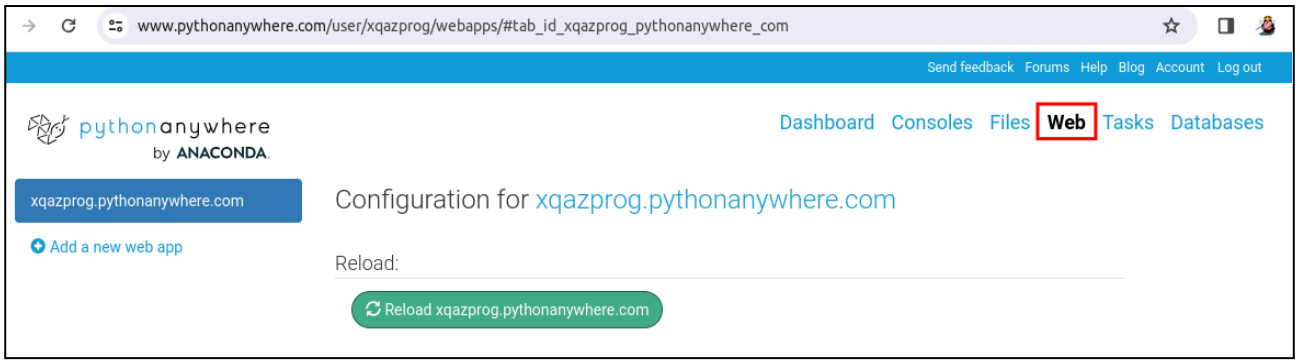
- `pip install -r requirements.txt`



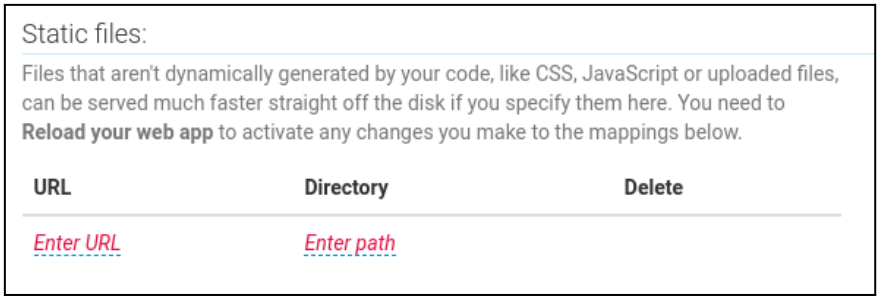
4. Configurar static files

Revisar: How to setup static files in Django:
https://help.pythonanywhere.com/pages/DjangoStaticFiles

Paso 1: Acceder a la pestaña web



Paso 2. Bajar hasta la sección "Static files"

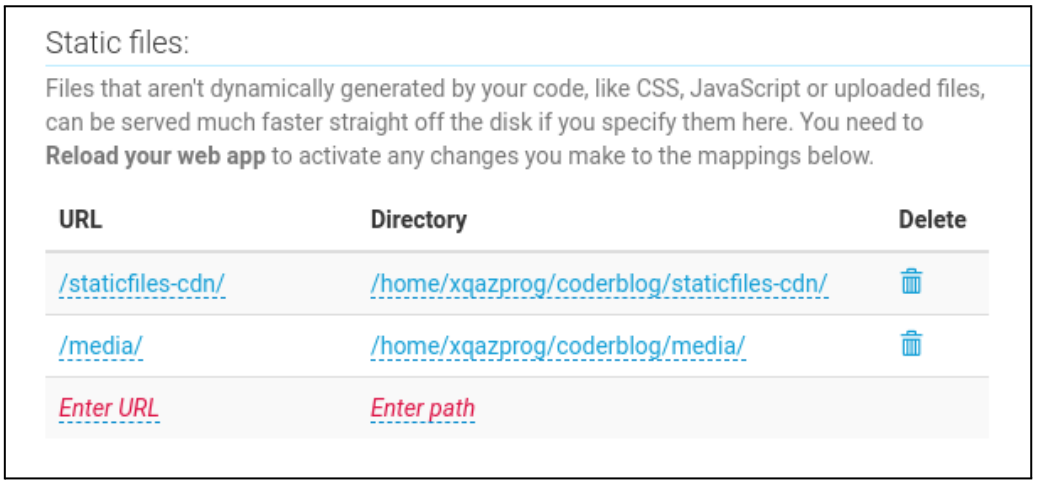


Agregar datos de static

- **URL:** `/staticfiles-cdn/`
- **Directory:** `/home/xqazprog/coderblog/staticfiles-cdn/`

Agregar datos de media

- **URL:** `/media/`
- **Directory:** `/home/xqazprog/coderblog/media/`



Paso 3. Configurar rutas estáticas en settings.py

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/3.2/howto/static-files/

STATIC_URL = '/static/'
STATICFILES_DIRS = [
    BASE_DIR / 'static', #os.path.join(BASE_DIR, 'static')
    os.path.join(BASE_DIR, 'static'),
]

STATIC_ROOT = BASE_DIR / 'staticfiles-cdn'
STATICFILES_STORAGE = 'whitenoise.storage.CompressedManifestStaticFilesStorage'

MEDIA_URL = 'media/'
MEDIA_ROOT = os.path.join(BASE_DIR, 'media/')
```

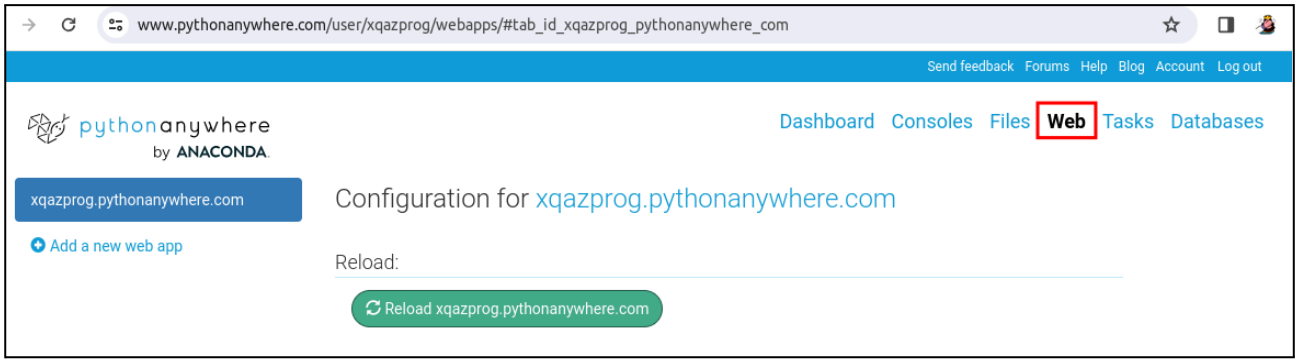
Paso 4. Recolectar estáticos

Luego, en la terminal insertar el comando para recolectar los archivos estáticos

- `python manage.py collectstatic`

5. Modificar WSGI

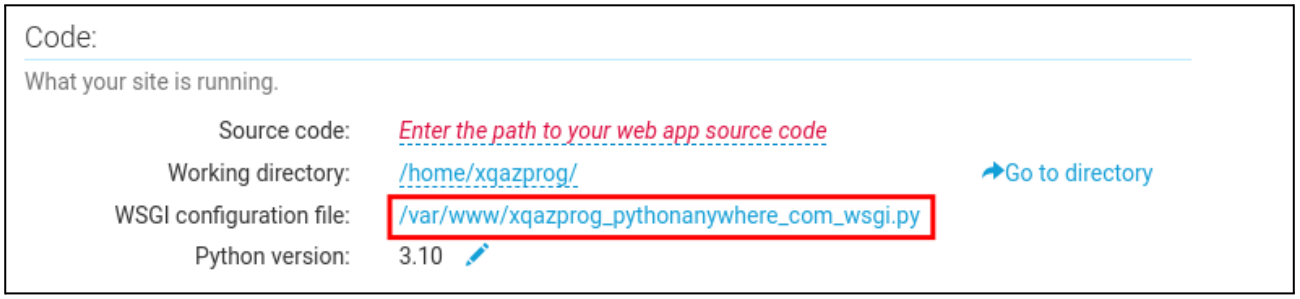
Paso 1. Acceder a la pestaña web



Paso 2. Bajar hasta la sección "Code"

Pinchar en WSGI configuration file para editar:

- `/var/www/proyecto_pythonanywhere_com_wsgi.py`



Paso 3. Editar archivo de configuración wsgi

Bajar hasta la sección de Django

- (Opcional): eliminar el código comentado no usado
- Estructura de `path`
    - `path = /home/[usuario]/[proyecto]/`
- Asegurarse de cambiar DJANGO_SETTINGS_MODULE'
    - `os.environ['DJANGO_SETTINGS_MODULE'] = '[proyecto].settings'`

Descomentar las líneas comentadas hasta conseguir un código similar al siguiente:

```
# ++++++++++ DJANGO ++++++++++
# To use your own django app use code like this:
import os
import sys

## assuming your django settings file is at
'/home/usuario/proyecto/core/settings.py'
## and your manage.py is is at '/home/usuario/proyecto/manage.py'
#path = '/home/[usuario]/[proyecto]

## assuming your django settings file is at
'/home/xqazprog/coderblog/core/settings.py'
## and your manage.py is is at '/home/xqazprog/coderblog/manage.py'
path = '/home/xqazprog/coderblog'
if path not in sys.path:
       sys.path.append(path)

#os.environ['DJANGO_SETTINGS_MODULE'] = '[project].settings'
os.environ['DJANGO_SETTINGS_MODULE'] = 'core.settings'

## then:
from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()
```


www.pythonanywhere.com/user/xqazprog/files/var/www/xqazprog_pythonanywhere_com_wsgi.py?edit
/var/www/xqazprog_pythonanywhere_com_wsgi.py
# This file contains the WSGI configuration required to serve up your
# web application at http://xqazprog.pythonanywhere.com/
# It works by setting the variable 'application' to a WSGI handler of some description.
# ++++++++++ GENERAL DEBUGGING TIPS ++++++++++
# getting imports and sys.path right can be fiddly!
# We've tried to collect some general tips here:
# https://help.pythonanywhere.com/pages/DebuggingImportError
# ++++++++++ DJANGO ++++++++++
# To use your own django app use code like this:
import os
import sys
assuming your django settings file is at '/home/usuario/proyecto/core/settings.py'
and your manage.py is is at '/home/usuario/proyecto/manage.py'
#path = '/home/[usuario]/[proyecto]
assuming your django settings file is at '/home/xqazprog/coderblog/core/settings.py'
and your manage.py is is at '/home/xqazprog/coderblog/manage.py'
path = '/home/xqazprog/coderblog'
if path not in sys.path:
sys.path.append(path)
#os.environ['DJANGO_SETTINGS_MODULE'] = '[project].settings'
os.environ['DJANGO_SETTINGS_MODULE'] = 'core.settings'
then:
from django.core.wsgi import get_wsgi_application
application = get_wsgi_application()

6. Otras configuraciones

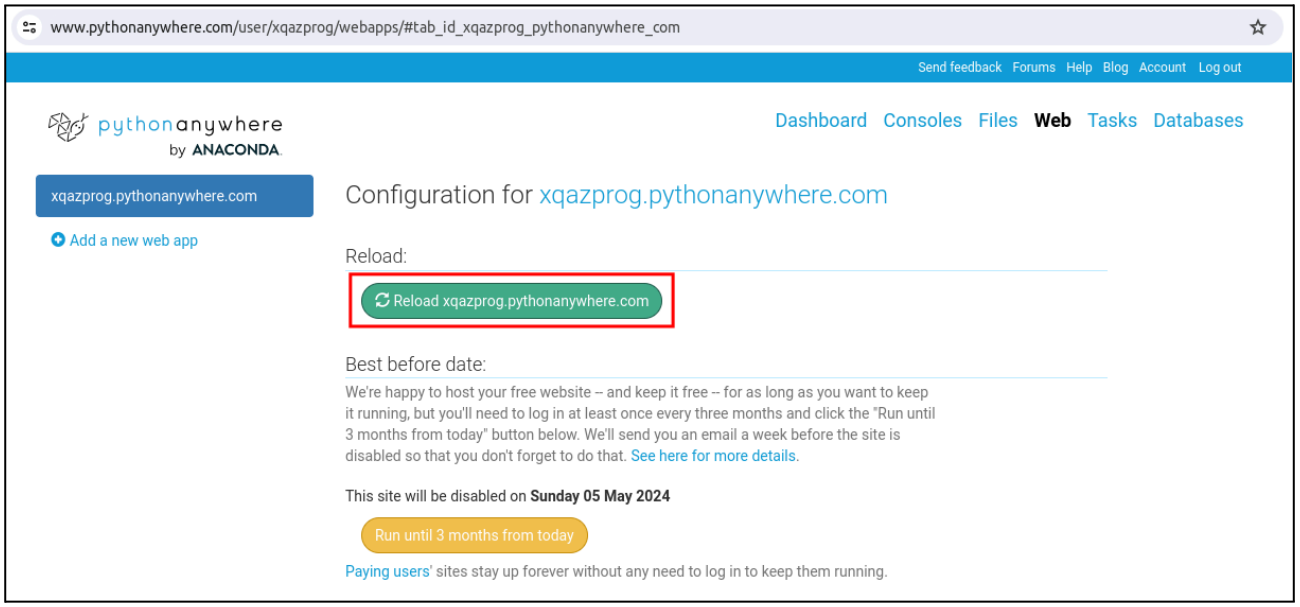
Modificar settings.py

- Agregar "`proyecto.pythonanywhere.com`" dentro de `allowed_hosts`
    - `ALLOWED_HOSTS = ['proyecto.pythonanywhere.com']`
- Cambiar `DEBUG = False` *a* `DEBUG = True`

Recargar web app

- https://www.pythonanywhere.com/user/xqazprog/webapps/
- Presionar botón "**Reload**" xqazprog.pythonanywhere.com
- Finalmente, para acceder al sitio del proyecto:
    - https://xqazprog.pythonanywhere.com/

> **Importante:** Cada vez que se hacen cambios a las configuraciones es necesario recargar la webapp.



7. Otra info

- **Deploying an existing Django project on PythonAnywhere:** https://help.pythonanywhere.com/pages/DeployExistingDjangoProject/
- **Using MySQL:** https://help.pythonanywhere.com/pages/UsingMySQL/
- **How To Migrate Sqlite To MySql Database In Django On PythonAnyWhere:** https://www.youtube.com/watch?v=e8fD9lU6qy4