1 Como pasar programas a pythonanywhere

1.1 Crear un directorio para los repositorios

1. En la pantalla Dashboard o panel de control de pythonanywhere, abrir una consola bash.
2. Ejecutar los siguientes comandos en la consola:
   (a) Para asegurarnos que estamos en el directorio raíz

   ```
   $ cd ~
   ```

   (b) Creamos el directorio Code (código en inglés)

   ```
   $ mkdir Code
   ```

   (c) Entramos en el directorio Code (código en inglés)

   ```
   $ cd Code
   ```

   (d) Vemos el path o camino,

   ```
   $ pwd
   ```

   que debería ser similar a

   ```
   /home/usuario_pythonanywhere/Code/
   ```

1.2 Como clonar un repositorio de Github

1. Obtener la dirección del repositorio (ver figura 1)
2. Dashboard o panel de control de pythonanywhere, abrir una consola bash.
3. Ejecutar los siguientes comandos en la consola:
   (a) Para asegurarnos que estamos en el directorio Code

   ```
   $ cd ~/Code
   ```

   (b) Clonamos el repo

   ```
   $ git clone https://github.com/usuario_github/nombrerepo.git
   ```

   (c) Entramos en el repo

   ```
   $ cd nombrerepo
   ```

   (d) Vemos el path o camino,

   ```
   $ pwd
   ```

   que debería ser similar a

   ```
   $ /home/usuario_pythonanywhere/Code/nombrerepo
   ```

   (e) Si no vemos este camino, podemos solucionarlo con

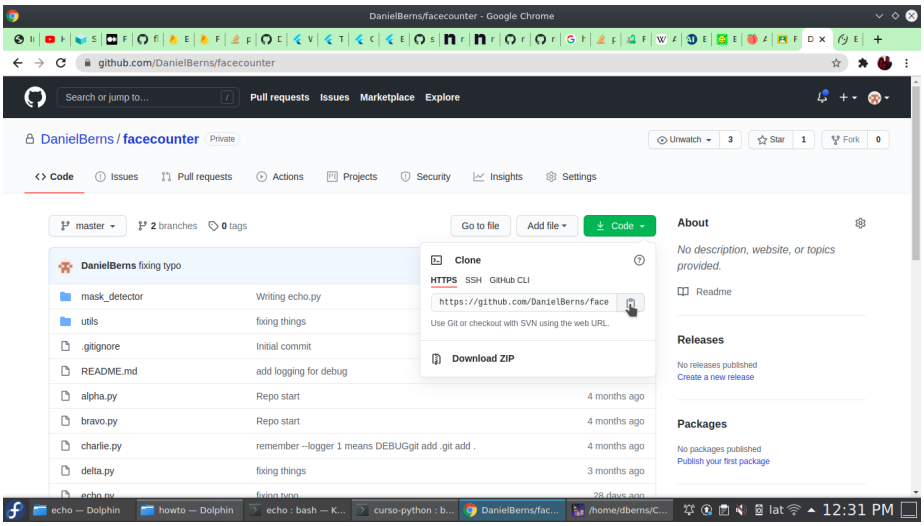

Figura 1: Donde está url repo

```
$ cd ~/Code/nombrerepo
```

(f) Una vez dentro del directorio del repo, podemos ver los archivos con el comando

```
$ ls
```

1.3 Como actualizar los repositorios en pythonanywhere

No nos conviene modificar los archivos de los repositorios en pythonanywhere, porque los editores son un tanto incómodos. Si quiere verificarlo, pruebe

```
$ vi test.txt
```

No pregunte nada sobre este punto en el foro. ¡No recibirá respuesta!

Por lo tanto, nos conviene modificar los archivos en nuestras computadoras, actualizar el repositorio en Github y y después actualizar el repositorio en pythonanywhere.

Si en sus computadoras están trabajando con Windows y Visual Studio pueden manejar la actualización del repositorio en Github con Visual Studio. Esto se explica en otro apunte y video.

También es posible usar la página del repo en Github (ver figura 2). Como vemos, podemos subir archivos directamente al repositorio. Es una forma lenta pero segura y simple si no se cuenta con un cliente de Git.

Finalmente, para actualizar el repositorio en pythonanywhere, en el Dashboard o panel de control de pythonanywhere abrirmos una consola bash y ejecutamos los siguientes comandos:

1. `$ cd ~/Code/nombrerepo`

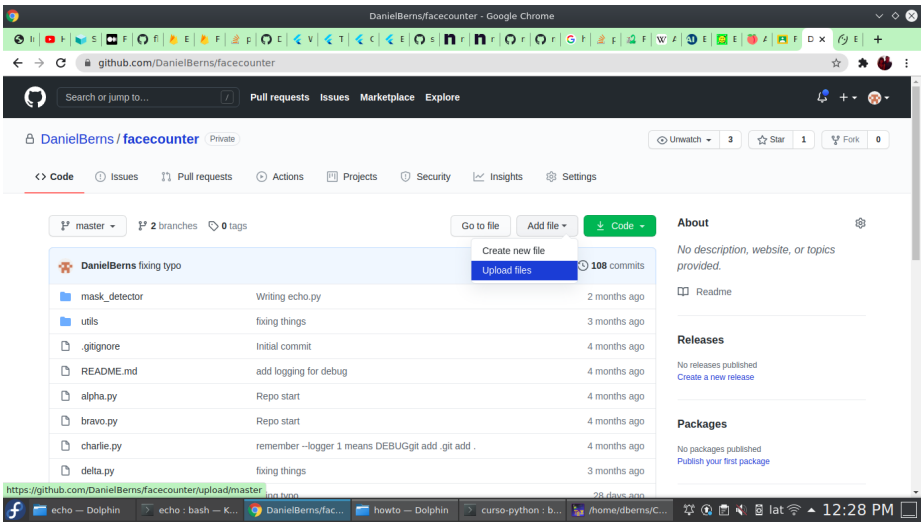

Figura 2: Como subir archivos en la página del repositorio

2. `$ git pull`

2 Como pasar archivos grandes a pythonanywhere

Supongamos que necesitamos transferir a pythonanywhere archivos binarios grandes (fotografías, planillas excel, bases de datos, etc), desde Google Drive o Dropbox. Suponemos también que tenemos los enlace de acceso a dichos archivos.

En el Dashboard o panel de control de pythonanywhere abrirmos una consola bash y ejecutamos el siguiente comando

```
$ wget -O test.pdf link
```

donde debemos reemplazar link por

```
https://drive.google.com/file/d/1HVBDQzbMmDPO5rmGFe65eI7Jlx--_BmZ/view?usp=sharing
```

Si ejecutamos el comando ls, debemos ver como resultado el archivo `test.pdf`.

3 Como extraer archivos desde pythonanywhere

Si ya clonaron el repositorio de este curso, pueden ver que en el directorio

```
curso-python/pythonanywhere/calendar/calendars
```

hay un archivo html y dos directorios con más archivos html generados por el programa `generate_two_hundred_years.py`. ¿Cómo podemos hacer para que estos archivos sean accesibles por Internet?

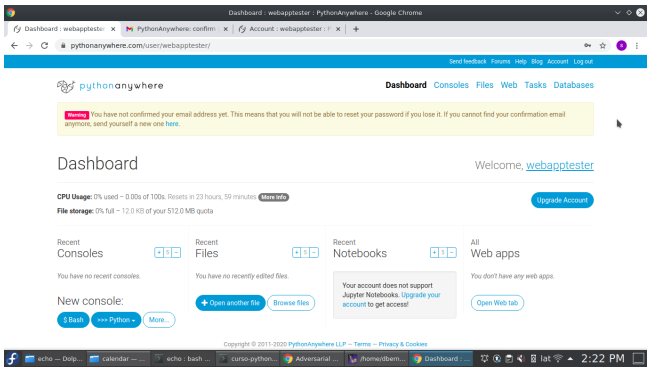

Figura 3: En Dashboard hay un enlace a la pantalla Web

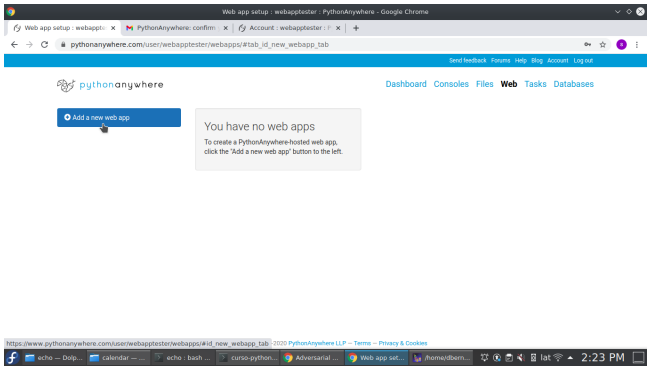

Figura 4: Aquí podemos crear una apliación web.

3.1 Crear una aplicación web en pythonanywhere

En la pantalla Dashboard aparece un enlace con el título Web (figura 3). Si seguimos ese enlace vamos a la pantalla Web donde podemos crear una aplicación web (figura 4).

Por ahora aceptamos el dominio por defecto (figura 5), seleccionamos Flask (figura 6), aceptamos el nombre por defecto del sitio web (figura 7) y la versión de python (figura 8).

En las pantallas (9), (10), (11), (12) vemos distintas partes del panel de control de la aplicación web. Para entender que es lo que estamos haciendo, tenemos que tener en cuenta que un sitio web o aplicación web tiene dos tipos de archivos: dinámicos y estáticos. La aplicación web recibe pedidos en distintas direcciones web o url (universal resource locator), y según los distintos pedidos y diferentes urls contesta con archivos dinámicos (generados con información de una base de datos o resultado de algún cálculo) o con archivos estáticos (cuyo contenido es inalterable).

En el repositorio, en `curso-python/pythonanywhere/calendar`, tenemos un directorio calendars con archivos html conteniendo los calendarios de los años 1900 al 2099. Estos archivos son inalterables (los calendarios no cambian), por lo tanto los vamos a publicar como archivos estáticos aprovechando que pythonanywhere nos permite definir los archivos con los que responde el servidor

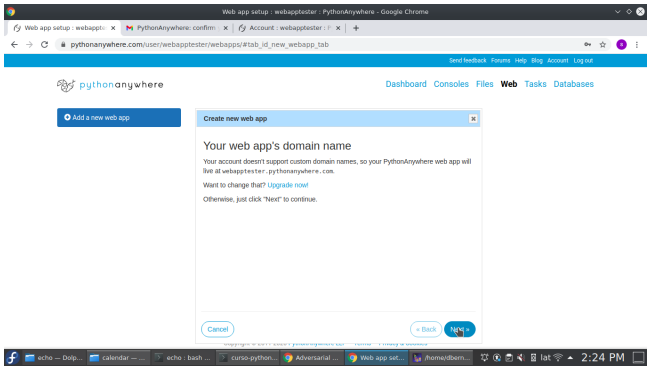

Figura 5: Dominio por defecto

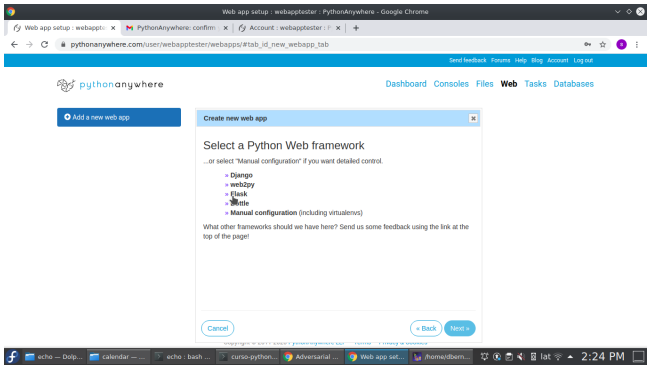

Figura 6: Seleccionamos Flask

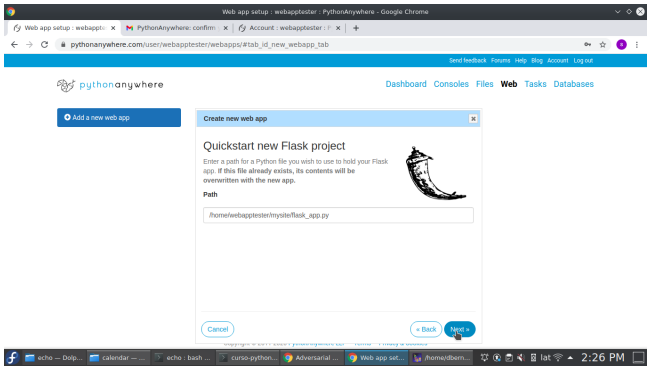

Figura 7: Aceptamos opciones por defecto

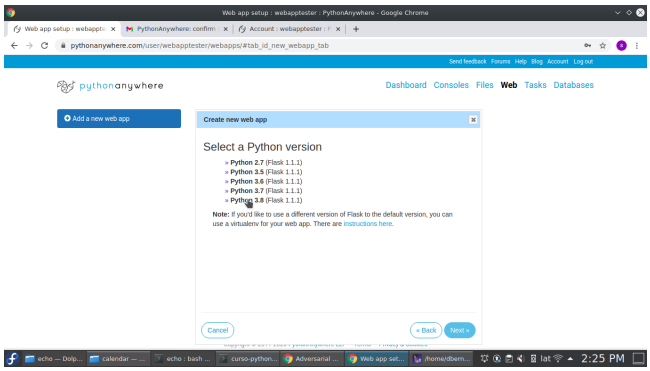

Figura 8: Versión de python

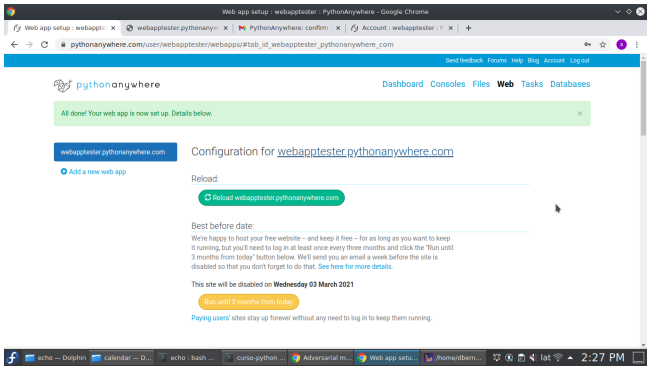

Figura 9: Panel de control de la aplicación web, parte 1

web cuando recibe determinados pedidos en ciertas urls (ver figura 11).

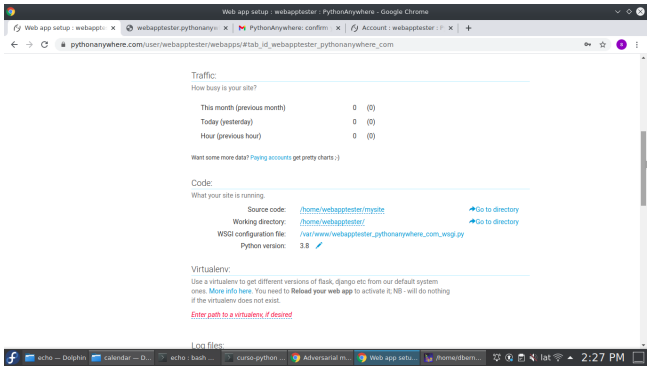

Figura 10: Panel de control de la aplicación web, parte 2

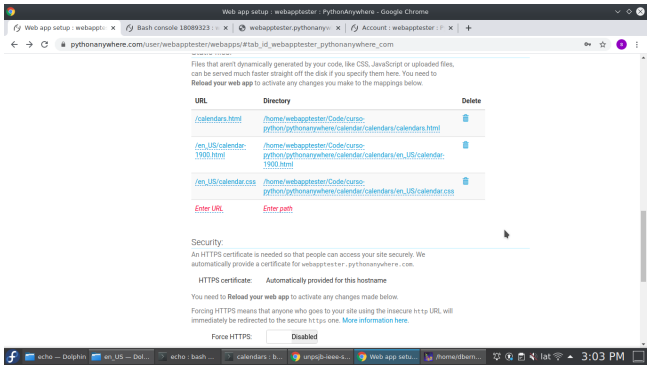

Figura 11: Panel de control de la aplicación web, archivos estáticos

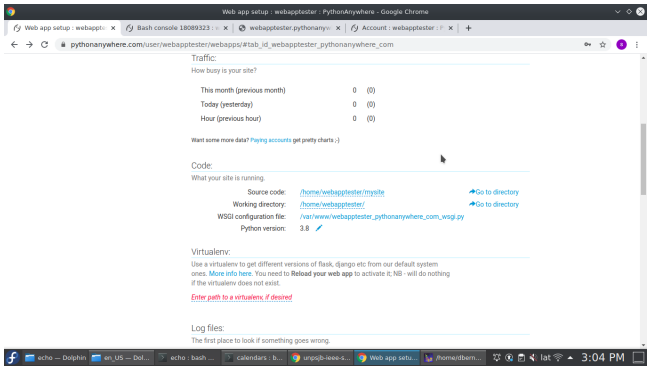

Figura 12: Panel de control de la aplicación web, donde está el código