# Lab 11 – Deployment

In questo laboratorio impareremo come eseguire il deploy del social network sviluppato nei laboratori precedenti nel cloud, consentendo così l'accesso da qualsiasi luogo tramite il web, non solo dal nostro PC. A tal proposito faremo uso di **PythonAnywhere** (https://www.pythonanywhere.com), una piattaforma di sviluppo Python accessibile da qualsiasi dispositivo dotato di browser.

Di seguito sono riportate le istruzioni dettagliate per completare il deploy con successo:

1. Crea un account gratuito sulla piattaforma: https://www.pythonanywhere.com/registration/register/beginner/

2. Comprimi il progetto del laboratorio 10[1] in un file .zip e caricalo tramite la pagina dei file accessibile dalla tua Dashboard personale. Clicca su "**Files**" (Figura 1) e troverai un pulsante "**Upload a file**" che ti consentirà di caricare il file .zip (Figura 2).

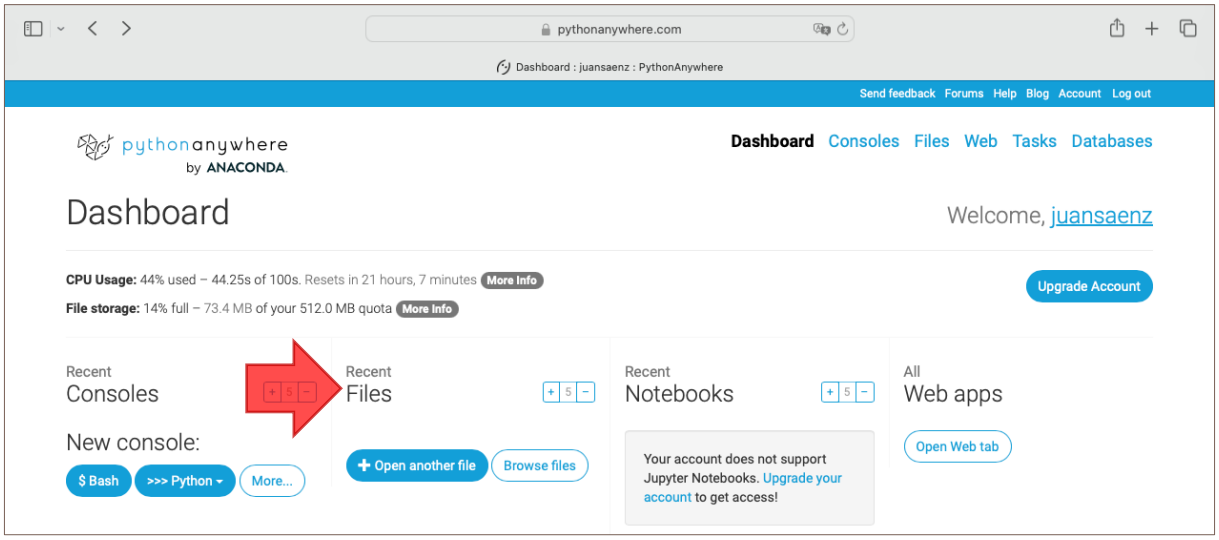



Figura 1 Dashboard PythonAnywhere

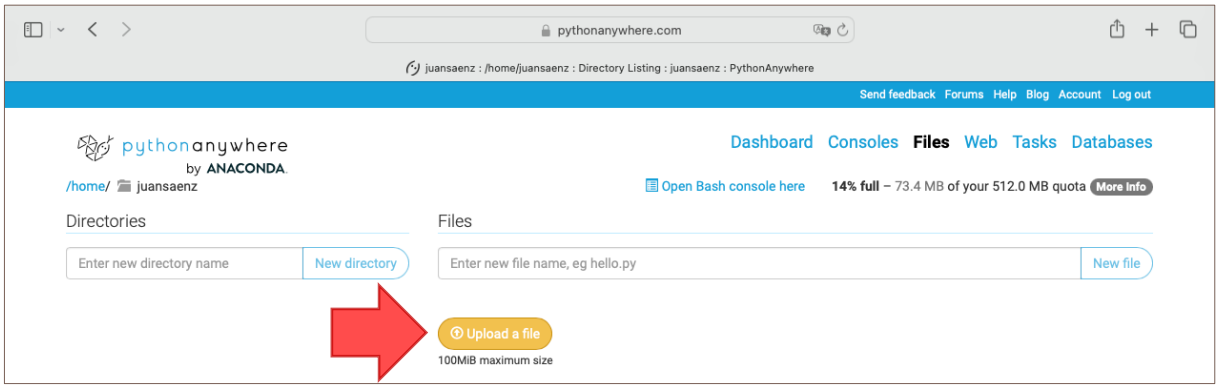



Figura 2 Schermo caricamento file

[1] Se vuoi, puoi usare la soluzione disponibile su GitHub: https://github.com/polito-iaw-2023/materiale/tree/main/laboratori/lab-10/soluzione.

3. Apri una nuova console Bash dalla Dashboard personale (Figura 3), ed esegui il comando "`unzip [nome del file]`" per scompattare il progetto appena caricato.

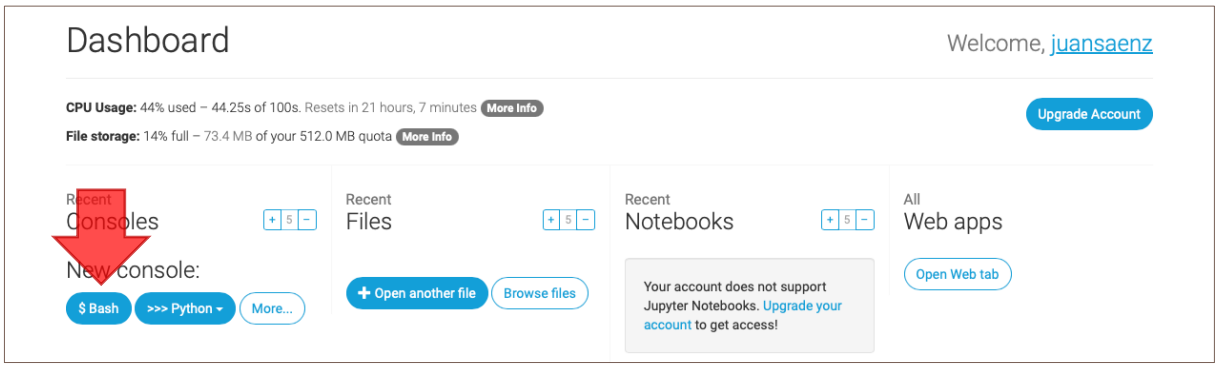



Figura 3 Creare una nuova consola Bash

4. Dalla stessa console, creare un nuovo ambiente virtuale eseguendo i seguenti comandi:

```
mkvirtualenv --python=/usr/bin/python3.10 my-virtualenv
pip install flask
```

5. Il prompt visualizzato sulla console dovrebbe a questo punto passare da `$` a `(my-virtualenv)$`: è così che si capisce che il `virtualenv` è attivo, in maniera analoga a quanto capitava con i virtual environment creati durante il corso.

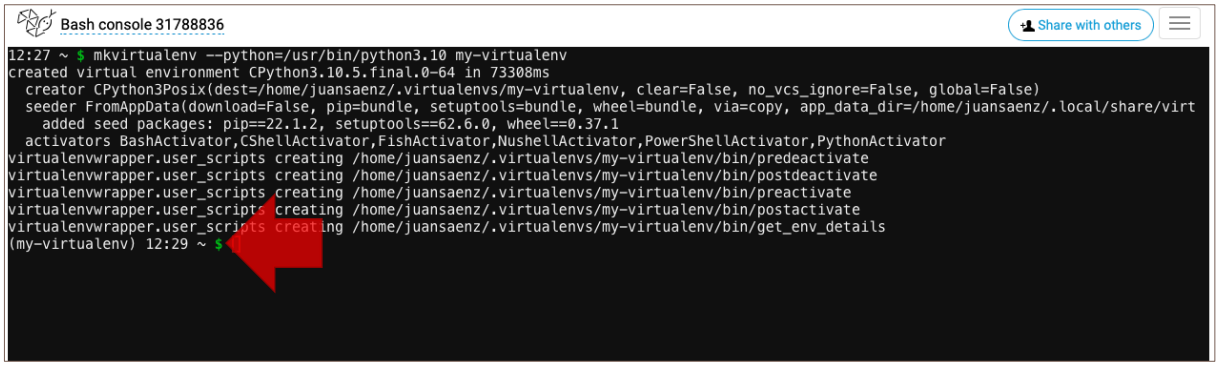



Figura 4 Creazione virtual environment dalla consola

Ogni volta che si vuole lavorare sul progetto nella console, bisogna assicurarsi che il virtualenv sia attivo. È possibile riattivarlo in un secondo momento con:

```
$ workon my-virtualenv
(my-virtualenv)$
```

Tramite la console, installa tutte le dipendenze utilizzate dal tuo social network attraverso il comando `pip`. Tipicamente, `Flask`, `flask-login` e `Pillow`.

6. Dalla dashboard personale, apri la sezione dedicata alle “**Web apps**” (raggiungibile direttamente da questo link: https://www.pythonanywhere.com/web_app_setup).

   Clicca su “**Add a new web app**” e segui i passaggi, avendo cura di selezionare “**Manual configuration**” quando è richiesto di selezionare il framework Python desiderato.

   **NOTA**: Assicurati di scegliere la stessa versione di Python utilizzata nell’ambiente virtuale creato al punto 4 (cioè la 3.10).

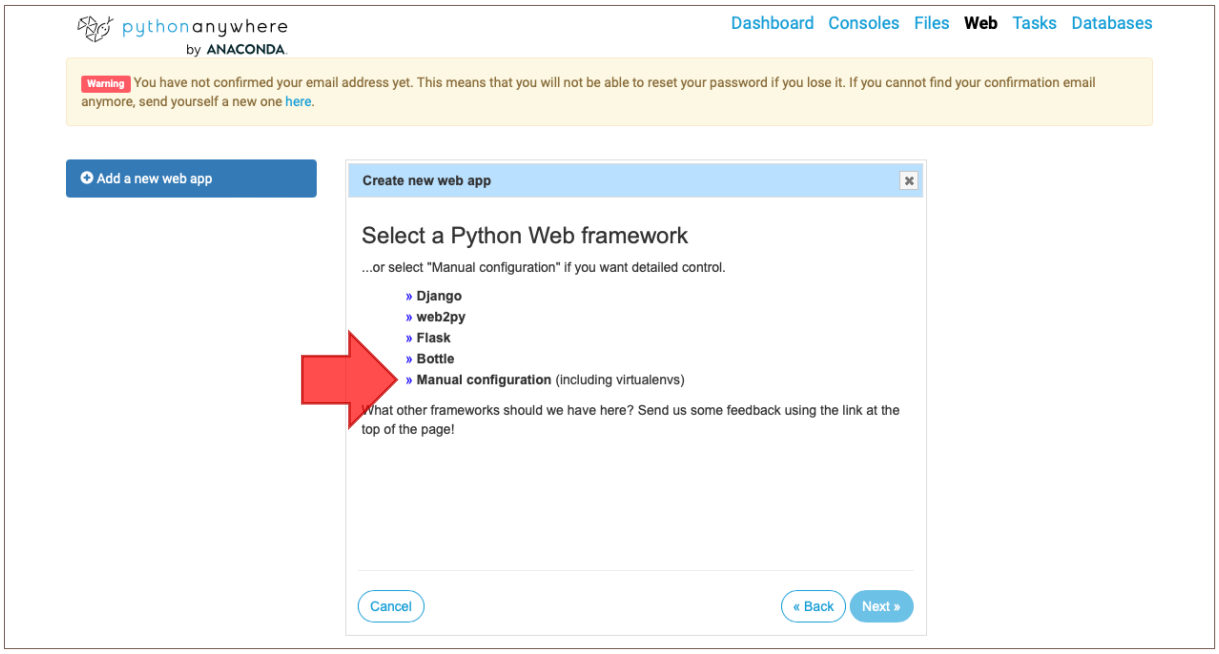



Figura 5 Menu per la creazione di una nuova Web app

7. Nella pagina che si apre, vai alla sezione **Code** e inserisci il path per il codice sorgente caricato sulla piattaforma (esempio: `/home/[username]/mysite`, dove “`mysite`” è il nome della cartella del progetto che è stata compressa.

   Se invece non è presente nessuna cartella “contenitore” che racchiude i file di progetto, il `path` sarà semplicemente `/home/[username]/`).

8. **Nella sezione Virtualenv**, invece, inserisci il nome del virtual environment creato al punto 4 (`my-virtualenv`, se hai seguito fedelmente le istruzioni precedenti).

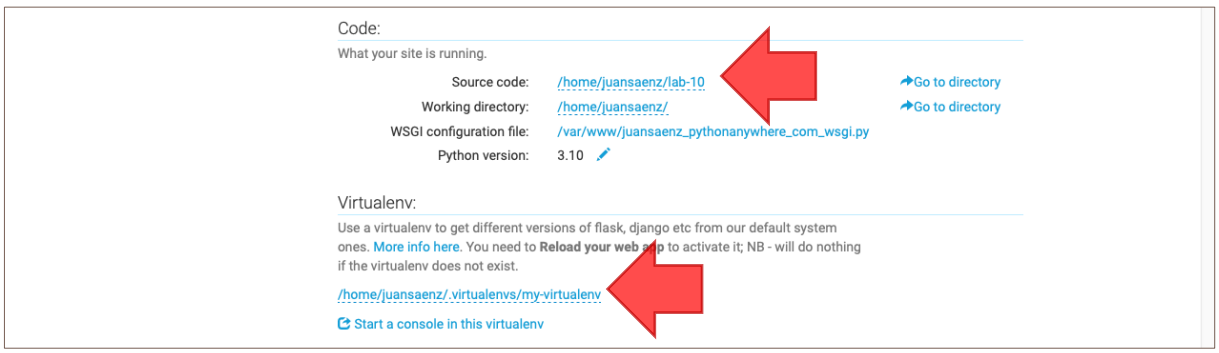
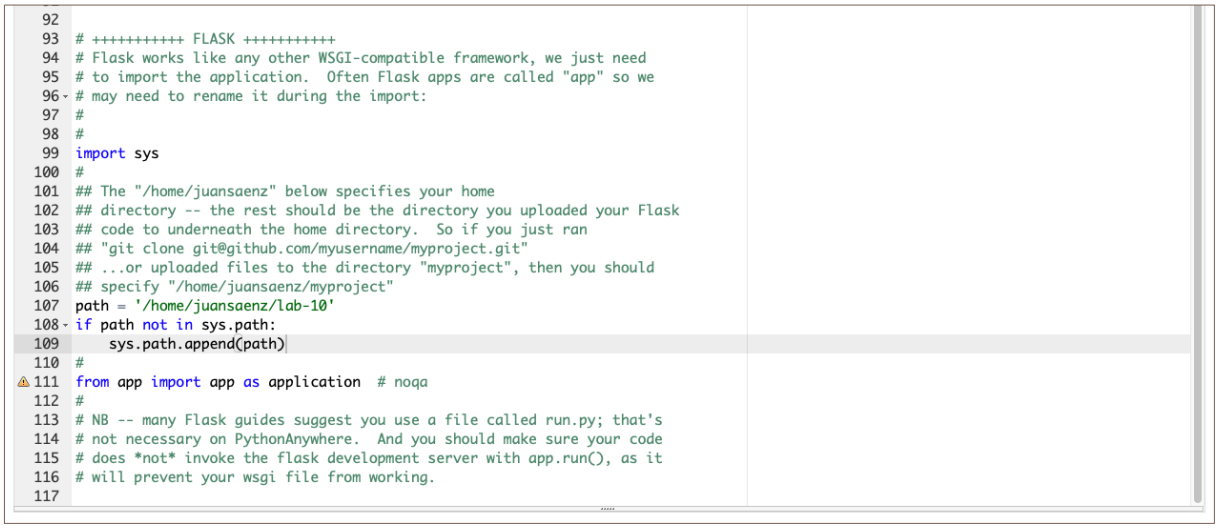



Figura 6 Impostazione del codice e del virtual environment

9. Dalla stessa sezione **Code**, apri il file di configurazione WSGI cliccando sull'apposito link. Nel file WSGI che si apre, vai alla sezione Flask e decommentala, rendendola simile a questa:



```
92
93  # ++++++++++ FLASK ++++++++++
94  # Flask works like any other WSGI-compatible framework, we just need
95  # to import the application.  Often Flask apps are called "app" so we
96  # may need to rename it during the import:
97  #
98  #
99  import sys
100 #
101 ## The "/home/juansaenz" below specifies your home
102 ## directory -- the rest should be the directory you uploaded your Flask
103 ## code to underneath the home directory.  So if you just ran
104 ## "git clone git@github.com/myusername/myproject.git"
105 ## ...or uploaded files to the directory "myproject", then you should
106 ## specify "/home/juansaenz/myproject"
107 path = '/home/juansaenz/lab-10'
108 if path not in sys.path:
109     sys.path.append(path)
110 #
111 from app import app as application  # noqa
112 #
113 # NB -- many Flask guides suggest you use a file called run.py; that's
114 # not necessary on PythonAnywhere.  And you should make sure your code
115 # does *not* invoke the flask development server with app.run(), as it
116 # will prevent your wsgi file from working.
117
```

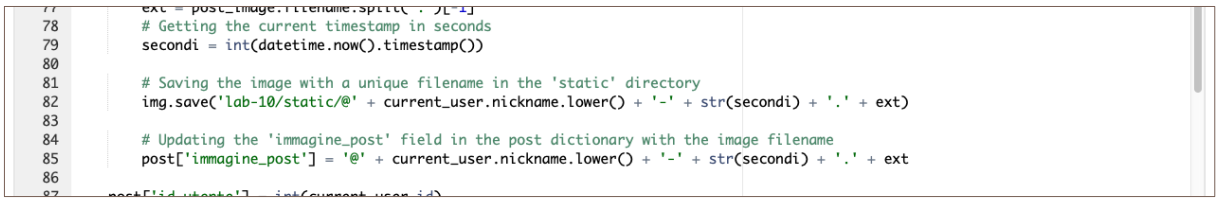
Figura 7 File di configurazione WSGI

**NOTA**: Se lanci il tuo progetto con "`app.run`" invece che con "flask run", rimuovi la riga corrispondente dal file principale.
Su PythonAnywhere, infatti, l'applicazione viene eseguita tramite il file di configurazione WSGI. Se lasci l'istruzione `app.run()` ed questa è al di fuori della clausula `if __name__ == '__main__':`, l'applicazione andrà in crash e sul sito verrà visualizzato un errore 504 o 502.

10. Poiché il progetto ora viene eseguito da un percorso diverso, devi aggiornare i percorsi della cartella static e del database all'interno del tuo file `app.py` e dei file DAO in modo che siano accessibili correttamente. Per farlo, torna alla pagina dedicata alle "**Web apps**" e nella sezione Code apri la source code cliccando su "**Go to directory**".

11. Modifica il percorso all'interno dei file aggiungendo "`mysite/`" all'inizio, dove "`mysite`" rappresenta sempre la cartella principale del progetto.

    Ad esempio, con la soluzione del laboratorio 10, nelle funzioni responsabili della gestione delle immagini, è stato inserito il prefisso "`lab-10/`" come mostrato nella Figura 8 (riga 82). Allo stesso modo, nei file DAO, è stato necessario adattare il percorso al database, poiché il file SQLite era situato nella cartella "`db`", come mostrato in Figura 9 (riga 20).



```
78          # Getting the current timestamp in seconds
79          secondi = int(datetime.now().timestamp())
80
81          # Saving the image with a unique filename in the 'static' directory
82          img.save('lab-10/static/@' + current_user.nickname.lower() + '-' + str(secondi) + '.' + ext)
83
84          # Updating the 'immagine_post' field in the post dictionary with the image filename
85          post['immagine_post'] = '@' + current_user.nickname.lower() + '-' + str(secondi) + '.' + ext
86
```

Figura 8 Esempio di modifica del percorso della cartella static (app.py)

```
18
19  def get_post(id):
20      conn = sqlite3.connect('lab-10/db/social_network.db')
21      conn.row_factory = sqlite3.Row
22      cursor = conn.cursor()
23
24      sql = 'SELECT posts.id, posts.data_pubblicazione, posts.testo, posts.immagine_post, posts.id_utente, utenti.nickname, utenti.immagine
25      cursor.execute(sql, (id,))
26      post = cursor.fetchone()
27
```

Figura 9 Esempio di modifica del percorso della cartella db. (posts_dao.py)

12. Se tutti i passaggi sono stati eseguiti correttamente, il social network è ora accessibile all'indirizzo `http://[username].pythonanywhere.com/`, ad esempio http://juansaenz.pythonanywhere.com.

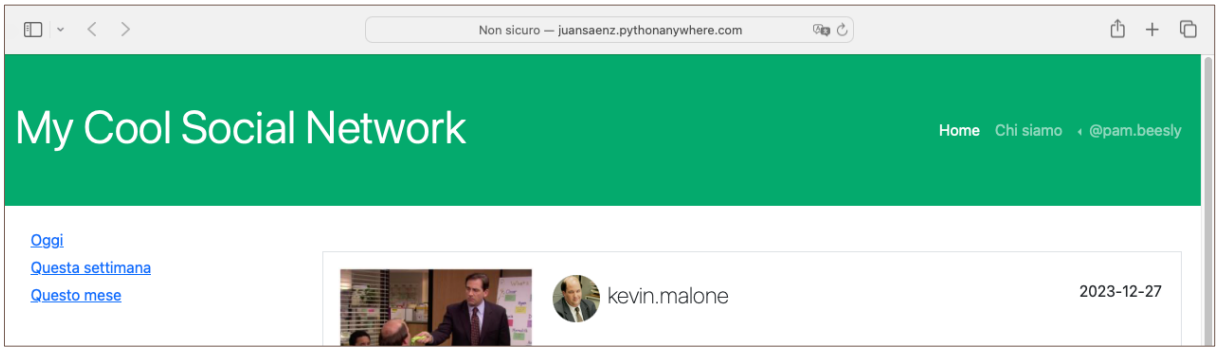



Figura 10 Deploy riuscito del social network su PythonAnywhere

**NOTE:**

1. Se al momento dell'accesso al sito ci fossero errori, dovresti consultare il registro degli errori (**Error log**) accessibile dalla pagina dedicata all'applicazione, nella sezione **Log files**, come indicato nella Figura 11.


Figura 11 Registro degli errori

2. Allo stesso modo, se apporti modifiche al codice, devi salvarlo e poi fare clic sul pulsante per ricaricare l'applicazione. È consigliabile attendere alcuni secondi tra il completamento del ricaricamento e l'aggiornamento del sito per vedere le modifiche riflesse.

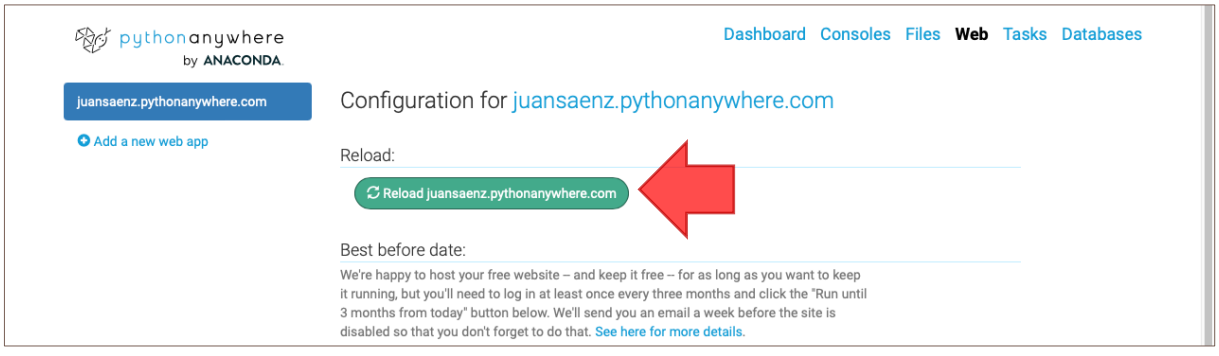



Figura 12 Ricaricare l'applicazione

3. In caso di dubbi o problemi, chiedi sempre ai docenti in laboratorio.