**PROJECT**

**PYTHON FUNDAMENTALS**

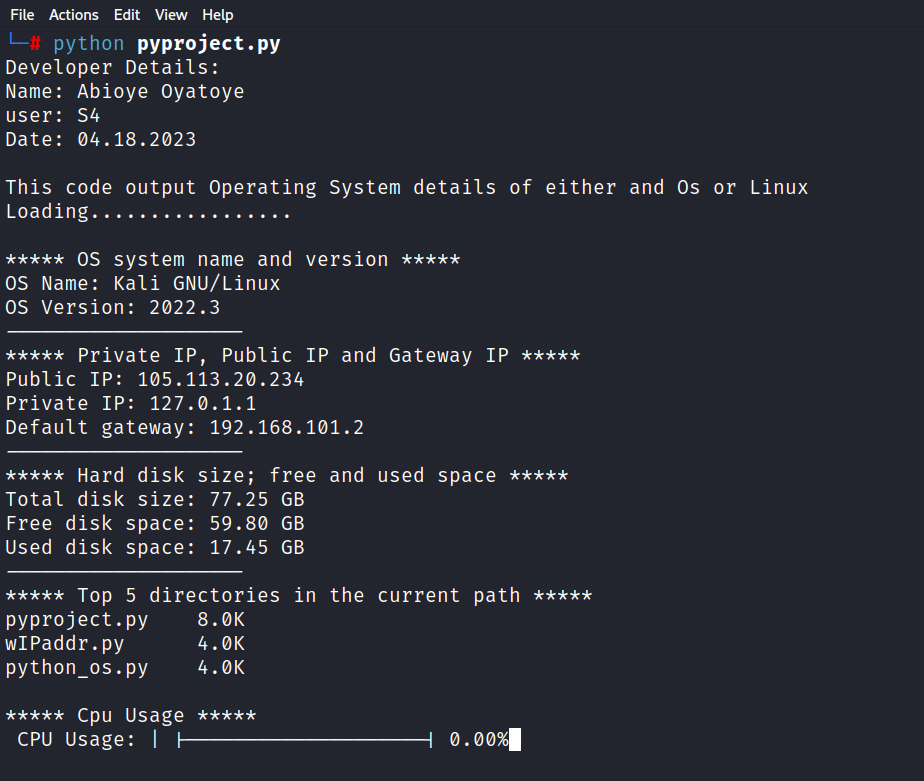
**OPERATING SYSTEM INFORMATION SCRIPT FOR BOTH LINUX AND WINDOWS SUMMARY OF THE SCRIPT DEVELOPMENT**

1. **I** splitted the codes into two different section using conditions for the script to automatically which operating system its being used on.
2. To only import libraries that are needed for the specific operating system.
3. Finally fit in each code to work seamlessly with either of the operating system without hassles.

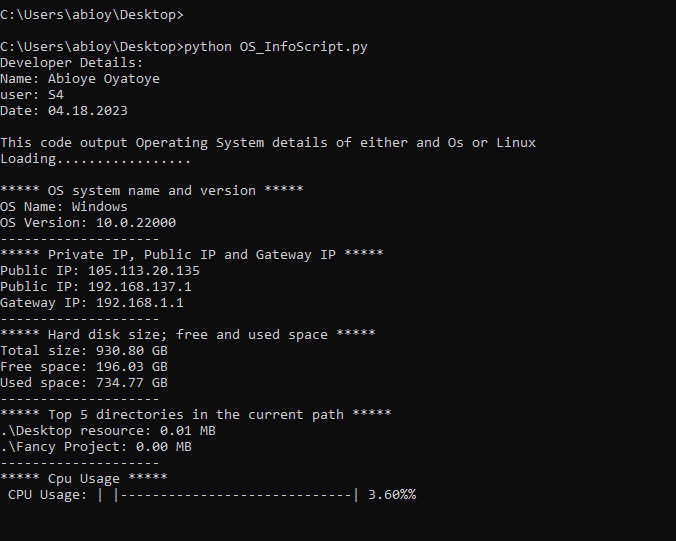
**Note: The code is properly commented to show what each code is achieving at any time!!!**

**Note: while submitting I have change the script name to 🡺 OS\_InfoScript.py**

1. **OUTPUT FOR LINUX**

****

1. **OUTPUT WHEN USED ON WINDOWS**

****

The python script I developed for this project is below:

#!/bin/python3

print('''Developer Details:

Name: Abioye Oyatoye

user: S4

Date: 04.18.2023

This code output Operating System details of either and Os or Linux''')

## need time module here!!!

import time

print("Loading.................\n")

time.sleep(5)

## Library work in both window and linux

import platform

## Code for Windows OS....................

if platform.system() == "Windows":

## import all neede libraries

import urllib.request as request

import socket

import subprocess

import shutil

import psutil

import os

#1. OS system name and version

# Get the operating system name

print("\*"\*5,"OS system name and version","\*"\*5)

os\_name = platform.system()

print('OS Name:', os\_name)

# Get the operating system version

os\_version = platform.version()

print('OS Version:', os\_version)

print("-" \* 20)

time.sleep(3)

#2. Display the private IP address, public IP address, and

#the default gateway

print("\*"\*5,"Private IP, Public IP and Gateway IP","\*"\*5)

try:

public\_ip = request.urlopen('https://ident.me').read().decode('utf8')

print('Public IP:', public\_ip)

except:

print("Unable to resolve the Public IP")

try:

private\_ip = socket.gethostbyname(socket.gethostname())

print('Public IP:', private\_ip)

except:

print("Unable to resolve the Public IP")

# Get the default gateway IP address using ipconfig command

result = subprocess.run('ipconfig', capture\_output=True, text=True)

ipconfig\_output = result.stdout.split('\n')

for line in ipconfig\_output:

#print(line)

if 'Default Gateway' in line:

gateway\_ip = line.split(': ')[-1].strip()

#print(type(gateway\_ip))

if type(gateway\_ip) == str:

#print(gateway\_ip)

continue

if gateway\_ip is not None:

print('Gateway IP:', gateway\_ip)

print("-" \* 20)

time.sleep(3)

else:

print('Unable to get gateway IP')

print("-" \* 20)

time.sleep(3)

#3. Display the hard disk size; free and used space

# Get information about the primary disk

print("\*"\*5,"Hard disk size; free and used space","\*"\*5)

total, used, free = shutil.disk\_usage('/')

# Display the size, free space, and used space

print(f'Total size: {total / (1024\*\*3):.2f} GB')

print(f'Free space: {free / (1024\*\*3):.2f} GB')

print(f'Used space: {used / (1024\*\*3):.2f} GB')

print("-" \* 20)

time.sleep(3)

#4. Display the top five (5) directories and their size

# Get the sizes of all directories in the current directory

print("\*"\*5,"Top 5 directories in the current path","\*"\*5)

dir\_sizes = [(os.path.getsize(entry.path), entry.path)

for entry in os.scandir('.')

if entry.is\_dir()]

#print(dir\_sizes)

# Sort the directories by size in descending order

dir\_sizes.sort(reverse=True)

#print(dir\_sizes)

# Display the top five directories and their size

for size, path in dir\_sizes[:5]:

print(f'{path}: {size / (1024\*\*2):.2f} MB')

print("-" \* 20)

time.sleep(3)

#5. Display the CPU usage; refresh every 10 seconds

print("\*"\*5,"Cpu Usage","\*"\*5)

def display\_usage(cpu\_usage,bars=50):

cpu\_percent = (cpu\_usage/100.0)

cpu\_bar = "▌" \* int(cpu\_percent) + "-" \* (bars - int(cpu\_percent \* bars))

print(f"\r CPU Usage: | |{cpu\_bar}| {cpu\_usage:.2f}%", end="")

while True:

display\_usage(psutil.cpu\_percent(),30)

time.sleep(0.5)

#################### THIS WORKS FOR LINUX ############################

elif platform.system() == "Linux":

### libraries to import

import distro

import urllib.request as request

import socket

import netifaces

import os

#1. OS system name and version

# Get the operating system name

print("\*"\*5,"OS system name and version","\*"\*5)

print(f"OS Name: {distro.name()}")

print(f"OS Version: {distro.version()}")

print("-" \* 20)

time.sleep(3)

#2. Display the private IP address, public IP address, and

#the default gateway

print("\*"\*5,"Private IP, Public IP and Gateway IP","\*"\*5)

try:

public\_ip = request.urlopen('https://ident.me').read().decode('utf8')

print('Public IP:', public\_ip)

except:

print("Unable to resolve the Public IP")

try:

private\_ip = socket.gethostbyname(socket.gethostname())

print('Private IP:', private\_ip)

except:

print("Unable to determine Public IP Address")

try:

gateways = netifaces.gateways()

default\_gateway = gateways['default'][netifaces.AF\_INET][0]

print(f'Default gateway: {default\_gateway}')

print("-" \* 20)

time.sleep(3)

except:

print("Unable to determine Default Gateway")

print("-" \* 20)

time.sleep(3)

#3. Display the hard disk size; free and used space

# Get information about the primary disk

print("\*"\*5,"Hard disk size; free and used space","\*"\*5)

# Get disk usage statistics

st = os.statvfs('/')

# Calculate disk size, free space, and used space

total\_space = st.f\_frsize \* st.f\_blocks

free\_space = st.f\_frsize \* st.f\_bfree

used\_space = total\_space - free\_space

# Print disk size, free space, and used space

print(f"Total disk size: {total\_space / (1024\*1024\*1024):.2f} GB")

print(f"Free disk space: {free\_space / (1024\*1024\*1024):.2f} GB")

print(f"Used disk space: {used\_space / (1024\*1024\*1024):.2f} GB")

print("-" \* 20)

time.sleep(3)

#4. Display the top five (5) directories and their size

# Get the sizes of all directories in the current directory

print("\*"\*5,"Top 5 directories in the current path","\*"\*5)

dir\_sizes = os.popen("du -h \*| sort -rh | head -n 5").read().split('\n')

for item in dir\_sizes:

if item:

size, path = item.split()

print(f"{path}\t{size}")

print()

#5. Display the CPU usage; refresh every 10 seconds

print("\*"\*5,"Cpu Usage","\*"\*5)

def display\_usage(cpu\_usage,bars=50):

cpu\_percent = (cpu\_usage/100.0)

cpu\_bar = "▌▌" \* int(cpu\_percent) + "-" \* (bars - int(cpu\_percent \* bars))

print(f"\r CPU Usage: | |{cpu\_bar}| {cpu\_usage:.2f}%", end="", flush=True)

while True:

cpu\_usage = float(os.popen("top -bn1 | awk '/^%Cpu/{print $2}'").read().strip())

display\_usage(cpu\_usage, 20)

time.sleep(0.5)

#print(f"\rCPU Usage: {cpu\_usage}%", end = "", flush=True)