import requests

import datetime

import time

API\_BASE\_URL = "https://your-miner-api.com/api"

MINER\_IDS = [1, 2, 3] # Replace with actual miner IDs

def set\_miner\_mode(miner\_id, mode):

endpoint = f"{API\_BASE\_URL}/miners/{miner\_id}/mode"

data = {"mode": mode}

response = requests.put(endpoint, json=data)

if response.status\_code == 200:

print(f"Miner {miner\_id} set to {mode} mode successfully.")

return True

else:

print(f"Failed to set mode for Miner {miner\_id}. Status code: {response.status\_code}")

return False

def get\_current\_time():

return datetime.datetime.now().time()

def schedule\_miner\_modes():

while True:

current\_time = get\_current\_time()

if current\_time >= datetime.time(0, 0) and current\_time < datetime.time(6, 0):

mode = "overclocked"

elif current\_time >= datetime.time(6, 0) and current\_time < datetime.time(12, 0):

mode = "normal"

elif current\_time >= datetime.time(12, 0) and current\_time < datetime.time(18, 0):

mode = "underclocked"

else:

mode = "curtailed"

for miner\_id in MINER\_IDS:

set\_miner\_mode(miner\_id, mode)

time.sleep(60) # Check and update every minute

if \_\_name\_\_ == "\_\_main\_\_":

schedule\_miner\_modes()

Updated code

import requests

import datetime

import time

API\_BASE\_URL = "http://localhost:5000/api" # Replace with actual API base URL

MINER\_IDS = [1, 2, 3] # Replace with actual miner IDs

def login(miner\_ip):

endpoint = f"{API\_BASE\_URL}/login"

data = {"miner\_ip": miner\_ip}

response = requests.post(endpoint, json=data)

if response.status\_code == 200:

print(f"Login successful for miner at {miner\_ip}.")

return True

else:

print(f"Login failed for miner at {miner\_ip}. Status code: {response.status\_code}")

return False

def logout(miner\_ip):

endpoint = f"{API\_BASE\_URL}/logout"

data = {"miner\_ip": miner\_ip}

response = requests.post(endpoint, json=data)

if response.status\_code == 200:

print(f"Logout successful for miner at {miner\_ip}.")

return True

else:

print(f"Logout failed for miner at {miner\_ip}. Status code: {response.status\_code}")

return False

def curtail\_miner(token, mode):

endpoint = f"{API\_BASE\_URL}/curtail"

data = {"token": token, "mode": mode}

response = requests.post(endpoint, json=data)

if response.status\_code == 200:

print(f"Curtail request successful. Mode: {mode}")

return True

else:

print(f"Curtail request failed. Status code: {response.status\_code}")

return False

def set\_profile(token, profile):

endpoint = f"{API\_BASE\_URL}/profileset"

data = {"token": token, "profile": profile}

response = requests.post(endpoint, json=data)

if response.status\_code == 200:

print(f"Profile set successfully. Profile: {profile}")

return True

else:

print(f"Failed to set profile. Status code: {response.status\_code}")

return False

def set\_miner\_mode(miner\_id, mode):

endpoint = f"{API\_BASE\_URL}/miners/{miner\_id}/mode"

data = {"mode": mode}

response = requests.put(endpoint, json=data)

if response.status\_code == 200:

print(f"Miner {miner\_id} set to {mode} mode successfully.")

return True

else:

print(f"Failed to set mode for Miner {miner\_id}. Status code: {response.status\_code}")

return False

def get\_current\_time():

return datetime.datetime.now().time()

def schedule\_miner\_modes():

while True:

current\_time = get\_current\_time()

if current\_time >= datetime.time(0, 0) and current\_time < datetime.time(6, 0):

mode = "overclocked"

elif current\_time >= datetime.time(6, 0) and current\_time < datetime.time(12, 0):

mode = "normal"

elif current\_time >= datetime.time(12, 0) and current\_time < datetime.time(18, 0):

mode = "underclocked"

else:

mode = "curtailed"

for miner\_id in MINER\_IDS:

set\_miner\_mode(miner\_id, mode)

time.sleep(60) # Check and update every minute

if \_\_name\_\_ == "\_\_main\_\_":

# Example usage

login("miner\_ip\_address")

schedule\_miner\_modes()

# You may add logout or other cleanup steps as needed