from luno\_python.client import Client

we use luno library to all apis of Luno SDK.

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

from selenium.webdriver.chrome.service import Service

from webdriver\_manager.chrome import ChromeDriverManager

from selenium.webdriver.chrome.options import Options

we use selenium to get exchange rate for Euro to RAND from XE site.

saAccount = Client(api\_key\_id='bdc6udywrcxdy',

               api\_key\_secret='6QhflvPxdqrhRDo1VU-qw3sdZXwCLlTCOIGBWyOkfeY')

itAccount = Client(api\_key\_id='fngu2pkxv37wu',

               api\_key\_secret='Yc40rf1DBmXMP9GZ\_6orCEila7-iYCCwq2Ffv44bzzE')

saAccount is luno library that handles api to SouthAfrica account. (for RAND)

itAccoutn is luno library that handles api to Italy(UK) account. (for Euro)

saEmail = "Amasallia2012@gmail.com"

itEmail = "mattmeabtc@gmail.com"

saEmail is for SouthAfrica account email to send and receive BTC from Italy account

itEmail is for UK account email to send and receive BTC from SouthAfrica account

itTradeFee = 0.0001

saTradeFee = 0.0001

it’s useless for now

##### Pre test for accounts are ready #########

    res = saAccount.get\_balances()

    saBTC = ''

    saZAR = ''

    if res['balance']:

        for bal in res['balance']:

            if bal['asset'] == 'XBT':

                saBTC = bal['account\_id']

            elif bal['asset'] == 'ZAR':

                saZAR = bal['account\_id']

    res = itAccount.get\_balances()

    itBTC = ''

    itEuro = ''

    if res['balance']:

        for bal in res['balance']:

            if bal['asset'] == 'XBT':

                itBTC = bal['account\_id']

            elif bal['asset'] == 'EUR':

                itEuro = bal['account\_id']

    print(saBTC, saZAR, itBTC, itEuro)

    #######################

This is pre-test for SouthAfrica account and UK account have BTC and Euro and RAND balance so we can start to trading.

And the main logics start. The main logic is to looping every 5 mins.

 print("###Trading Loop Begin####", loopCount)

            loopCount+=1

            ## get exchange rate between zar and euro

            '''

            response = requests.get("https://xecdapi.xe.com/v1/convert\_from/?from=EUR&to=ZAR&amount=1", auth=HTTPBasicAuth('liming419944535', 'qajfi3hr0ug3g71ulc3n25ben8'))

            xrate = 0

            if response.status\_code == 200:

                xrateJson = response.json()

                xrate = float(xrateJson['to'][0]['mid'])

                print(xrate)

            else:

                print("Can't fetch exchange rate, wait until get exact exchange rate ....")

                continue

            '''

            driver.get("https://www.xe.com/currencyconverter/convert/?Amount=1&From=EUR&To=ZAR")

            value = driver.find\_element(By.CLASS\_NAME, 'iGrAod')

            price = value.text.replace("South African Rand", "")

            xrate = float(price)

            print("Exchange EURO-ZAR Rate:", xrate)

This is where to check the Exchange rate between Euro and RAND.

res = saAccount.get\_ticker(pair='XBTZAR')

            \_br = float(res['last\_trade']) # BTC to ZAR

            res = itAccount.get\_ticker(pair='XBTEUR')

            \_be = float(res['last\_trade']) # BTC to EURO

            print("BTC to ZAR:", \_br, "BTC to EURO:", \_be)

            arbitrageRate = (\_br - \_be \* xrate) / (\_be \* xrate) \* 100.0

            print( "arbitrageRate", arbitrageRate)

This is where to get the rate of BTC-Euro and BTC-RAND and calculate the arbitrage rate

## when arbitrage rate is below than 1%, then send BTC to italy

            if arbitrageRate <= 1:

                res = saAccount.get\_balances(assets='ZAR')

                saZarBalance = float(res["balance"][0]["balance"])

                saZAR = res["balance"][0]["account\_id"]

                try:

                    ### selling BTC to ZAR in South Africa

                    if saZarBalance > 10:

                        try:

                            orderResp = saAccount.post\_market\_order(pair="XBTZAR", type="BUY", counter\_account\_id=saZAR, counter\_volume=saZarBalance)

                            orderId = orderResp["order\_id"]

                            print("Buy BTC in South Africa Success, OrderID: ", orderId, "RAND Amount:", saZarBalance)

                            while True:

                                orderDetail = saAccount.get\_order(orderId)

                                print("Waiting for Buy BTC in South Africa....", orderId, orderDetail['state'])

                                if orderDetail['state'] == 'COMPLETE':

                                    break

                                time.sleep(10)

                        except Exception as e:

                            print("Error while buying BTC in South Africa", e, saZarBalance)

                    ### send BTC to italy

                    res = saAccount.get\_balances(assets='XBT')

                    saBTCBalance = float(res["balance"][0]["balance"])

                    sendAmount = round(saBTCBalance, 8)

                    if sendAmount > 0:

                        saAccount.send(address=itEmail, amount=sendAmount, currency="XBT")

                        print("Send BTC to Italy Success", sendAmount)

                    ## wait until BTC arrived

                    while True:

                        res = saAccount.get\_balances(assets='XBT')

                        reservedBalance = float(res["balance"][0]["reserved"])

                        if reservedBalance == 0:

                            break

                        time.sleep(10)

                    time.sleep(10)

                    ### exchange to EURO

                    res = itAccount.get\_balances(assets='XBT')

                    itBTCBalance = float(res["balance"][0]["balance"])

                    itBTC = res["balance"][0]["account\_id"]

                    baseVolume = round(itBTCBalance-itTradeFee, 4)

                    if baseVolume > 0:

                        orderResp = itAccount.post\_market\_order(pair="XBTEUR", type="SELL", base\_account\_id=itBTC, base\_volume=baseVolume)

                        orderId = orderResp["order\_id"]

                        while True:

                            orderDetail = itAccount.get\_order(orderId)

                            print("Waiting for Sell BTC in Italy....", orderId, orderDetail['state'])

                            if orderDetail['state'] == 'COMPLETE':

                                break

                            time.sleep(10)

                except Exception as e:

                    print("Error while sending BTC to Italy", e, round(saBTCBalance, 8))

and then arbitrage rate is below than 1%, then send the BTC to UK(Italy) account.

And then if the arbitrage rate is upper than 2% then send the BTC to SouthAfrica account.

so let’s get into deeper.

res = saAccount.get\_balances(assets='ZAR')

                saZarBalance = float(res["balance"][0]["balance"])

                saZAR = res["balance"][0]["account\_id"]

first get the RAND balance of SouthAfrica account and then get the id of RAND account,

                   if saZarBalance > 10:

                        try:

                            orderResp = saAccount.post\_market\_order(pair="XBTZAR", type="BUY", counter\_account\_id=saZAR, counter\_volume=saZarBalance)

                            orderId = orderResp["order\_id"]

                            print("Buy BTC in South Africa Success, OrderID: ", orderId, "RAND Amount:", saZarBalance)

                            while True:

                                orderDetail = saAccount.get\_order(orderId)

                                print("Waiting for Buy BTC in South Africa....", orderId, orderDetail['state'])

                                if orderDetail['state'] == 'COMPLETE':

                                    break

                                time.sleep(10)

                        except Exception as e:

                            print("Error while buying BTC in South Africa", e, saZarBalance)

we can only do trading when the RAND Amount is more than 10ZAR, so we check the balance first

and then post\_market\_order to exchange RAND to BTC.

And we should wait until the order is completed.

                    ### send BTC to italy

                    res = saAccount.get\_balances(assets='XBT')

                    saBTCBalance = float(res["balance"][0]["balance"])

                    sendAmount = round(saBTCBalance, 8)

                    if sendAmount > 0:

                        saAccount.send(address=itEmail, amount=sendAmount, currency="XBT")

                        print("Send BTC to Italy Success", sendAmount)

after the completion, we sould send the BTC to UK account. We use the email declared on the top.

## wait until BTC arrived

                    while True:

                        res = saAccount.get\_balances(assets='XBT')

                        reservedBalance = float(res["balance"][0]["reserved"])

                        if reservedBalance == 0:

                            break

                        time.sleep(10)

                    time.sleep(10)

Sending BTC takes some time, so we should wait until the order is completed…

### exchange to EURO

                    res = itAccount.get\_balances(assets='XBT')

                    itBTCBalance = float(res["balance"][0]["balance"])

                    itBTC = res["balance"][0]["account\_id"]

                    baseVolume = round(itBTCBalance-itTradeFee, 4)

                    if baseVolume > 0:

                        orderResp = itAccount.post\_market\_order(pair="XBTEUR", type="SELL", base\_account\_id=itBTC, base\_volume=baseVolume)

                        orderId = orderResp["order\_id"]

                        while True:

                            orderDetail = itAccount.get\_order(orderId)

                            print("Waiting for Sell BTC in Italy....", orderId, orderDetail['state'])

                            if orderDetail['state'] == 'COMPLETE':

                                break

                            time.sleep(10)

BTC is arrived to UK account, so we need to exchange it to Euro.

So we get the amount of BTC of UK account. If the volume amount is positive number, we post\_market\_order to exchange BTC to euro.

And then wait until the order is completed….

UK->SouthAfrica logic is same as above.