به نام خدا

گزارش تسک مربوط به web-socket binance

آدرس github :

<https://github.com/yunes1375/web-socket-binance>

نویسنده : یونس حسن زاده

شماره تماس : 09392806852

برنامه طراحی شده دارای دو اسکریپت binance.py و main.py میباشد.

کتابخانه های مورد نیاز در فایل requirements.txt موحود میباشد.

**اسکریپت binance.py:**

این اسکریپت برای دریافت اطلاعات از طریق api بایننس طراحی شده است.

برای تست اطلاعات معاملاتی اخیر برای پنج رمز ارز BTCUSD,ETCUSD,LTCUSD,TRXUSD,VETUSD از طریق api بایننس )اطلاعات (aggregated trades دریافت میشود.لازم به ذکر است در صورتیکه معامله جدیدی انجام شود در هربار درخواست api اطلاعات دریافت خواهند شد.

Api درخواستی به شکل زیر است:

#Binance API string

socket = 'wss://stream.binance.com:9443/stream?streams=btcusdt@aggTrade/etcusdt@aggTrade/ltcusdt@aggTrade/trxusdt@aggTrade/vetusdt@aggTrade'

ws = websocket.WebSocketApp(socket,on\_message=on\_message,on\_close=on\_close)

با استفاده از دستور :

ws.run\_forever()

اسکریپت به صورت پیوسته اجرا خواهد شد و اطلاعات دریافتی را به تابع on\_message پاس خواهد داد.

اطلاعات دریافت شده برای ریشه @aggTrade به شرح زیر است :

{

"e": "aggTrade", // Event type

"E": 123456789, // Event time

"s": "BNBBTC", // Symbol

"a": 12345, // Aggregate trade ID

"p": "0.001", // Price

"q": "100", // Quantity

"f": 100, // First trade ID

"l": 105, // Last trade ID

"T": 123456785, // Trade time

"m": true, // Is the buyer the market maker?

"M": true // Ignore

}

تابع on\_message به شرح زیر است:

#function for inserting data in mongodb and calcuting value of trades

def on\_message(ws, message):

    res = json.loads(message)

    price=float(res['data']['p'])

    quantity=float(res['data']['q'])

    value=price\*quantity

    res['data']['value']=value

    mycol.insert\_one(res['data'])

این تابع مقادیر قیمت معاملاتی و حجم معاملاتی را در یکدیگر ضرب میکند و ارزش معاملاتی هر معامله محاسبه میشود،سپس ارزش معاملاتی را به اطلاعات دریافت شده از web-socket اضافه کرده و وارد دیتابیس mongodb میکند.

اطلاعات دیتابیس و کالکشن برای mongodb به شرح زیر تعریف شده است:

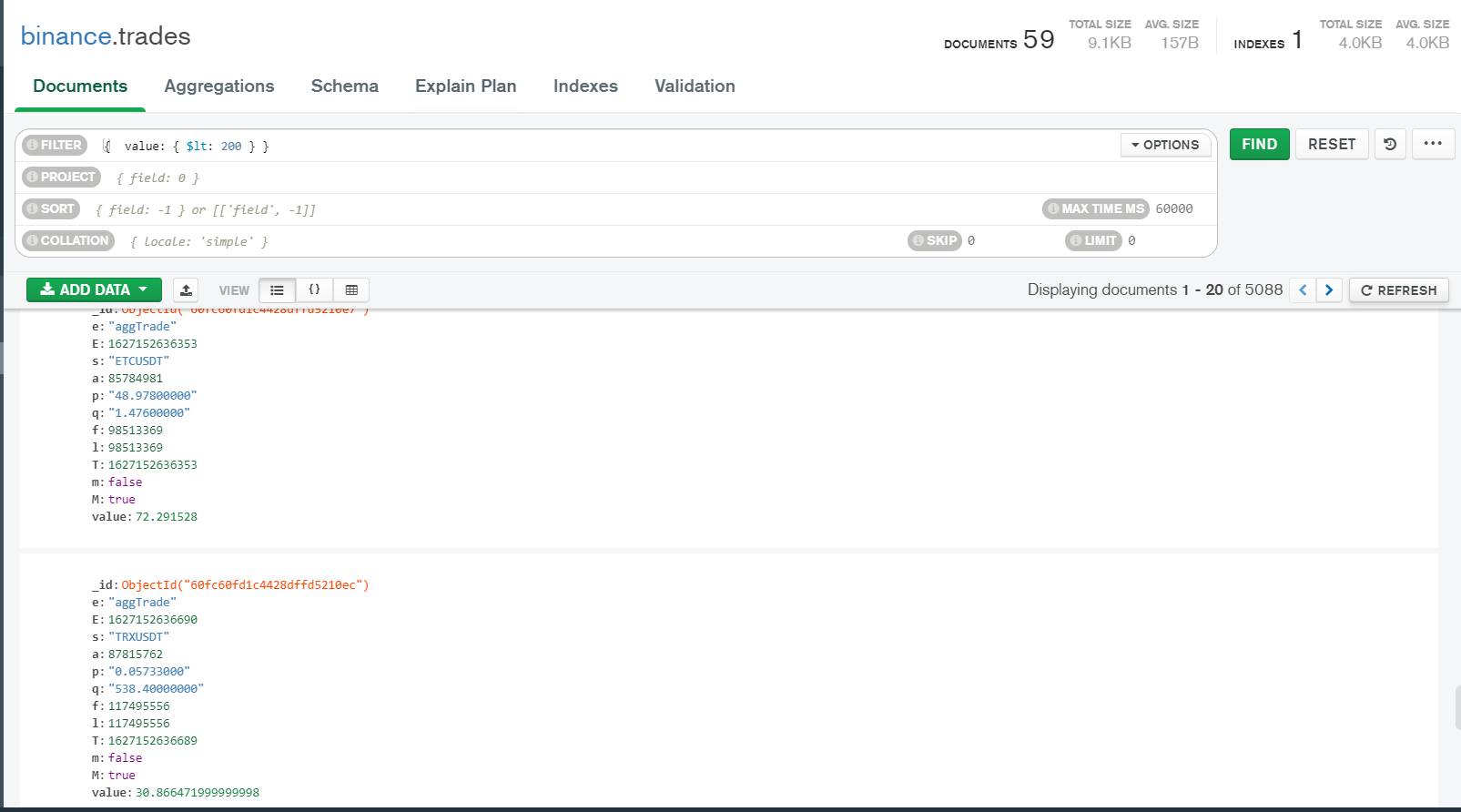
#connection string formongodb

myclient = pymongo.MongoClient("mongodb://localhost:27017/")

mydb = myclient["binance"]

mycol = mydb["trades"]

با اجرای اسکریپت اطلاعات در دیتابیس به شکل زیر دخیره شده اند:



**اسکریپت :main.py**

این اسکریپت از طریق فریمورک فلاسک نوشته شده و به سادگی قابل پیاده سازی از طریق Django هم میباشد.

برای ریشه /besttrades/ در آی پی : 127.0.0.1:80 تابع زیر تعریف شده است:

@app.route('/besttrades/', methods=['GET', 'POST'])

def testfn():

    # GET request

    if request.method == 'GET':

        #getting data of 10 best trade data from mongo by mongodata function as dictionary and converting to json

        return json.dumps(json.dumps(mongodata()))

    # POST request

    if request.method == 'POST':

        return 'Sucesss', 200

این تابع با هر بار که درخواست از سمت کاربر اجرا شود اطلاعات را از طریق تابع mongodata() فراخوانی کرده و پس از تبدیل به json به سمت کاربر ارسال میکند.

تابع mongodata() به شرح زیر تعریف شده است:

def mongodata():

    data={}

    #calcuting time for Query

    deltatime1=deltatime()

    myclient = pymongo.MongoClient("mongodb://localhost:27017/")

    mydb = myclient["binance"]

    mycol = mydb["trades"]

    #Query string for selection objects for last 10 hour

    mydoc = mycol.find({"T": {"$gte": deltatime1}}).sort("value",-1)

    #selection 10 best trades or any number of best trades

    for x in range(0,10):

       dic1={}

       #select time of trade,Coin name,Price,quantity and value of best trades

       dic1['time']=str(datetime.fromtimestamp(int(mydoc[x]["E"])/1000))

       dic1['Crypto']=mydoc[x]['s']

       dic1['Price']=mydoc[x]['p']

       dic1['quantity']=mydoc[x]['q']

       dic1['value']=mydoc[x]['value']

       data[x]=dic1

    return data

این تابع در هر بار فراخوانی اطلاعات موجود در دیتابیس را بر اساس زمان معاملات که در این پروژه 10 ساعت در نظر گرفته شده فراخوانی میکند، سپس اطلاعات دریافتی را بر اساس ارزش معاملاتی از بزرگترین ارزش معاملاتی به کوچکترین ارزش معاملاتی مرتب میکند،اطلاعات زمان معامله، ارز معاملاتی، قیمت معاملاتی،حجم معاملاتی و ارزش معاملاتی را برای هر معامله برتر انتخاب کرده و در قالب فرمت دیکشنری برمیگرداند.

لازم به ذکر است فرمت زمانی معامله از شکل ثانیه از زمان مبدا به تاریخ و زمان تقویمی تبدیل میشود.

از آنجاییکه فرمت زمانی معاملات از طریق بایننس به صورت میلی ثانیه سپری شده از مبدا زمان میباشد برای محاسبه زمان در 10 ساعت گذشته از تابع deltatime() استفاده شده است.

تابع deltatime() به شرح زیر است:

def deltatime():

    nowtime=time.time()

    #calcuting time for 10 hours later or any other time

    querytime=(int(nowtime)-10\*60\*60)\*1000

    return querytime

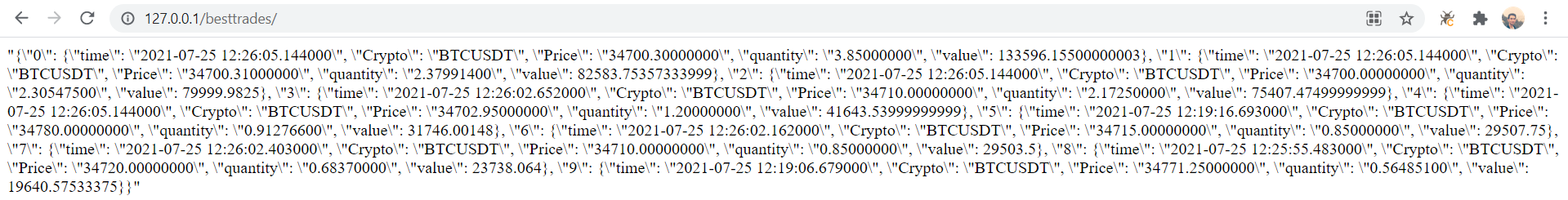
این تابع زمان فعلی را براساس ثانیه از مبدا زمان دریافت کرده،زمان در 10 ساعت گذشته را محاسبه کرده و به صورت میلی ثانیه برمیگرداند.

در نهایت با استفاده از API زیر نتیجه زیر حاصل میشود:

API:

<http://127.0.0.1/besttrades/>

result:



|  |  |
| --- | --- |
| Top of Form  Line wrap  Bottom of Form | |
|  | "{\"0\": {\"time\": \"2021-07-25 12:26:05.144000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34700.30000000\", \"quantity\": \"3.85000000\", \"value\": 133596.15500000003}, \"1\": {\"time\": \"2021-07-25 12:26:05.144000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34700.31000000\", \"quantity\": \"2.37991400\", \"value\": 82583.75357333999}, \"2\": {\"time\": \"2021-07-25 12:26:05.144000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34700.00000000\", \"quantity\": \"2.30547500\", \"value\": 79999.9825}, \"3\": {\"time\": \"2021-07-25 12:26:02.652000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34710.00000000\", \"quantity\": \"2.17250000\", \"value\": 75407.47499999999}, \"4\": {\"time\": \"2021-07-25 12:26:05.144000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34702.95000000\", \"quantity\": \"1.20000000\", \"value\": 41643.53999999999}, \"5\": {\"time\": \"2021-07-25 12:19:16.693000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34780.00000000\", \"quantity\": \"0.91276600\", \"value\": 31746.00148}, \"6\": {\"time\": \"2021-07-25 12:26:02.162000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34715.00000000\", \"quantity\": \"0.85000000\", \"value\": 29507.75}, \"7\": {\"time\": \"2021-07-25 12:26:02.403000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34710.00000000\", \"quantity\": \"0.85000000\", \"value\": 29503.5}, \"8\": {\"time\": \"2021-07-25 12:25:55.483000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34720.00000000\", \"quantity\": \"0.68370000\", \"value\": 23738.064}, \"9\": {\"time\": \"2021-07-25 12:19:06.679000\", \"Crypto\": \"BTCUSDT\", \"Price\": \"34771.25000000\", \"quantity\": \"0.56485100\", \"value\": 19640.57533375}}" |

سورس کد ها و فایل توضیحات از طریق آدرس زیر در دسترس است:

<https://github.com/yunes1375/web-socket-binance>