Web scraping code used for scrap 2 years of historical data(1 january 2023 to 31 december 2024)

```
import json, time, sys, os
from urllib import request as req
from urllib import parse
from bs4 import BeautifulSoup

# Please change date range before starting
bdate = "01-Jan-2024"
edate = "31-Dec-2024"
csv_path = sys.path[0] + os.path.sep + "all-funds-nav.csv"


def fund_fund_names():
    while True:
        try:
            fund_fund_names_list = {}

            request =
req.Request("https://www.amfiindia.com/net-asset-value/nav-history")
            response = req.urlopen(request)
            soup = BeautifulSoup(response.read().decode('utf-8'),
'html.parser')

            for elem in soup.select("#NavHisMFName")[0].select('option'):
                if elem['value']:
                    # Dictionary structure - key: mfID & value: fund house
name
                    fund_fund_names_list[elem['value']] = elem.get_text()

            return fund_fund_names_list
        except:
            # Wait 10 seconds till internet connection is back
            time.sleep(10)
            continue

def find_schemes(mfID):
    # Finds all schemes run by a MF using its mfID
    while True:
```

```
        try:
            request =
req.Request("https://www.amfiindia.com/modules/NavHistorySchemeNav")
            request.add_header('X-Requested-With', 'XMLHttpRequest')
            request.add_header('User-Agent', 'Mozilla/5.0 (X11; Linux
i686) AppleWebKit/537.17 (KHTML, like Gecko) Chrome/24.0.1312.27
Safari/537.17')
            data = parse.urlencode({"ID": mfID})
            data = data.encode('ascii')
            response = req.urlopen(request, data)

            dictionary = {}
            for scheme in json.loads(response.read().decode('utf-8')):
                dictionary[scheme["Value"]] = scheme["Text"].strip("\xa0")
            return dictionary
        except:
            # Wait till connection is back
            time.sleep(10)
            continue


def download_nav_raw_data(mfID, scID):
    # When looking for NAV of a particular scheme of a fund house data is
provided in HTML format that HTML string is returned by this function
    while True:
        try:
            request =
req.Request("https://www.amfiindia.com/modules/NavHistoryPeriod")
            request.add_header('User-Agent', 'Mozilla/5.0 (X11; Linux
i686) AppleWebKit/537.17 (KHTML, like Gecko) Chrome/24.0.1312.27
Safari/537.17')
            data = parse.urlencode({"mfID": mfID,"scID": scID,"fDate":
bdate,"tDate": edate})
            data = data.encode('ascii')

            response = req.urlopen(request, data)
            answer = response.read().decode('utf-8')
            return answer
        except:
            # Wait till connection is back
```

```
            time.sleep(3)
            continue


def add_funds_to_file(fp, mfid, scid):
    # Function parses the HTML string sent by AMFI & extracts NAV of a
particular fund from the tabular HTML data
    soup = BeautifulSoup(download_nav_raw_data(mfid,scid), 'html.parser')
    table = soup.find_all("tr")

    for element in table:
        lines = []
        for i in  element.find_all('td'):
            lines+=i.contents
        if (len(lines) != 0) and lines[0] != '\n':
            # Printing lines into the file
            print(lines[-1], fund_house_name[mfid], mfid,
scheme_name[scid], scid, lines[0], file=fp, sep=',')

with open(csv_path, 'w') as fp:
    try:
        print("Date,Fund House,Fund HouseID,Scheme,SchemeID,NAV", file=fp)
# CSV file header
        fund_house_name = fund_fund_names()
        mfdone = 0
        for mfID, mf_house_name in fund_house_name.items():
            scheme_name = find_schemes(mfID)
            mfdone += 1
            sdone = 0
            for ids, names in scheme_name.items():
                add_funds_to_file(fp, mfID, ids)
                sdone += 1
                print("MF", mfdone, "/", len(fund_house_name), "SCHEME",
sdone, "/", len(scheme_name),fund_house_name[mfID], scheme_name[ids])
            time.sleep(1) # Wait for a while after a MF is done so website
doesn't ban us for traffic
    except KeyboardInterrupt:
        fp.flush()
        fp.close()
        quit("User Exit")
```