NAMA: Zanuar Rikza Aditiya
NIM: 230411100087
MATA KULIAH: EKSTRAKSI INFORMASI A

Code ini dugunakan untuk mengambil URL halaman detail setiap putusan dari list putusan
parameter yang disiapkan adalah URL posisi awal munculnya list putusan sesuai kriteria yang akan di ambil dan URL halaman terakhir yang memuat list putusan
output dari code ini adalah file dengan nama hasilListURLPage.txt yang berisi list URL untuk melihat detail setiap putusan

In [2]:
```
import requests
import urllib.request
import time
import os
import warnings
from bs4 import BeautifulSoup
```

fungsi getURLfromWeb untuk Mendapatkan url putusan dari website
Penjelasan lainya berada pada komentar code

In [3]:
```
def getURLfromWeb(url):

    response = requests.get(url, verify=False)
    print(response)

    htmlCode1 = BeautifulSoup(response.text, 'html.parser')
    result1=htmlCode1.findAll('a')

    urlHasil=[]
    for eachResult1 in result1:
        cariURLawal=str(eachResult1).find('https://putusan3.mahkamahagung.go.id/direktori/putusan/')
        cariURLakhir=str(eachResult1).find('html">Putusan') + 4
        #print(cariURLakhir)
        if cariURLawal == 9 and cariURLakhir >= 4:
            #print(str(eachResult1)[cariURLawal:cariURLakhir])
            cariURL=str(eachResult1)[cariURLawal:cariURLakhir]
            urlHasil.append(cariURL)

    print(urlHasil)
    return(urlHasil)
```

Penjelasan lainya berada pada komentar code

In [ ]:
```
def main():

    warnings.filterwarnings('ignore')

    data_path = "./data/"
    #url1 = 'https://putusan3.mahkamahagung.go.id/direktori/kategori/jenis/pencurian-1.html'
    #url2 = 'https://putusan3.mahkamahagung.go.id/direktori/kategori/jenis/pencurian-1/page/'

    url1 = 'https://putusan3.mahkamahagung.go.id/direktori/index/pengadilan/pn-pekalongan/kategori/pidana-umum-1.html
    url2 = 'https://putusan3.mahkamahagung.go.id/direktori/index/pengadilan/pn-pekalongan/kategori/pidana-umum-1/page

    listHasil = []
    # Saya ingin mengambil putusan hingga page 50
    ulang = 50

    # Saya menambahkan last index
    # agar saat code gagal karena suatu alasan saya tidak mengulangi lagi dari awal
    if not os.path.exists(data_path + "lastindex.txt"):
        with open(data_path + "lastindex.txt", "w", encoding="UTF-8") as file_index:
            file_index.write(str(19))
    try:
        last_index = int(open("./data/lastindex.txt", "r", encoding="UTF-8").read())
    except:
        last_index = 19

    startTime = time.time()

    # file_hasil = open(data_path + "hasilListURLPage.txt", "w", encoding='UTF8')

    # Scraping di mulai dari index terakhir yang telah di cek + 1
    for i in range(last_index, ulang):
        if i == 0:
            url = url1
            print(1)
        else:
            n=i+1
            url = url2+str(n)+'.html'
            print(2)

        listHasil = getURLfromWeb(url)

        print(listHasil)
```

```
        # menambahkan list url ke file hasilListURLPage.txt
        # agar jika gagal tidak berulang dari awal
        with open(data_path + "hasilListURLPage.txt", "a", encoding="UTF-8") as file_hasil:
            for listURL in listHasil:
                file_hasil.write(listURL+"\n")

        print(last_index)
        # Menyimpan index terakhir yang telah di cek + 1
        # agar kita tahu dari mana kita harus mulai lagi tanpa mengulangi dari awal
        with open(data_path + "lastindex.txt", "w", encoding="UTF-8") as file_index:
            file_index.write(str(i + 1))

    # file_hasil.close()
    endTime = time.time()
    print(listHasil)
    print('Time Processing : ', endTime-startTime, ' Second')

main();
```

EKSTRAK META INFO

In [ ]:
```
import requests
import urllib.request
import time
import os
import warnings
import re

from bs4 import BeautifulSoup

import csv
import os.path
```

fungsi generateFileCvs untuk menyimpan data yang di dapatkan dari ektraksi informasi

Penjelasan lain nya berupa komentar di code

In [ ]:
```
def generateFileCSV(listHasil,csvName1):

        csvFolrder = "./data/"
        csvName = csvName1

        # Mengecek apkah file csv ada atau tidak
        # jika ada akan membuka file dengan mode 'a' untuk menambahkan data
        # jika tidak akan membuka file dengan mode 'w' untuk membuat baru file nya
        # dengan kolom kolom yang di tentukan
        if os.path.exists(csvName):

                f = open(csvName, 'a', newline='\n')
                print(f)
                print("ada")
                w = csv.writer(f)

        else:

                f = open(csvName, 'w', newline='\n')
                print(f)
                print("tidak ada")
                w = csv.writer(f)
                w.writerow(("terdakwa", "penuntut_umum", "nomor", "tingkat_proses", "klasifikasi", "kata_kunci", "tah
               "lembaga_peradilan", "jenis_lembaga_peradilan", "hakim_ketua", "hakim_anggota", "panitera", "amar",
               "amar_lainnya", "catatan_amar", "tanggal_musyawarah", "tanggal_dibacakan", "kaidah", "abstrak", "url"))

        # menulis file csv
        print(listHasil)
        for s in listHasil:
                w.writerow(s)

        f.close()
        berhasil = "\nCreate Csv file Berhasil\n"
        return berhasil
```

fungsi generateMeta untuk mengekstraksi informasi penting dari

In [ ]:
```
def generateMeta(urlMeta):

    url = str(urlMeta).strip()
    # url = 'https://putusan3.mahkamahagung.go.id/direktori/putusan/zaedca28e81ec25e8cb4313634373336.html'
    response = requests.get(url, verify=False)
    # print(url)
    print(response)

    soup = BeautifulSoup(response.text, 'html.parser')
    cleanTags = re.compile('<.*?>')

    listMetaHead = ["terdakwa", "penuntut_umum", "nomor", "tingkat_proses", "klasifikasi", "kata_kunci", "tahun", "ta
```

```
            "lembaga_peradilan", "jenis_lembaga_peradilan", "hakim_ketua", "hakim_anggota", "panitera", "amar",
            "amar_lainnya", "catatan_amar", "tanggal_musyawarah", "tanggal_dibacakan", "kaidah", "abstrak", "url"]

    # initialisasi
    listMeta = []

    # for i in range(len(listMetaHead)):
        #listMeta.append("")

    rowsMETA1 = soup.find("ul", {"class": "portfolio-meta nobottommargin"}).find("table").findAll("tr")
    rowsMETA2 = soup.findAll("ul", {"class": "portfolio-meta nobottommargin"})

    #print('-------------------------')
    for row in rowsMETA1:
        coll = row.findAll("td")

        cleantext2 =''
        cleantext1 =''

        if len(coll) > 1:
            cleantext2 = (re.sub(cleanTags, '', str(coll[1]))).strip()
            listMeta.append(cleantext2.replace('\n',' '))
            #print(cleantext2.replace('\n',' '))

        else:
            cleantext1 = (re.sub(cleanTags, ' ', str(coll[0]))).strip()
            # untuk putusan pidana akan muncul terdakwa dan penuntut umum pada meta
            # sedangkan untuk putusan selain pidana tidak muncul

            pidorpdt = re.search( r'(.*)/Pdt.(.*)',str(cleantext1), re.M|re.I)

            # check dokumen putusannya pidana atau perdata
            if pidorpdt == None:
                entTerdakwah = re.search( r'(.*)Terdakwa:(.*)',str(cleantext1), re.M|re.I)
                entPenuntut = re.search( r'Penuntut Umum:(.*)Terdakwa:',str(cleantext1), re.M|re.I)
            else:
                entTerdakwah = re.search( r'(.*)Tergugat:(.*)',str(cleantext1), re.M|re.I)
                entPenuntut = re.search( r'Penggugat:(.*)Tergugat:',str(cleantext1), re.M|re.I)

            if entTerdakwah == None:
                listMeta.append("")
            else:
                listMeta.append(entTerdakwah.group(2))

            if entPenuntut == None:
                listMeta.append("")
            else:
                listMeta.append(entPenuntut.group(1))

            #print(coll[0])
            #print(entTerdakwah.group(2))
            #print(entPenuntut.group(1))

    urlDL = rowsMETA2[1].findAll("li")
    urlDLStr = str(urlDL[4])
    listMeta.append(urlDLStr[urlDLStr.find("https"):urlDLStr.find('">')])
    #print(urlDLStr[urlDLStr.find("https"):urlDLStr.find('">')])
    #print(listMeta)

    return listMeta
```

fungsi main menjalankan fungsi fungsi sebelumnya

Penjelasan lainya berada pada komentar code

In [ ]:

```
def main():

    warnings.filterwarnings('ignore')

    # folderListURL = pathFile
    data_path = "./data/"
    fileListURL = "./data/hasilListURLPage.txt"
    fileMetaCSV = "./data/metaPidanaUmum.csv"

    # Saya menambahkan last index
    # agar saat code gagal karena suatu alasan saya tidak perlu menghitung
    # jumlah data yang terkumpul secara manual dan juga agar tidak salah hitung
    # karena sudah dihitung oleh program
    if not os.path.exists(data_path + "lastindexmeta.txt"):
        with open(data_path + "lastindexmeta.txt", "w", encoding="UTF-8") as file_index:
            file_index.write(str(0))

    try:
        last_index = int(open(data_path + "lastindexmeta.txt", "r", encoding="UTF-8").read())
    except:
        last_index = 0

    startTime = time.time()
    openfileListURL = open(fileListURL, "r", encoding='UTF8')
    bacaListURL = openfileListURL.readlines()
```

```
openfileListURL.close()

i = 1

# Melakukan Perulangan dari index terakhir yang dicek + 1 samapi
for index in range(last_index, len(bacaListURL)):
    try:
        #print(str(barisURL))
        print(bacaListURL[last_index])
        # membungkus hasil generate meta ke dalam list
        # karena fungsi generateFileCsv mengolah list dalam list untuk menambahkan data
        hasil = [generateMeta(str(bacaListURL[index]))]
        print("======= ROW HASIL =======",i)
        # Langsung menulis hasil yang ditemukan agar tidak diulangi dari awal saat gagal
        createFile = generateFileCSV(hasil, fileMetaCSV)
    except Exception as e:
        print(f"Error Get Meta Inf, {e}")

    # Menyimpan index terakhir yang telah di cek + 1
    # agar kita tahu dari mana kita dapat mulai lagi mengekstrak infromasi
    with open(data_path + "lastindexmeta.txt", "w", encoding="UTF-8") as file_index:
        file_index.write(str(index + 1))

    i=i+1

endTime = time.time()
# print(listHasil)
print('Time Processing : ', endTime-startTime, ' Second')
```

```
main();
```