**UAS**

**PEMROGRAMAN KOMPUTER K-01**

**NAMA : MUHAMMAD NAUFAL AQIL ZUHDI**

**NIM : 12220051**

**TANGGAL : 17 DESEMBER 2021**

**DESKRIPSI**

1. **CODE PROGRAM DALAM BAHASA PYTHON**

"""

Aplikasi Streamlit untuk menggambarkan statistik jumlah produksi minyak mentah di berbagai negara

Sumber data berasal dari file ujian akhir semester IF2112 Pemrograman Komputer

Referensi API Streamlit: https://docs.streamlit.io/library/api-reference

Nama        : Muhammad Naufal Aqil Zuhdi

NIM         : 12220103

Kelas       : 01

Mata Kuliah : IF2112 Pemrograman Komputer

UJIAN AKHIR SEMESTER

"""

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from matplotlib import cm

import json

import streamlit as st

from PIL import Image

#fungsi mengconvert data yang ada di csv ke list

def convert\_DataToList():

    df = pd.read\_csv('produksi\_minyak\_mentah.csv')

    list\_datatolist = [[row[col] for col in df.columns] for row in df.to\_dict('records')]

    return list\_datatolist

#fungsi yang menghasilkan list yang isinya negara yang kode negara pada csv terdapat di json

def Nama\_lengkapNegara():

    List\_data = convert\_DataToList()

    ListKodeNegara = []

    ListNamaLengkapNegara = []

    for i in List\_data:

        if i[0] not in ListKodeNegara:

            ListKodeNegara.append(i[0])

    with open("kode\_negara\_lengkap.json") as f:

        data = json.load(f)

    for i in range(len(ListKodeNegara)):

        for Negara in data:

            if ListKodeNegara[i] == Negara["alpha-3"]:

                ListNamaLengkapNegara.append([Negara["alpha-3"], Negara["name"]])

                break

    return ListNamaLengkapNegara

#fungsi list semua nama negara

def List\_NamaNegara():

    List = []

    List\_NamaNegara = Nama\_lengkapNegara()

    for i in List\_NamaNegara:

        List.append(i)

    return List

#fungsi convert nama negara menjadi kode negara untuk keperluan input pada parameter fungsi grafik

def convert\_NamaNegaraToKodeNegara(Nama\_Lengkap):

    List = Nama\_lengkapNegara()

    for i in List:

        if i[1] == Nama\_Lengkap:

            return i[0]

#fungsi list data produksi setiap kode negara pada csv

def get\_data(kode\_negara):

    List = convert\_DataToList()

    List\_Data = []

    for i in range(len(List)):

        if List[i][0] == kode\_negara:

            List\_Data.append(List[i])

    return List\_Data

#fungsi list semua negara untuk di pengaturan tampilan

List\_semua\_negara = []

for i in Nama\_lengkapNegara():

    List\_semua\_negara.append(i[1])

#Bagian A

#untuk mendapatkan semua warna pada bar plot

cmap\_name = 'tab20'

cmap = cm.get\_cmap(cmap\_name)

colors = cmap.colors[:len(List\_semua\_negara)]

#fungsi grafik jumlah produksi minyak mentah untuk negara N

def Grafik\_JumlahProduksiMinyak(nama\_lengkap):

    List\_tahun\_xy = get\_data(convert\_NamaNegaraToKodeNegara(nama\_lengkap))

    X = []

    for i in range(len(List\_tahun\_xy)):

        X.append(List\_tahun\_xy[i][1])

    Y = []

    for i in range(len(List\_tahun\_xy)):

        Y.append(List\_tahun\_xy[i][2])

    if P == 'Chart Tipe 1':

        plt.title("Line Plot Merepresentasikan Jumlah Produksi Minyak Mentah untuk Setiap Tahun di Negara " +str(N)+ "\n")

        plt.plot(X,Y, color = "black")

        plt.xlabel('Tahun')

        plt.ylabel('Produksi')

        plt.legend(labels = ["Jumlah Produksi Minyak Mentah"], loc = "best")

        st.pyplot(plt)

    else:

        plt.clf()

        plt.scatter(X, Y, marker="o", color = "black")

        plt.title("Scatter Plot Merepresentasikan Jumlah Produksi Minyak Mentah untuk Setiap Tahun di Negara " +str(N)+ "\n")

        plt.xlabel('Tahun')

        plt.ylabel('Produksi')

        plt.legend(labels = ["Jumlah Produksi\nMinyak Mentah"], loc = "best")

        st.pyplot(plt)

#Bagian B

#fungsi untuk mendapatkan data produksi negara pada tahun tertentu

def get\_tahun(Tahun):

    List = convert\_DataToList()

    List\_Tahun = []

    for i in List:

        if i[1] == Tahun:

            List\_Tahun.append(i)

    return List\_Tahun

#fungsi transisi untuk mengambil elemen ketiga pada suatu list

def take\_tirth(elem):

    return elem[2]

#fungsi transisi untuk mengambil elemen kedua pada suatu list

def take\_second(elem):

    return elem[1]

#fungsi untuk data produksi negara yang disort pada tahun tertentu

def get\_NbesarNegara(N, tahun):

    List\_NamaLengkap\_Negara = Nama\_lengkapNegara()

    List\_data = get\_tahun(tahun)

    sorted\_list = sorted(List\_data, key=take\_tirth)

    List\_Nterbesar = []

    count = 0

    penunjuk = 0

    while count < N:

        for i in List\_NamaLengkap\_Negara:

            if sorted\_list[len(sorted\_list) - 1 - penunjuk][0] == i[0]:

                List\_Nterbesar.append([i[1], sorted\_list[len(sorted\_list) - 1 - penunjuk][2]])

                count += 1

                break

        penunjuk += 1

    return List\_Nterbesar

#fungsi untuk list produksi negara pada tahun tertentu

def get\_ValueNbesarNegara(N, tahun):

    List = get\_NbesarNegara(N, tahun)

    List\_ValueNbesarNegara = []

    for i in List:

        List\_ValueNbesarNegara.append(i[1])

    return List\_ValueNbesarNegara

#fungsi untuk list nama negara pada tahun tertentu

def get\_NamaLengkap\_NbesarNegara(N, tahun):

    List = get\_NbesarNegara(N, tahun)

    List\_NamaLengkap\_NbesarNegara = []

    for i in List:

        List\_NamaLengkap\_NbesarNegara.append(i[0])

    return List\_NamaLengkap\_NbesarNegara

#fungsi grafik B besar negara yang memproduksi minyak mentah

def Grafik\_NbesarNegara(N, Tahun):

    X = get\_NamaLengkap\_NbesarNegara(N, Tahun)

    Y = get\_ValueNbesarNegara(N, Tahun)

    if P == 'Chart Tipe 1':

        plt.clf()

        plt.bar(X, Y, alpha = 1.0, width=0.9, bottom=None, align="center", color=colors, data=None, zorder=3)

        plt.grid(True, color="grey", linewidth="0.7", linestyle="-.", zorder=0)

        for i in range(len(Y)):

            plt.annotate(f'{Y[i]}\n', xy=(X[i], Y[i]),ha="center", va="center", size=9, rotation = 20)

        plt.legend(labels=["Total Produksi Minyak Mentah di\n" + str(int(N))+" Besar Negara"], loc="best")

        plt.title("Bar Plot Merepresentasikan Grafik Produksi Minyak Mentah di " +str(int(N))+ " Besar Negara\n")

        plt.xlabel("Negara")

        plt.xticks(size=8, rotation = 65)

        plt.ylabel("Produksi")

        st.pyplot(plt)

    else:

        cmap\_name1 = 'tab20'

        cmap1 = cm.get\_cmap(cmap\_name1)

        colors1 = cmap1.colors[:len(X)]

        plt.clf()

        plt.scatter(X, Y, alpha=1.0, marker="o", color=colors1)

        plt.grid(True, color="grey", linewidth="0.7", linestyle="-.", zorder=0)

        for i in range(len(Y)):

            plt.annotate(f'{Y[i]}\n', xy=(X[i], Y[i]),ha="center", va="center", size=9, rotation = 20)

        plt.xlabel("Negara")

        plt.xticks(size=8, rotation = 65)

        plt.ylabel("Produksi")

        plt.title("Scatter Plot Merepresentasikan Grafik Produksi Minyak Mentah di " +str(int(N))+ " Besar Negara\n")

        plt.legend(labels=["Total Produksi Minyak Mentah di\n" + str(int(N))+ " Besar Negara"], loc="best")

        st.pyplot(plt)

#Bagian C

#fungsi untuk mendapatkan data produksi negara pada keseluruhan tahun

def get\_JumlahProduksiKumulatif(N):

    List = convert\_DataToList()

    List\_NamaLengkap\_Negara = Nama\_lengkapNegara()

    kode\_negara = List[0][0]

    List\_KumulatifNegara = []

    Value = []

    Value2 = []

    Nbesar = []

    sum = 0

    count = 0

    index = 0

    #Menghasilkan List Jumlah produksi Kumulatif dari setiap Negara

    for ListOfData in List:

        if index != len(List) - 1:

            if kode\_negara == ListOfData[0]:

                sum += ListOfData[2]

            else:

                sum = round(sum, 3)

                List\_KumulatifNegara.append([kode\_negara, sum])

                Value.append(sum)

                Value2.append(sum)

                kode\_negara = ListOfData[0]

                sum = 0

                sum += ListOfData[2]

        else:

            sum += ListOfData[2]

            sum = round(sum, 3)

            List\_KumulatifNegara.append([kode\_negara, sum])

        index += 1

    sorted\_list = sorted(List\_KumulatifNegara, key=take\_second)

    #Menghasilkan List berupa N besar Jumlah produksi Kumulatif sebelumnya

    penunjuk = 0

    while count < N:

        for j in List\_NamaLengkap\_Negara:

            if sorted\_list[len(sorted\_list) - 1 - penunjuk][0] == j[0]:

                Nbesar.append([j[1], sorted\_list[len(sorted\_list) - 1 - penunjuk][1]])

                count += 1

                break

        penunjuk += 1

    return Nbesar

#fungsi untuk mendapatkan jumlah produksi negara pada keseluruhan tahun

def get\_Value\_JumlahProduksiKumulatif(N):

    List = get\_JumlahProduksiKumulatif(N)

    List\_Value = []

    for i in List:

        List\_Value.append(i[1])

    return List\_Value

#fungsi untuk mendapatkan nama negara pada keseluruhan tahun

def get\_NamaLengkapnegara\_JumlahProduksiKumulatif(N):

    List = get\_JumlahProduksiKumulatif(N)

    List\_NamaLengkapNegara = []

    for i in List:

        List\_NamaLengkapNegara.append(i[0])

    return List\_NamaLengkapNegara

#fungsi grafik untuk B besar negara pada keseluruhan tahun

def Grafik\_NbesarNegaraKumulatif(N):

    X = get\_NamaLengkapnegara\_JumlahProduksiKumulatif(N)

    Y = get\_Value\_JumlahProduksiKumulatif(N)

    if P == "Chart Tipe 1":

        plt.clf()

        plt.bar(X, Y, alpha = 1.0, width=0.9, bottom=None, align="center", color=colors, data=None, zorder=3)

        plt.grid(True, color="grey", linewidth="0.7", linestyle="-.", zorder=0)

        for i in range(len(Y)):

            plt.annotate(f'{Y[i]}\n', xy=(X[i], Y[i]),ha="center", va="center", size=9, rotation = 20)

        plt.legend(labels=["Total Produksi Minyak Mentah di\n" +str(int(N))+ " Besar Negara Secara Kumulatif"], loc="best")

        plt.title("Bar Plot Merepresentasikan Grafik Produksi Minyak Mentah di " + str(int(N)) + " Besar Negara Secara Kumulatif\n", size = 10)

        plt.xlabel("Negara")

        plt.xticks(size=8, rotation = 65)

        plt.ylabel("Produksi Kumulatif")

        st.pyplot(plt)

    else:

        cmap\_name2 = 'tab20'

        cmap2 = cm.get\_cmap(cmap\_name2)

        colors2 = cmap2.colors[:len(X)]

        plt.clf()

        plt.scatter(X, Y, alpha=1.0, marker="o", color=colors2)

        plt.grid(True, color="grey", linewidth="0.7", linestyle="-.", zorder=0)

        for i in range(len(Y)):

            plt.annotate(f'{Y[i]}\n', xy=(X[i], Y[i]), ha="center", va="center", size=9, rotation = 20)

        plt.xlabel("Negara")

        plt.xticks(size=8, rotation = 65)

        plt.ylabel("Produksi Kumulatif")

        plt.title("Scatter Plot Merepresentasikan Grafik Produksi Minyak Mentah di " +str(int(N))+ " Besar Negara Secara Kumulatif\n", size = 10)

        plt.legend(labels=["Total Produksi Minyak Mentah di\n" +str(int(N))+ " Besar Negara Secara Kumulatif"], loc="best")

        st.pyplot(plt)

# Bagian D

#fungsi untuk mendapatkan data minimum pada tahun tertentu

def getMinimumPadaTahun(Tahun):

    List = convert\_DataToList()

    List\_NamaLengkap\_Negara = Nama\_lengkapNegara()

    List\_data = get\_tahun(Tahun)

    List\_Nilai = [List[i][2] for i in range(len(List))]

    List\_jumlahProduksi = []

    for j in List\_data:

        List\_jumlahProduksi.append(j[2])

    List\_Nterkecil = []

    Minimum = min(List\_jumlahProduksi)

    while Minimum == 0:

        List\_jumlahProduksi.remove(Minimum)

        Minimum = min(List\_jumlahProduksi)

    for i in List\_NamaLengkap\_Negara:

        if List[List\_Nilai.index(Minimum)][0] == i[0]:

            List\_Nterkecil.append([i[1], Minimum])

            break

    List\_jumlahProduksi.remove(Minimum)

    return List\_Nterkecil

#fungsi untuk mendapatkan data nol pada tahun tertentu

def getListNol(Tahun):

    List\_NegaraNol = []

    List\_NamaLengkapNegaraNol = []

    List = convert\_DataToList()

    for i in List:

        if i[2] == 0 and i[1] == Tahun:

            List\_NegaraNol.append([i[0], i[2]])

    for j in List\_NegaraNol:

        for i in Nama\_lengkapNegara():

            if i[0] == j[0]:

                List\_NamaLengkapNegaraNol.append([i[1], j[1]])

    return List\_NamaLengkapNegaraNol

#fungsi untuk informasi data maks, min, dan nol pada tahun tertentu

def informasi\_Tahuntertentu(Tahun, extreme):

    Informasi = []

    List = []

    if extreme == "maksimum":

        List.extend(get\_NbesarNegara(1, Tahun))

    elif extreme == "minimum":

        List.extend(getMinimumPadaTahun(Tahun))

    elif extreme == "Nol":

        List.extend(getListNol(Tahun))

    with open("kode\_negara\_lengkap.json") as f:

        data = json.load(f)

    for i in range(len(List)):

        for Negara in data:

            if Negara["name"] == List[i][0]:

                if extreme != "Nol":

                    Informasi.extend([Negara["name"], Negara["alpha-3"], Negara["region"], Negara["sub-region"]])

                    break

                else:

                    Informasi.append([Negara["name"], Negara["alpha-3"], Negara["region"], Negara["sub-region"]])

    return Informasi

#fungsi untuk data maks, min, dan nol pada tahun tertentu

def getExtremeAll(ModeExtreme):

    List = convert\_DataToList()

    List\_NamaLengkap\_Negara = Nama\_lengkapNegara()

    kode\_negara = List[0][0]

    List\_KumulatifNegara = []

    Value = []

    Value2 = []

    Extreme = []

    sum = 0

    index = 0

    # Menghasilkan List Jumlah produksi Kumulatif dari setiap Negara

    for ListOfData in List:

        if index != len(List) - 1:

            if kode\_negara == ListOfData[0]:

                sum += ListOfData[2]

            else:

                sum = round(sum, 3)

                List\_KumulatifNegara.append([kode\_negara, sum])

                Value.append(sum)

                Value2.append(sum)

                kode\_negara = ListOfData[0]

                sum = 0

                sum += ListOfData[2]

        else:

            sum += ListOfData[2]

            sum = round(sum, 3)

            List\_KumulatifNegara.append([kode\_negara, sum])

            Value.append(sum)

            Value2.append(sum)

        index += 1

    if (ModeExtreme == "maksimum"):

        Extreme = get\_JumlahProduksiKumulatif(1)

    elif (ModeExtreme == "minimum"):

        Minimum = min(Value)

        # cek bahwa nilai minimum != 0

        while Minimum == 0:

            Value.remove(Minimum)

            Minimum = min(Value)

        for i in List\_NamaLengkap\_Negara:

            if List\_KumulatifNegara[Value2.index(Minimum)][0] == i[0]:

                Extreme.append([i[1], Minimum])

                break

            else:

                while Minimum == 0:

                    Value.remove(Minimum)

                    Minimum = min(Value)

    elif (ModeExtreme == "Nol"):

        List\_NegaraProduksiNol = []

        for i in List\_KumulatifNegara:

            if i[1] == 0:

                List\_NegaraProduksiNol.append(i)

        for i in List\_NegaraProduksiNol:

            for j in List\_NamaLengkap\_Negara:

                if i[0] == j[0]:

                    Extreme.append([j[1], 0])

                    break

    return Extreme

#fungsi untuk informasi data maks, min, dan nol pada tahun tertentu

def Informasi\_ForAll(extreme):

    Informasi = []

    List = getExtremeAll(extreme)

    with open("kode\_negara\_lengkap.json") as f:

        data = json.load(f)

    for i in range(len(List)):

        for Negara in data:

            if Negara["name"] == List[i][0]:

                if extreme != "Nol":

                    Informasi.extend([Negara["name"], Negara["alpha-3"], Negara["region"], Negara["sub-region"]])

                    break

                else:

                    Informasi.append([Negara["name"], Negara["alpha-3"], Negara["region"], Negara["sub-region"]])

    return Informasi

############### title ###############

st.set\_page\_config(layout="wide")  # this needs to be the first Streamlit command called

st.title("APLIKASI UJIAN AKHIR SEMESTER IF2112 PEMROGRAMAN KOMPUTER")

st.title("Statistik Jumlah Produksi Minyak Mentah di Berbagai Negara")

st.markdown("\*Sumber data berasal dari file Ujian Akhir Semester IF2112 Pemrograman Komputer\*")

st.markdown("Oleh: Muhammad Naufal Aqil Zuhdi")

st.markdown("NIM: 12220103")

st.markdown("Kelas: 01")

st.markdown("Mata Kuliah: IF2112 Pemrograman Komputer")

st.write("\n")

st.write("\n")

st.write("\n")

############### title ###############

############### sidebar ###############

image = Image.open('TM\_2020.jpeg')

st.sidebar.image(image)

st.sidebar.title("Pengaturan")

## User inputs on the control panel

st.sidebar.subheader("Pengaturan Konfigurasi Tampilan")

P = st.sidebar.selectbox('Pilih Tipe Chart',['Chart Tipe 1','Chart Tipe 2'])

N = st.sidebar.selectbox('Pilih Nama Negara', List\_semua\_negara)

B = st.sidebar.number\_input("Pilih Berapa Besar Negara", min\_value=1, max\_value=137, value = 5)

T = st.sidebar.number\_input("Pilih Tahun Produksi", min\_value=1971, max\_value=2015)

############### sidebar ###############

############### content ###############

st.subheader("Tabel Data Informasi Jumlah Produksi Minyak Mentah di Negara " +str(N))

st.markdown("Adapun tabel data tersebut adalah sebagai berikut.")

list\_produk = get\_data(convert\_NamaNegaraToKodeNegara(N))

df = pd.DataFrame(list\_produk, columns = ['Kode Negara', 'Tahun', 'Produksi'])

st.write(df)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Tabel Data Nama Negara pada File \*csv\* yang Tersedia")

st.markdown("Adapun tabel data tersebut adalah sebagai berikut.")

list\_neg = List\_semua\_negara

df2 = pd.DataFrame(list\_neg, columns = ['Nama Negara'])

st.write(df2)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Grafik Data Jumlah Produksi Minyak Mentah di Negara " +str(N))

st.markdown("Adapun grafik tersebut adalah sebagai berikut.")

convert\_DataToList()

convert\_NamaNegaraToKodeNegara(N)

Nama\_lengkapNegara()

List\_NamaNegara()

get\_data(convert\_NamaNegaraToKodeNegara(N))

Grafik\_JumlahProduksiMinyak(N)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Grafik Data Jumlah Produksi Minyak Mentah di " +str(int(B))+ " Besar Negara pada Tahun " +str(int(T)))

st.markdown("Adapun grafik tersebut adalah sebagai berikut.")

get\_tahun(T)

get\_NbesarNegara(B, T)

get\_ValueNbesarNegara(B, T)

get\_NamaLengkap\_NbesarNegara(B, T)

Grafik\_NbesarNegara(B, T)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Grafik Data Jumlah Kumulatif Produksi Minyak Mentah di "+str(int(B))+ " Besar Negara pada Keseluruhan Tahun")

st.markdown("Adapun grafik tersebut adalah sebagai berikut.")

get\_JumlahProduksiKumulatif(B)

get\_Value\_JumlahProduksiKumulatif(B)

get\_NamaLengkapnegara\_JumlahProduksiKumulatif(B)

Grafik\_NbesarNegaraKumulatif(B)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Terbesar pada Tahun " +str(int(T)))

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

getMinimumPadaTahun(T)

getListNol(T)

list\_maks\_T = informasi\_Tahuntertentu(T, "maksimum")

df3 = pd.DataFrame([list\_maks\_T], columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region'])

st.write(df3)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Terkecil pada Tahun " +str(int(T)))

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

list\_min\_T = informasi\_Tahuntertentu(T, "minimum")

df4 = pd.DataFrame([list\_min\_T], columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region'])

st.write(df4)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Sama Dengan Nol pada Tahun " +str(int(T)))

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

list\_nol\_T = informasi\_Tahuntertentu(T, "Nol")

df5 = pd.DataFrame(list\_nol\_T)

df5.columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region']

st.write(df5)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Terbesar pada Keseluruhan Tahun")

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

list\_maks\_allT = Informasi\_ForAll("maksimum")

df6 = pd.DataFrame([list\_maks\_allT], columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region'])

st.write(df6)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Terkecil pada Keseluruhan Tahun")

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

list\_min\_allT = Informasi\_ForAll("minimum")

df7 = pd.DataFrame([list\_min\_allT], columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region'])

st.write(df7)

st.write("\n")

st.write("\n")

st.write("\n")

st.subheader("Informasi Negara dengan Jumlah Produksi Minyak Mentah Sama Dengan Nol pada Keseluruhan Tahun")

st.markdown("Adapun informasi tersebut adalah sebagai berikut.")

list\_nol\_allT = Informasi\_ForAll("Nol")

df8 = pd.DataFrame(list\_nol\_allT)

df8.columns = ['Nama Negara', 'Kode Negara', 'Region', 'Sub-Region']

st.write(df8)

############### content ###############

1. **DESKRIPSI APLIKASI**

Chart

Description automatically generated