import pickle

import streamlit as st

import pandas as pd

from streamlit\_option\_menu import option\_menu

# loading the saved models

diabetes\_model = pickle.load(open('C:/Users/Legion/Downloads/project/diabetes\_model.sav', 'rb'))

heart\_disease\_model = pickle.load(open('C:/Users/Legion/Downloads/project/heart\_disease\_model.sav','rb'))

parkinsons\_model = pickle.load(open('C:/Users/Legion/Downloads/project/parkinsons\_model.sav', 'rb'))

# sidebar for navigation

with st.sidebar:

selected = option\_menu('Multiple Disease Prediction System',

['Diabetes Prediction',

'Heart Disease Prediction',

'Parkinsons Prediction'],

icons=['activity','heart','person'],

default\_index=0)

# Diabetes Prediction Page

if (selected == 'Diabetes Prediction'):

# page title

st.title('Diabetes Prediction using ML')

# getting the input data from the user

col1, col2, col3 = st.columns(3)

with col1:

Pregnancies = st.text\_input('Number of Pregnancies')

with col2:

Glucose = st.text\_input('Glucose Level')

with col3:

BloodPressure = st.text\_input('Blood Pressure value')

with col1:

SkinThickness = st.text\_input('Skin Thickness value')

with col2:

Insulin = st.text\_input('Insulin Level')

with col3:

BMI = st.text\_input('BMI value')

with col1:

DiabetesPedigreeFunction = st.text\_input('Diabetes Pedigree Function value')

with col2:

Age = st.text\_input('Age of the Person')

# code for Prediction

diab\_diagnosis = ''

if st.button('Diabetes Test Result'):

diab\_prediction = diabetes\_model.predict([[Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age]])

if (diab\_prediction[0] == 1):

diab\_diagnosis = 'The person is diabetic'

else:

diab\_diagnosis = 'The person is not diabetic'

st.subheader("OR")

uploaded\_file = st.file\_uploader("Upload CSV file", type="csv")

if uploaded\_file is not None:

# Read the uploaded CSV file

data = pd.read\_csv(uploaded\_file)

# Display the uploaded data

st.dataframe(data)

# creating a button for Prediction

if st.button('Test Result'):

diab\_prediction = diabetes\_model.predict(data)

if (diab\_prediction[0] == 1):

diab\_diagnosis = 'The person is diabetic'

else:

diab\_diagnosis = 'The person is not diabetic'

st.success(diab\_diagnosis)

# Heart Disease Prediction Page

if (selected == 'Heart Disease Prediction'):

# page title

st.title('Heart Disease Prediction using ML')

col1, col2, col3 = st.columns(3)

with col1:

age = st.text\_input('Age')

with col2:

sex = st.text\_input('Sex')

with col3:

cp = st.text\_input('Chest Pain types(Pain Rate 0-3) ')

with col1:

trestbps = st.text\_input('Resting Blood Pressure')

with col2:

chol = st.text\_input('Serum Cholestoral in mg/dl')

with col3:

fbs = st.text\_input('Fasting Blood Sugar > 120 mg/dl(yes=1 and no=1)')

with col1:

restecg = st.text\_input('Resting Electrocardiographic results')

with col2:

thalach = st.text\_input('Maximum Heart Rate achieved')

with col3:

exang = st.text\_input('Exercise Induced Angina')

with col1:

oldpeak = st.text\_input('ST depression induced by exercise')

with col2:

slope = st.text\_input('Slope of the peak exercise ST segment')

with col3:

ca = st.text\_input('Major vessels colored by flouroscopy(yes=1 and No=0)')

with col1:

thal = st.text\_input('thal: 0 = normal; 1 = fixed defect; 2 = reversable defect')

# code for Prediction

heart\_diagnosis = ''

# creating a button for Prediction

if st.button('Heart Disease Test Result'):

heart\_prediction = heart\_disease\_model.predict([[int(age),int(sex),int(cp),int(trestbps),int(chol),int(fbs),int(restecg),int(thalach),int(exang),float(oldpeak),int(slope),int(ca),int(thal)]])

if (heart\_prediction[0] == 1):

heart\_diagnosis = 'The person is having heart disease'

else:

heart\_diagnosis = 'The person does not have any heart disease'

st.subheader("OR")

uploaded\_file = st.file\_uploader("Upload CSV file", type="csv")

if uploaded\_file is not None:

# Read the uploaded CSV file

data = pd.read\_csv(uploaded\_file)

# Display the uploaded data

st.dataframe(data)

# creating a button for Prediction

if st.button('Test Result'):

heart\_prediction = heart\_disease\_model.predict(data)

if (heart\_prediction[0] == 1):

heart\_diagnosis = 'The person is having heart disease'

else:

heart\_diagnosis = 'The person does not have any heart disease'

st.success(heart\_diagnosis)

# Parkinson's Prediction Page

if (selected == "Parkinsons Prediction"):

# page title

st.title("Parkinson's Disease Prediction using ML")

col1, col2, col3, col4, col5 = st.columns(5)

with col1:

fo = st.text\_input('MDVP Fo(Hz)')

with col2:

fhi = st.text\_input('MDVP Fhi(Hz)')

with col3:

flo = st.text\_input('MDVP Flo(Hz)')

with col4:

Jitter\_percent = st.text\_input('MDVP Jitter(%)')

with col5:

Jitter\_Abs = st.text\_input('MDVP Jitter(Abs)')

with col1:

RAP = st.text\_input('MDVP RAP')

with col2:

PPQ = st.text\_input('MDVP PPQ')

with col3:

DDP = st.text\_input('Jitter DDP')

with col4:

Shimmer = st.text\_input('MDVP Shimmer')

with col5:

Shimmer\_dB = st.text\_input('MDVP Shimmer(dB)')

with col1:

APQ3 = st.text\_input('Shimmer(APQ3)')

with col2:

APQ5 = st.text\_input('Shimmer(APQ5)')

with col3:

APQ = st.text\_input('MDVP APQ')

with col4:

DDA = st.text\_input('Shimmer(DDA)')

with col5:

NHR = st.text\_input('NHR')

with col1:

HNR = st.text\_input('HNR')

with col2:

RPDE = st.text\_input('RPDE')

with col3:

DFA = st.text\_input('DFA')

with col4:

spread1 = st.text\_input('spread1')

with col5:

spread2 = st.text\_input('spread2')

with col1:

D2 = st.text\_input('D2')

with col2:

PPE = st.text\_input('PPE')

# code for Prediction

parkinsons\_diagnosis = ''

# creating a button for Prediction

if st.button("Parkinson's Test Result"):

parkinsons\_prediction = parkinsons\_model.predict([[fo, fhi, flo, Jitter\_percent, Jitter\_Abs, RAP, PPQ,DDP,Shimmer,Shimmer\_dB,APQ3,APQ5,APQ,DDA,NHR,HNR,RPDE,DFA,spread1,spread2,D2,PPE]])

if (parkinsons\_prediction[0] == 1):

parkinsons\_diagnosis = "The person has Parkinson's disease"

else:

parkinsons\_diagnosis = "The person does not have Parkinson's disease"

st.subheader("OR")

uploaded\_file = st.file\_uploader("Upload CSV file", type="csv")

if uploaded\_file is not None:

# Read the uploaded CSV file

data = pd.read\_csv(uploaded\_file)

# Display the uploaded data

st.dataframe(data)

# creating a button for Prediction

if st.button("Test Result"):

parkinsons\_prediction = parkinsons\_model.predict(data)

if (parkinsons\_prediction[0] == 1):

parkinsons\_diagnosis = "The person has Parkinson's disease"

else:

parkinsons\_diagnosis = "The person does not have Parkinson's disease"

st.success(parkinsons\_diagnosis)