**import** **pickle**

**import** **streamlit** **as** **st**

**from** **streamlit\_option\_menu** **import** option\_menu

**from** **sklearn** **import** datasets

**import** **requests**

**from** **streamlit\_lottie** **import** st\_lottie

# loading the saved models

diabetes\_model = pickle.load(open(

'C:/Users/Saketh/Desktop/Multi\_Disease\_Prediction/Diabetes/diabetes\_model.sav', 'rb'))

heart\_disease\_model = pickle.load(open(

'C:/Users/Saketh/Desktop/Multi\_Disease\_Prediction/Heart/V1/heart\_disease\_model.sav', 'rb'))

parkinsons\_model = pickle.load(open(

'C:/Users/Saketh/Desktop/Multi\_Disease\_Prediction/Parkinsons/parkinsons\_model.sav', 'rb'))

# sidebar for navigation

**with** st.sidebar:

selected = st.selectbox('Multiple Disease Prediction System',

['Home',

'Diabetes Prediction',

'Heart Disease Prediction',

'Parkinsons Prediction'],)

**if** (selected == 'Home'):

st.title("Multiple Disease Prediction using Machine Learning")

**def** **load\_lottieurl**(url: str):

r = requests.get(url)

**if** r.status\_code != **200**:

**return** None

**return** r.json()

lottie\_url\_hello = "https://assets3.lottiefiles.com/packages/lf20\_0ssane8p.json"

lottie\_hello = load\_lottieurl(lottie\_url\_hello)

st\_lottie(lottie\_hello, key="hello")

# Diabetes Prediction Page

**if** (selected == 'Diabetes Prediction'):

# page title

st.title('Diabetes Prediction using ML')

# getting the input data from the user

colm1, colm2, colm3 = st.columns(**3**)

**with** colm1:

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

**with** colm2:

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

**with** colm3:

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

**with** colm1:

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

**with** colm2:

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

**with** colm3:

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

**with** colm1:

DiabetesPedigreeFunction = st.text\_input(

'Diabetes Pedigree Function value')

**with** colm2:

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

# code for Prediction

diab\_diagnosis = ''

# creating a button for Prediction

**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.success(diab\_diagnosis)

# Heart Disease Prediction Page

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

# page title

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

colm1, colm2, colm3 = st.columns(**3**)

**with** colm1:

age = st.number\_input('Age', step=**1**, max\_value=**110**)

**with** colm2:

sex = st.selectbox('Sex (female=0, male=1)', (**0**, **1**))

**with** colm3:

cp = st.selectbox('Chest pain type', (**0**, **1**, **2**, **3**))

**with** colm1:

trestbps = st.number\_input('Resting Blood Pressure', step=**1**)

**with** colm2:

chol = st.number\_input('Serum Cholestoral in mg/dl', step=**1**)

**with** colm3:

fbs = st.selectbox('Fasting blood sugar', (**0**, **1**))

**with** colm1:

restecg = st.number\_input(

'Resting Electrocardiographic results', step=**1**)

**with** colm2:

thalach = st.number\_input('Maximum Heart Rate achieved', step=**1**)

**with** colm3:

exang = st.number\_input('Exercise Induced Angina', step=**1**)

**with** colm1:

oldpeak = st.number\_input(

'ST depression induced by exercise', step=**0.1**)

**with** colm2:

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

**with** colm3:

ca = st.selectbox('Major vessels colored by flourosopy', (**0**, **1**, **2**, **3**))

**with** colm1:

thal = st.selectbox(

'thal: 0 = normal; 1 = fixed defect; 2 = reversable defect', (**0**, **1**, **2**))

# code for Prediction

heart\_diagnosis = ''

# creating a button for Prediction

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

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

[[age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, 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.success(heart\_diagnosis)

# Parkinson's Prediction Page

**if** (selected == "Parkinsons Prediction"):

# page title

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

colm1, colm2, colm3, col4, col5 = st.columns(**5**)

**with** colm1:

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

**with** colm2:

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

**with** colm3:

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** colm1:

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

**with** colm2:

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

**with** colm3:

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** colm1:

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

**with** colm2:

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

**with** colm3:

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

**with** col4:

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

**with** col5:

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

**with** colm1:

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

**with** colm2:

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

**with** colm3:

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

**with** col4:

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

**with** col5:

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

**with** colm1:

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

**with** colm2:

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.success(parkinsons\_diagnosis)