**Simple Framework for Heart Disease Detection Application**

# Import necessary libraries

import numpy as np

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import accuracy\_score

# Load your dataset (replace 'your\_dataset.csv' with your dataset)

data = pd.read\_csv('your\_dataset.csv')

# Preprocess the data (cleaning, handling missing values, feature extraction, etc.)

# Split the data into features (X) and target (y)

X = data.drop('target', axis=1)

y = data['target']

# Split the dataset into training and testing sets

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Initialize and train a machine learning model (Random Forest in this example)

model = RandomForestClassifier(random\_state=42)

model.fit(X\_train, y\_train)

# Make predictions on the test set

y\_pred = model.predict(X\_test)

# Evaluate the model's accuracy

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Model Accuracy: {accuracy \* 100:.2f}%")

# Now, you can create a simple function to make predictions based on user input

def predict\_heart\_disease(symptoms):

# Preprocess user input and convert it into the format expected by the model

user\_input = np.array(symptoms).reshape(1, -1)

# Make predictions

prediction = model.predict(user\_input)

# Return the prediction (1 for heart disease, 0 for no heart disease)

return prediction[0]

# Example usage

user\_symptoms = [63, 1, 3, 145, 233, 1, 0, 150, 0, 2.3, 0, 0, 1]

result = predict\_heart\_disease(user\_symptoms)

if result == 1:

print("The model predicts heart disease.")

else:

print("The model predicts no heart disease.")