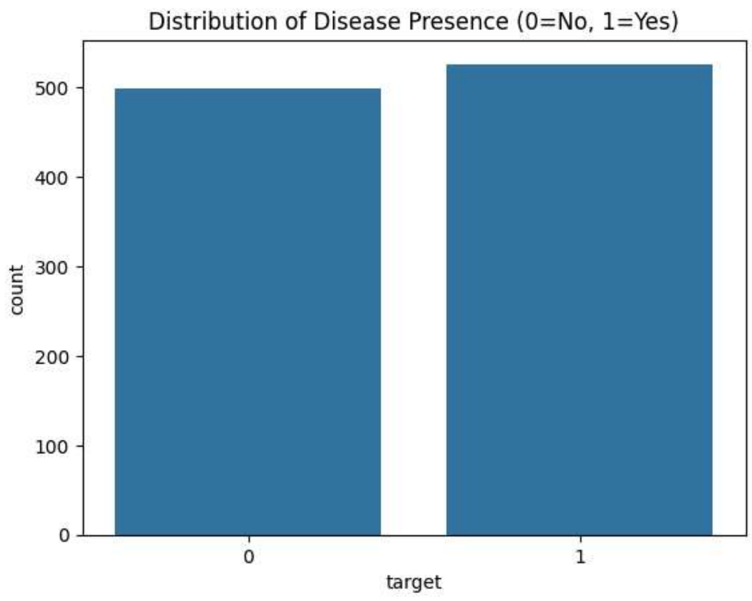
```
import pandas as pd

df = pd.read_csv('/content/HeartDiseaseTrain-Test.csv')

# 3️⃣ Data Exploration
print("\nFirst 5 rows of the dataset:")
print(df.head())

print("\nShape of the dataset:", df.shape)
print("\nColumns:", df.columns.tolist())
print("\nData Types & Missing Values:")
df.info()

print("\nSummary Statistics:")
print(df.describe())

print("\nMissing values per column:\n", df.isnull().sum())
print("\nDuplicate rows:", df.duplicated().sum())

# 4️⃣ Data Visualization (Optional, for better understanding)
import seaborn as sns
import matplotlib.pyplot as plt

# Example: Plot distribution of target variable (assuming 'target' is the disease indicator)
sns.countplot(x='target', data=df)
plt.title('Distribution of Disease Presence (0=No, 1=Yes)')
plt.show()

# 5️⃣ Prepare Features and Target
target = 'target'  # Change to actual target column if named differently
features = df.columns.drop(target)
print("\nFeatures:", features.tolist())

# 6️⃣ Convert Categorical Columns
categorical_cols = df.select_dtypes(include=['object']).columns.tolist()
print("\nCategorical Columns:", categorical_cols)

df_encoded = pd.get_dummies(df, drop_first=True)

# 3️⃣ Check for Missing Values
print("\n🟠 Missing values per column:\n", df.isnull().sum())

# Fill missing numeric columns with median
numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns
for col in numeric_cols:
    if df[col].isnull().sum() > 0:
        median_val = df[col].median()
        df[col].fillna(median_val, inplace=True)
        print(f"Filled missing values in '{col}' with median: {median_val}")

# Fill missing categorical columns with mode
categorical_cols = df.select_dtypes(include=['object']).columns
for col in categorical_cols:
    if df[col].isnull().sum() > 0:
        mode_val = df[col].mode()[0]
        df[col].fillna(mode_val, inplace=True)
        print(f"Filled missing values in '{col}' with mode: {mode_val}")

# 4️⃣ Check for Duplicates
duplicates = df.duplicated().sum()
print(f"\n🟠 Duplicate rows: {duplicates}")
if duplicates > 0:
    df.drop_duplicates(inplace=True)
    print("✅ Duplicates removed.")

# 5️⃣ Detect and Handle Outliers (Optional: here we cap them using IQR method)
def cap_outliers(column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower = Q1 - 1.5 * IQR
    upper = Q3 + 1.5 * IQR
    df[column] = df[column].clip(lower, upper)
    print(f"✅ Outliers capped for '{column}'")

for col in numeric_cols:
    cap_outliers(col)

# 6️⃣ Encode Categorical Features
print("\n✅ Categorical Columns:", categorical_cols.tolist())
df_encoded = pd.get_dummies(df, drop_first=True)

# 7️⃣ Feature Scaling
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
X = df_encoded.drop('target', axis=1)  # Replace 'target' with your actual target column
y = df_encoded['target']

X_scaled = scaler.fit_transform(X)

# 8️⃣ Train-Test Split
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# 9️⃣ Model Training
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report

model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# 🔟 Predictions
y_pred = model.predict(X_test)

# 🔍 Evaluation
print("\nAccuracy:", accuracy_score(y_test, y_pred))
print("\nConfusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# 🔮 Predicting a new patient
# Example: Replace with actual input values
new_patient = {
    'age': 55,
    'sex': 1,
    'cp': 3,
    'trestbps': 140,
    'chol': 250,
    'fbs': 0,
    'restecg': 1,
    'thalach': 150,
    'exang': 0,
    'oldpeak': 2.3,
    'slope': 0,
    'ca': 0,
    'thal': 2
}

# Convert input to DataFrame
new_df = pd.DataFrame([new_patient])

# Combine with original data to ensure same columns
df_temp = pd.concat([df.drop(target, axis=1), new_df], ignore_index=True)
df_temp_encoded = pd.get_dummies(df_temp, drop_first=True)

# Reindex to match training columns
df_temp_encoded = df_temp_encoded.reindex(columns=df_encoded.drop(target, axis=1).columns, fill_value=0)

# Scale new input
new_input_scaled = scaler.transform(df_temp_encoded.tail(1))

# Make prediction
predicted_disease = model.predict(new_input_scaled)

print("\nPredicted Disease Presence (1=Yes, 0=No):", predicted_disease[0])
```

```
First 5 rows of the dataset:
   age     sex chest_pain_type  resting_blood_pressure  cholestoral  \
0   52    Male  Typical angina                     125          212
1   53    Male  Typical angina                     140          203
2   70    Male  Typical angina                     145          174
3   61    Male  Typical angina                     148          203
4   62  Female  Typical angina                     138          294

      fasting_blood_sugar               rest_ecg  Max_heart_rate  \
0    Lower than 120 mg/ml  ST-T wave abnormality             168
1  Greater than 120 mg/ml                 Normal             155
2    Lower than 120 mg/ml  ST-T wave abnormality             125
3    Lower than 120 mg/ml  ST-T wave abnormality             161
4  Greater than 120 mg/ml  ST-T wave abnormality             106

  exercise_induced_angina  oldpeak        slope vessels_colored_by_flourosopy  \
0                      No      1.0  Downsloping                           Two
1                     Yes      3.1    Upsloping                          Zero
2                     Yes      2.6    Upsloping                          Zero
3                      No      0.0  Downsloping                           One
4                      No      1.9         Flat                         Three

         thalassemia  target
0  Reversable Defect       0
1  Reversable Defect       0
2  Reversable Defect       0
3  Reversable Defect       0
4      Fixed Defect       0

Shape of the dataset: (1025, 14)

Columns: ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'cholestoral', 'fasting_blood_sugar', 'rest_ecg', 'Max_heart_rat

Data Types & Missing Values:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1025 entries, 0 to 1024
Data columns (total 14 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   age                            1025 non-null   int64  
 1   sex                            1025 non-null   object 
 2   chest_pain_type                1025 non-null   object 
 3   resting_blood_pressure         1025 non-null   int64  
 4   cholestoral                    1025 non-null   int64  
 5   fasting_blood_sugar            1025 non-null   object 
 6   rest_ecg                       1025 non-null   object 
 7   Max_heart_rate                 1025 non-null   int64  
 8   exercise_induced_angina        1025 non-null   object 
 9   oldpeak                        1025 non-null   float64
 10  slope                          1025 non-null   object 
 11  vessels_colored_by_flourosopy  1025 non-null   object 
 12  thalassemia                    1025 non-null   object 
 13  target                         1025 non-null   int64  
dtypes: float64(1), int64(5), object(8)
memory usage: 112.2+ KB

Summary Statistics:
               age  resting_blood_pressure  cholestoral  Max_heart_rate  \
count  1025.000000             1025.000000   1025.00000     1025.000000
mean     54.434146              131.611707    246.00000      149.114146
std       9.072290               17.516718     51.59251       23.005724
min      29.000000               94.000000    126.00000       71.000000
25%      48.000000              120.000000    211.00000      132.000000
50%      56.000000              130.000000    240.00000      152.000000
75%      61.000000              140.000000    275.00000      166.000000
max      77.000000              200.000000    564.00000      202.000000

           oldpeak       target
count  1025.000000  1025.000000
mean      1.071512     0.513171
std       1.175053     0.500070
min       0.000000     0.000000
25%       0.000000     0.000000
50%       0.800000     1.000000
75%       1.800000     1.000000
max       6.200000     1.000000

Missing values per column:
 age                              0
sex                              0
chest_pain_type                  0
resting_blood_pressure           0
cholestoral                      0
```

```
fasting_blood_sugar           0
rest_ecg                      0
Max_heart_rate                0
exercise_induced_angina       0
oldpeak                       0
slope                         0
vessels_colored_by_flourosopy 0
thalassemia                   0
target                        0
dtype: int64

Duplicate rows: 723
```



```
Features: ['age', 'sex', 'chest_pain_type', 'resting_blood_pressure', 'cholestoral', 'fasting_blood_sugar', 'rest_ecg', 'Max_heart_ra

Categorical Columns: ['sex', 'chest_pain_type', 'fasting_blood_sugar', 'rest_ecg', 'exercise_induced_angina', 'slope', 'vessels_color

🟠 Missing values per column:
 age                           0
sex                           0
chest_pain_type               0
resting_blood_pressure        0
cholestoral                   0
fasting_blood_sugar           0
rest_ecg                      0
Max_heart_rate                0
exercise_induced_angina       0
oldpeak                       0
slope                         0
vessels_colored_by_flourosopy 0
thalassemia                   0
target                        0
dtype: int64

🟠 Duplicate rows: 723
✅ Duplicates removed.
✅ Outliers capped for 'age'
✅ Outliers capped for 'resting_blood_pressure'
✅ Outliers capped for 'cholestoral'
✅ Outliers capped for 'Max_heart_rate'
✅ Outliers capped for 'oldpeak'
✅ Outliers capped for 'target'

✅ Categorical Columns: ['sex', 'chest_pain_type', 'fasting_blood_sugar', 'rest_ecg', 'exercise_induced_angina', 'slope', 'vessels_co

Accuracy: 0.8360655737704918

Confusion Matrix:
 [[24  8]
 [ 2 27]]

Classification Report:
               precision    recall  f1-score   support

           0       0.92      0.75      0.83        32
           1       0.77      0.93      0.84        29

    accuracy                           0.84        61
   macro avg       0.85      0.84      0.84        61
```

```
weighted avg       0.85      0.84      0.84        61


Predicted Disease Presence (1=Yes, 0=No): 1
```

```
!pip install gradio
```

```
Requirement already satisfied: gradio in /usr/local/lib/python3.11/dist-packages (5.29.0)
Requirement already satisfied: aiofiles<25.0,>=22.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (24.1.0)
Requirement already satisfied: anyio<5.0,>=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.9.0)
Requirement already satisfied: fastapi<1.0,>=0.115.2 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.115.12)
Requirement already satisfied: ffmpy in /usr/local/lib/python3.11/dist-packages (from gradio) (0.5.0)
Requirement already satisfied: gradio-client==1.10.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (1.10.0)
Requirement already satisfied: groovy~=0.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.2)
Requirement already satisfied: httpx>=0.24.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.28.1)
Requirement already satisfied: huggingface-hub>=0.28.1 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.31.1)
Requirement already satisfied: jinja2<4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.1.6)
Requirement already satisfied: markupsafe<4.0,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.0.2)
Requirement already satisfied: numpy<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.0.2)
Requirement already satisfied: orjson~=3.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (3.10.18)
Requirement already satisfied: packaging in /usr/local/lib/python3.11/dist-packages (from gradio) (24.2)
Requirement already satisfied: pandas<3.0,>=1.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.2.2)
Requirement already satisfied: pillow<12.0,>=8.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (11.2.1)
Requirement already satisfied: pydantic<2.12,>=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.11.4)
Requirement already satisfied: pydub in /usr/local/lib/python3.11/dist-packages (from gradio) (0.25.1)
Requirement already satisfied: python-multipart>=0.0.18 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.0.20)
Requirement already satisfied: pyyaml<7.0,>=5.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (6.0.2)
Requirement already satisfied: ruff>=0.9.3 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.11.9)
Requirement already satisfied: safehttpx<0.2.0,>=0.1.6 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.1.6)
Requirement already satisfied: semantic-version~=2.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (2.10.0)
Requirement already satisfied: starlette<1.0,>=0.40.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.46.2)
Requirement already satisfied: tomlkit<0.14.0,>=0.12.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.13.2)
Requirement already satisfied: typer<1.0,>=0.12 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.15.3)
Requirement already satisfied: typing-extensions~=4.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (4.13.2)
Requirement already satisfied: uvicorn>=0.14.0 in /usr/local/lib/python3.11/dist-packages (from gradio) (0.34.2)
Requirement already satisfied: fsspec in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (2025.3.2)
Requirement already satisfied: websockets<16.0,>=10.0 in /usr/local/lib/python3.11/dist-packages (from gradio-client==1.10.0->gradio) (1
Requirement already satisfied: idna>=2.8 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (3.10)
Requirement already satisfied: sniffio>=1.1 in /usr/local/lib/python3.11/dist-packages (from anyio<5.0,>=3.0->gradio) (1.3.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (2025.4.26)
Requirement already satisfied: httpcore==1.* in /usr/local/lib/python3.11/dist-packages (from httpx>=0.24.1->gradio) (1.0.9)
Requirement already satisfied: h11>=0.16 in /usr/local/lib/python3.11/dist-packages (from httpcore==1.*->httpx>=0.24.1->gradio) (0.16.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (3.18.0)
Requirement already satisfied: requests in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (2.32.3)
Requirement already satisfied: tqdm>=4.42.1 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (4.67.1)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.0 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.28.1->gradio) (1
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2.9.0.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0.7
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (2.33
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio) (0
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (8.1.8)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (1.5.4)
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (13.9.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas<3.0,>=1.0->gradi
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12->g
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=0.12-
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.28.1->gr
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0->typer<1
```

```
import gradio as gr
import joblib
import pandas as pd

# Load the trained model
#model = joblib.load("heart_disease_model.pkl")

# Prediction function
def predict_heart_disease(age, sex, chest_pain_type, resting_blood_pressure, cholestoral,
                          fasting_blood_sugar, rest_ecg, max_heart_rate, exercise_induced_angina,
                          oldpeak, slope, vessels_colored_by_flourosopy, thalassemia):

    input_data = pd.DataFrame({
        "age": [age],
        "sex": [sex],
        "chest_pain_type": [chest_pain_type],
        "resting_blood_pressure": [resting_blood_pressure],
        "cholestoral": [cholestoral],
        "fasting_blood_sugar": [fasting_blood_sugar],
        "rest_ecg": [rest_ecg],
        "Max_heart_rate": [max_heart_rate],
        "exercise_induced_angina": [exercise_induced_angina],
            "oldpeak": [oldpeak],
            "slope": [slope],
            "vessels_colored_by_flourosopy": [vessels_colored_by_flourosopy],
            "thalassemia": [thalassemia]
        })

        prediction = model.predict(input_data)[0]
        return "🔴 High Risk of Heart Disease" if prediction == 1 else "🟢 Low Risk of Heart Disease"

    # Gradio interface
    demo = gr.Interface(
        fn=predict_heart_disease,
        inputs=[
```