

# Heart Disease Prediction : HeartWise

HeartWise.

Home

Advices

contact

Login

Get Started

## Stay One Step Ahead of Heart Disease with Smart Predictions

Discover personalized insights tailored to your unique heart health. Gain valuable predictions to identify risks early and take proactive measures.

Join us

Explore



Réalisé par : Zayneb Hamdi

Groupe : INGA2- groupe1- sousgroupe2

Année universitaire :2024/2025

## Table des matières

I.	Introduction :	3
II.	Base de données :	3
III.	Backend : le modèle de Prédiction	5
IV.	Frontend : les interfaces de l'application	5
V.	Etapes :	7
VI.	Améliorations futures :	11
VII.	Conclusion :	12

## I. Introduction :

Les maladies cardiaques demeurent parmi les principales causes de mortalité dans le monde. La détection précoce peut sauver la vie des malades. Ce projet vise à développer une application capable de prédire le risque de maladies cardiaques en fonction de divers paramètres médicaux, tels que l'âge, le sexe, le cholestérol, la pression artérielle, etc.

Vous trouverez dans cette application :

- ✓ Une interface utilisateur permettant la saisie de données
- ✓ Un modèle de machine Learning entraîné sur des données cliniques
- ✓ Une base de données pour stocker les données nécessaires à évaluer la précision de prédiction

Objectifs de l'application :

- ✓ Atteindre une précision supérieure à 80%
- ✓ Concevoir une interface simple accessible à des utilisateurs non techniques
- ✓ Stocker les données afin de suivre la performance de l'application

Outils de développement :

- ✓ Python : pour l'entraînement de modèle de machine Learning et le backend
- ✓ Flask : Framework léger pour le backend
- ✓ MongoDB : stocker les données de prédictions
- ✓ Scikit-learn : librairie python utilisée pour entraîner le modèle de prédiction
- ✓ Html/css : la conception des interfaces utilisateurs

## II. Base de données :

La base de donnée utilisée est MongoDB (NoSQL) pour stocker les données cliniques ainsi que le résultat de prédiction obtenues pour suivre la précision de modèle afin de le développer.

- ✓ DB : « prédiction »

- ✓ Collection : « cases » : pour chaque cas l'id, les données cliniques associés et le résultat

Pour l'entraînement du modèle on utilisera une dataset  
« heart\_disease\_data.csv » :

- ✓ Age : l'âge de patient en année
- ✓ Sex : le genre de patient :
  - 0 : female
  - 1 : male
- ✓ cp : chest pain type
  - 0 = Typical angina
  - 1 = Atypical angina
  - 2 = Non-anginal pain
  - 3 = Asymptomatic
- ✓ trestbps - resting blood pressure
- ✓ chol - serum cholesterol
- ✓ fbs - fasting blood sugar
  - 1 = True
  - 0 = False
- ✓ restecg - resting electrocardiographic results
  - 0 = Normal
  - 1 = Having ST-T wave abnormality
  - ✓ 2 = Showing probable or definite left ventricular
- ✓ thalach - maximum heart rate achieved
- ✓ exang - exercise-induced angina
  - 1 = Yes
  - 0 = No
- ✓ oldpeak - ST depression
- ✓ slope - slope of the peak exercise ST segment
  - 0 = Upsloping
  - 1 = Flat
  - 2 = Downsloping
- ✓ ca - number of major vessels :(0-3)
- ✓ thal – thalassemia  
A blood disorder condition :

- 1 = Normal
- 2 = Fixed defect
- 3 = Reversible defect
- ✓ target - diagnosis of heart disease
  - 1 = Presence of heart disease
  - 0 = Absence of heart disease

### III. Backend : le modèle de Prédiction

- ✓ NumPy : Pour manipuler les données sous forme de tableaux.
- ✓ Pandas : Pour charger et analyser les données tabulaires.
- ✓ Scikit-learn : Pour diviser les données, entraîner le modèle et mesurer ses performances.
- ✓ Flask : Pour créer le backend et exposer les fonctionnalités via une interface web.
- ✓ PyMongo : Pour interagir avec MongoDB et enregistrer les données utilisateur.
- ✓ MongoDB : Pour stocker les données et les résultats des prédictions.

Régression Logistique (Logistic Regression) : Algorithme de classification supervisée, bien adapté à ce type de problème binaire (0 = Pas de risque, 1 = Risque).

### IV. Frontend : les interfaces de l'application

Cette application comporte 3 interfaces :

- ✓ Page d'accueil « index.html » : d'après la page d'accueil l'utilisateur peut savoir le but et la façon d'utiliser l'application. le métier : la santé .


## Stay One Step Ahead of Heart Disease with Smart Predictions

Discover personalized insights tailored to your unique heart health. Gain valuable predictions to identify risks early and take proactive measures.

[Join us](#)[Explore](#)

- ✓ Page de formulaire « formulaire.html » : avec des labels et des phrases claires, l'utilisateur peut saisir ses données cliniques facilement. Le formulaire est guidé et facile à utiliser

### Heart Disease Risk Assessment Form



Age:	<input type="text"/>	Sex:	<input type="text" value="Male"/>
Resting Blood Pressure (mmHg):	<input type="text"/>	Chest Pain Type:	<input type="text" value="Typical Angina"/>
Cholesterol Level (mg/dL):	<input type="text"/>	Fasting Blood Sugar > 120 mg/dL:	<input type="text" value="Yes"/>
Maximum Heart Rate Achieved:	<input type="text"/>	Resting ECG Results:	<input type="text" value="Normal"/>
ST Depression:	<input type="text"/>	Exercise-Induced Angina:	<input type="text" value="Yes"/>
Number of Major Vessels (0-4):	<input type="text"/>	Slope of ST Segment:	<input type="text" value="Upsloping"/>
Thalassemia:	<input type="text" value="Normal"/>	<input type="button" value="Get Result"/>	

- ✓ Page de résultat « resultat.html » : contient le résultat de la prédiction ainsi que le pourcentage de précision de prédiction. S'il y a un risque l'application affiche un message qui l'incite à contacter son médecin.



**Good heart !**

this result is about 82% accurate

try again



**heart disease risk please check with doctor!**

this result is about 82% accurate

try again

## V. Etapes :

1. Installer les dépendances nécessaires :

```
PS C:\Users\hamdi\OneDrive\Desktop\heart disease prediction> pip install pandas numpy scikit-learn flask pymongo  
Requirement already satisfied: pandas in c:\users\hamdi\appdata\local\programs\python\python312\lib\site-packages (2.2.3)
```

2. Importer les bibliothèques :

```

1  import numpy as np
2  import pandas as pd
3  from sklearn.model_selection import train_test_split
4  from sklearn.linear_model import LogisticRegression
5  from sklearn.metrics import accuracy_score
6  from flask import Flask,render_template,request
7  from pymongo import mongo_client
8

```

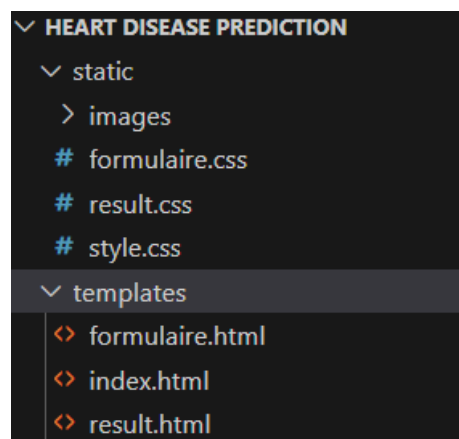
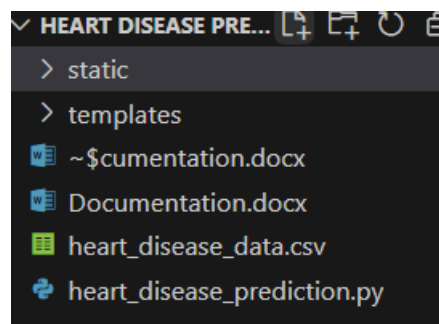
3. Créer une application Flask et créer la base de données

```

8
9  app=Flask(__name__)
10 client = mongo_client.MongoClient("mongodb://localhost:27017/")
11 db=client["prediction_db"]
12 cases=db["cases"]
13

```

4. L'arborescence de projet est le suivant



5. Avec pandas lire les données de dataset



```

19 heart_data=pd.read_csv('heart_disease_data.csv')
20 """heart_data.head()
21 heart_data.tail()
22 heart_data.shape
23 heart_data.info()
24 heart_data.isnull().sum()
25 heart_data.describe()
26 heart_data['target'].value_counts()"""
27

```

## 6. Séparer les données pour créer des variables d'entrées

```

28 x=heart_data.drop('target',axis=1)
29 y=heart_data['target']
30 #print(x)
31 #print(y)

```

## 7. Séparer les données pour former les données d'entraînement des données de test

```

x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,stratify=y,random_state=2)
#print(x.shape,x_train.shape,x_test.shape)

```

## 8. Entraîner le modèle

```

34 model=LogisticRegression()
35 model.fit(x_train,y_train)
36 x_train_prediction=model.predict(x_train)

```

## 9. Vérifier la précision du modèle

```

36 x_train_prediction=model.predict(x_train)
37 training_data_accuracy=accuracy_score(x_train_prediction,y_train)
38 print("accuracy on training data: ",training_data_accuracy)
39 x_test_prediction=model.predict(x_test)
40 test_data_accuracy=accuracy_score(x_test_prediction,y_test)
41 print("accuracy on test data",test_data_accuracy)
42 #*****

```

```

accuracy on training data: 0.8512396694214877
accuracy on test data 0.819672131147541

```

## 10.Gérer les routes de l'application Flask

```
45 @app.route("/")
46 def home():
47     return render_template("index.html")
48
49 @app.route("/formulaire.html")
50 def formulaire():
51     return render_template("formulaire.html")
52
```

## 11.Récupérer les données nécessaires de formulaire

```
53 @app.route("/submit",methods=["POST"])
54 def submit():
55     age = int(request.form['age'])
56     sex = int(request.form['sex'])
57     cp = int(request.form['cp'])
58     trestbps = int(request.form['trestbps'])
59     chol = int(request.form['chol'])
60     fbs = int(request.form['fbs'])
61     restecg = int(request.form['restecg'])
62     thalach = int(request.form['thalach'])
63     exang = int(request.form['exang'])
64     oldpeak = float(request.form['oldpeak'])
65     slope = int(request.form['slope'])
66     ca = int(request.form['ca'])
67     thal = int(request.form['thal'])
68     input_data = (age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal)
```

## 12.Convertir en tableau NumPy

```
69 input_data_as_numpy_array=np.asarray(input_data)
70 input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
71 prediction=model.predict(input_data_reshaped)
72 print(prediction)
```

## 13.Prédiction avec le modèle entraîné

```
73 if(prediction[0]==0):
74     result='Good heart !'
75 else:
76     result='heart disease risk please check with doctor!'
77
```

```

78     object = {
79         "age": age,
80         "sex": sex,
81         "cp": cp,
82         "trestbps": trestbps,
83         "chol": chol,
84         "fbs": fbs,
85         "restecg": restecg,
86         "thalach": thalach,
87         "exang": exang,
88         "oldpeak": oldpeak,
89         "slope": slope,
90         "ca": ca,
91         "thal": thalach,
92         "result": result,
93     }
94     cases.insert_one(object)
95     return render_template("result.html", resultat=result)
96

```

#### 14. Run l'application Flask

```

115     if __name__ == "__main__":
116         app.run(debug=True)

```

#### 15. Exécuter le programme

```

\heart disease prediction> py .\heart_disease_prediction.py
ams\Python\Python312\Lib\site-packages\sklearn\linear_model\_logistic
DEBUG: model on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit

```

### VI. Améliorations futures :

- ✓ Visualisation des données
- ✓ Ajout d'une authentification des patients
- ✓ Hébergement de l'application sur un service cloud

## VII. Conclusion :

Ce mini projet est basé sur l'entraînement d'un modèle de machine Learning qui fait la prédiction des maladies cardiaques. Grâce à l'utilisation de la régression logistique, le modèle offre une précision raisonnable basée sur des données fiables.