

BÁO CÁO ĐỒ ÁN MÔN HỌC

Lớp: IE221.O23.CNCL

SINH VIÊN THỰC HIỆN:

Mã sinh viên: 21522747

Họ và tên: Trịnh Tuấn Tú

TÊN ĐỀ TÀI: XÂY DỰNG ÚNG DỤNG DỰ ĐOÁN BỆNH TIM BẰNG HỌC MÁY

1. Giới thiệu đồ án

Với sự phát triển mạnh mẽ của con người, công nghệ thông tin đã trở thành một phần không thể thiếu trong ngành y tế. Trong lĩnh vực y tế, việc áp dụng học máy để dự đoán các chứng bệnh đã trở nên phổ biến và hữu ích. Trong đó, bệnh tim là một trong những nguy cơ sức khỏe hàng đầu của con người. Việc phát hiện và dự đoán sớm các vấn đề về tim có thể cứu sống rất nhiều người.

Đề tài "Xây dựng ứng dụng dự đoán bệnh tim bằng học máy" nhằm mục tiêu phát triển một hệ thống máy học thông minh có khả năng dự đoán nguy cơ bệnh tim của một người dựa trên các thông số y tế cơ bản. Cung cấp khả năng dự đoán bệnh tim, cho phép người dùng nhập liệu về các yếu tố như huyết áp, mức độ hoạt động, nhịp tim và các yếu tố khác.

Bằng phương pháp máy học RandomForestClassifier, hệ thống sẽ phân tích dữ liệu đầu vào và đưa ra dự đoán về nguy cơ mắc bệnh tim của người dùng. Kết quả được hiển thị thông qua biểu đồ và báo cáo. Từ đó, đưa ra được các khuyến cáo cho người dùng.

2. Quá trình thực hiện

a. Tuần 1:

❖ Tiền xử lý dữ liệu:

- i. Sử dụng bộ dataset: <https://www.kaggle.com/datasets/johnsmith88/heart-disease-dataset>

- ii. Import thư viện

- iii. Load dataset: xem các thông tin cơ bản của dataset như shape, info, missing value

- iv. Khám phá dữ liệu:

❖ Trực quan hóa dữ liệu:

➤ Vẽ dữ liệu địa lý với dạng đồ thị phân tán scatter.

➤ Vẽ biểu đồ phân phối của các biến số trong dataset.

➤ Sử dụng Skewness để xem độ lệch của một biến số so với phân phối chuẩn.

➤ Loại bỏ các giá trị Outlier (là các điểm dữ liệu mà có giá trị rất khác biệt, cách xa so với phần còn lại của tập dữ liệu.)

❖ Chuẩn bị dữ liệu:

➤ Co giãn thuộc tính (feature scaling)

➤ Tạo các feature mới từ dataset ban đầu để cải thiện hiệu suất (đã tạo 3 feature mới là age_group, blood_pressure_group, categorize_cholesterol)

➤ Handling categorical data (biến đổi dữ liệu được biểu diễn dưới dạng “categorical variables” thành dạng phù hợp với mô hình). Ở bộ dataset này có 3 loại categorical:

- Nominal Features:

- ◆ cp (chest pain type): 4 values (0,1,2,3)

- ◆ restecg (resting electrocardiographic results): 3 values (0,1,2)

- ◆ Thal : 0 = normal, 1 = fixed defect, 2 = reversible defect

- Ordinal Features:

- ◆ age_group : 0 = Young, 1 = Middle-aged, 2 = 'Elderly', 3 = 'Very Elderly'

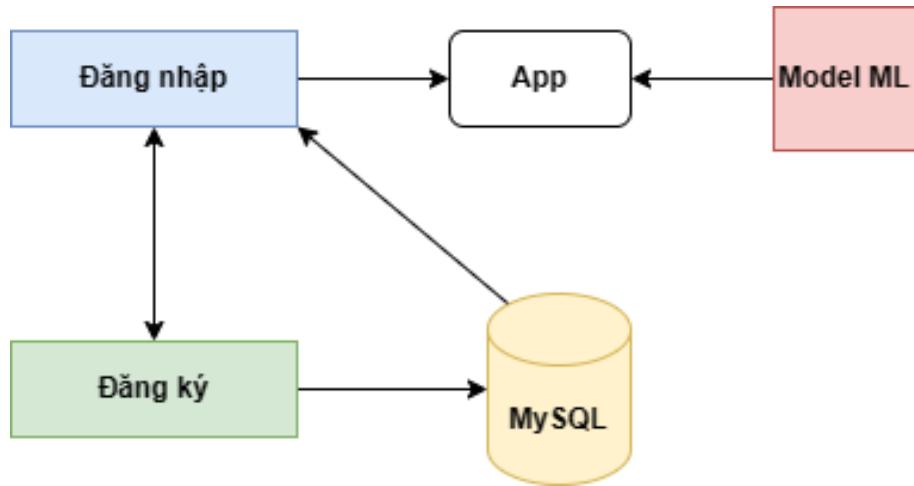
- ◆ blood_pressure_group : 0 = Low, 1 = Normal, 2 = High

- ◆ categorize_cholesterol : 0 = Desirable, 1 = Borderline High, 2 = High

- Binary
 - ◆ Sex (1 = Male, 0 = Female)
 - ◆ Fasting blood sugar > 120 mg/dl (True, False)
 - ◆ Exercise induced angina (True, False)
- OneHotEncoding cho Nominal.
- Encode ordinal categories
- Chia dữ liệu để train, test (tỉ lệ 0,7 train; 0,3 test)
- ❖ Xây dựng model:
 - Huấn luyện trên nhiều phương pháp máy học: Logistic Regression, Decision Tree Classifier, Random Forest Classifier, SVM
 - Sử dụng các độ đo: accuracy, roc auc, f1-weighted để đánh giá và so sánh độ chính xác của các mô hình.
 - Phương pháp có độ chính xác cao nhất là Random Forest Classifier. Chọn phương pháp này để phát triển đề tài.
 - Xây dựng model.py, Run.py để chạy thử nghiệm và data_preprocessing.py để tự động xử lý dữ liệu khi người dùng nhập vào từ giao diện.

b. Tuần 2:

- ❖ Thiết kế giao diện.
- ❖ Pipeline:



- ❖ Lập trình giao diện trang chủ ở file main.py
 - Tìm kiếm hình ảnh
 - Lập trình các ô nhập dữ liệu
 - Lập trình các button Login, Logout, Save, Analysis, info
 - Hiển thị các biểu đồ mẫu
- ❖ Lập trình giao diện trang đăng nhập (login.py), đăng ký (SignUp.py)
 - Lập trình giao diện đăng nhập
 - Lập trình các ô nhập thông tin: Username, Password
 - Lập trình button Login, button SignUp nếu chưa có tài khoản
 - Lập trình giao diện đăng ký
 - Lập trình các ô nhập thông tin Username, Password, Conform password
 - Lập trình button SignUp, button Login nếu đã có tài khoản
- ❖ Lập trình trang con info.py chứa mô tả các thông tin về dataset. Trang này mô tả các thông tin của dữ liệu mà người dùng cần nhập vào.

c. Tuần 3:

- ❖ Gộp 2 file model.py và data_preprocessing.py thành file heart_prediction.py.
 - File heart_prediction.py sẽ có các phương thức tiền xử lý dữ liệu dataset, huấn luyện mô hình, tiền xử lý dữ liệu mới (do người dùng nhập vào).
- ❖ Lập trình các ràng buộc để cho người dùng nhập đầy đủ thông tin cần thiết để dự đoán, in ra thông báo lỗi nếu người dùng nhập thiếu thông tin để dự đoán.
- ❖ Lập trình chức năng dự đoán
 - Import heart_prediction vào main.py
 - Xây dựng phương thức Analysis
 - Lấy thông tin người dùng nhập vào.
 - Dùng phương thức preprocess_data (heart_prediction.py) để tiền xử lý dữ liệu dataset.
 - Dùng phương thức train_model (heart_prediction.py) để huấn luyện mô hình.

- Dùng phương thức preprocess_new_data (heart_prediction.py) để tiền xử lý dữ liệu mới.
- Sử dụng model đã train để dự đoán dữ liệu mới
- Nếu kết quả là 1 thì mắc bệnh tim và ngược lại, 0 là bình thường. In ra màn hình thông báo kết quả dự đoán.

➤ Khi click vào button Analysis thì sẽ hiển thị kết quả dự đoán lên màn hình.

❖ Connect vào MySQL

❖ Xây dựng database lưu thông tin người dùng

➤ Lập trình file database.py. Trong file này có class DatabaseManager gồm các phương thức.

- `__init__(self, host, user, password, database_name)`: Phương thức khởi tạo một đối tượng DatabaseManager thiết lập các thuộc tính cơ bản như host, user, password, database_name, connection, và cursor.
- `connect(self)`: Phương thức này kết nối đến cơ sở dữ liệu MySQL sử dụng thông tin được cung cấp.
- `create_tables(self)`: Phương thức này tạo các bảng trong cơ sở dữ liệu nếu chúng chưa tồn tại.
- `save_data(self, name, today, age, B2, C2, D2, E2, F2, G2, H2, I2, J2, K2, L2, M2, result)`: Phương thức này lưu thông tin của một bệnh nhân vào cơ sở dữ liệu.
- `login(self, username, password)`: Phương thức này thêm một tài khoản người dùng mới vào cơ sở dữ liệu.
- `close_connection(self)`: Phương thức này đóng kết nối với cơ sở dữ liệu.

➤ Tạo database heart_data.

➤ Phương thức connect: kết nối đến cơ sở dữ liệu MySQL sử dụng thông tin được cung cấp.

➤ Tạo các table trong phương thức create_tables

- Tạo table data lưu thông tin tên, tuổi, ngày dự đoán, các thông số người dùng nhập vào và kết quả dự đoán bệnh tim.
- Tạo table login lưu tài khoản người dùng.

- Trong phương thức save_data: lập trình các câu lệnh insert dữ liệu của người dùng vào database, nếu không thành công thì báo lỗi.
 - Trong phương thức login: lập trình các câu lệnh thêm username, password mới mà người dùng vừa đăng ký tài khoản vào database, nếu không thành công thì báo lỗi.
 - Trong phương thức close_connection: lập trình các câu lệnh đóng kết nối cơ sở dữ liệu.
- ❖ Lập trình chức năng lưu thông tin
- Import database vào main.py
 - Viết phương thức Save
 - Dùng phương thức connect (database.py) để kết nối vào cơ sở dữ liệu.
 - Dùng phương thức create_tables (database.py) để tạo các table.
 - Lấy các thông tin của người dùng
 - Dùng phương thức save_data (database.py) để lưu thông tin người dùng.
 - Click chuột vào button (hình download) thông tin người dùng sẽ được lưu vào database và hiển thị thông báo lưu thành công. Nếu không thành công sẽ báo lỗi.
- ❖ Lập trình chức năng đăng nhập
- Trong file login.py
 - Lập trình phương thức login_user để đăng nhập
 - Lập trình phương thức limit_login để giới hạn số lần nhập sai. Nhập sai 3 lần thì sẽ đóng màn hình login.
 - Import login vào main.py
 - Khi người dùng nhấn vào button đăng nhập. Màn hình đăng nhập sẽ hiện ra. Nếu đăng nhập sai quá 3 lần sẽ đóng màn hình đăng nhập.

d. Tuần 4

- ❖ Điều chỉnh database
- Ở phương thức login: tạo ràng buộc username là duy nhất
 - Ở phương thức create_tables (database.py) tạo table recommendations lưu các khuyến cáo cho người dùng.

- Lập trình phương thức insert_recommendations vào file database.py để thêm các dữ liệu khuyến cáo của người dùng.
- ❖ Lập trình chức năng đăng ký
 - Import login vào SignUp.py
 - Trong file SignUp.py
 - Lập trình phương thức signup_user để kết nối với database và các ràng buộc về đăng ký tài khoản như trùng tên user, xác nhận mật khẩu không đúng. Nếu đăng ký thành công thì insert tài khoản vào cơ sở dữ liệu.
 - Khi người dùng nhập chọn SignUp ở màn hình đăng nhập, màn hình đăng ký hiện ra, đăng ký thành công thì tiếp tục tiến hành đăng nhập.
- ❖ Lập trình chức năng đăng xuất
 - Khi người dùng đăng nhập thành công, giao diện phần đăng nhập sẽ thay đổi, một button người dùng và một button (hình nút nguồn) đăng xuất sẽ hiện ra.
 - Dưới button người dùng sẽ có dòng chữ xin chào <tên đăng nhập>
 - Khi click vào button (hình nút nguồn) đăng xuất, thì sẽ trở lại trang ban đầu chưa đăng nhập, các thông tin người dùng sẽ biến mất
- ❖ Lập trình chức năng xem thông tin người dùng
 - Khi người dùng nhập các thông số, tên, tuổi và tiến hành dự đoán kết quả thì toàn bộ thông số người dùng nhập vào sẽ được diễn giải rõ ràng về tình trạng hiện tại và kết quả dự đoán sẽ được lưu vào trong một trang mới.
 - Khi click vào button người dùng thì sẽ hiển thị thông tin người dùng.
- ❖ Lập trình chức năng xuất báo cáo ở main.py (phương thức export_to_pdf)
 - Tạo button xuất báo cáo
 - Lấy tất cả thông tin người dùng ở phần thông tin người dùng
 - Khi người dùng đã dự đoán thành công, người dùng có thể click vào button xuất báo để xuất ra một file pdf.
- ❖ Xây dựng chức năng vẽ biểu đồ phân tích các số liệu sau khi đã có kết quả dự đoán.
(trong main.py ở phương thức Analysis)

- Lấy các thông số người dùng nhập vào.
 - Lấy các thông số chuẩn của người bình thường không bị bệnh.
 - Vẽ biểu đồ so sánh
 - Hiển thị biểu đồ lên trang web
- ❖ Chức năng khuyến cáo (trong main.py ở phương thức display_recommendations)
- Khi người dùng dự đoán bệnh tim, một button khuyến cáo sẽ được tạo ra
 - Viết phương thức để xuất các khuyến cáo nên làm để giảm bệnh cho người mắc bệnh
 - Hiển thị trên một trang mới
 - Lưu khuyến cáo cùng button với lưu thông tin bệnh nhân. (click save thì lưu cả thông tin người dùng và khuyến cáo)
- ❖ Lập trình các ràng buộc cho người dùng
- Yêu cầu đăng nhập trước khi dự đoán, lưu thông tin, xuất báo cáo
 - Yêu cầu mật khẩu và xác nhận mật khẩu phải trùng khớp
 - Yêu cầu tên người dùng nếu đã có phải nhập lại tên mới
 - Yêu cầu phải nhập đầy đủ tất cả thông tin
 - Nếu thiếu thông tin người dùng không lưu được, không xuất báo cáo được
- ❖ Viết docstring
3. Kết quả đạt được
 - a. Tuần 1:
 - Dataset đã được tiền xử lý
 - Xây dựng được file huấn luyện nhiều mô hình, so sánh mô hình. (Heart-Disease-Prediction-using-ML.ipynb)
 - Xây dựng được mô hình dự đoán bệnh tim (model.py)
 - Xây dựng được tiền xử lý dữ liệu mới (data_preprocessing.py)
 - b. Tuần 2:
 - Thiết kế được giao diện trang chủ
 - Giao diện trang chủ (main.py)
 - Giao diện trang đăng nhập, đăng ký (login.py, SignUp.py)

- Giao diện trang mô tả dữ liệu (info.py)
- c. Tuần 3:
 - Chức năng đăng nhập
 - Chức năng dự đoán bệnh tim
 - Chức năng lưu thông tin người dùng
 - Database heart_data
 - Các ràng buộc về nhập thông tin để dự đoán.
- d. Tuần 4:
 - Database hoàn chỉnh
 - Chức năng đăng ký
 - Chức năng đăng xuất
 - Chức năng xem thông tin người dùng
 - Chức năng xuất báo cáo
 - Chức năng vẽ biểu đồ phân tích số liệu
 - Chức năng khuyến cáo
 - Các ràng buộc cho người dùng nhập
 - Viết docstring
 - Kiểm thử
 - Hoàn thành được tất cả nội dung công việc trong file đăng ký.
 - Link github: https://github.com/TuanTu1007/AppHeart_Prediction.github.io/

4. Tài liệu tham khảo

- [1] Sách tham khảo “Hands-on Machine Learning with Scikit-Learn, Keras &TensorFlow, 2nd Edition” của tác giả “Aurélien Géron”.
- [2] Python GUI Programming With Tkinter, URL: <https://realpython.com/python-gui-tkinter/>
- [3] tkinterPython interface to Tcl/Tk, URL:
<https://docs.python.org/3/library/tkinter.html>
- [4] Python GUI – tkinter, URL: <https://www.geeksforgeeks.org/python-gui-tkinter/>

[5] GUI Python Project, URL:

https://www.youtube.com/watch?v=CQZ19G_SwVc&list=PLI316cKxhMxtOWHa88kDqm42uWz1aqGfD&index=53

[6] Student Registration System with Database Using Python, URL:

<https://www.youtube.com/watch?v=JUGEkFDeuwg&list=PLI316cKxhMxtOWHa88kDqm42uWz1aqGfD&index=45>

[7] How To Create Login Form for Apps Using Python, URL:

https://www.youtube.com/watch?v=X9reTI_Mckk&list=PLI316cKxhMxtOWHa88kDqm42uWz1aqGfD&index=34

5. Phụ lục 1: Giới thiệu (demo) kết quả

a. Tuần 1:

- File xử lý data và train các mô hình

The screenshot shows a Jupyter Notebook interface with three tabs at the top: 'model.py', 'data_preprocessing.py', and 'Heart-Disease-Prediction-using-ML.ipynb'. The 'Heart-Disease-Prediction-using-ML.ipynb' tab is active. The notebook has two code cells. Cell [1] contains code to import numpy, pandas, seaborn, matplotlib, and sklearn. Cell [2] contains code to read a CSV file named 'heart.csv' and print its head. Below the code cells, the dataset is displayed as a table with 13 columns: age, sex, cp, trestbps, chol, fbs, restecg, thalach, exang, oldpeak, slope, ca, thal, and target. The first five rows of the dataset are shown. At the bottom, there is a text cell with the instruction 'Kích thước dataset' and a code cell with 'data.shape'.

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split, cross_val_score, learning_curve
```

```
[1]
```

```
data = pd.read_csv('heart.csv')
data.head()
```

```
[2]
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0

Kích thước dataset

```
data.shape
```

The screenshot shows a Jupyter Notebook interface with several tabs at the top: Run.py, Heart-Disease-Prediction-using-ML.ipynb (active), model.py 9+, and data_preprocessing.py 9+. Below the tabs, there's a toolbar with buttons for Code, Markdown, Run All, Clear All Outputs, and Outline. On the right, there's a "Select Kernel" dropdown.

```

model.fit(X_train,y_train)
y_pred = model.predict(X_test)
acc_score.append(accuracy_score(y_test, y_pred))
roc_score.append(roc_auc_score(y_test, y_pred))
f1.append(f1_score(y_test, y_pred, average='weighted'))
name_model.append(type(model).__name__)

result = pd.DataFrame(
    {'Model Name' : name_model,
     'accuracy' : acc_score,
     'roc auc' : roc_score,
     'f1-weighted' : f1}
)

result.sort_values('f1-weighted', ascending=False)

```

The code block ends with a table output:

	Model Name	accuracy	roc auc	f1-weighted
2	RandomForestClassifier	0.988764	0.989726	0.988775
1	DecisionTreeClassifier	0.977528	0.979452	0.977564
4	XGBClassifier	0.977528	0.979452	0.977564
3	SVC	0.880150	0.879797	0.880228
0	LogisticRegression	0.850187	0.848155	0.850073

At the bottom, there are tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL (selected), PORTS, and JUPYTER. The terminal tab shows command-line logs:

```

NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.

```

- File model.py

The screenshot shows a Jupyter Notebook interface with tabs for Run.py, model.py 9+ (active), and data_preprocessing.py 9+. Below the tabs is a toolbar with Code, Output, Debug Console, Terminal, Ports, and Jupyter.

```

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import OneHotEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
from DoAn.Model.data_preprocessing import *
data = pd.read_csv('D:\Subject\IE221_Python\DoAn\heart.csv')

# print(data.head())
# print(data.shape)
# print(data.info())
# data.isnull().sum()

def remove_outliers_iqr(df, column_name, lower_bound_factor=1.5, upper_bound_factor=1.5):
    Q1 = df[column_name].quantile(0.25)
    Q3 = df[column_name].quantile(0.75)
    IQR = Q3 - Q1

    lower_bound = Q1 - lower_bound_factor * IQR
    upper_bound = Q3 + upper_bound_factor * IQR

```

The code block ends with a command-line log:

```

NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.

```

```
Run.py model.py 9+ data_preprocessing.py 9+
DoAn > Model > model.py > remove_outliers_iqr
19     def remove_outliers_iqr(df, column_name, lower_bound_factor=1.5, upper_bound_factor=1.5):
20         df_filtered = df[(df[column_name] >= lower_bound) & (df[column_name] <= upper_bound)]
21         return df_filtered
22
23
24     data_no_outlier = data.copy()
25     numeric_columns = ['age','trestbps','chol','thalach','oldpeak','slope','ca']
26
27     for column in numeric_columns:
28         data_no_outlier = remove_outliers_iqr(data_no_outlier, column)
29
30     data_no_outlier.reset_index(inplace=True, drop=True)
31
32     #
33     scaler = MinMaxScaler()
34     numeric_df = data_no_outlier[numeric_columns]
35
36     numeric_scaled_df = pd.DataFrame(scaler.fit_transform(numeric_df), columns = scaler.get_feature_names_out())
37
38     data2 = data_no_outlier
39     data2[numeric_columns] = numeric_scaled_df
40
41     def assign_age_group(age):
42
43         age_groups = {
44             (0, 40): 'Young',
45             (41, 60): 'Middle-aged',
46             (61, 80): 'Elderly',
47             (81, float('inf')): 'Very Elderly'
48         }
49
50
51         for age_range, group in age_groups.items():
52             if age_range[0] <= age <= age_range[1]:
53                 return group
54
55     data2['age_group'] = data['age'].apply(assign_age_group)
56
57     def categorize_blood_pressure(resting_blood_pressure):
58         if resting_blood_pressure < 90:
59             return "Low"
60         elif 90 <= resting_blood_pressure <= 120:
61             return "Normal"
62         else:
63             return "High"
64
65     data2['blood_pressure_group'] = data['trestbps'].apply(categorize_blood_pressure)
66
67     def categorize_cholesterol(cholesterol):
68         if cholesterol < 200:
69             return "Desirable"
70         elif 200 <= cholesterol <= 239:
71
72
73
74
75
76
77
78
79
79
80
81
82
83
84
85
86
87
88
```

PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.

```
Run.py model.py 9+ data_preprocessing.py 9+
DoAn > Model > model.py > remove_outliers_iqr
60     def assign_age_group(age):
61
62         age_groups = {
63             (0, 40): 'Young',
64             (41, 60): 'Middle-aged',
65             (61, 80): 'Elderly',
66             (81, float('inf')): 'Very Elderly'
67         }
68
69
70         for age_range, group in age_groups.items():
71             if age_range[0] <= age <= age_range[1]:
72                 return group
73
74     data2['age_group'] = data['age'].apply(assign_age_group)
75
76     def categorize_blood_pressure(resting_blood_pressure):
77         if resting_blood_pressure < 90:
78             return "Low"
79         elif 90 <= resting_blood_pressure <= 120:
80             return "Normal"
81         else:
82             return "High"
83
84     data2['blood_pressure_group'] = data['trestbps'].apply(categorize_blood_pressure)
85
86     def categorize_cholesterol(cholesterol):
87         if cholesterol < 200:
88             return "Desirable"
89         elif 200 <= cholesterol <= 239:
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
209
210
211
212
213
214
215
216
217
218
219
219
220
221
222
223
224
225
226
227
228
229
229
230
231
232
233
234
235
236
237
238
238
239
240
241
242
243
244
245
246
247
248
249
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1195
1196
1197
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1217
1218
1218
1219
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1317
1318
1318
1319
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1345
1346
1347
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1375
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1425
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1435
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1445
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1455
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1475
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1485
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1525
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1535
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1545
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1575
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1585
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1625
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1635
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1645
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1675
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1685
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1725
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1735
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1745
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1775
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1785
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1825
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1835
1836
1837
1837
1838
1838
1839
1839
1840
1841
1842
1843
1844
1845
1846
1847
1847
1848
1848
1849
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1867
1868
1868
1869
1869
1870
1871
1872
1873
1874
1875
1876
1877
1877
1878
1878
1879
1879
1880
1881
1882
1883
1884
1885
1886
1887
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1917
1918
1918
1919
1919
1920
1921
1922
1923
1924
1925
1926
1927
1927
1928
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1937
1938
1938
1939
1939
1940
1941
1942
1943
1944
1945
1946
1947
1947
1948
1948
1949
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1967
1968
1968
1969
1969
1970
1971
197
```

```
Run.py model.py 9+ data_preprocessing.py 9+
DoAn > Model > model.py > ...
85 def categorize_cholesterol(cholesterol):
86     if cholesterol < 160:
87         return "Borderline High"
88     else:
89         return "High"
90
91 data2['cholesterol_group'] = data['chol'].apply(categorize_cholesterol)
92
93 nominal_columns = ['cp', 'restecg', 'thal']
94 ohc = OneHotEncoder(sparse_output=False, drop='first')
95
96 dummies_df = pd.DataFrame(ohc.fit_transform(data2[nominal_columns]), columns = ohc.get_feature_names_out())
97
98 age_group_mapping = {'Young': 0, 'Middle-aged': 1, 'Elderly': 2, 'Very Elderly' : 3}
99 bloodp_mapping = {'Low': 0, 'Normal': 1, 'High': 2}
100 chol_mapping = {'Desirable': 0, 'Borderline High': 1, 'High': 2}
101
102 # Create Ordinal Dataframe
103 ordinal_df = {
104     'age_group' : data2['age_group'],
105     'blood_pressure' : data2['blood_pressure_group'],
106     'cholesterol_group' : data2['cholesterol_group']}
107 ordinal_df = pd.DataFrame(ordinal_df)
108
109 # Encode ordinal categories
110 ordinal_df['age_group'] = ordinal_df['age_group'].map(age_group_mapping)
111 ordinal_df['blood_pressure'] = ordinal_df['blood_pressure'].map(bloodp_mapping)
112 ordinal_df['cholesterol_group'] = ordinal_df['cholesterol_group'].map(chol_mapping)
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
```

PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

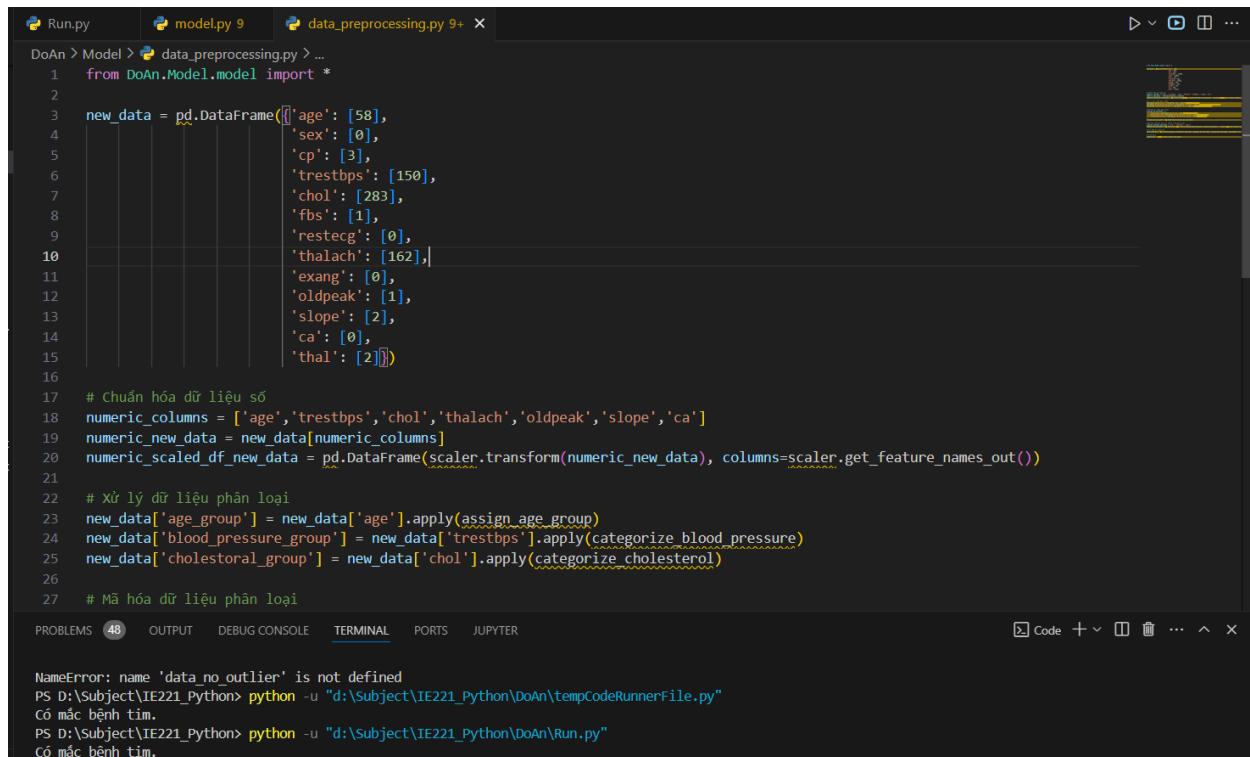
NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.

```
Run.py model.py 9+ data_preprocessing.py 9+
DoAn > Model > model.py > ...
112 # Encode ordinal categories
113 ordinal_df['age_group'] = ordinal_df['age_group'].map(age_group_mapping)
114 ordinal_df['blood_pressure'] = ordinal_df['blood_pressure'].map(bloodp_mapping)
115 ordinal_df['cholesterol_group'] = ordinal_df['cholesterol_group'].map(chol_mapping)
116
117 last_data = pd.concat([numeric_scaled_df, dummies_df, ordinal_df, data2[['sex', 'fbs', 'exang', 'target']]], axis=1)
118
119 # print(last_data.head())
120
121 X = last_data.drop('target', axis=1)
122 y = last_data['target']
123
124 X_train, X_test, y_train, y_test = train_test_split(X,y,stratify=y, test_size=.3,random_state=42)
125
126 model = RandomForestClassifier()
127 model.fit(X_train, y_train)
128
129 # accuracy on test
130 # X_test_prediction = model.predict(X_test)
131 # test_accuracy = accuracy_score(X_test_prediction, y_test)
132 # print(test_accuracy)
133
```

PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.

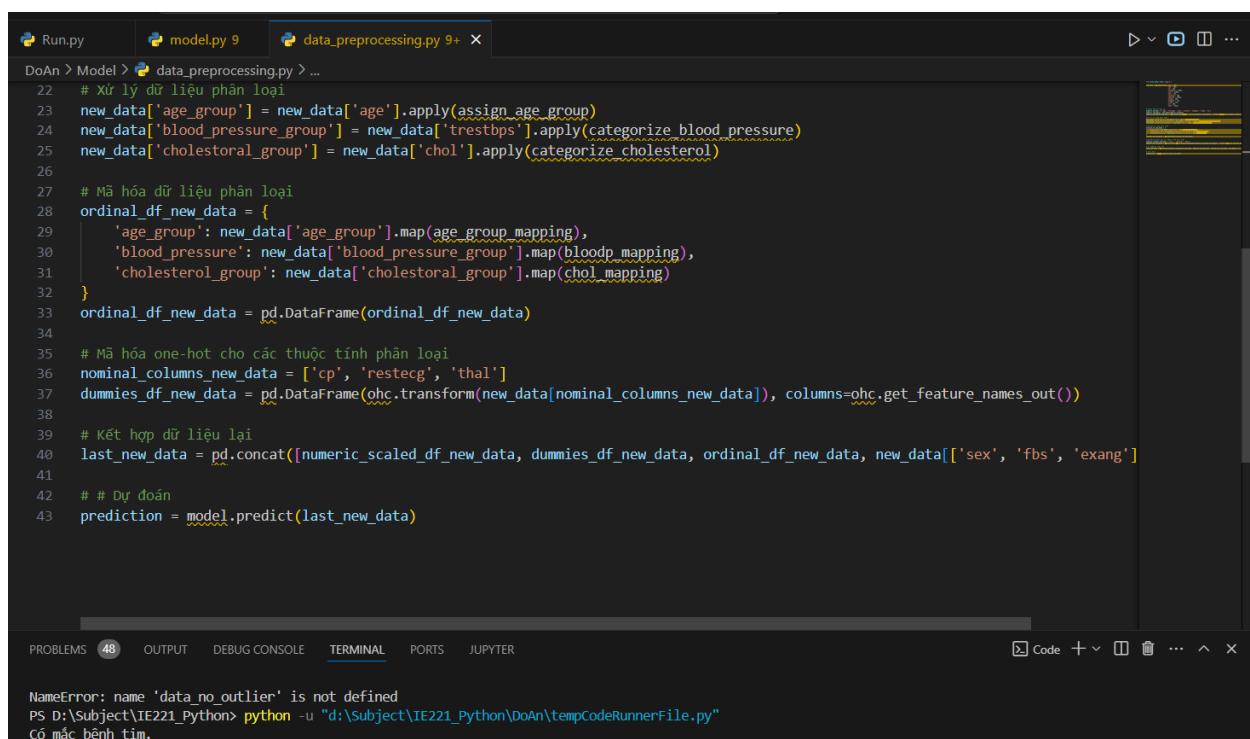
- File data_preprocessing.py (chưa sử dụng OOP), dự đoán với dữ liệu mới.



```
DoAn > Model > data_preprocessing.py > ...
1  from DoAn.Model.model import *
2
3  new_data = pd.DataFrame([{'age': [58],
4      'sex': [0],
5      'cp': [3],
6      'trestbps': [150],
7      'chol': [283],
8      'fbs': [1],
9      'restecg': [0],
10     'thalach': [162],
11     'exang': [0],
12     'oldpeak': [1],
13     'slope': [2],
14     'ca': [0],
15     'thal': [2]}])
16
17 # Chuẩn hóa dữ liệu số
18 numeric_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'slope', 'ca']
19 numeric_new_data = new_data[numeric_columns]
20 numeric_scaled_df_new_data = pd.DataFrame(scaler.transform(numeric_new_data), columns=scaler.get_feature_names_out())
21
22 # Xử lý dữ liệu phân loại
23 new_data['age_group'] = new_data['age'].apply(assign_age_group)
24 new_data['blood_pressure_group'] = new_data['trestbps'].apply(categorize_blood_pressure)
25 new_data['cholesterol_group'] = new_data['chol'].apply(categorize_cholesterol)
26
27 # Mã hóa dữ liệu phân loại
```

PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.
```



```
DoAn > Model > data_preprocessing.py > ...
22 # Xử lý dữ liệu phân loại
23 new_data['age_group'] = new_data['age'].apply(assign_age_group)
24 new_data['blood_pressure_group'] = new_data['trestbps'].apply(categorize_blood_pressure)
25 new_data['cholesterol_group'] = new_data['chol'].apply(categorize_cholesterol)
26
27 # Mã hóa dữ liệu phân loại
28 ordinal_df_new_data = {
29     'age_group': new_data['age_group'].map(age_group_mapping),
30     'blood_pressure': new_data['blood_pressure_group'].map(bloodp_mapping),
31     'cholesterol_group': new_data['cholesterol_group'].map(chol_mapping)
32 }
33 ordinal_df_new_data = pd.DataFrame(ordinal_df_new_data)
34
35 # Mã hóa one-hot cho các thuộc tính phân loại
36 nominal_columns_new_data = ['cp', 'restecg', 'thal']
37 dummies_df_new_data = pd.DataFrame(ohc.transform(new_data[nominal_columns_new_data]), columns=ohc.get_feature_names_out())
38
39 # Kết hợp dữ liệu lại
40 last_new_data = pd.concat([numeric_scaled_df_new_data, dummies_df_new_data, ordinal_df_new_data, new_data[['sex', 'fbs', 'exang']]])
41
42 # # Dự đoán
43 prediction = model.predict(last_new_data)
```

PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER

```
NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
```

- File Run.py

The screenshot shows a code editor interface with several tabs at the top: Run.py (active), Heart-Disease-Prediction-using-ML.ipynb, model.py 9, and data_preprocessing.py 9+. The Run.py tab contains the following Python code:

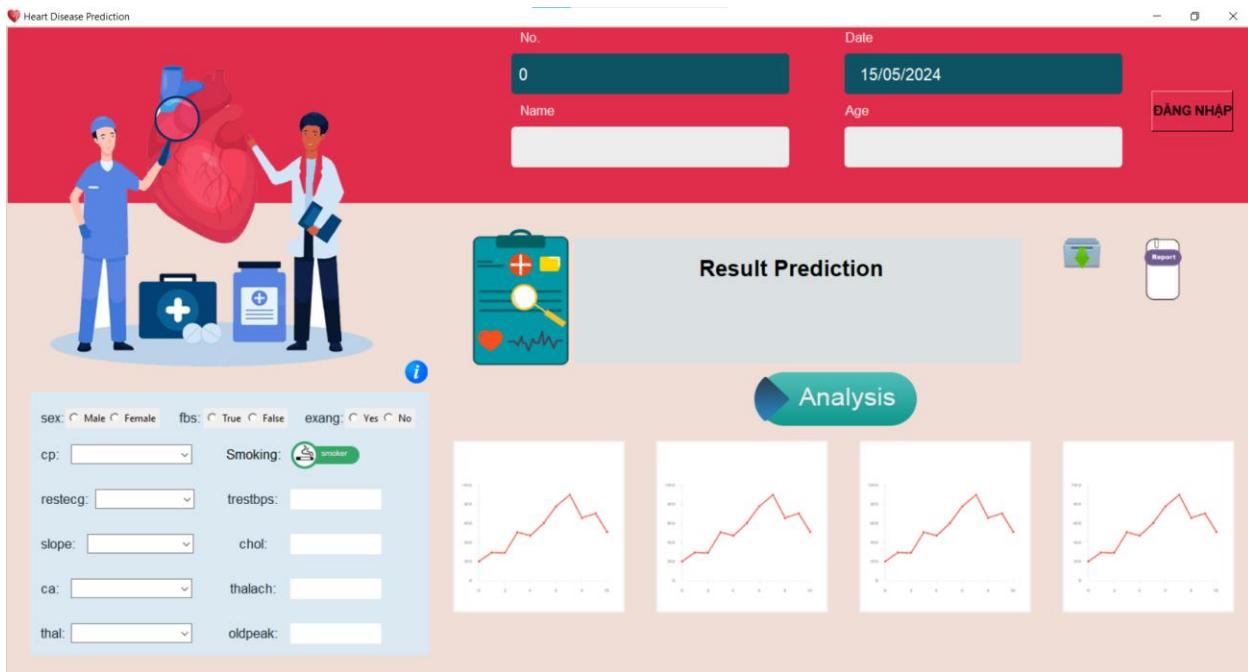
```
DoAn > Model > Run.py
1  from DoAn.Model.data_preprocessing import *
2  from DoAn.Model.model import *
3
4  dict(last_new_data)
5
6  if prediction[0] == 0:
7      print("không mắc bệnh tim.")
8  else:
9      print("có mắc bệnh tim.")
```

Below the code editor is a terminal window showing command-line output:

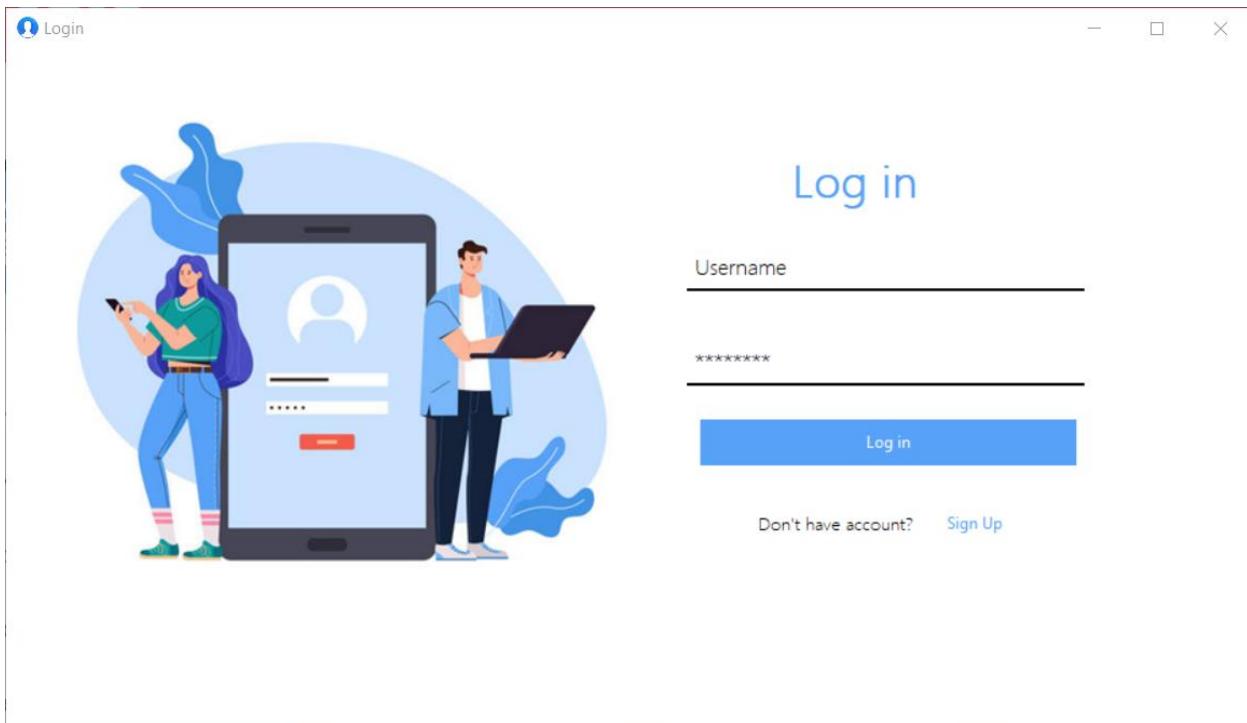
```
PROBLEMS 48 OUTPUT DEBUG CONSOLE TERMINAL PORTS JUPYTER
NameError: name 'data_no_outlier' is not defined
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\tempCodeRunnerFile.py"
Có mắc bệnh tim.
PS D:\Subject\IE221_Python> python -u "d:\Subject\IE221_Python\DoAn\Run.py"
Có mắc bệnh tim.
```

b. Tuần 2:

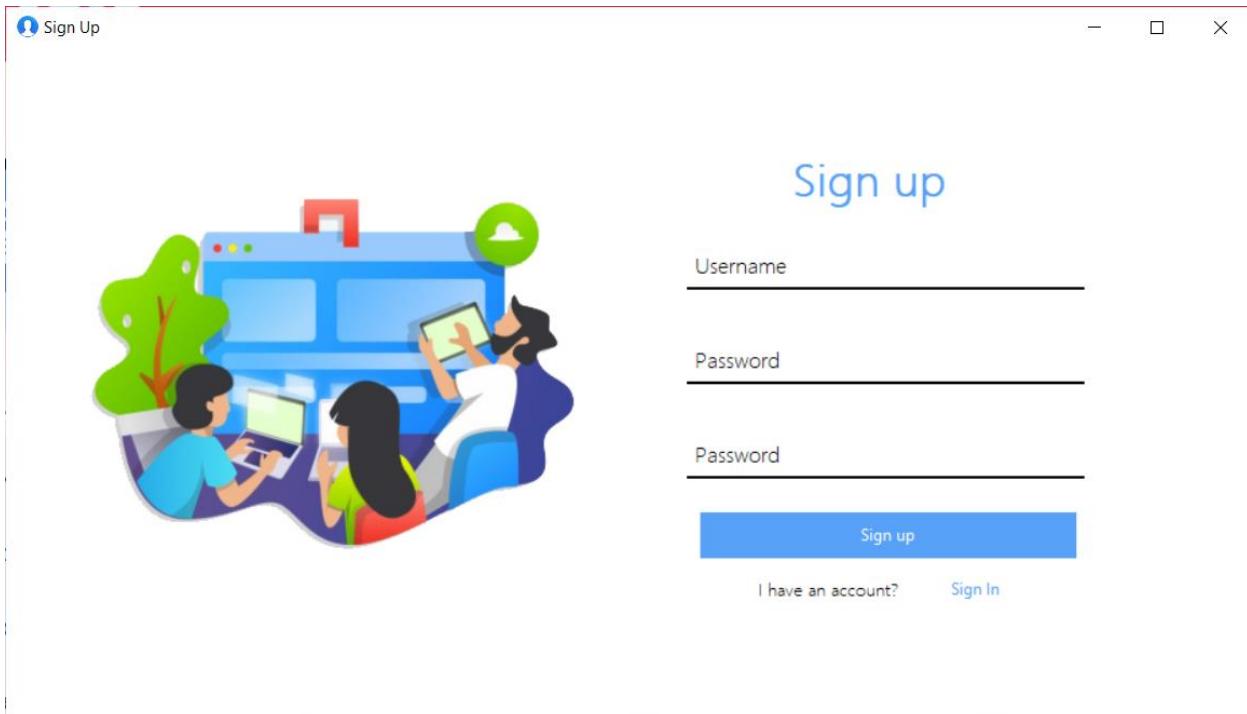
- Giao diện trang chủ:



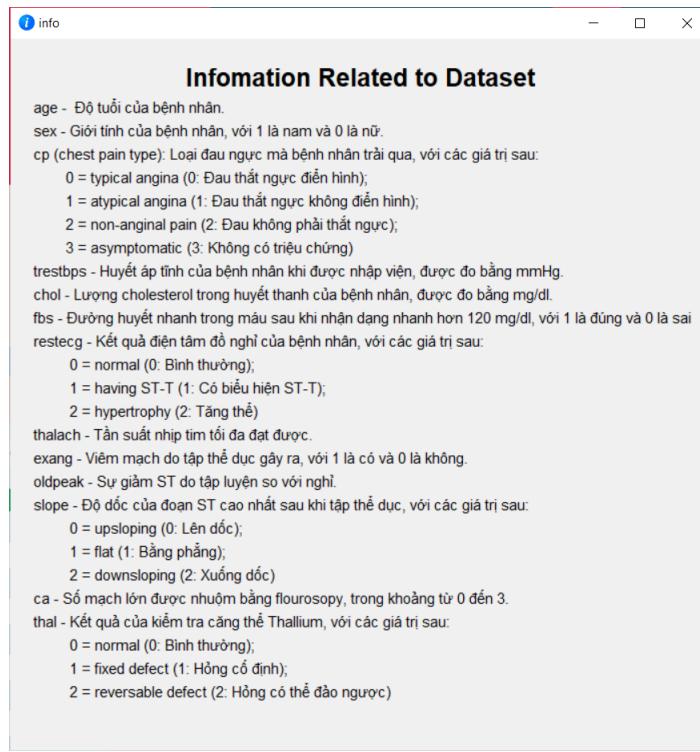
- Giao diện trang đăng nhập



- Giao diện trang đăng ký



- Trang mô tả thông tin dữ liệu khi click chuột vào biểu tượng chữ i ở trang chủ sẽ hiện ra.



c. Tuần 3:

- Chức năng dự đoán

Heart Disease Prediction

No. 0 Date 08/05/2024

Name Trịnh Tuấn Tú Age 21

Result Prediction

Trịnh Tuấn Tú, bạn không mắc bệnh tim

Analysis

SBP: ♂ Male ♀ Female fbs: ⚡ True ⚡ False exang: ⚡ Yes ⚡ No

cp: 2 = non-anginal pa Smoking: non-smoker

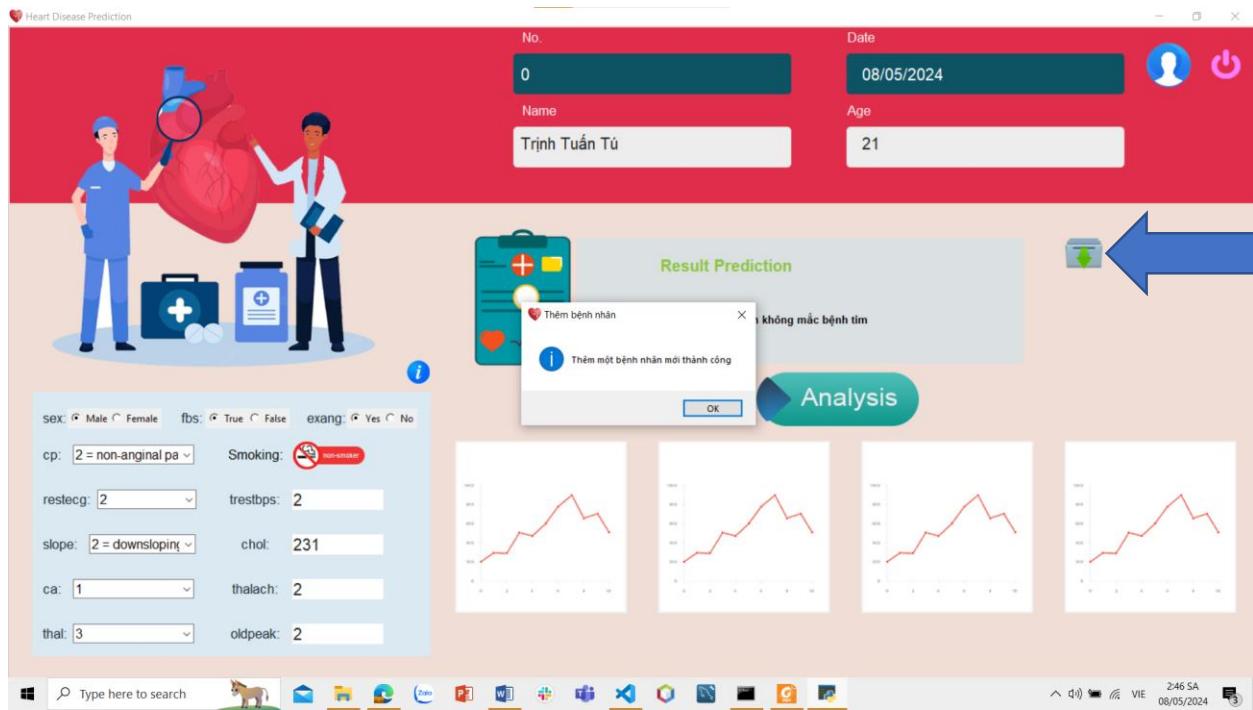
restecg: 2 trestbps: 2

slope: 2 = downsloping chol: 231

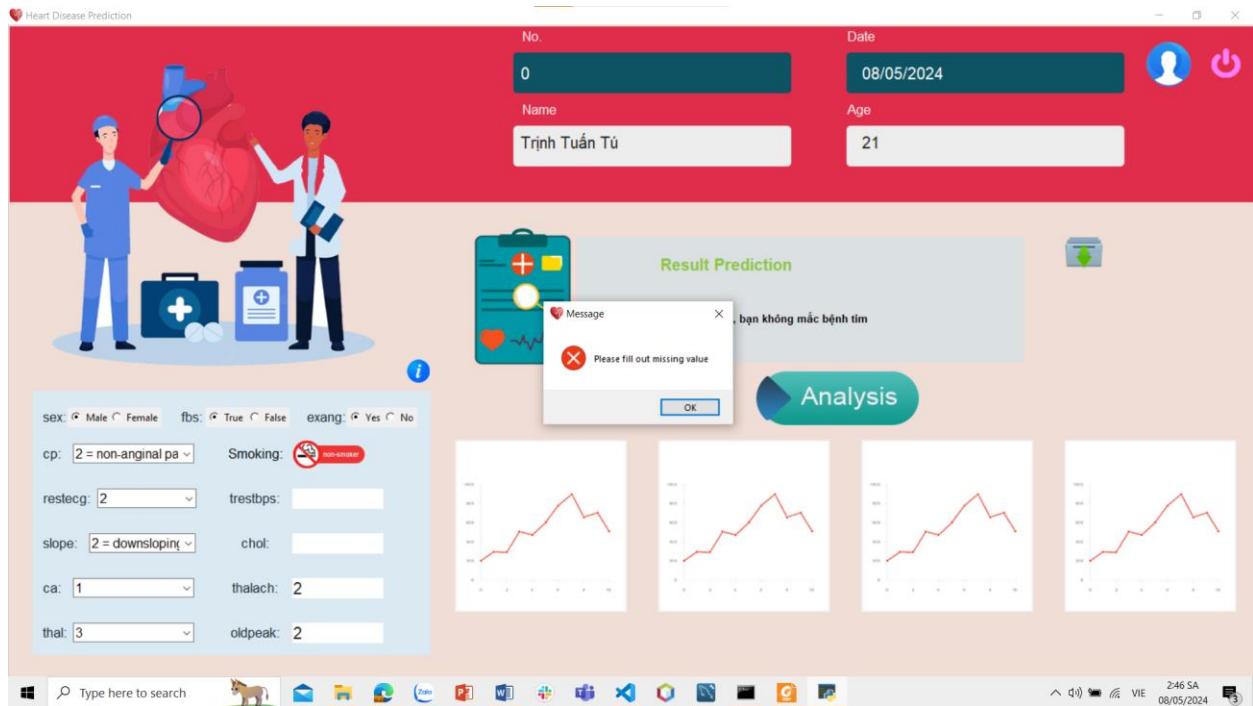
ca: 1 thalach: 2

thal: 3 oldpeak: 2

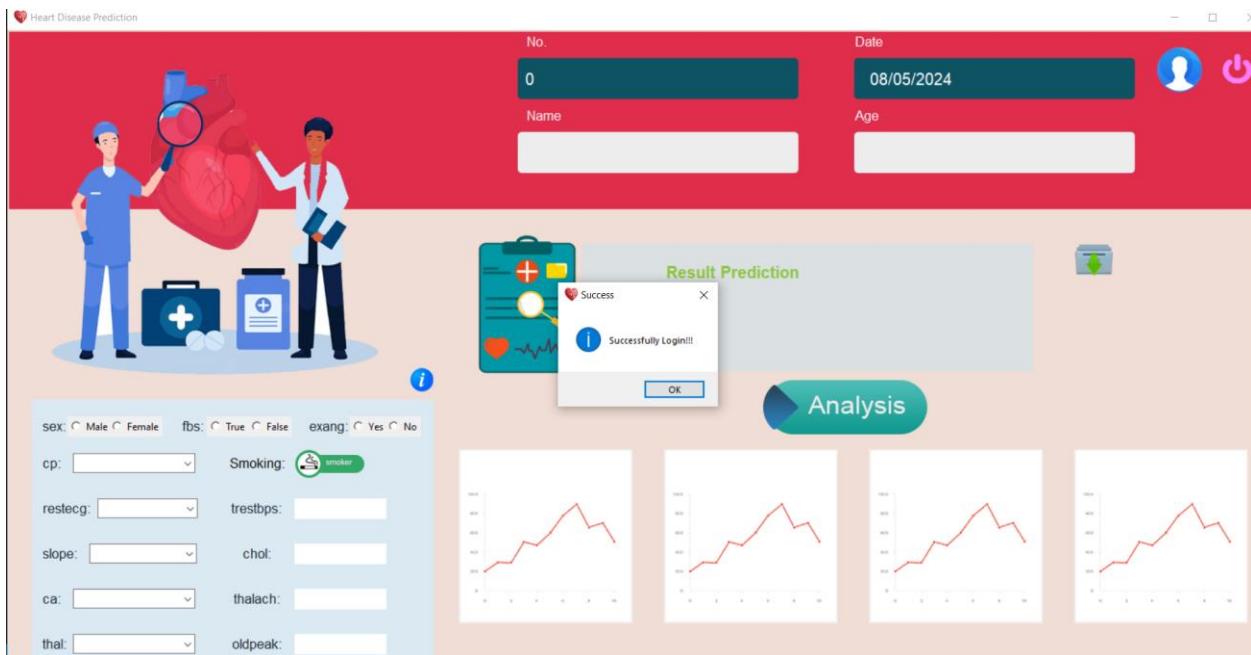
- Chức năng lưu



- Ràng buộc thiếu thông tin



- Chức năng đăng nhập



- Database

```
mysql> use heart_data;
Database changed
mysql> select * from data;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| user | Name      | Date       | BD    | sex   | cp    | trestbps | chol  | fbs   | restecg | thalach | exang | oldpeak | slope | ca    | thal  | result |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1   | Tr?nh Tu?n T? | 08/05/2024 | 21   | 1    | 2    | 231    | 1     | 2     | 2     | 2     | 1     | 2     | 2     | 1    | 3    | 0    |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> select * from login;
+-----+-----+
| user | Username | Password |
+-----+-----+
| 1   | admin    | admin    |
+-----+-----+
1 row in set (0.00 sec)
```

- Hàm preprocessed (heart_prediction.py)

```
def preprocessed(new_data):
    new_data_dict = [
        'age': [new_data[0]],
        'sex': [new_data[1]],
        'cp': [new_data[2]],
        'trestbps': [new_data[3]],
        'chol': [new_data[4]],
        'fbs': [new_data[5]],
        'restecg': [new_data[6]],
        'thalach': [new_data[7]],
        'exang': [new_data[8]],
        'oldpeak': [new_data[9]],
        'slope': [new_data[10]],
        'ca': [new_data[11]],
        'thal': [new_data[12]]
    ]

    df = pd.DataFrame(new_data_dict)
    # Chuẩn hóa dữ liệu số
    scaler = MinMaxScaler()
    numeric_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'slope', 'ca']
    numeric_scaled_df_new_data = pd.DataFrame(scaler.fit_transform(df[numeric_columns]), columns=numeric_columns)

    # Xử lý dữ liệu phân loại
    df['age_group'] = df['age'].apply(assign_age_group)
    df['blood_pressure_group'] = df['trestbps'].apply(categorize_blood_pressure)
    df['cholesterol_group'] = df['chol'].apply(categorize_cholesterol)

    # Mã hóa dữ liệu phân loại
    ordinal_df_new_data = {
        'age_group': df['age_group'].map(age_group_mapping),
        'blood_pressure': df['blood_pressure_group'].map(bloodp_mapping),
        'cholesterol_group': df['cholesterol_group'].map(chol_mapping)
    }
    ordinal_df_new_data = pd.DataFrame(ordinal_df_new_data)

    # Mã hóa one-hot cho các thuộc tính phân loại
    nominal_columns_new_data = ['cp', 'restecg', 'thal']
    dummies_df_new_data = pd.DataFrame(ohc.transform(df[nominal_columns_new_data]), columns=ohc.get_feature_names_out())

    # Kết hợp dữ liệu lại
    last_new_data = pd.concat([numeric_scaled_df_new_data, dummies_df_new_data, ordinal_df_new_data, df[['sex', 'fbs', 'exang']]], axis=1)

    return last_new_data
```

- Hàm lấy thông tin, các ràng buộc nhập dữ liệu, xử lý dữ liệu, dự đoán và hiển thị kết quả dự đoán

```
def Analysis(self):
    global prediction
    try:
        A = int(self.BD.get())
    except ValueError:
        messagebox.showerror("Error", "Invalid birth year. Please enter a valid year.")
        return

    try:
        B = self.selecGender()
    except:
        messagebox.showerror("Message", "Please select Gender")
        return

    try:
        F = self.selecfbs()
    except:
        messagebox.showerror("Message", "Please select fbs")
        return

    try:
        I = self.selecxang()
    except:
        messagebox.showerror("Message", "Please select exang")
        return

    return

    try:
        K = self.selecslope()
    except:
        messagebox.showerror("Message", "Please select slope")
        return

    try:
        G = int(self.restecg_combobox.get())
    except:
        messagebox.showerror("Message", "Please select restecg")
        return

    try:
        L = int(self.ca_combobox.get())
    except:
        messagebox.showerror("Message", "Please select ca")
        return

    try:
        M = int(self.thal_combobox.get())
    except:
        messagebox.showerror("Message", "Please select thal")
        return

    try:
        D = int(self.trestbps.get())
        E = int(self.chol.get())
        H = int(self.thalach.get())
        J = int(self.oldpeak.get())
    except:
        messagebox.showerror("Message", "Please fill out missing value")
        return
```

```
input_new = (A,B,C,D,E,F,G,H,I,J,K,L,M)

# Dự đoán
last_input_new = preprocessed(input_new)
prediction = model.predict(last_input_new)

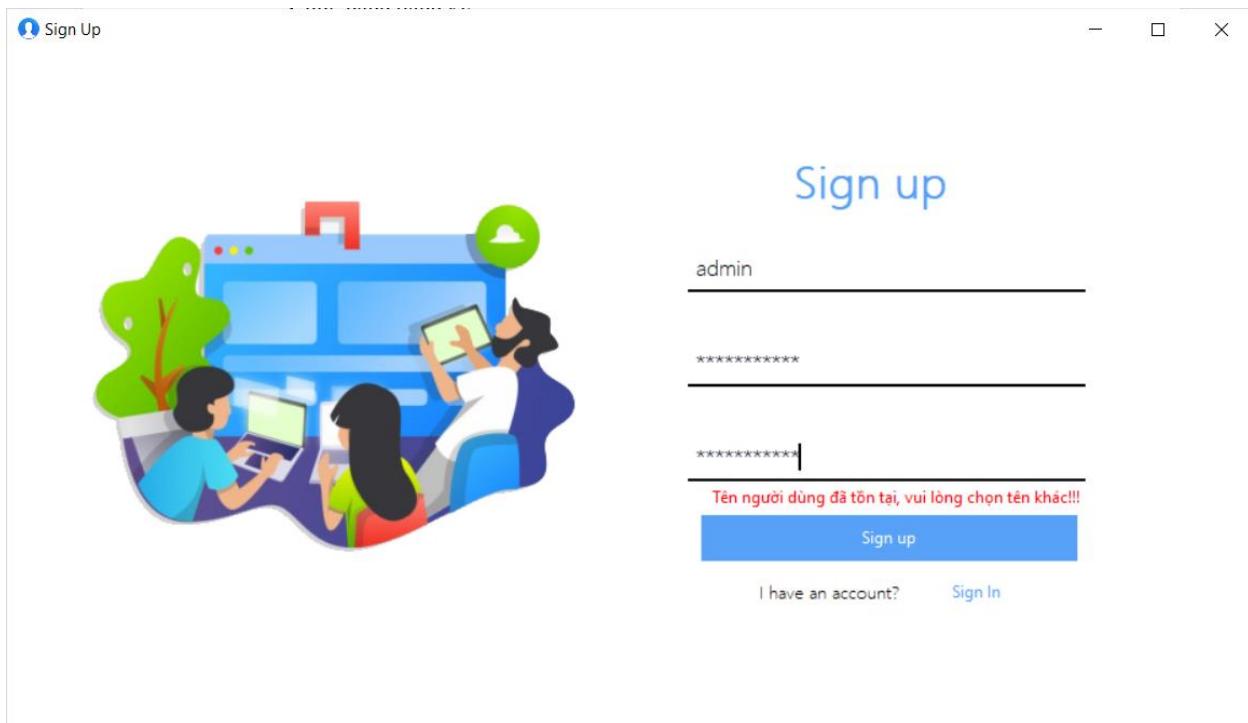
if prediction[0] == 0:
    self.report1.config(text=f"{self.Name.get()}, bạn không mắc bệnh tim")
else:
    self.report1.config(text=f"{self.Name.get()}, bạn có mắc bệnh tim")
```

d. Tuần 4:

- Chức năng đăng ký

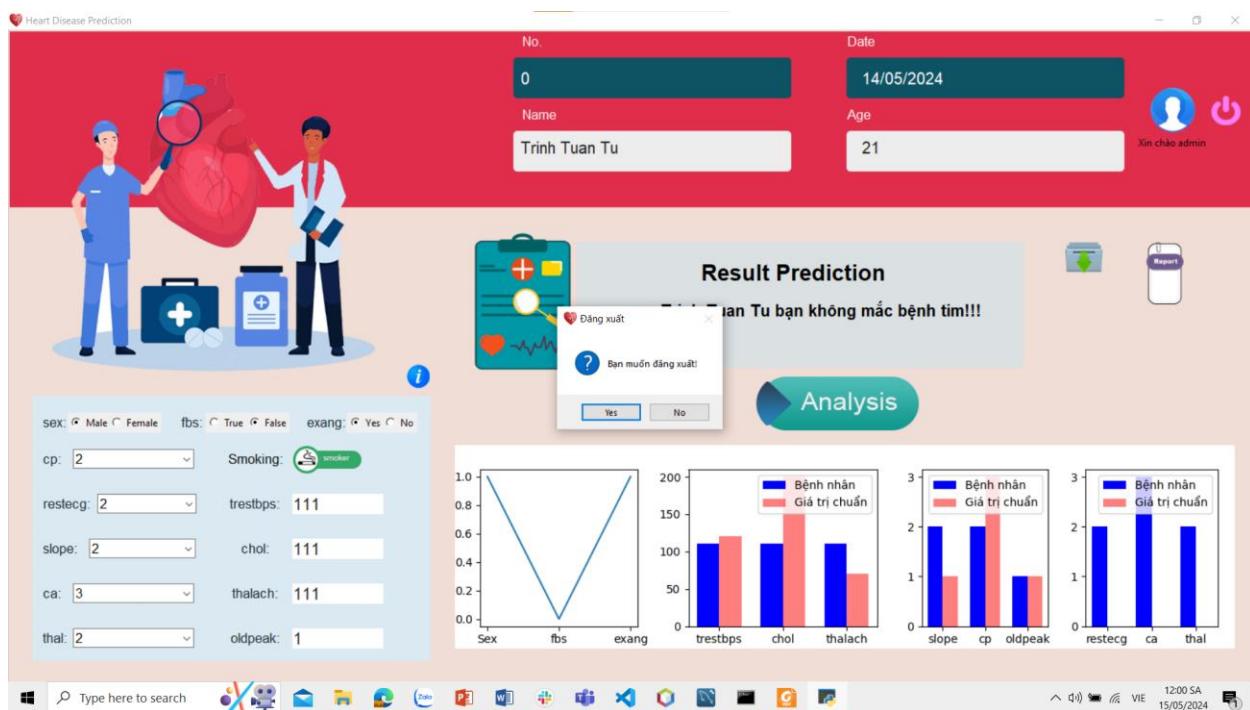
+ Ràng buộc đăng ký

The screenshot shows a sign-up interface. At the top left is a 'Sign Up' button. The main area features a colorful illustration of three people using laptops and tablets. To the right, the word 'Sign up' is displayed in large blue letters. Below it is a text input field containing 'tutrinh'. There are two password input fields, both marked with a red error message: 'Xác nhận mật khẩu không đúng!!!'. A blue 'Sign up' button is positioned below these fields. At the bottom, there are links for 'I have an account?' and 'Sign In'.

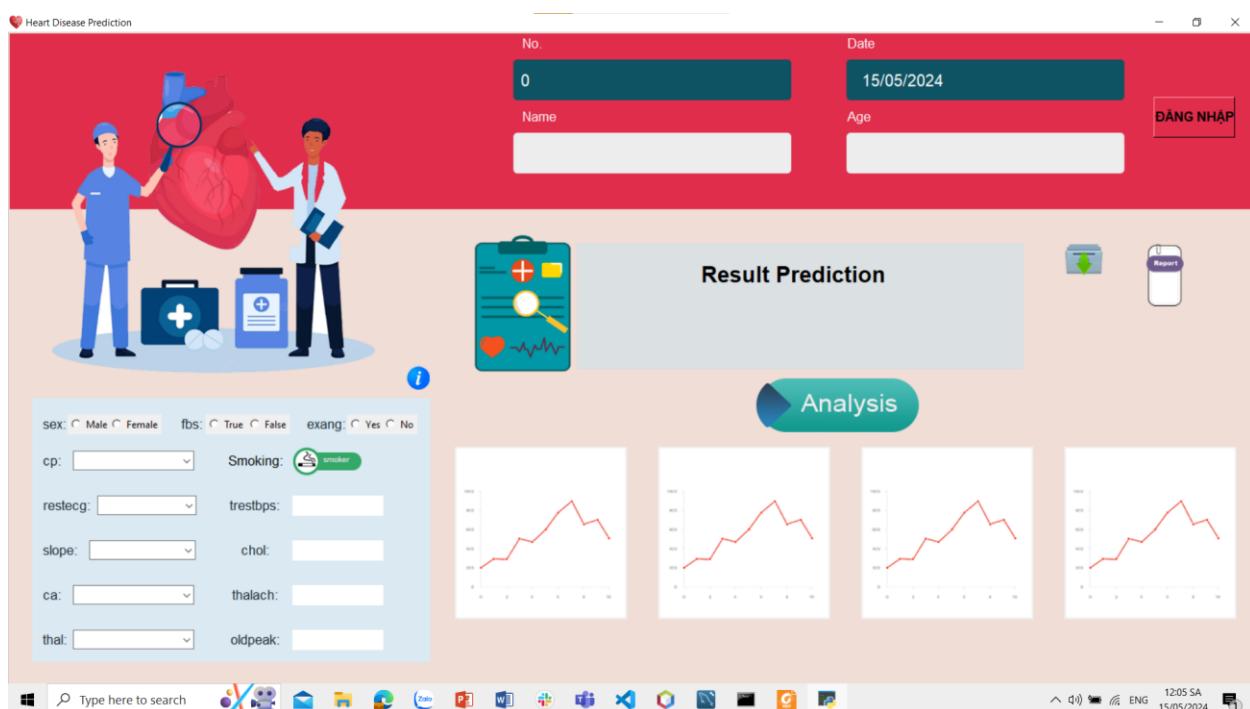


+ Đăng ký thành công

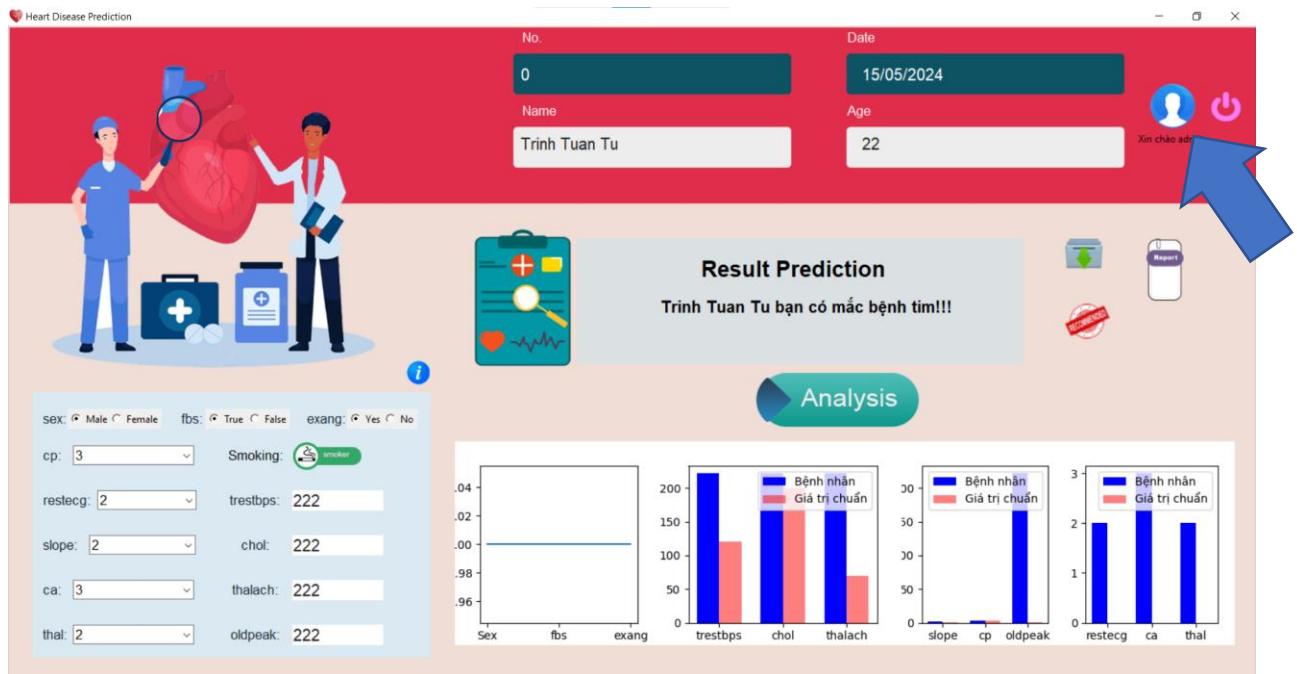
- Chức năng đăng xuất



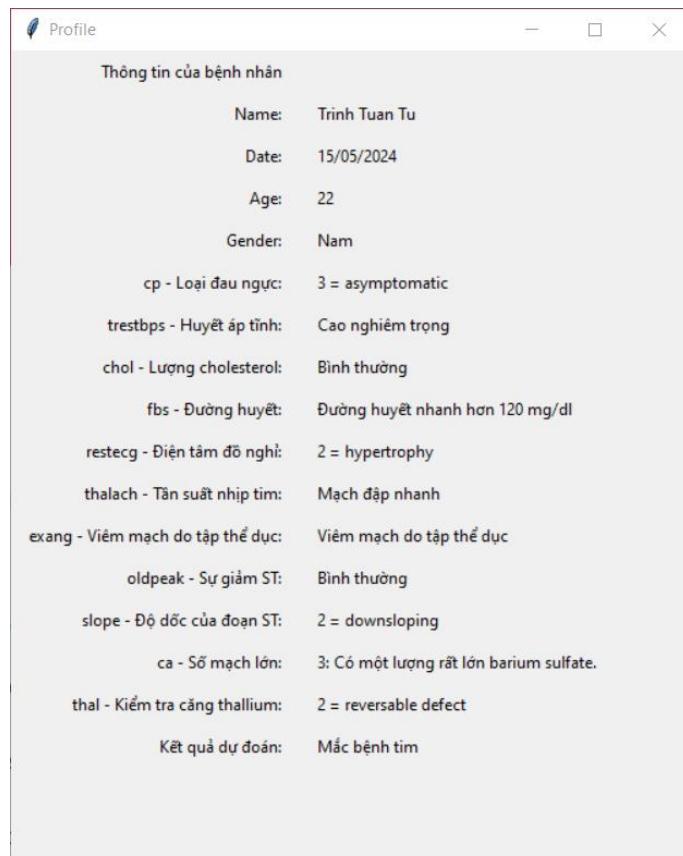
- Sau khi đăng xuất



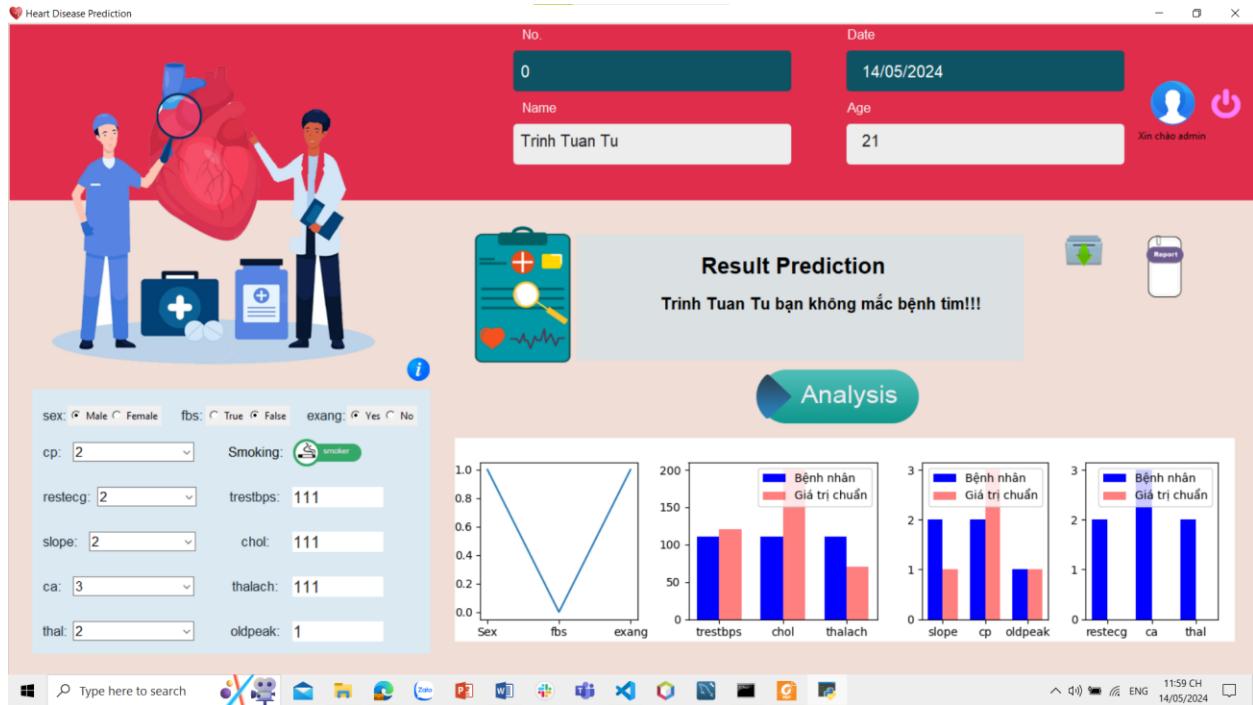
- Chức năng xem thông tin người dùng



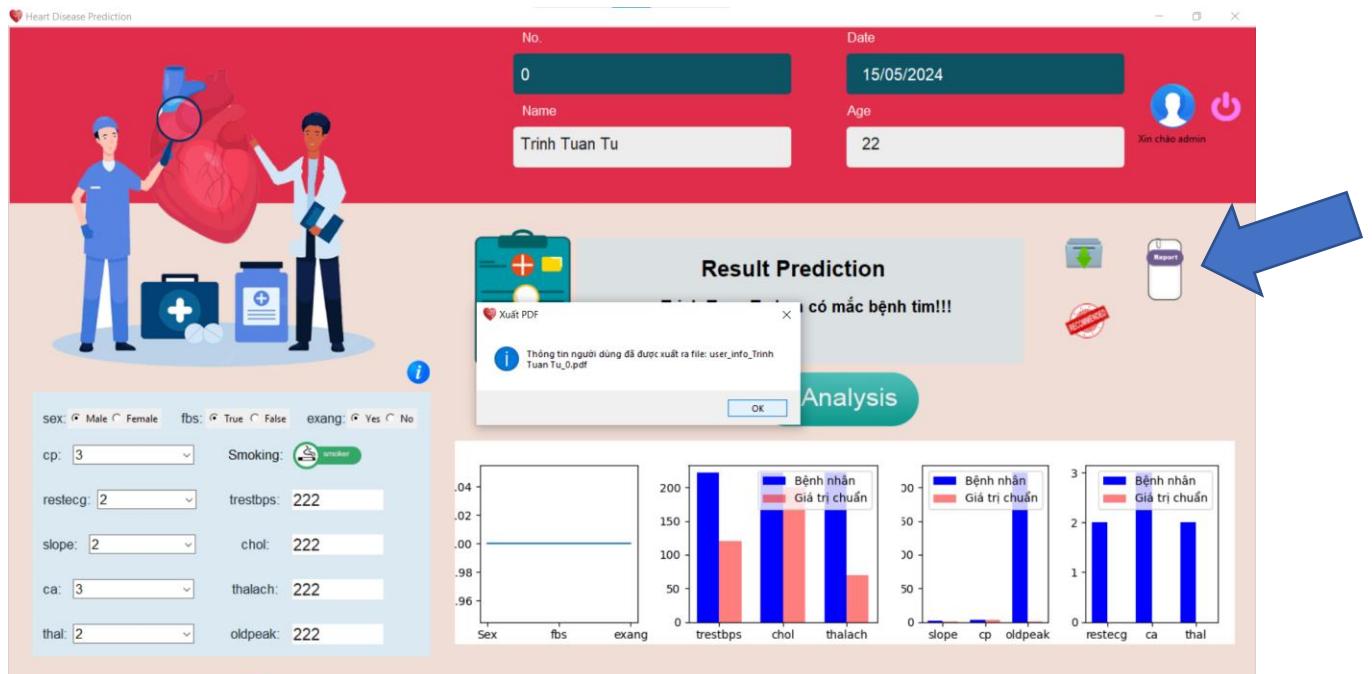
- Thông tin người dùng sẽ hiện ra

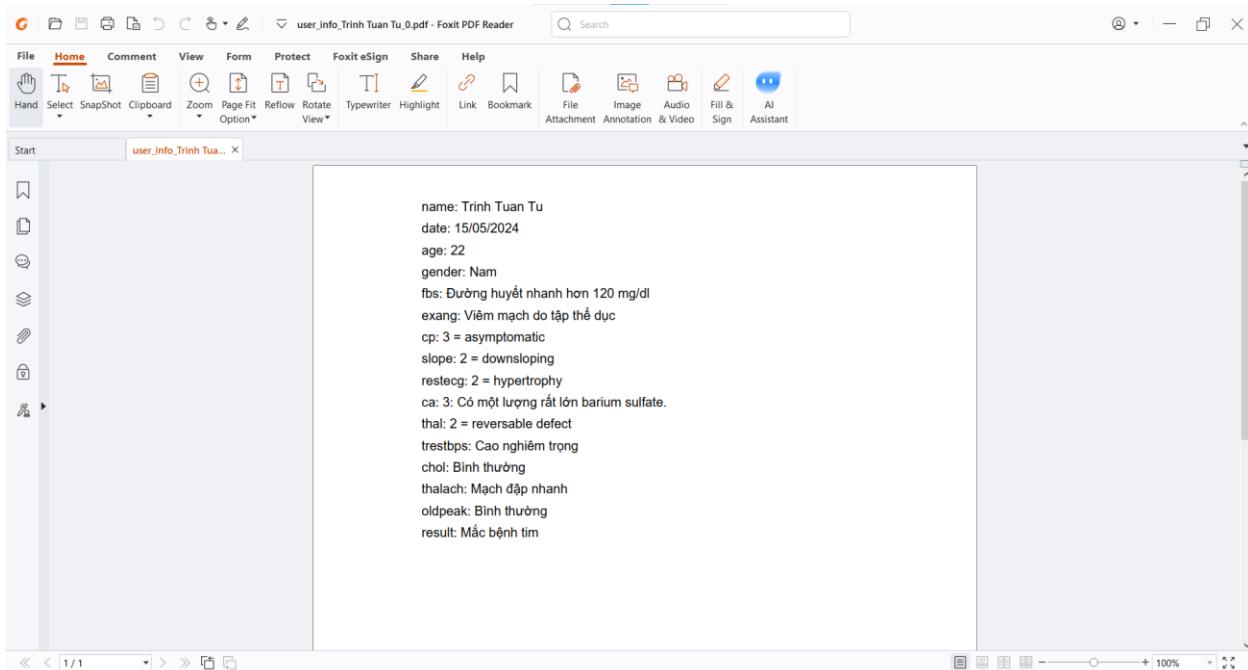


- Chức năng vẽ biểu đồ

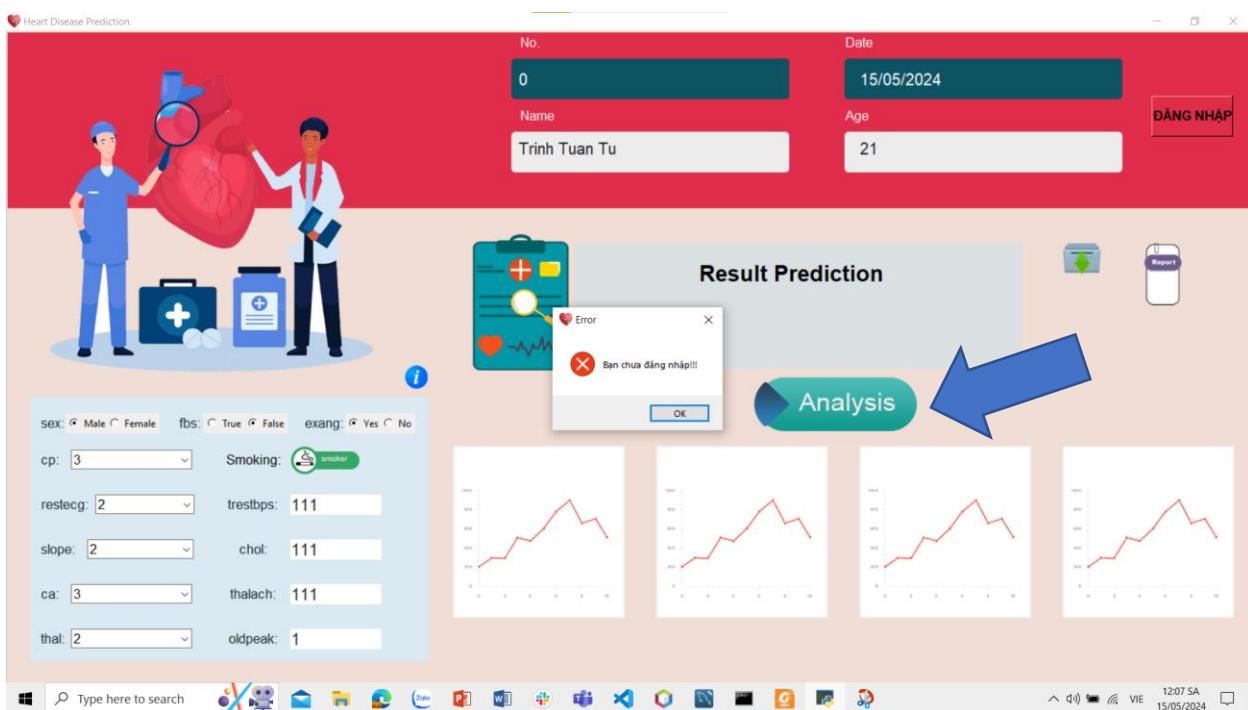


- Chức năng xuất báo cáo





- Ràng buộc đăng nhập



- Database

```
Command Prompt - mysql -u root -p

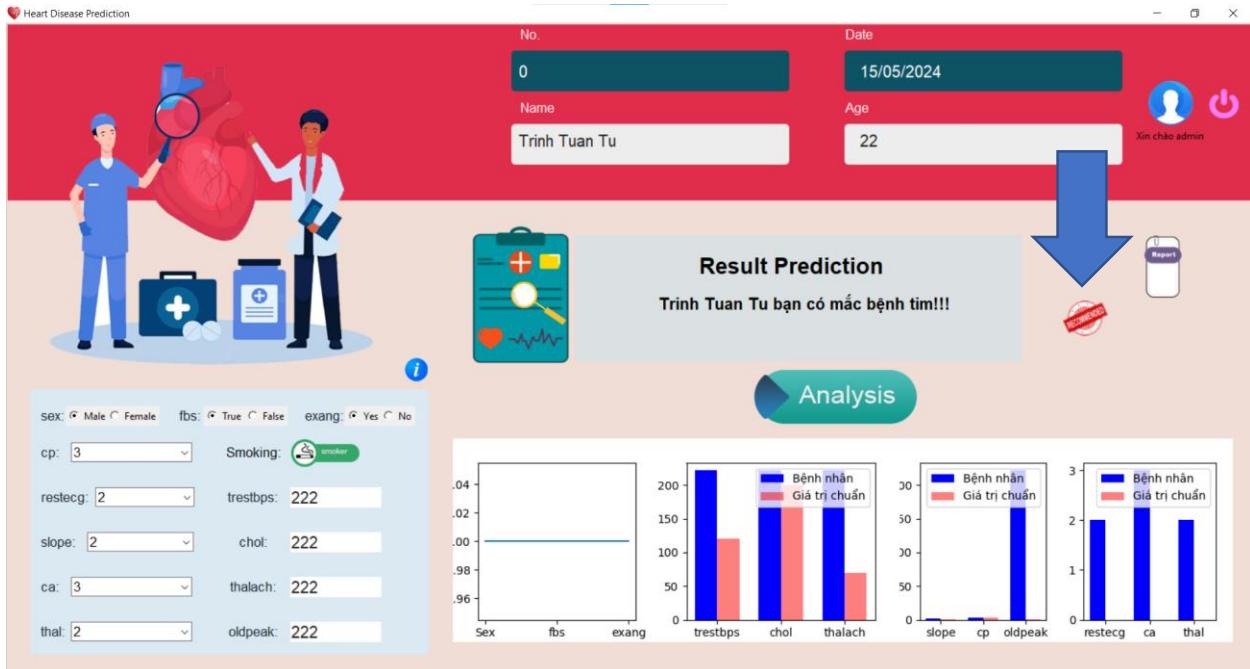
mysql> select * from data;
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| user | Name | Date | BD | sex | cp | trestbps | chol | fbs | restecg | thalach | exang | oldpeak | slope | ca | thal | result |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| 1 | Trinh Tu | 08/05/2024 | 21 | 1 | 2 | 2 | 231 | 1 | 2 | 2 | 1 | 2 | 2 | 1 | 3 | 0 |
| 2 | Tu | 11/05/2024 | 21 | 1 | 3 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1 |
| 3 | MR A | 13/05/2024 | 22 | 1 | 2 | 111 | 111 | 0 | 2 | 111 | 1 | 1 | 2 | 3 | 3 | 0 |
| 4 | test | 14/05/2024 | 12 | 1 | 2 | 111 | 111 | 0 | 2 | 111 | 1 | 1 | 2 | 3 | 3 | 1 |
| 5 | df | 14/05/2024 | 21 | 1 | 2 | 111 | 111 | 1 | 2 | 111 | 1 | 1 | 2 | 3 | 2 | 1 |
| 6 | tu | 14/05/2024 | 21 | 1 | 1 | 111 | 111 | 1 | 2 | 111 | 1 | 1 | 2 | 3 | 2 | 1 |
| 7 | tu | 14/05/2024 | 22 | 1 | 1 | 111 | 111 | 1 | 1 | 111 | 1 | 1 | 2 | 3 | 2 | 0 |
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
7 rows in set (0.01 sec)

mysql> select * from login;
+-----+-----+-----+
| user | Username | Password |
+-----+-----+-----+
| 1 | admin | admin |
| 4 | admin1 | admin |
| 5 | t | t |
| 6 | tu | tu |
| 8 | tutrinh | trinhtuantu |
+-----+-----+-----+
5 rows in set (0.01 sec)

mysql>
```

```
Command Prompt - mysql -u root -p
mysql> use heart_data;
Database changed
mysql> select * from recommendations;
+-----+-----+
| id | recommendation |
+-----+-----+
| 1 | 1. Hãy thử khám thi?ng xuyên kiểm tra s?c kh?e v?i b?p s? ch?y?n khoa tim m?ch. |
| 2 | 2. Tuần th? ch? ?? ?n ưng lanh m?nh và t?ng c?m?ng ho?t ??m? th? ch?t. |
| 3 | 3. Tuần th? li?u ph?p v?i ?i?u tr?: ?i?u quan tr?ng nh?t l? tuân th? ?ung li?u ph?p ???c ch? ??nh b?i b?p s?. |
| 4 | ?i?u n?y c? th? bao g?m vi?c s? d?ng thu?c theo ?ung h?ng d?, tuân th? ch? ?? ?n u?ng v? t?p th? d?c ph?u h?p. |
| 5 | 4. Ch? ?? ?n ưng lanh m?nh: Hãy ?n it ch?t béo b?p h?o và cholesterol, h?n ch? natri, ???ng v? th?c ?n ch? bi?n. |
| 6 | Thêm vào ?, hãy t?ng c?m?ng ti?u th? rau c?, hoa qu?, h?t v? các protein lanh m?nh nh? c? v? g??. |
| 7 | 5. T?p th? d?c: Th?c hi?n c?c ho?t ??ng th? ch?t nh? ?i b?, ?p xe, b?i l?i, yoga ho?c c?c bài t?p aerobic nh? nh?ng. |
| 8 | Tuy nh?n, tr?i?c kh? b?t k? ch?ng trình t?p luy?n m?i n?o, hãy th?o lu?n v?i b?p s? ?? ?m b?o r?ng n? l? ph?u h?p v?i tinh tr?ng s?c kh?e c?a b?p. |
| 9 | 6. K?m soát c?n n?ng: ?i?i v?i nh?ng g??i m?c b?p tim, vi?c g?i c?n n?ng ? m?c ?n ??nh l? r?t quan tr?ng. |
| 10 | 7. Qu?n lý c?ng th?ng: C?ng th?ng c? th? g?y ra t?ng huy?i áp v? t?ng nguy c? b?p?n tim. |
| 11 | 8. Ki?m tra ??nh k?: ?i?u n?y r?t quan tr?ng ?? theo d?i tri?n c?a b?p?n tim v? ?i?u ch?n h?i l?u ph?p khi c?n thi?t. |
| 12 | 9. Hãy ?m b?p b?p ??n ki?m tra s?c kh?e ??nh k? theo h?ng d?n c?a b?p s?. |
| 13 | 10. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 14 | Nếu b?p h?p th?c, hãy t?p ki?m s? h? tr? ?? d?ng l?p. |
| 15 | 11. H?p ch? c?m: U?ng ri?u c?m c? th? t?ng nguy c? b?p tim. Hãy h?p ch? ti?u th? c?n ho?c t?t nh?t l? tr?nh hoàn toàn. |
| 16 | 12. Hãy g?i: m?t t?p tr?ng tich c?c: T?p tr?ng tich c?c c? th? c? l?p?i cho s?c kh?e tim m?ch. |
| 17 | 13. Hãy t?p ki?m s? h? tr? t? b?p be, g?ia ?inh ho?c c?c nh?n h? tr? n?u c?n. |
| 18 | 14. Hãy d?ng h?p th?c: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 19 | 15. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 20 | 16. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 21 | 17. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 22 | 18. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 23 | 19. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch. |
| 24 | 20. Hãy t?p h?p: H?p th?c l? m?t trong nh?ng y?u t? r?i ro l?n ??i v?i s?c kh?e tim m?ch.
```

- Chức năng khuyến cáo

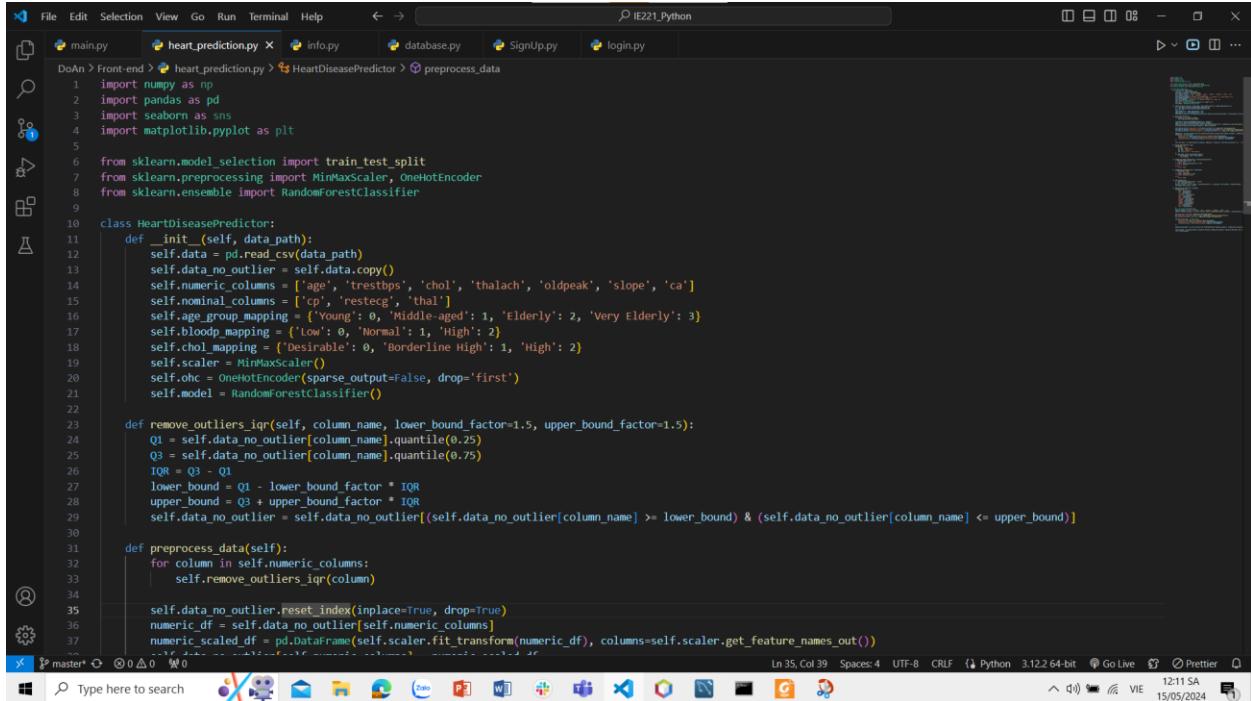


Recommendations

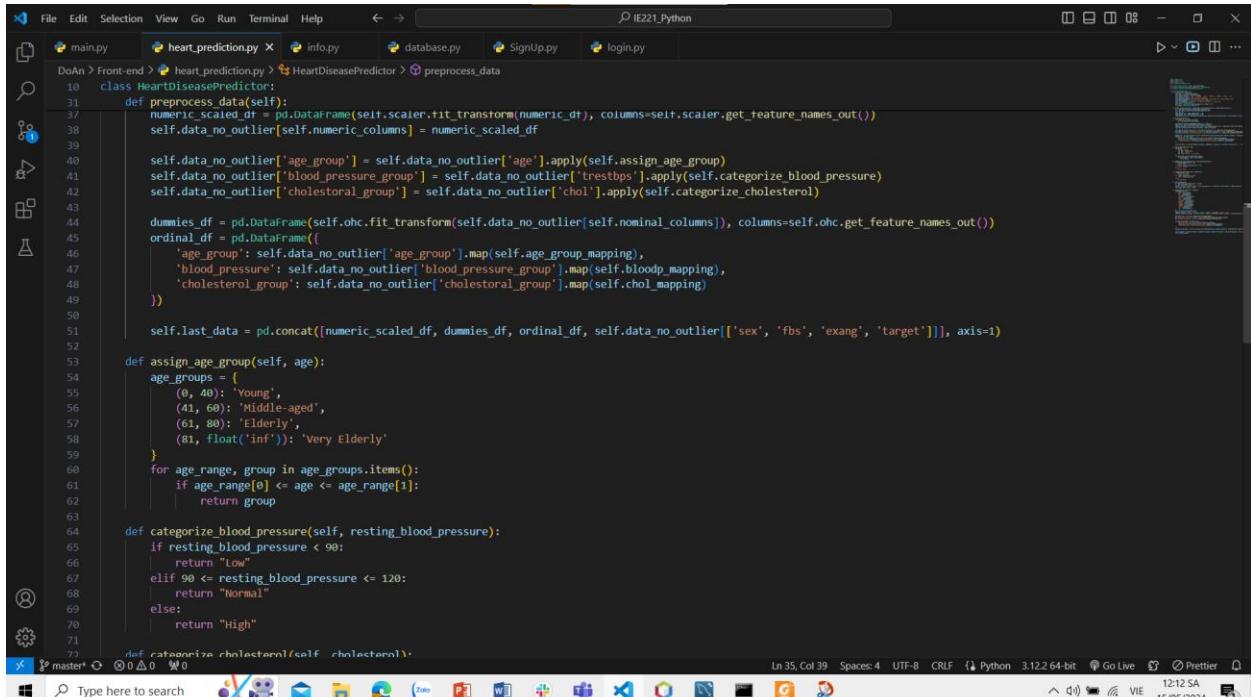
Dựa trên kết quả dự đoán, dưới đây là một số khuyến cáo

- Hãy thực hiện thường xuyên kiểm tra sức khỏe với bác sĩ chuyên khoa tim mạch.
- Tuân thủ chế độ ăn uống lành mạnh và tăng cường hoạt động thể chất.
- Tuân thủ liệu pháp và điều trị: Điều quan trọng nhất là tuân thủ đúng liệu pháp được chỉ định bởi bác sĩ. Điều này có thể bao gồm việc sử dụng thuốc theo đúng hướng dẫn, tuân thủ chế độ ăn uống và tập thể dục phù hợp.
- Chế độ ăn uống lành mạnh: Hãy ăn ít chất béo hòa và cholesterol, hạn chế natri, đường và thức ăn chế biến. Thêm vào đó, hãy tăng cường tiêu thụ rau củ, hoa quả, hạt và các nguồn protein lành mạnh như cá và gà.
- Tập thể dục: Thực hiện các hoạt động thể chất như đi bộ, đạp xe, bơi lội, yoga hoặc các bài tập aerobic nhẹ nhàng. Tuy nhiên, trước khi bắt đầu bất kỳ chương trình tập luyện nào, hãy thảo luận với bác sĩ để đảm bảo rằng nó là phù hợp với tình trạng sức khỏe của bạn.
- Kiểm soát cân nặng: Đối với những người mắc bệnh tim, việc giữ cân nặng ở mức ổn định là rất quan trọng. Hãy tìm hiểu về chế độ ăn uống cân đối và duy trì một lối sống lành mạnh để giúp kiểm soát cân nặng.
- Quản lý căng thẳng: Căng thẳng có thể gây ra tăng huyết áp và tăng nguy cơ bệnh tim. Hãy tìm hiểu các kỹ thuật giảm căng thẳng như thiền, yoga, hoặc các phương pháp thư giãn khác để giúp giảm căng thẳng và cải thiện tâm trạng.
- Kiểm tra định kỳ: Điều này rất quan trọng để theo dõi tiến triển của bệnh tim và điều chỉnh liệu pháp khi cần thiết. Hãy đảm bảo bạn đến kiểm tra sức khỏe định kỳ theo hướng dẫn của bác sĩ.
- Hãy dừng hút thuốc: Hút thuốc lá là một trong những yếu tố rủi ro lớn đối với sức khỏe tim mạch. Nếu bạn hút thuốc, hãy tìm kiếm sự hỗ trợ để dừng lại.
- Hạn chế cồn: Uống rượu cồn có thể tăng nguy cơ bệnh tim. Hãy hạn chế tiêu thụ cồn hoặc tốt nhất là tránh hoàn toàn.
- Hãy giữ một tâm trạng tích cực: Tâm trạng tích cực có thể có lợi cho sức khỏe tim mạch. Hãy tìm kiếm sự hỗ trợ từ bạn bè, gia đình hoặc các nhóm hỗ trợ nếu cần.

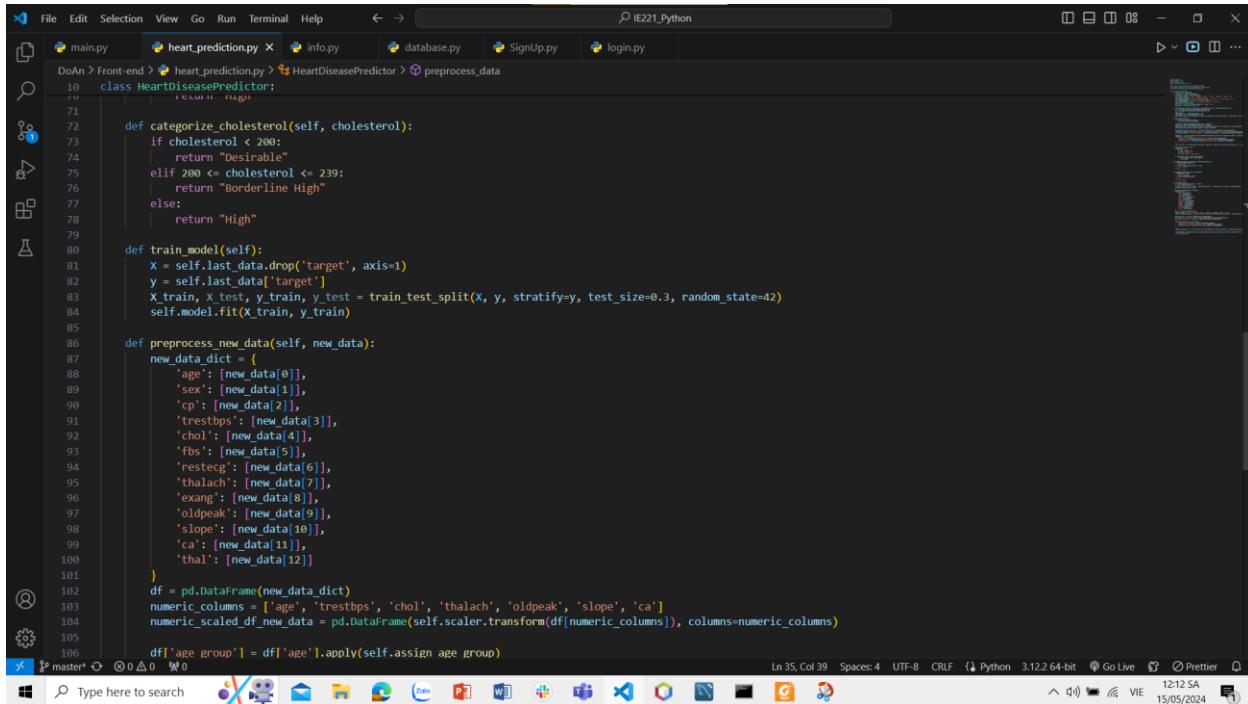
- File heart_prediction.py



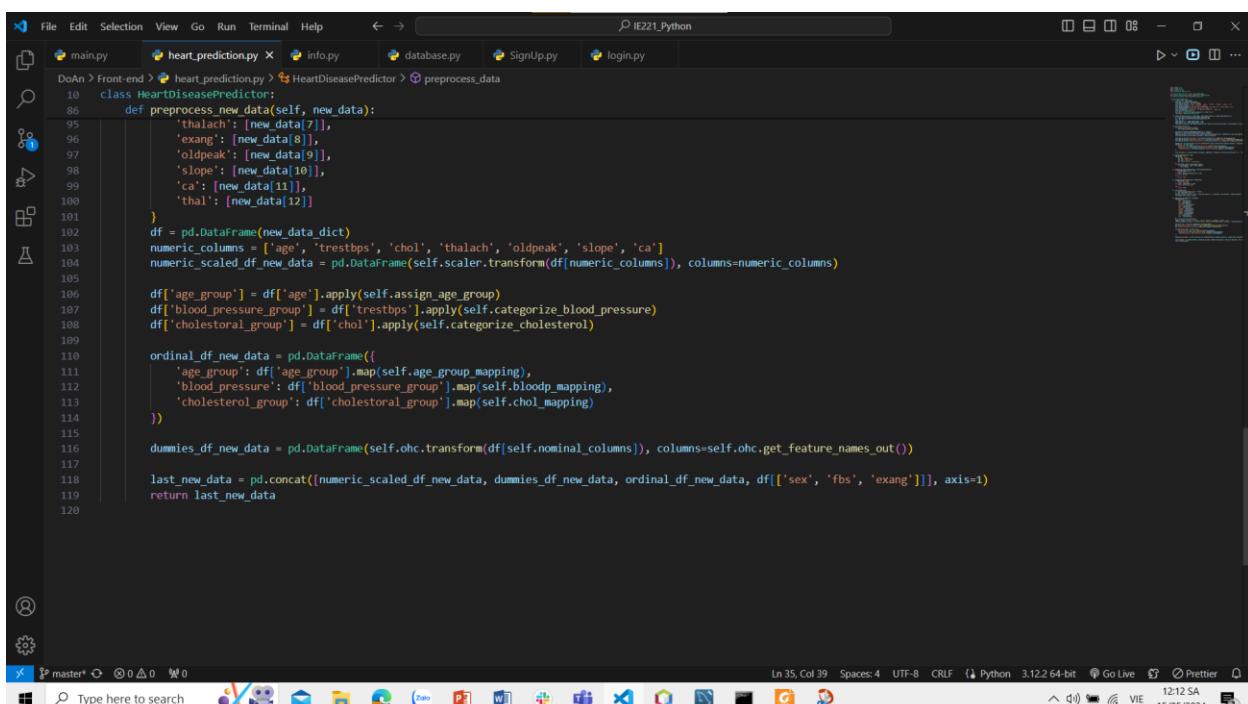
```
File Edit Selection View Go Run Terminal Help ↵ → IE221_Python
main.py heart_prediction.py info.py database.py SignUp.py login.py
DoAn > Front-end > heart_prediction.py > HeartDiseasePredictor > preprocess_data
1 import numpy as np
2 import pandas as pd
3 import seaborn as sns
4 import matplotlib.pyplot as plt
5
6 from sklearn.model_selection import train_test_split
7 from sklearn.preprocessing import MinMaxScaler, OneHotEncoder
8 from sklearn.ensemble import RandomForestClassifier
9
10 class HeartDiseasePredictor:
11     def __init__(self, data_path):
12         self.data = pd.read_csv(data_path)
13         self.data_no_outlier = self.data.copy()
14         self.numeric_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'slope', 'ca']
15         self.nominal_columns = ['cp', 'restecg', 'thal']
16         self.age_group_mapping = { 'young': 0, 'Middle-aged': 1, 'Elderly': 2, 'Very Elderly': 3}
17         self.bloodp_mapping = { 'low': 0, 'Normal': 1, 'high': 2}
18         self.chol_mapping = { 'desirable': 0, 'Borderline High': 1, 'High': 2}
19         self.scaler = MinMaxScaler()
20         self.ohc = OneHotEncoder(sparse_output=False, drop='first')
21         self.model = RandomForestClassifier()
22
23     def remove_outliers_iqr(self, column_name, lower_bound_factor=1.5, upper_bound_factor=1.5):
24         Q1 = self.data_no_outlier[column_name].quantile(0.25)
25         Q3 = self.data_no_outlier[column_name].quantile(0.75)
26         IQR = Q3 - Q1
27         lower_bound = Q1 - lower_bound_factor * IQR
28         upper_bound = Q3 + upper_bound_factor * IQR
29         self.data_no_outlier = self.data_no_outlier[(self.data_no_outlier[column_name] >= lower_bound) & (self.data_no_outlier[column_name] <= upper_bound)]
30
31     def preprocess_data(self):
32         for column in self.numeric_columns:
33             self.remove_outliers_iqr(column)
34
35         self.data_no_outlier.reset_index(inplace=True, drop=True)
36         numeric_df = self.data_no_outlier[self.numeric_columns]
37         numeric_scaled_df = pd.DataFrame(self.scaler.fit_transform(numeric_df), columns=self.scaler.get_feature_names_out())
38
39         dummies_df = pd.DataFrame(self.ohc.fit_transform(self.data_no_outlier[self.nominal_columns]), columns=self.ohc.get_feature_names_out())
40         ordinal_df = pd.DataFrame({
41             'age_group': self.data_no_outlier['age'].apply(self.assign_age_group),
42             'blood_pressure_group': self.data_no_outlier['trestbps'].apply(self.categorize_blood_pressure),
43             'cholesterol_group': self.data_no_outlier['chol'].apply(self.categorize_cholesterol)
44         })
45
46         self.last_data = pd.concat([numeric_scaled_df, dummies_df, ordinal_df, self.data_no_outlier[['sex', 'fbs', 'exang', 'target']]], axis=1)
47
48     def assign_age_group(self, age):
49         age_groups = {
50             (0, 40): 'Young',
51             (41, 60): 'Middle-aged',
52             (61, 80): 'Elderly',
53             (81, float('inf')): 'Very Elderly'
54         }
55         for age_range, group in age_groups.items():
56             if age_range[0] <= age <= age_range[1]:
57                 return group
58
59     def categorize_blood_pressure(self, resting_blood_pressure):
60         if resting_blood_pressure < 90:
61             return "low"
62         elif 90 <= resting_blood_pressure <= 120:
63             return "Normal"
64         else:
65             return "High"
66
67     def categorize_cholesterol(self, cholesterol):
68
69
70
71
72
73
74
75
76
77
```



```
File Edit Selection View Go Run Terminal Help ↵ → IE221_Python
main.py heart_prediction.py info.py database.py SignUp.py login.py
DoAn > Front-end > heart_prediction.py > HeartDiseasePredictor > preprocess_data
10 class HeartDiseasePredictor:
11     def preprocess_data(self):
12         numeric_scaled_df = pd.DataFrame(self.scaler.fit_transform(self.data_no_outlier[self.numeric_columns]), columns=self.scaler.get_feature_names_out())
13         self.data_no_outlier[self.numeric_columns] = numeric_scaled_df
14
15         self.data_no_outlier['age_group'] = self.data_no_outlier['age'].apply(self.assign_age_group)
16         self.data_no_outlier['blood_pressure_group'] = self.data_no_outlier['trestbps'].apply(self.categorize_blood_pressure)
17         self.data_no_outlier['cholesterol_group'] = self.data_no_outlier['chol'].apply(self.categorize_cholesterol)
18
19         dummies_df = pd.DataFrame(self.ohc.fit_transform(self.data_no_outlier[self.nominal_columns]), columns=self.ohc.get_feature_names_out())
20         ordinal_df = pd.DataFrame({
21             'age_group': self.data_no_outlier['age'].map(self.assign_age_group),
22             'blood_pressure': self.data_no_outlier['blood_pressure_group'].map(self.bloodp_mapping),
23             'cholesterol_group': self.data_no_outlier['cholesterol_group'].map(self.chol_mapping)
24         })
25
26         self.last_data = pd.concat([numeric_scaled_df, dummies_df, ordinal_df, self.data_no_outlier[['sex', 'fbs', 'exang', 'target']]], axis=1)
27
28     def assign_age_group(self, age):
29         age_groups = {
30             (0, 40): 'Young',
31             (41, 60): 'Middle-aged',
32             (61, 80): 'Elderly',
33             (81, float('inf')): 'Very Elderly'
34         }
35         for age_range, group in age_groups.items():
36             if age_range[0] <= age <= age_range[1]:
37                 return group
38
39     def categorize_blood_pressure(self, resting_blood_pressure):
40         if resting_blood_pressure < 90:
41             return "low"
42         elif 90 <= resting_blood_pressure <= 120:
43             return "Normal"
44         else:
45             return "High"
46
47     def categorize_cholesterol(self, cholesterol):
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
79
```



```
DoAn > Front-end > heart_prediction.py > HeartDiseasePredictor > preprocess_data
10  class HeartDiseasePredictor:
11      ...
12
13      def categorize_cholesterol(self, cholesterol):
14          if cholesterol < 200:
15              return "Desirable"
16          elif 200 <= cholesterol <= 239:
17              return "Borderline High"
18          else:
19              return "High"
20
21      def train_model(self):
22          X = self.last_data.drop('target', axis=1)
23          y = self.last_data['target']
24          X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y, test_size=0.3, random_state=42)
25          self.model.fit(X_train, y_train)
26
27
28      def preprocess_new_data(self, new_data):
29          new_data_dict = {
30              'age': [new_data[0]],
31              'sex': [new_data[1]],
32              'cp': [new_data[2]],
33              'trestbps': [new_data[3]],
34              'chol': [new_data[4]],
35              'fbs': [new_data[5]],
36              'restecg': [new_data[6]],
37              'thalach': [new_data[7]],
38              'exang': [new_data[8]],
39              'oldpeak': [new_data[9]],
40              'slope': [new_data[10]],
41              'ca': [new_data[11]],
42              'thal': [new_data[12]]
43          }
44          df = pd.DataFrame(new_data_dict)
45          numeric_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'slope', 'ca']
46          numeric_scaled_df_new_data = pd.DataFrame(self.scaler.transform(df[numeric_columns]), columns=numeric_columns)
47
48          df['age_group'] = df['age'].apply(self.assign_age_group)
49
50
51
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
```



```
DoAn > Front-end > heart_prediction.py > HeartDiseasePredictor > preprocess_data
10  class HeartDiseasePredictor:
11      ...
12
13      def preprocess_new_data(self, new_data):
14          new_data_dict = {
15              'thalach': [new_data[7]],
16              'exang': [new_data[8]],
17              'oldpeak': [new_data[9]],
18              'slope': [new_data[10]],
19              'ca': [new_data[11]],
20              'thal': [new_data[12]]
21          }
22          df = pd.DataFrame(new_data_dict)
23          numeric_columns = ['age', 'trestbps', 'chol', 'thalach', 'oldpeak', 'slope', 'ca']
24          numeric_scaled_df_new_data = pd.DataFrame(self.scaler.transform(df[numeric_columns]), columns=numeric_columns)
25
26          df['age_group'] = df['age'].apply(self.assign_age_group)
27          df['blood_pressure_group'] = df['trestbps'].apply(self.categorize_blood_pressure)
28          df['cholesterol_group'] = df['chol'].apply(self.categorize_cholesterol)
29
30          ordinal_df_new_data = pd.DataFrame({
31              'age_group': df['age_group'].map(self.age_group_mapping),
32              'blood_pressure': df['blood_pressure_group'].map(self.blood_group_mapping),
33              'cholesterol_group': df['cholesterol_group'].map(self.chol_mapping)
34          })
35
36          dummies_df_new_data = pd.DataFrame(self.ohe.transform(df[self.nominal_columns]), columns=self.ohe.get_feature_names_out())
37
38          last_new_data = pd.concat([numeric_scaled_df_new_data, dummies_df_new_data, ordinal_df_new_data, df[['sex', 'fbs', 'exang']]], axis=1)
39
40          return last_new_data
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
109
110
111
112
113
114
115
116
117
118
119
120
```

6. Phụ lục 2: docstring

- File main.py

Import thư viện

HeartDiseasePredictionApp:

Ứng dụng Dự đoán Bệnh tim.

Attributes:

root (tk.Tk): Cửa sổ chính của ứng dụng.

image_icon (tk.PhotoImage): Hình ảnh frame của ứng dụng.

header (tk.PhotoImage): Hình ảnh tiêu đề của ứng dụng.

prediction (int): Kết quả dự đoán bệnh tim.

is_logged_in (bool): Trạng thái đăng nhập của người dùng.

Phương thức:

__init__(self, root): Phương thức khởi tạo đối tượng HeartDiseasePredictionApp với các thuộc tính cần thiết và cấu hình giao diện người dùng.

create_heading_entry_frame(self): Tạo frame tiêu đề và nhập thông tin chung của bệnh nhân.

create_detail_entry_frame(self): Tạo frame để nhập thông tin chi tiết về các thông số của bệnh nhân.

create_report_frame(self): Tạo frame để hiển thị báo cáo dự đoán kết quả với các thông số đã nhập.

create_graphs(self): Tạo các đồ thị dự đoán mặc định và hiển thị chúng trên giao diện.

create_buttons(self): Tạo và thiết lập các nút điều khiển trên giao diện người dùng.

LoginForm(self): Hiển thị giao diện đăng nhập và xử lý sự kiện đăng nhập.

Logout(self): Đăng xuất và hiển thị giao diện đăng nhập.

change_mode(self): Chuyển đổi giữa chế độ hút thuốc và không hút thuốc.

Clear(self): Xóa trống các trường thông tin trong ứng dụng sau khi đăng xuất.

selecGender(self): Lấy thông tin giới tính từ RadioButton.

selecfbs(self): Lấy thông tin fbs từ RadioButton.

selecxang(self): Lấy thông tin exang từ RadioButton.

seleccp(self): Lấy thông tin cp từ Combobox.

selecslope(self): Lấy thông tin slope từ Combobox.

check_login(self): Kiểm tra trạng thái đăng nhập của người dùng.

create_bar_chart(self, figure_size, position, patient_data, standard_values, labels, x, y):
Tạo biểu đồ cột.

Analysis(self): Phương thức này thực hiện phân tích dữ liệu bệnh tim của người dùng và hiển thị kết quả dự đoán, cùng với các biểu đồ thống kê.

Save(self): Phương thức này lưu dữ liệu phân tích bệnh tim của người dùng vào cơ sở dữ liệu.

User(self): Phương thức này hiển thị thông tin của người dùng trong cửa sổ Profile.

display_recommendations(self): Phương thức này hiển thị khuyến cáo cho người mắc bệnh tim.

export_to_pdf(self): Phương thức này xuất thông tin của người dùng ra file PDF.

__init__:

Khởi tạo đối tượng HeartDiseasePredictionApp.

Args:

root (Tk): Cửa sổ gốc của ứng dụng.

Hàm này khởi tạo một đối tượng ứng dụng dự đoán bệnh tim mạch. Nó cài đặt các thuộc tính cần thiết và cấu hình giao diện người dùng.

Các bước thực hiện bao gồm:

- Thiết lập cửa sổ gốc với tiêu đề "Heart Disease Prediction" và kích thước 1550x800 pixels.
- Cấu hình màu nền của ứng dụng.
- Đặt biểu tượng cho ứng dụng.
- Hiển thị header của ứng dụng.

- Tạo các khung chứa các thành phần giao diện khác nhau như thông tin đầu vào, báo cáo, biểu đồ và nút điều khiển.
- Khởi tạo một số biến quan trọng như prediction (dự đoán) và is_logged_in (đã đăng nhập hay chưa).

create_heading_entry_frame:

Tạo khung tiêu đề và nhập các thông tin chung của bệnh nhân.

Hàm này tạo một khung tiêu đề bao gồm các trường nhập liệu cho các thông tin chung của bệnh nhân như số thứ tự, ngày tháng hiện tại, tên và tuổi.

Các trường thông tin bao gồm:

- Số thứ tự(No.)
- Ngày (Date)
- Tên (Name)
- Tuổi (Age)

create_detail_entry_frame:

Tạo frame nhập thông tin các thông số của bệnh nhân.

Hàm này tạo một frame để nhập thông tin chi tiết về các thông số của bệnh nhân, bao gồm giới tính, đường huyết, tình trạng tập thể dục, cảm giác đau ngực, kết quả điện tâm đồ nghỉ, độ dốc của đoạn ST, số mạch và hình dạng của mạch và các chỉ số huyết áp, cholesterol, nhịp tim tối đa, cũng như đỉnh giảm ST.

Các bước thực hiện bao gồm:

- Tạo một frame với kích thước và màu nền nhất định để chứa các thành phần.
- Tạo các nút radio cho giới tính, đường huyết nhanh và tình trạng tập thể dục.

- Tạo các hộp combobox để chọn các thông số khác như cảm giác đau ngực, kết quả điện tâm đồ nghỉ, độ dốc của đoạn ST, số mạch, và hình dạng mạch.
- Tạo các ô nhập dữ liệu cho các chỉ số huyết áp, cholesterol, nhịp tim tối đa và đỉnh giảm ST.

create_report_frame:

Tạo frame hiển thị báo cáo dự đoán kết quả.

Hàm này tạo một frame để hiển thị báo cáo dự đoán kết quả với các thông số đã nhập.

Các bước thực hiện bao gồm:

- Tạo một hình ảnh nền cho frame hiển thị báo cáo.
- Tạo một frame với kích thước và màu nền để chứa các phần tử hiển thị báo cáo.
- Hiển thị tiêu đề "Result Prediction".
- Hiển thị nội dung báo cáo dự đoán kết quả.

create_graphs:

Tạo các đồ thị dự đoán mặc định, hiển thị khi người dùng chưa dự đoán.

Hàm này tạo các đồ thị dự đoán và hiển thị chúng trên giao diện.

Các bước thực hiện bao gồm:

- Tạo một hình ảnh đồ thị từ tệp hình ảnh đã cho.
- Tạo các nhãn hình ảnh cho các đồ thị và thiết lập hình ảnh của chúng.
- Đặt vị trí của các nhãn hình ảnh trên giao diện để hiển thị đồ thị.

create_buttons:

Tạo các nút điều khiển trên giao diện.

Hàm này tạo và thiết lập các nút điều khiển trên giao diện người dùng.

Các bước thực hiện bao gồm:

- Tạo nút ĐĂNG NHẬP với các thuộc tính như chữ, font, màu sắc, và vị trí.
- Tạo nút Đăng Xuất với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút Prediction với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút Info với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút Save với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút Kiểm Tra Hút Thuốc với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút User với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.
- Tạo nút Xuất PDF với hình ảnh và thiết lập các thuộc tính như màu sắc và vị trí.

LoginForm:

Hiển thị giao diện đăng nhập và xử lý sự kiện đăng nhập.

Hàm này mở cửa sổ đăng nhập và xử lý sự kiện đăng nhập bằng cách gọi hàm `login_callback`.

Args:

success (bool): Kết quả đăng nhập thành công hoặc không.

username (str): Tên người dùng đăng nhập thành công.

Logout:

Đăng xuất và hiển thị giao diện đăng nhập.

Hàm này yêu cầu xác nhận từ người dùng trước khi đăng xuất và xóa thông tin người dùng đã đăng nhập.

Returns:

None

change_mode:

Chuyển đổi giữa chế độ hút thuốc và không hút thuốc.

Hàm này thay đổi hình ảnh của nút và cập nhật lựa chọn tương ứng.

Returns:

None

Clear:

Xóa trắng các trường thông tin trong ứng dụng sau khi đăng xuất.

Returns:

None

selecGender:

Lấy thông tin giới tính từ RadioButton.

Returns:

int or None: Trả về 1 nếu là Nam, 0 nếu là Nữ, None nếu không có lựa chọn.

selecfbs:

Lấy thông tin fbs từ RadioButton.

Returns:

int: Trả về 1 nếu có, 0 nếu không.

selecexang:

Lấy thông tin exang từ RadioButton.

Returns:

int: Trả về 1 nếu có, 0 nếu không.

seleccp:

Lấy thông tin cp từ Combobox.

Returns:

int: Giá trị cp được chọn.

selecslope:

Lấy thông tin slope từ Combobox.

Returns:

int: Giá trị slope được chọn.

check_login:

Kiểm tra trạng thái đăng nhập của người dùng.

Returns:

bool: True nếu đã đăng nhập, False nếu chưa đăng nhập.

create_bar_chart:

Tạo biểu đồ cột.

Args:

figure_size (tuple): Kích thước của hình vẽ.

position (tuple): Vị trí của biểu đồ trên cửa sổ.

patient_data (dict): Dữ liệu của bệnh nhân.

standard_values (dict): Giá trị chuẩn.

labels (list): Danh sách nhãn.

x (int): Tọa độ x của biểu đồ trên cửa sổ.

y (int): Tọa độ y của biểu đồ trên cửa sổ.

Analysis:

Phân tích dữ liệu bệnh tim của người dùng và hiển thị kết quả dự đoán, cùng với các biểu đồ thống kê.

Hàm này thực hiện các bước sau:

1. Kiểm tra xem người dùng đã đăng nhập chưa, nếu chưa thì kết thúc hàm.
2. Nhận dữ liệu nhập từ giao diện người dùng và kiểm tra tính hợp lệ của dữ liệu.
3. Tiến hành dự đoán bệnh tim dựa trên dữ liệu nhập và hiển thị kết quả.
4. Tạo và hiển thị các biểu đồ thống kê về các thuộc tính liên quan đến bệnh tim.
5. Loại bỏ các nhãn của biểu đồ thống kê nếu chúng đã tồn tại.

Returns:

None

Raises:

ValueError: Nếu không nhập tên.

Save:

Lưu dữ liệu phân tích bệnh tim của người dùng vào cơ sở dữ liệu.

Hàm này thực hiện các bước sau:

1. Kiểm tra xem người dùng đã đăng nhập chưa, nếu chưa thì kết thúc hàm.

2. Nhận dữ liệu nhập từ giao diện người dùng và kiểm tra tính hợp lệ của dữ liệu.
3. Kết nối đến cơ sở dữ liệu và tạo bảng nếu chưa tồn tại.
4. Lưu dữ liệu vào cơ sở dữ liệu.

Returns:

None

Raises:

None

User:

Hiển thị thông tin của người dùng trong cửa sổ Profile.

Hàm này thực hiện các bước sau:

1. Tạo một cửa sổ con (Toplevel) để hiển thị thông tin người dùng.
2. Tạo và định dạng các nhãn để hiển thị thông tin chi tiết của người dùng, bao gồm:
 - Tên, Ngày sinh, Tuổi, Giới tính,
 - Loại đau ngực, Huyết áp tĩnh, Lượng cholesterol, Đường huyết,
 - Điện tâm đồ nghỉ, Tần suất nhịp tim, Viêm mạch do tập thể dục,
 - Sự giảm ST, Độ dốc của đoạn ST, Sô mạch lớn, Kiểm tra cảng thallium,
 - Kết quả dự đoán về mắc bệnh tim.
3. Hiển thị các giá trị tương ứng với thông tin của người dùng.

Returns:

None

Raises:

None

display_recommendations:

Hiển thị các khuyến cáo dựa trên kết quả dự đoán về bệnh tim.

Parameters:

self (object): Đối tượng của lớp chứa phương thức.

Returns:

None

Ghi chú:

Hàm này tạo một cửa sổ mới hoặc cửa sổ con để hiển thị các khuyến cáo liên quan đến sức khỏe tim mạch.

Nếu dự đoán là mắc bệnh tim, hàm sẽ hiển thị các khuyến cáo cụ thể về kiểm tra sức khỏe, chế độ ăn uống,

tập thể dục, kiểm soát cân nặng, quản lý căng thẳng và các hành vi khác có lợi cho sức khỏe tim mạch.

Nếu không mắc bệnh tim, hàm sẽ hiển thị thông báo cho biết sức khỏe đang ổn định và khuyến khích duy trì một lối sống lành mạnh.

export_to_pdf:

Xuất thông tin của người dùng ra file PDF.

Hàm này thực hiện các bước sau:

1. Kiểm tra xem người dùng đã đăng nhập chưa, nếu chưa thì kết thúc hàm.
2. Tạo dữ liệu từ các thông tin của người dùng.
3. Đăng ký font chữ và tạo file PDF.
4. Lưu thông tin của người dùng vào file PDF và thông báo thành công.

Returns:

None

Raises:

None

- File heart_prediction.py

HeartDiseasePredictor:

Lớp này dùng để dự đoán bệnh tim dựa trên dữ liệu từ tệp CSV.

Thuộc tính:

- data: Dữ liệu gốc từ tệp CSV.
- data_no_outlier: Dữ liệu đã loại bỏ các giá trị ngoại lai.
- numeric_columns: Các cột số trong dữ liệu.
- nominal_columns: Các cột nominal trong dữ liệu.
- age_group_mapping: nhóm tuổi thành các giá trị số.
- bloodp_mapping: nhóm huyết áp thành các giá trị số.
- chol_mapping: nhóm cholesterol thành các giá trị số.
- scaler: Bộ chuẩn hóa dữ liệu MinMaxScaler.
- ohc: Bộ mã hóa OneHotEncoder cho các cột nominal.
- model: Mô hình phân loại RandomForestClassifier.

Phương thức:

- __init__(self, data_path): Khởi tạo đối tượng HeartDiseasePredictor.
- remove_outliers_iqr(self, column_name, lower_bound_factor=1.5, upper_bound_factor=1.5): Loại bỏ các giá trị ngoại lai trong cột dữ liệu dựa trên phương pháp IQR.
- assign_age_group(self, age): Phân nhóm tuổi dựa trên giá trị tuổi.
- categorize_blood_pressure(self, resting_blood_pressure): Phân loại nhóm huyết áp dựa trên giá trị huyết áp nghỉ ngơi.

- categorize_cholesterol(self, cholesterol): Phân loại nhóm cholesterol dựa trên giá trị cholesterol.
- preprocess_data(self): Tiền xử lý dữ liệu bao gồm loại bỏ giá trị ngoại lai, chuẩn hóa dữ liệu số, mã hóa các biến nominal và biến hạng.
- train_model(self): Huấn luyện mô hình RandomForestClassifier trên dữ liệu đã tiền xử lý.
- preprocess_new_data(self, new_data): Tiền xử lý dữ liệu mới để dự đoán.

__init__:

Khởi tạo đối tượng HeartDiseasePredictor.

Tham số:

- data_path: Đường dẫn đến tệp CSV chứa dữ liệu.

remove_outliers_iqr:

Loại bỏ các giá trị ngoại lai trong cột dữ liệu dựa trên phương pháp IQR.

Tham số:

- column_name: Tên cột cần loại bỏ giá trị ngoại lai.
- lower_bound_factor: Hệ số cho cận dưới của IQR.
- upper_bound_factor: Hệ số cho cận trên của IQR.

assign_age_group:

Phân nhóm tuổi dựa trên giá trị tuổi.

Tham số:

- age: Tuổi của bệnh nhân.

Trả về:

- Nhóm tuổi tương ứng.

categorize_blood_pressure:

Phân loại nhóm huyết áp dựa trên giá trị huyết áp nghỉ ngơi.

Tham số:

- resting_blood_pressure: Giá trị huyết áp nghỉ ngơi.

Trả về:

- Nhóm huyết áp tương ứng.

categorize_cholesterol:

Phân loại nhóm cholesterol dựa trên giá trị cholesterol.

Tham số:

- cholesterol: Giá trị cholesterol.

Trả về:

- Nhóm cholesterol tương ứng.

preprocess_data:

Tiền xử lý dữ liệu bao gồm: loại bỏ giá trị ngoại lai, chuẩn hóa dữ liệu số, mã hóa các biến nominal và biến hạng.

Hàm này thực hiện các bước sau:

1. Loại bỏ giá trị ngoại lai cho các biến số.
2. Chuẩn hóa dữ liệu số sử dụng phép chia tỷ lệ.
3. Mã hóa biến nominal và biến hạng thành dạng số.
4. Ghép các DataFrame sau xử lý thành một DataFrame duy nhất.

Returns:

None

Raises:

None

train_model:

Huấn luyện mô hình RandomForestClassifier trên dữ liệu đã tiền xử lý.

Hàm này thực hiện các bước sau:

1. Tách dữ liệu thành các tập huấn luyện và kiểm tra.
2. Tiến hành huấn luyện mô hình RandomForestClassifier trên tập huấn luyện.

Returns:

None

Raises:

None

preprocess_new_data:

Tiền xử lý dữ liệu mới để dự đoán.

Tham số:

- new_data: Dữ liệu mới dưới dạng danh sách.

Trả về:

- Dữ liệu mới đã tiền xử lý.

- File info.py

InfoData:

Class để hiển thị thông tin dataset.

Attributes:

icon_info (Toplevel): Cửa sổ để hiển thị thông tin.

Phương thức:

__init__(self): Khởi tạo đối tượng InfoData và thiết lập giao diện người dùng.

setup_ui(self): Thiết lập giao diện người dùng cho cửa sổ thông tin.

Info:

Hiển thị cửa sổ thông tin về dataset.

Returns:

None

__init__:

Khởi tạo đối tượng InfoData và thiết lập giao diện người dùng.

setup_ui:

Thiết lập giao diện người dùng cho cửa sổ thông tin.

Gồm các bước sau:

1. Thiết lập biểu tượng cho cửa sổ.
2. Tạo khung chứa thông tin.
3. Hiển thị tiêu đề.
4. Hiển thị các nhãn thông tin về dataset.

Returns:

None

- File database.py

DatabaseManager:

class này quản lý cơ sở dữ liệu

Attributes:

host (str): Địa chỉ host của cơ sở dữ liệu.

user (str): Tên người dùng để truy cập cơ sở dữ liệu.

password (str): Mật khẩu để truy cập cơ sở dữ liệu.

database_name (str): Tên cơ sở dữ liệu.

connection: Kết nối đến cơ sở dữ liệu.

cursor: Con trỏ để thực hiện các truy vấn SQL.

Phương thức:

__init__(self, host, user, password, database_name): Phương thức khởi tạo một đối tượng DatabaseManager

và thiết lập các thuộc tính cơ bản như host, user, password, database_name, connection, và cursor.

connect(self): Phương thức này kết nối đến cơ sở dữ liệu MySQL sử dụng thông tin được cung cấp.

create_tables(self): Phương thức này tạo các bảng trong cơ sở dữ liệu nếu chúng chưa tồn tại.

save_data(self, name, today, age, B2, C2, D2, E2, F2, G2, H2, I2, J2, K2, L2, M2, result):
Phương thức này lưu thông tin của một bệnh nhân vào cơ sở dữ liệu.

login(self, username, password): Phương thức này thêm một tài khoản người dùng mới vào cơ sở dữ liệu.

close_connection(self): Phương thức này đóng kết nối với cơ sở dữ liệu.

__init__:

Khởi tạo một đối tượng DatabaseManager.

Args:

host (str): Địa chỉ host của cơ sở dữ liệu.

user (str): Tên người dùng để truy cập cơ sở dữ liệu.

password (str): Mật khẩu để truy cập cơ sở dữ liệu.

database_name (str): Tên cơ sở dữ liệu.

Returns:

None

connect:

Kết nối đến cơ sở dữ liệu.

Raises:

messagebox.showerror: Hiển thị thông báo lỗi nếu kết nối không thành công.

Returns:

None

create_tables:

Tạo các bảng trong cơ sở dữ liệu nếu chúng chưa tồn tại.

Returns:

None

save_data:

Lưu thông tin của một bệnh nhân vào cơ sở dữ liệu.

Args:

name: Tên của bệnh nhân.

today: Ngày.

age: Tuổi của bệnh nhân.

B2: Giới tính của bệnh nhân.

C2: Loại đau ngực.

D2: Huyết áp tĩnh.

E2: Lượng cholesterol.

F2: Đường huyết.

G2: Điện tâm đồ nghỉ.

H2: Tần suất nhịp tim.

I2: Viêm mạch do tập thể dục.

J2: Sự giảm ST.

K2: Độ dốc của đoạn ST.

L2: Số mạch lớn.

M2: Kiểm tra căng thallium.

result: Kết quả dự đoán.

Returns:

None

insert_recommendations:

Chèn các khuyến cáo vào bảng cơ sở dữ liệu.

Parameters:

cursor: Đối tượng con trỏ của MySQL.

recommendations (list): Danh sách các khuyến cáo cần chèn vào cơ sở dữ liệu.

login:

Thêm một tài khoản người dùng mới vào cơ sở dữ liệu.

Args:

username: Tên người dùng.

password: Mật khẩu.

Returns:

None

close_connection:

Đóng kết nối với cơ sở dữ liệu.

Returns:

None

- File SignUp.py

SignUpForm:

Lớp SignUpForm chịu trách nhiệm tạo và hiển thị form đăng ký người dùng.

Attributes:

callback (function): Hàm gọi lại khi đăng ký thành công.

root (Toplevel): Cửa sổ Toplevel cho form đăng ký.

Phương thức:

__init__(self, callback): Phương thức khởi tạo, tạo một đối tượng SignUpForm với callback như một tham số.

setup_ui(self): Thiết lập giao diện người dùng cho form đăng ký.

`setup_entries(self)`: Thiết lập các ô nhập liệu cho tên người dùng, mật khẩu và xác nhận mật khẩu.

`setup_buttons(self)`: Thiết lập các nút bấm cho form đăng ký.

`create_entry(self, placeholder, y, show=None)`: Tạo một ô nhập liệu với placeholder và vị trí xác định.

`login_form(self)`: Chuyển sang form đăng nhập.

`signup_user(self)`: Xử lý đăng ký người dùng.

Sign_Up:

Khởi tạo và hiển thị form đăng ký.

Hàm này tạo ra một instance của lớp SignUpForm và hiển thị nó.

Args:

`callback (function)`: Hàm callback được gọi khi người dùng đăng ký thành công.

__init__:

Khởi tạo một đối tượng SignUpForm.

Args:

`callback`: Hàm gọi lại khi đăng ký thành công.

setup_ui:

Thiết lập giao diện người dùng cho form đăng ký.

Hàm này thiết lập các thành phần giao diện người dùng bao gồm hình ảnh nền, biểu tượng, khung chứa form đăng ký, nhãn tiêu đề, nhãn thông báo lỗi, và gọi các hàm để thiết lập các ô nhập liệu và nút bấm.

setup_entries:

Thiết lập các ô nhập liệu cho tên người dùng, mật khẩu và xác nhận mật khẩu.

Hàm này tạo ra các ô nhập liệu cho tên người dùng, mật khẩu và xác nhận mật khẩu, và đặt chúng ở các vị trí xác định trong khung chúa.

setup_buttons:

Thiết lập các nút bấm cho form đăng ký.

Hàm này tạo ra nút "Sign up" để đăng ký và nút "Sign In" để chuyển sang form đăng nhập, và đặt chúng ở các vị trí xác định trong khung chúa.

create_entry:

Tạo một ô nhập liệu với placeholder và vị trí xác định.

Hàm này tạo ra một ô nhập liệu với placeholder, đặt vị trí của ô trên khung chúa và thiết lập

các sự kiện focus vào và ra cho ô nhập liệu.

Args:

placeholder (str): Văn bản hiển thị mặc định trong ô nhập liệu.

y (int): Vị trí theo trục y để đặt ô nhập liệu.

show (str, optional): Ký tự để hiển thị thay cho ký tự nhập vào.

Returns:

Entry: Ô nhập liệu đã được tạo.

login_form:

Chuyển sang form đăng nhập.

Hàm này đóng cửa sổ hiện tại và mở form đăng nhập mới.

signup_user:

Xử lý đăng ký người dùng.

Hàm này lấy thông tin từ các ô nhập liệu, kiểm tra tính hợp lệ của thông tin và thực hiện đăng ký người dùng vào cơ sở dữ liệu. Nếu có lỗi xảy ra, hiển thị thông báo lỗi tương ứng.

- File login.py

LoginForm:

Lớp LoginForm chịu trách nhiệm tạo và hiển thị form đăng nhập.

Thuộc tính:

callback (function): Hàm callback được gọi khi người dùng đăng nhập thành công.

limit (int): Giới hạn số lần thử đăng nhập không thành công.

root (Toplevel): Cửa sổ Toplevel cho form đăng nhập.

Phương thức:

__init__(self, callback): Khởi tạo form đăng nhập với các thuộc tính và thiết lập giao diện người dùng.

setup_ui(self): Thiết lập giao diện người dùng cho form đăng nhập.

setup_entries(self): Thiết lập các ô nhập liệu cho tên người dùng và mật khẩu.

setup_buttons(self): Thiết lập các nút bấm cho form đăng nhập.

enter_user(self, e): Xóa văn bản mặc định trong ô nhập liệu tên người dùng khi có sự kiện focus.

leave_user(self, e): Đặt lại văn bản mặc định trong ô nhập liệu tên người dùng khi mất sự kiện focus nếu trống.

`enter_passwd(self, e)`: Xóa văn bản mặc định và thiết lập chế độ hiển thị mật khẩu khi có sự kiện focus.

`leave_passwd(self, e)`: Đặt lại văn bản mặc định trong ô nhập liệu mật khẩu khi mất sự kiện focus nếu trống.

`signup(self)`: Chuyển sang form đăng ký.

`login_user(self)`: Xử lý đăng nhập người dùng.

`limit_login(self)`: Giới hạn số lần thử đăng nhập không thành công.

`Login_Form`:

Khởi tạo và hiển thị form đăng nhập.

Hàm này tạo ra một instance của lớp LoginForm và gọi hàm khởi tạo của nó.

Args:

`callback (function)`: Hàm callback được gọi khi người dùng đăng nhập thành công.

`__init__`:

Khởi tạo một instance của LoginForm.

Hàm này thiết lập hàm callback, giới hạn số lần thử đăng nhập, tạo cửa sổ Toplevel cho form

đăng nhập và gọi hàm thiết lập giao diện người dùng.

Args:

`callback (function)`: Hàm callback được gọi khi người dùng đăng nhập thành công.

`setup_ui`:

Thiết lập giao diện người dùng cho form đăng nhập.

Hàm này thiết lập các thành phần giao diện người dùng bao gồm hình ảnh nền, biểu tượng, khung chứa form đăng nhập, nhãn tiêu đề, nhãn thông báo lỗi, và gọi các hàm để thiết lập các ô nhập liệu và nút bấm.

setup_entries:

Thiết lập các ô nhập liệu cho tên người dùng và mật khẩu.

Hàm này tạo ra các ô nhập liệu cho tên người dùng và mật khẩu, đặt chúng ở các vị trí xác định

trong khung chứa và thiết lập các sự kiện focus vào và ra cho các ô nhập liệu.

setup_buttons:

Thiết lập các nút bấm cho form đăng nhập.

Hàm này tạo ra nút "Log in" để đăng nhập và nút "Sign Up" để chuyển sang form đăng ký, và đặt chúng ở các vị trí xác định trong khung chứa.

enter_user:

Xóa văn bản mặc định trong ô nhập liệu tên người dùng khi có sự kiện focus.

Args:

e (Event): Sự kiện focus.

leave_user:

Đặt lại văn bản mặc định trong ô nhập liệu tên người dùng khi mất sự kiện focus nếu trống.

Args:

e (Event): Sự kiện mất focus.

enter_passwd:

Xóa văn bản mặc định và thiết lập chế độ hiển thị mật khẩu khi có sự kiện focus.

Args:

e (Event): Sự kiện focus.

leave_passwd:

Đặt lại văn bản mặc định trong ô nhập liệu mật khẩu khi mất sự kiện focus nếu trống.

Args:

e (Event): Sự kiện mất focus.

signup:

Chuyển sang form đăng ký.

Hàm này đóng cửa sổ hiện tại và mở form đăng ký mới.

login_user:

Xử lý đăng nhập người dùng.

Hàm này lấy thông tin từ các ô nhập liệu, kiểm tra tính hợp lệ của thông tin và thực hiện kiểm tra thông tin đăng nhập trong cơ sở dữ liệu. Nếu thông tin không hợp lệ hoặc sai, hiển thị

thông báo lỗi và giới hạn số lần thử đăng nhập.

limit_login:

Giới hạn số lần thử đăng nhập không thành công.

Hàm này tăng biến đếm số lần thử đăng nhập không thành công. Nếu số lần thử vượt quá 3, hiển thị thông báo và đóng cửa sổ đăng nhập.