

CR Projet Final

LELEU Nicolas

ITS1

But du projet : Créer une interface homme machine qui permet à l'utilisateur de rentrer les informations ainsi que les symptômes d'un patient pour obtenir les médicaments à lui prescrire.

Pour ce faire , le programme se sépare en plusieurs programme , un pour chaque fenêtre créer.

Lien GitHub où un fichier .rar où tout le contenu du projet est téléchargeable :

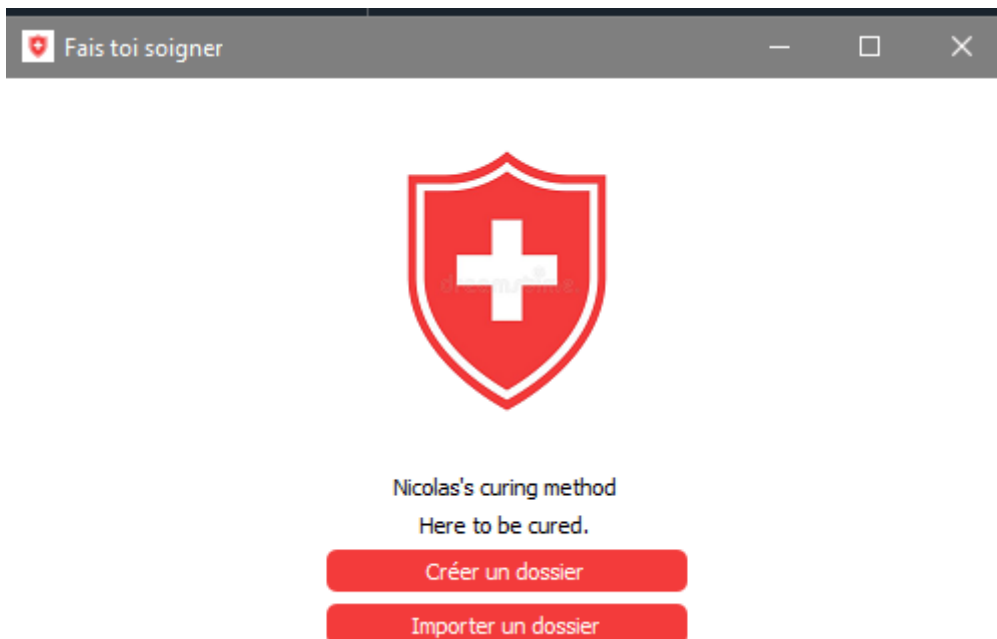
https://github.com/NicolasKerso/projet_final_POO

1^{ère} fenêtre : L'accueil

Import de tout les modules nécessaires :

```
import dbm
import sys
from PyQt5.QtWidgets import (QLineEdit, QPushButton, QVBoxLayout, QHBoxLayout,
                             QApplication, QMainWindow, QWidget, QLabel, QRadioButton)
from PyQt5.QtGui import QPixmap, QIcon
from PyQt5 import QtCore
from fenetre2 import FenetreDossier
```

Capture d'écran de la fenêtre d'accueil :



Ajout des boutons , zones de textes , images ... :

```
class View(QWidget):
    def __init__(self):
        super().__init__()

        self.f2 = FenetreDossier(self)

        self.bcreer = QPushButton('Créer un dossier')
        self.bcreer.setStyleSheet("background-color: rgb(243,59,59) ; border-style: outset ; border-width: 0 px; border-radius: 5px; color : white ; padding: 4px")
        self.bimport = QPushButton('Importer un dossier')
        self.bimport.setStyleSheet("background-color: rgb(243,59,59) ; border-style: outset ; border-width: 0 px; border-radius: 5px ; color : white ; padding: 4px")

        self.label = QLabel(self)
        self.pixmap = QPixmap('im.png')
        self.label.setPixmap(self.pixmap)
        self.label.setAlignment(QtCore.Qt.AlignCenter)

        self.text = QLabel("Nicolas's curing method")
        self.text.setAlignment(QtCore.Qt.AlignCenter)
        self.text2 = QLabel("Here to be cured.")
        self.text2.setAlignment(QtCore.Qt.AlignCenter)

        self.init_ui1()

        self.show()
```

1/L'image centré est ajouté grâce à ces lignes (module QPixmap) :

```
self.label = QLabel(self)
self.pixmap = QPixmap('im.png')
self.label.setPixmap(self.pixmap)
self.label.setAlignment(QtCore.Qt.AlignCenter)
```

2/ Les zones de textes sont créés grâce à des « QLabel » que l'on a centré

```
self.text = QLabel("Nicolas's curing method")
self.text.setAlignment(QtCore.Qt.AlignCenter)
self.text2 = QLabel("Here to be cured.")
self.text2.setAlignment(QtCore.Qt.AlignCenter)
```

3/Les boutons poussoirs sont créés grâce à des « QPushButton » , auxquels on a changé leur couleur de fond ainsi que le texte affiché dessus.

```
self.bcreer = QPushButton('Créer un dossier')
self.bcreer.setStyleSheet("background-color: rgb(243,59,59) ; color : white")
self.bimport = QPushButton('Importer un dossier')
self.bimport.setStyleSheet("background-color: rgb(243,59,59) ; color : white")
```

Afin de pouvoir aligner les différents Widget , une méthode « init_ui1 » a été créée , pour utiliser des boîtes verticales et horizontales.

Voici les lignes de codes présente dans cette méthode :

```
def init_ui(self):
    h_box = QHBoxLayout()

    v_box = QVBoxLayout()
    v_box.setAlignment(Qt.AlignCenter)
    v_box.addLayout(h_box)
    v_box.addWidget(self.label)
    v_box.addWidget(self.text)
    v_box.addWidget(self.text2)
    v_box.addWidget(self.bcree)
    v_box.addWidget(self.bimport)

    self.setLayout(v_box)

    self.setGeometry(400,400,500,200)
    self.setWindowTitle('Fais toi soigner')
    self.setWindowIcon(QIcon('im.png'))

    self.bcree.clicked.connect(self.btn1_click)

    self.btn1_click(self)
```

1/ Les alignement verticaux sont gérés par ces lignes, ils sont tous alignés au centre grâce à QtAlignCenter

2/ La taille de la fenêtre lors de son ouverture , son titre et son icone sont respectivement gérés par ces commandes :

```
self.setGeometry(400,400,500,200)
self.setWindowTitle('Fais toi soigner')
self.setWindowIcon(QIcon('im.png'))
```

```
def btn1_click(self):
    self.hide()
    self.f2.show()

app = QApplication(sys.argv)
interface = View()
sys.exit(app.exec_())
```

Dans cette méthode de notre classe View , lors d'une pression du bouton « créer un dossier » , le fenêtre se cache , et en montre une nouvelle.

Pour en montrer une nouvelle , nous avons du instancier « f2 » , qui au départ est un attribut de classe , appelant une classe d'un autre programme.

```

16     from fenetre2 import FenetreDossier
17
18     class View(QWidget):
19         def __init__(self):
20
21             super().__init__()
22
23             self.f2 = FenetreDossier(self)

```

Grâce à cette instanciation , nous allons pouvoir relier les différents programmes.

Nous allons donc désormais voir , cette classe « FenetreDossier »

2^{ème} fenêtre : La fiche du patient / Le dossier

Capture d'écran de la fenêtre d'accueil (après avoir appuyer sur le bouton « Créer un dossier »)

I/Partie interface visuelle du code :

Création des différents Widget :

-Création des « Label » (rien de spécial ici)

```

self.setStyleSheet("background-color: rgb(230,230,230);")
self.nom = QLabel("Nom")
self.lenom = QLineEdit("")

self.prenom = QLabel("Prenom")
self.leprenom = QLineEdit("")

self.age = QLabel("Age")
self.leage = QLineEdit("")

self.sexe = QLabel("Sexe")
self.male = QRadioButton("Homme")
self.female = QRadioButton("Femme")

self.TEsymptome = QTextEdit()
self.TEsymptome.setStyleSheet("background-color: white ; color : black")
self.TEmed = QTextEdit()
self.TEmed.setReadOnly(True)

self.btRetour = QPushButton("Fermer")
self.btRetour.setFont(QFont('Arial', 11))
self.btRetour.setToolTip("Accueil")
self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

self.bHisto=QPushButton("Historique")
self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
self.bHisto.setToolTip("Historique du patient")
self.btRetour.setFont(QFont('Arial', 11))

self.bSave = QPushButton("Enregistrer")
self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
self.btRetour.setFont(QFont('Arial', 11))
self.bSave.setToolTip("Enregistrez votre dossier")

self.msg = QMessageBox()
self.msg.setIcon(QMessageBox.Critical)
self.msg.setText("Le patient est déjà enregistré , allez voir son historique")

self.init_fenDossier()
self.fprec = fprec

```

-Création des « QLineEdit » (rien de spécial non plus ici)

Chaque « QLineEdit » est éditable , un est utilisé pour le nom du patient, un autre pour son prénom et le dernier pour son âge

```

self.setStyleSheet("background-color: rgb(230,230,230);")
self.nom = QLabel("Nom")
self.lenom = QLineEdit("")

self.prenom = QLabel("Prenom")
self.leprenom = QLineEdit("")

self.age = QLabel("Age")
self.leage = QLineEdit("")

self.sexe = QLabel("Sexe")
self.male = QRadioButton("Homme")
self.female = QRadioButton("Femme")

self.TEsymptome = QTextEdit()
self.TEsymptome.setStyleSheet("background-color: white ; color : black")
self.TEmed = QTextEdit()
self.TEmed.setReadOnly(True)

self.btRetour = QPushButton("Fermer")
self.btRetour.setFont(QFont('Arial', 11))
self.btRetour.setToolTip("Accueil")
self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

self.bHisto=QPushButton("Historique")
self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
self.bHisto.setToolTip("Historique du patient")
self.btRetour.setFont(QFont('Arial', 11))

self.bSave = QPushButton("Enregistrer")
self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
self.btRetour.setFont(QFont('Arial', 11))
self.bSave.setToolTip("Enregistrez votre dossier")

self.msg = QMessageBox()
self.msg.setIcon(QMessageBox.Critical)
self.msg.setText("Le patient est déjà enregistré , allez voir son historique")

self.init_fenDossier()
self.fprec = fprec

```

-Création du « RadioButton »

Le « RadioButton » permet de faire un choix , ici c'est pour le sexe de la personne.

```

self.setStyleSheet("background-color: rgb(230,230,230);")
self.nom = QLabel("Nom")
self.lenom = QLineEdit("")

self.prenom= QLabel("Prenom")
self.leprenom = QLineEdit("")

self.age = QLabel("Âge")
self.leage = QLineEdit("")

self.sexe = QLabel("Sexe")
self.male = QRadioButton("Homme")
self.female = QRadioButton("Femme")

self.TEsymptome = QTextEdit()
self.TEsymptome.setStyleSheet("background-color: white ; color : black")
self.TEmed = QTextEdit()
self.TEmed.setReadOnly(True)

self.btRetour = QPushButton("Fermer")
self.btRetour.setFont(QFont('Arial', 11))
self.btRetour.setToolTip("Accueil")
self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

self.bHisto=QPushButton("Historique")
self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
self.bHisto.setToolTip("Historique du patient")
self.btRetour.setFont(QFont('Arial', 11))

self.bSave = QPushButton("Enregistrer")
self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
self.btRetour.setFont(QFont('Arial', 11))
self.bSave.setToolTip("Enregistrez votre dossier")

self.msg = QMessageBox()
self.msg.setIcon(QMessageBox.Critical)
self.msg.setText("Le patient est déjà enregistré , allez voir son historique")

self.init_fenDossier()
self.fprec = fprec

```

-Création des «TextEdit»

```

27 self.setStyleSheet("background-color: rgb(230,230,230);")
28 self.nom = QLabel("Nom")
29 self.lenom = QLineEdit("")
30
31 self.prenom= QLabel("Prenom")
32 self.leprenom = QLineEdit("")
33
34 self.age = QLabel("Âge")
35 self.leage = QLineEdit("")
36
37 self.sexe = QLabel("Sexe")
38 self.male = QRadioButton("Homme")
39 self.female = QRadioButton("Femme")
40
41
42 self.TEsymptome = QTextEdit()
43 self.TEsymptome.setStyleSheet("background-color: white ; color : black")
44 self.TEsymptome.setPlaceholderText("Entrez les symptômes")
45 self.TEmed = QTextEdit()
46 self.TEmed.setReadOnly(True)
47 self.TEmed.setPlaceholderText("Liste de médicaments pour ces symptômes : ")
48
49
50
51 self.btRetour = QPushButton("Fermer")
52 self.btRetour.setFont(QFont('Arial', 11))
53 self.btRetour.setToolTip("Accueil")
54 self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")
55
56
57 self.bHisto=QPushButton("Historique")
58 self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
59 self.bHisto.setToolTip("Historique du patient")
60 self.btRetour.setFont(QFont('Arial', 11))
61
62 self.bSave = QPushButton("Enregistrer")
63 self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
64 self.btRetour.setFont(QFont('Arial', 11))
65 self.bSave.setToolTip("Enregistrez votre dossier")
66
67 self.msg = QMessageBox()
68 self.msg.setIcon(QMessageBox.Critical)
69 self.msg.setWindowTitle("Erreur")
70 self.msg.setText("Le patient est déjà enregistré , allez voir son historique")
71
72
73 self.init_fenDossier()
74 self.fprec = fprec
75

```

Ici , les TextEdit sont des zones de textes editables , mais on en rend un des deux non éditables avec la commande « setReadOnly(True) ». L'éritable correspond aux symptômes du patient que l'on écrira tandis que le « TextEdit » non éditable sera la liste des médicaments proposé en fonction des symptômes entrés.

-Création des « QPushButton »

```
self.setStyleSheet("background-color: rgb(230,230,230);")
self.nom = QLabel("Nom")
self.lenom = QLineEdit("")

self.prenom= QLabel("Prenom")
self.leprenom = QLineEdit("")

self.age = QLabel("Âge")
self.leage = QLineEdit("")

self.sexe = QLabel("Sexe")
self.male = QRadioButton("Homme")
self.female = QRadioButton("Femme")

self.TEsymptome = QTextEdit()
self.TEsymptome.setStyleSheet("background-color: white ; color : black")
self.TEmed = QTextEdit()
self.TEmed.setReadOnly(True)

self.btRetour = QPushButton("Fermer")
self.btRetour.setFont(QFont('Arial', 11))
self.btRetour.setToolTip("Accueil")
self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

self.bHisto=QPushButton("Historique")
self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
self.bHisto.setToolTip("Historique du patient")
self.btRetour.setFont(QFont('Arial', 11))

self.bSave = QPushButton("Enregistrer")
self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
self.btRetour.setFont(QFont('Arial', 11))
self.bSave.setToolTip("Enregistrez votre dossier")

self.msg = QMessageBox()
self.msg.setIcon(QMessageBox.Critical)
self.msg.setText("Le patient est déjà enregistré , allez voir son historique")

self.init_fenDossier()
self.fprec = fprec
```

QFont permet de changer la police d'écriture du bouton

ToolTip permet d'obtenir une zone de texte lorsque l'on positionne sa souris sur le bouton ,sans pour autant cliquer dessus. Ici, on l'utilise pour donner une information supplémentaire à l'utilité du bouton.

En voici un exemple pour le bouton enregistrer.

Fais toi soigner : Fiche patient

Nom

Prenom

Âge

Sexe ☐ Homme ☐ Femme

Entrez les symptômes

-Création du « QMessageBox »

```
self.setStyleSheet("background-color: rgb(230,230,230);")
self.nom = QLabel("Nom")
self.lenom = QLineEdit("")

self.prenom= QLabel("Prenom")
self.leprenom = QLineEdit("")

self.age = QLabel("Age")
self.leage = QLineEdit("")

self.sexe = QLabel("Sexe")
self.male = QRadioButton("Homme")
self.female = QRadioButton("Femme")

self.TEsymptome = QTextEdit()
self.TEsymptome.setStyleSheet("background-color: white ; color : black")
self.TEmed = QTextEdit()
self.TEmed.setReadOnly(True)

self.btRetour = QPushButton("Fermer")
self.btRetour.setFont(QFont('Arial', 11))
self.btRetour.setToolTip("Accueil")
self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

self.bHisto=QPushButton("Historique")
self.bHisto.setStyleSheet("background-color: rgb(243,59,59) ; color :black ")
self.bHisto.setToolTip("Historique du patient")
self.btRetour.setFont(QFont('Arial', 11))

self.bSave = QPushButton("Enregistrer")
self.bSave.setStyleSheet("background-color: rgb(243,59,59) ; color : black ")
self.btRetour.setFont(QFont('Arial', 11))
self.bSave.setToolTip("Enregistrez votre dossier")

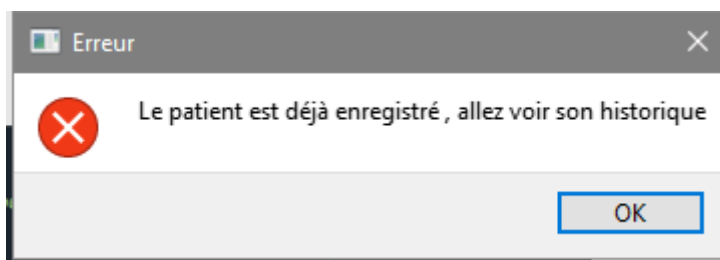
self.msg = QMessageBox()
self.msg.setIcon(QMessageBox.Critical)
self.msg.setWindowTitle("Erreur")
self.msg.setText("Le patient est déjà enregistré , allez voir son historique")

self.init_fenetre()
```

« QMessageBox » permet d'ouvrir une fenêtre pop-up lorsqu'on le souhaite.

Il nous sert ici lorsque le patient a déjà été enregistré dans la base de donnée et que l'on essaye encore de l'enregistrer

Capture d'écran de la fenêtre :



Initialisation de l'alignement des Widgets :

```
self.init_fenDossier()
self.fprec = fprec

def init_fenDossier(self):
    h_radio = QHBoxLayout()
    h_radio.addWidget(self.male)
    h_radio.addWidget(self.female)

    v_label = QVBoxLayout()
    v_label.addWidget(self.nom)
    v_label.addWidget(self.prenom)
    v_label.addWidget(self.age)
    v_label.addWidget(self.sexe)

    v_cases = QVBoxLayout()
    v_cases.addWidget(self.lenom)
    v_cases.addWidget(self.leprenom)
    v_cases.addWidget(self.leage)
    v_cases.addLayout(h_radio)

    h_textedit = QHBoxLayout()
    h_textedit.addWidget(self.TEsymptome)
    h_textedit.addWidget(self.TEmed)

    h_quatre = QHBoxLayout()
    h_quatre.addWidget(self.bSave)
    h_quatre.addWidget(self.btRetour)

    v_trois = QVBoxLayout()
    v_trois.addWidget(self.bHisto)
    v_trois.addWidget(self.TEmed)
    v_trois.addWidget(self.btRetour)

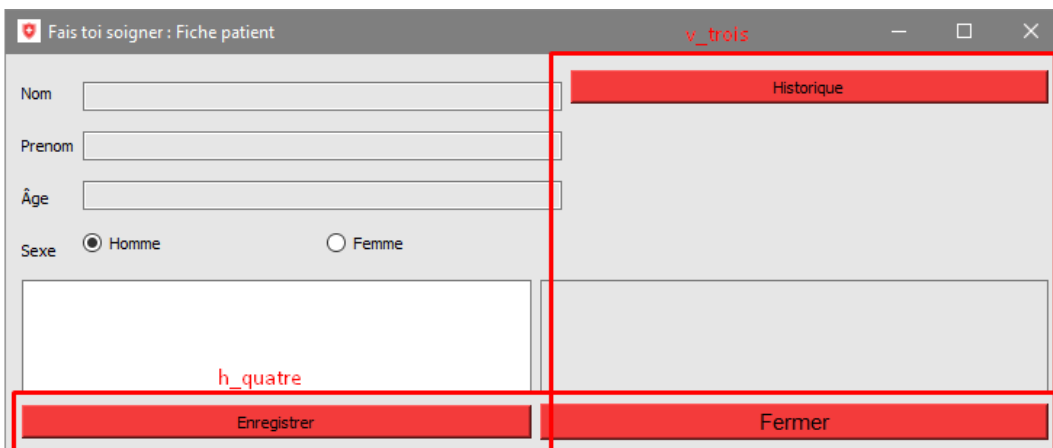
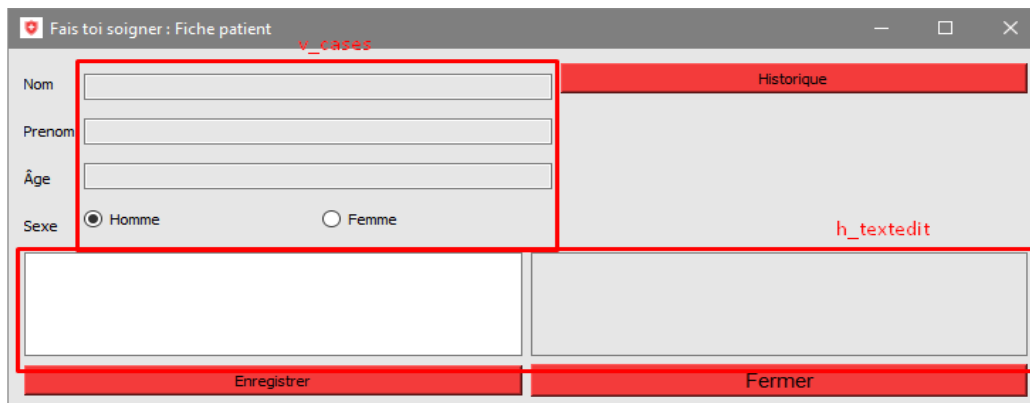
    h_infos = QHBoxLayout()
    h_infos.addLayout(v_label)
    h_infos.addLayout(v_cases)
    h_infos.addLayout(v_trois)

    v_box = QVBoxLayout()
    v_box.addLayout(h_infos)
    v_box.addLayout(h_textedit)
    v_box.addLayout(h_quatre)
    #v_box.addWidget(self.TEsymptome)
    self.setLayout(v_box)

    self.setGeometry(200,200,800,300)
    self.setWindowTitle('Fais toi soigner : Fiche patient')
    self.setWindowIcon(QIcon('im.png'))
```

Ici , chaque boite verticale et horizontale sont gérés pour pouvoir aligner les Widgets de la manière dont on le souhaite , on peut identifier différentes boites horizontales et verticales.

The screenshot shows a Qt application window titled "Fais toi soigner : Fiche patient". The window contains a form with several input fields and buttons. A red box highlights the "v_label" vertical layout containing the "Nom", "Prenom", "Âge", and "Sexe" labels and their corresponding input fields. Another red box highlights the "H_radio" horizontal layout containing the "Homme" and "Femme" radio buttons. The "Historique" button is also highlighted in red. The "Enregistrer" and "Fermer" buttons are at the bottom.



II/ Fonctionnement des boutons

Bouton «Fermer » :

Rôle : Le bouton fermer permet de retourner à la fenêtre d'accueil.

Fonctionnement :

```
self.btRetour.clicked.connect(self.btn1_click)
self.bSave.clicked.connect(self.btn2_click)
self.bHisto.clicked.connect(self.btn3_click)

def btn1_click(self):
    self.hide()
    self.fprec.show()
    def btn2_click(self):
```

Le bouton « Fermer » est lié à la méthode « btn1_click »

« fprec » est une variable qui devient une variable d'entrée de constructeur de la class « FenetreDossier »(class gérant la fenêtre du patient/dossier)

```
class FenetreDossier(QWidget):
    global var
    var = 0
    def __init__(self, fprec):
```

« fprec » est donc l'instanciation d'une ancienne fenêtre.

Bouton «Enregistrer» :

Rôle : Le bouton « Enregistrer » permet d'enregistrer les données du patient rentrés par l'utilisateur dans une base de données sur une machine distante.

Si la fiche du patient a déjà été créée , l'utilisateur sera invité à aller dans l'historique , et la fiche ne sera pas recréée.

Fonctionnement :

```
132 def btn2_click(self):
133     db = dbm.open('DBMed', 'c')
134
135     self.text_symp = self.TEsymptome.toPlainText()
136
137     if self.male.isChecked():
138         sexe = "Homme"
139     else :
140         sexe = "Femme"
141     cond = 0
142     for j in db.keys():
143         fichep = str(db[j].decode('UTF-8'))
144         #print('fichep : ' + fichep)
145         np = fichep.split('/')[0]
146         #print('np : ' + np)
147
148     infos = fichep.split('/')[1]
149     print("infos : " + infos)
150
151     if str(self.lenom.text().strip()) + ',' + str(self.leprenom.text().strip()) == np:
152         cond = 1
153         cle_patient = j
154     else :
155         cond = 0
156
157     if cond == 1 :
158         print("Le patient est déjà enregistré")
159         x = self.msg.exec()
160         db[cle_patient]=str(self.lenom.text().strip()) + ',' + str(self.leprenom.text().strip()) \
161             + '/' + str(self.leage.text().strip()) + ',' + str(sexe) + \
162             + str(self.text_symp)
163
164     else:
165         long = len(db.keys()) + 1
166         db[str(long)] = str(self.lenom.text().strip()) + ',' + str(self.leprenom.text().strip()) \
167             + '/' + str(self.leage.text().strip()) + ',' + str(sexe) + \
168             + str(self.text_symp)
169
170     db.close()
171     sftp.put('DBMed.dir', '/home/student/Medecin/DBMed.dir')
172     sftp.put('DBMed.dat', '/home/student/Medecin/DBMed.dat')
173     sftp.put('DBMed.dat', '/home/student/Medecin/DBMed.bak')
```

Au départ , on ouvre la base de données sans l'écraser et on récupère le texte du « TextEdit » des symptômes.

1/ Donne l'information sur le radio bouton sous forme de chaine de caractère.

On initialise la variable « cond » qui nous servira plus tard.

2/

La boucle crée parcourt la base de donnée en fonction de ses clés.

« fichep » est la variable qui stock les informations du patient , pour ne pas avoir de soucis de format et avoir une belle chaine de caractère , on utilise « decode('UTF-8')

« np » est la variable qui récupère seulement le nom et le prénom de patient , elle servira pour savoir si le patient est déjà enregistré.

```
nom    prenom  age  sexe  symptomes
fichep : testnom, testprenom, 65, Homme, test
np : testnom, testprenom
Le patient est déjà enregistré
```

3/ On récupère ici les infos qui sont après la délimitation du nom et du prénom.

On utilise un « / » , ainsi que la fonction split() pour ce faire.

```
np : test, test
fichep : testnom, testprenom/65, Homme, test
np : testnom, testprenom
```

4/ On pose une condition pour savoir si le nom et le prénom sont déjà enregistrés dans la base de données .

Ici , la fonction strip sert à récupérer clairement le text des « TextEdit » du nom et du prénom entré par l'utilisateur. La virgule en plus permet de reproduire exactement la même chaine de caractère s'il y a bien correspondance entre la variable « np » et cette dernière.

Si c'est le cas, on passe la variable « cond » à 1 et on identifie la cle du patient dans la base de donnée par la variable d'itération de la boucle (en l'occurrence ici c'est « j »).

5/ La condition ou cond==1 est lorsque le patient est déjà enregistré dans la base de donnée,

Dans ce cas là, un message d'erreur est affiché en associant la valeur de clé trouvée précédemment par toutes les informations déjà récupéré auparavant.

Sinon , une nouvelle clé est crée pour le nouveau patient et toutes les informations écrites par l'utilisateur lui sont associées.

5/ On ferme ensuite la base de donnée et on ajoute les fichiers de la base de donnée dans une machine distante.

Bouton «Historique» :

Rôle : Le bouton historique permet d'obtenir l'historique de la fiche du patient, utile pour savoir ses antécédents.

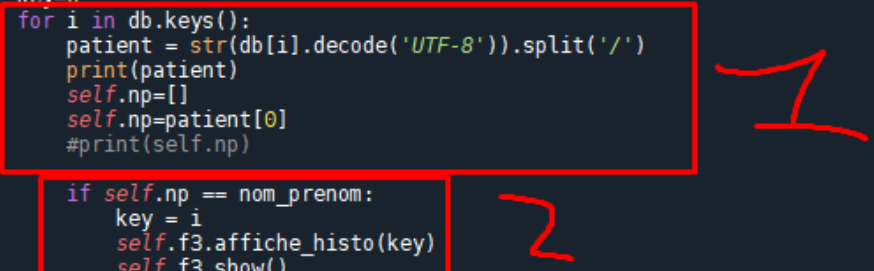
Fonctionnement :

Le bouton poussoir « Historique » est lié à la méthode « btn3_click ».

Tout d'abord , on ouvre la base de donnée , on récupère le nom et le prénom depuis les zones de texte éditables.

1/ On parcourt la base de donnée en obtenant toutes les informations de chaque patient dans la variable « patient »

```
def btn3_click(self):  
    db = dbm.open('DBMed' , 'c')  
    nom_prenom = self.lenom.text().strip() + ',' + self.leprenom.text().strip()  
    key=0  
    for i in db.keys():  
        patient = str(db[i].decode('UTF-8')).split('/')  
        print(patient)  
        self.np=[]  
        self.np=patient[0]  
        #print(self.np)  
        if self.np == nom_prenom:  
            key = i  
            self.f3.affiche_histo(key)  
            self.f3.show()
```



The diagram illustrates the data flow in the btn3_click method. A red box highlights the loop where patient data is retrieved and stored in the self.np attribute. A red arrow points from this box to another red box below it, which contains the conditional logic that checks if self.np matches the entered nom_prenom. A red '1' is next to the first box, and a red '2' is next to the second box, indicating the sequence of operations.

L'attribut « np » récupère le nom et le prénom du patient

2/Si la personne a déjà été enregistré précédemment , alors on peut ouvrir l'historique.

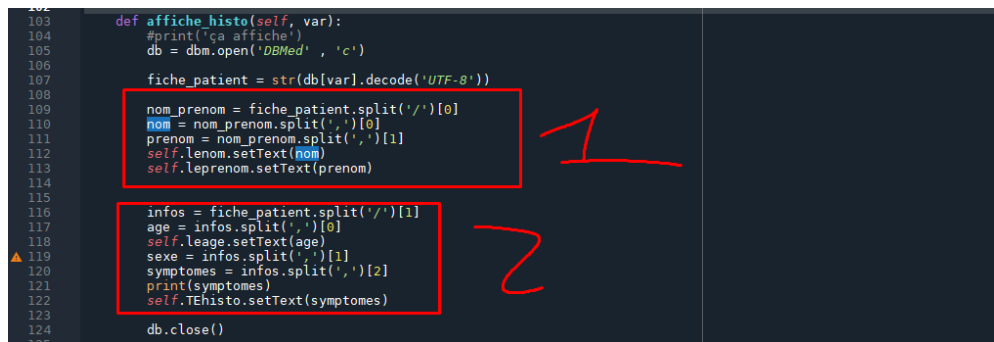
La variable key est la valeur de clé où le nom et le prénom correspondent à celle dans la base de donnée.

Pour ce faire , une nouvelle instantiation a du se faire car un autre programme a été crée spécialement pour cette fenêtre , ainsi que l'importation de la classe de la nouvelle fenêtre

```
IHM_Medecin.py x fenetre2.py x fenetre3.py x  
1  # -*- coding: utf-8 -*-  
2  """  
3  Created on Tue Apr 26 16:42:57 2022  
4  
5  @author: leleu  
6  """  
7  
8  from PyQt5.QtWidgets import (QLineEdit, QPushButton, QVBoxLayout, Q  
9                                     QApplication, QMessageBox, QMainWindow,  
10 from PyQt5.QtGui import QPixmap , QFont , QIcon  
11  
12 from PyQt5 import QtCore  
13 import dbm  
14 import paramiko  
15 import sys  
16  
17 from fenetre3 import FenetreHisto  
18  
19
```

Nous nous retrouvons donc désormais dans le programme de la fenêtre d'historique , qui a donc un lien avec notre bouton « Historique » tout en étant dans un autre programme , car on utilise une méthode de la classe « FenetreHisto »

```
103 def affiche_histo(self, var):
104     #print('ça affiche')
105     db = dbm.open('DBMed', 'c')
106
107     fiche_patient = str(db[var].decode('UTF-8'))
108
109     nom_prenom = fiche_patient.split('/')[0]
110     nom = nom_prenom.split(',')[0]
111     prenom = nom_prenom.split(',')[1]
112     self.lenom.setText(nom)
113     self.leprenom.setText(prenom)
114
115
116     infos = fiche_patient.split('/')[1]
117     age = infos.split(',')[0]
118     self.leage.setText(age)
119     sexe = infos.split(',')[1]
120     symptomes = infos.split(',')[2]
121     print(symptomes)
122     self.Tehisto.setText(symptomes)
123
124     db.close()
```



La variable « key » qui appartient à la méthode du bouton historique, devient « var » dans la méthode de la classe gérant la fenêtre historique.

On commence par ouvrir la base de donnée ainsi qu'à récupérer toutes les infos du patient via la variable « fiche_patient »

1/ On récupère les informations du nom et prénom qu'on place dans leur zone de texte éditable respectif

2/ On récupère le reste des infos du patient qu'on place dans leur zone de texte respectif.

Durant toutes ces extractions d'informations , on a utilisé des « , » pour faciliter l'extraction.

III/ Import de modules et commandes de connexion à distance nécessaires :

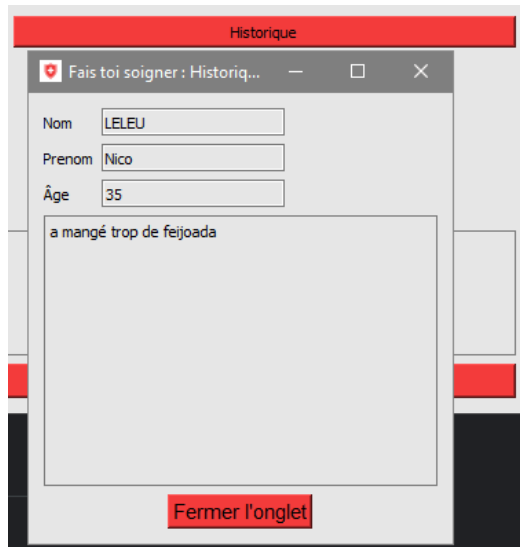
```
from PyQt5.QtWidgets import (QLineEdit, QPushButton, QVBoxLayout, QHBoxLayout,
                             QApplication, QMessageBox, QMainWindow, QWidget, QLabel, QRadioButton, QGridLayout, QGroupBox, QTextEdit, QToolTip)
from PyQt5.QtGui import QPixmap, QFont, QIcon

from PyQt5 import QtCore
import dbm
import paramiko
import sys
```

```
207
208
209 s = paramiko.SSHClient()
210 s.set_missing_host_key_policy(paramiko.AutoAddPolicy())
211 s.connect('192.168.8.14',22,username="student",password='vitygtr',timeout=4)
212 sftp = s.open_sftp()
213
```

3^{ème} fenêtre : L'historique du patient

Capture d'écran de la fenêtre d'historique :



1/Partie interface visuelle du code :

```
class FenetreHisto(QWidget):
    def __init__(self, fdoss):
        super().__init__()
        db = dbm.open('DBMed', 'c')
        self.setStyleSheet("background-color: rgb(230,230,230);")
        self.nom = QLabel("Nom")
        self.lenom = QLineEdit("")
        self.lenom.setText("test")
        self.lenom.setReadOnly(True)

        self.prenom = QLabel("Prenom")
        self.leprenom = QLineEdit("")
        self.leprenom.setReadOnly(True)

        self.age = QLabel("Âge")
        self.leage = QLineEdit("")
        self.leage.setReadOnly(True)

        self.TEhisto = QTextEdit()
        self.TEhisto.setReadOnly(True)

        self.btRetour = QPushButton("Fermer l'onglet")
        self.btRetour.setMaximumHeight(100)
        self.btRetour.setMaximumWidth(300)
        self.btRetour.setFont(QFont('Arial', 11))
        self.btRetour.setToolTip("Retourner au dossier")
        self.btRetour.setStyleSheet("background-color: rgb(243,59,59) ; color : black")

        self.init_fenHisto()
        self.fdoss = fdoss
```

```

def init_fenHisto(self):

    v_label = QVBoxLayout()
    v_label.addWidget(self.nom)
    v_label.addWidget(self.prenom)
    v_label.addWidget(self.age)

    v_cases = QVBoxLayout()
    v_cases.addWidget(self.lenom)
    v_cases.addWidget(self.leprenom)
    v_cases.addWidget(self.leage)

    h_quatre=QHBoxLayout()
    h_quatre.addWidget(self.btRetour)
    h_quatre.setAlignment(Qt.AlignCenter)

    v_trois = QVBoxLayout()
    v_trois.addWidget(self.btRetour)

    h_textedit = QHBoxLayout()
    h_textedit.addWidget(self.TEhisto)

    h_infos = QHBoxLayout()
    h_infos.addLayout(v_label)
    h_infos.addLayout(v_cases)
    h_infos.addLayout(v_trois)

    v_box = QVBoxLayout()
    v_box.addLayout(h_infos)
    v_box.addLayout(h_textedit)
    v_box.addLayout(h_quatre)

    self.setLayout(v_box)
    self.setWindowTitle('Fais toi soigner : Historique ')
    self.setWindowIcon(QIcon('im.png'))

```

(Tout les Widget et les alignements ont déjà pu être expliqués plus tôt)

II/ Fonctionnement du bouton fermer

```

def __init__(self, fdoss ):
    super().__init__()

    self.fdoss = fdoss

    self.btRetour.clicked.connect(self.btn1_click)

```

L'instanciation de « fdoss » fait appel à la fenêtre gérant le dossier du patient

```

127
128     def btn1_click(self):
129         self.hide()
130         self.fdoss.show()

```

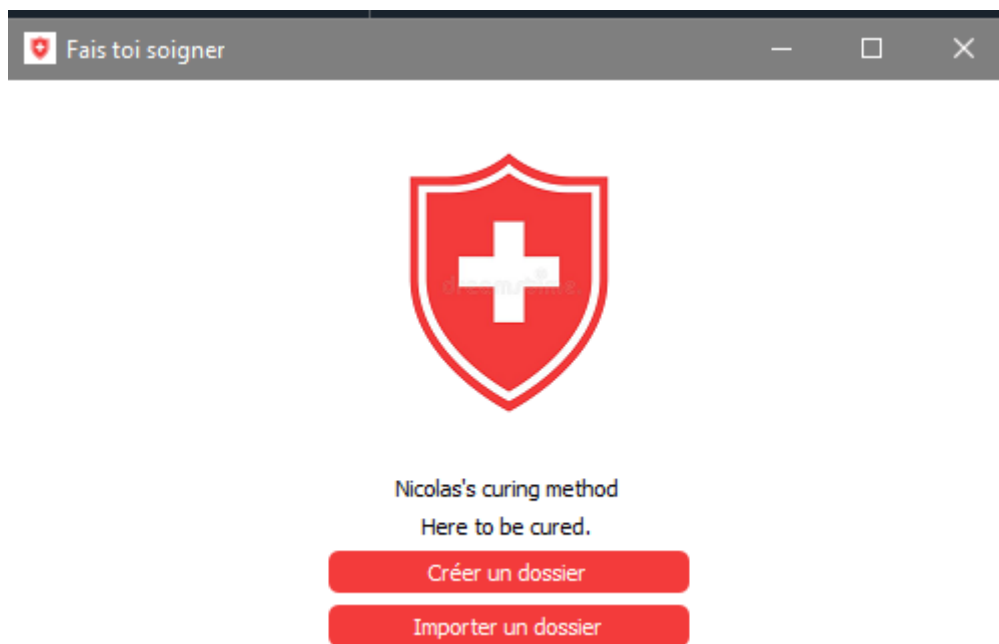
Lorsque le bouton 1 est pressé , la fenêtre se cache et on appelle (montre) la fenêtre de dossier du patient.

Scénario du programme :

Prérequis :

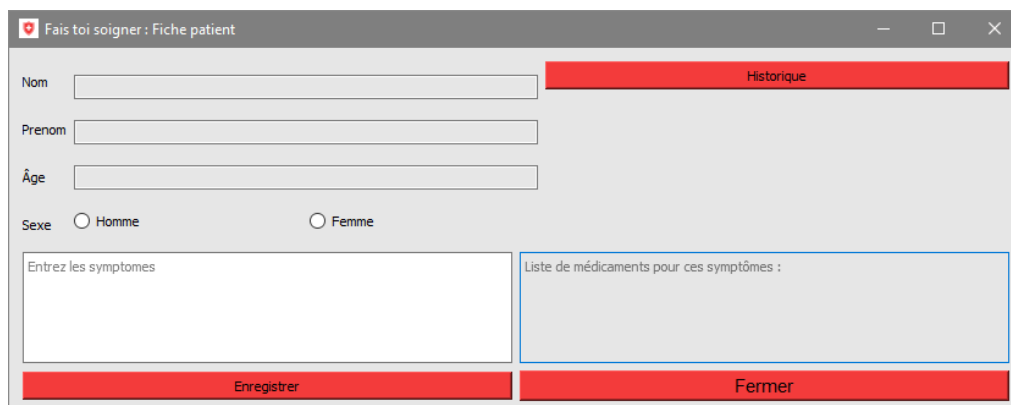
- Avoir créer lancer la base de donnée au moins une fois auparavant , sinon modifier un « c » lors de l'ouverture d'une base de donnée par un « n »
- Avoir sa machine distante allumée.
- Avoir l'image « im.png » dans le même répertoire que les programmes.

1/Lancement du programme :



On clique ensuite sur « Créer un dossier »

2/ On arrive sur la fenêtre de création de dossier



On remplit les informations du patient et on appuie sur le bouton enregistrer

Fais toi soigner : Fiche patient

Nom: LELEU

Prenom: Nicolas

Âge: 21

Sexe: ☒ Homme ☐ Femme

mal de ventre

Historique

Liste de médicaments pour ces symptômes :

Enregistrer Fermer

2 cas possibles désormais :

1-le patient n'est pas reconnu dans la base de donnée : aucun soucis , il est bien enregistré

2-Le patient est reconnu dans la base de donnée : un message d'erreur est affiché

Fais toi soigner : Fiche patient

Nom: LELEU

Prenom: Nicolas

Âge: 21

Sexe: ☒ Homme ☐ Femme

mal de ventre

Historique

Liste de médicaments pour ces symptômes :

Enregistrer Fermer

Erreur

Le patient est déjà enregistré , allez voir son historique

OK

On clique sur le bouton « Historique » :

Fais toi soigner : Fiche patient

Nom: LELEU

Prenom: Nicolas

Âge: 21

Sexe: ☒ Homme ☐ Femme

mal de ventre

Historique

Liste de médicaments pour ces symptômes :

Enregistrer Fermer

Fais toi soigner : Historique...

Nom: LELEU

Prenom: Nicolas

Âge: 21

mal de ventre

Fermer l'onglet

Les informations du patient sont bien retransmises dans l'historique de ce dernier.

Perspectives du programme :

J'ai pris beaucoup de temps à réaliser le fonctionnement du bouton d'enregistrement et d'historique.

Je n'ai pas eu le temps de réaliser la recherche en temps réel des médicaments disponibles , cependant j'ai réfléchi à comment la réaliser , ce qui me semble important.

Avec la fonction « TextChanged » de PyQt5 , on peut réaliser des actions dès que le texte d'un Widget est modifié. Lorsque le texte du « TextEdit » symptômes est modifié , nous allons faire une recherche dans une base de données dbm. Cette base de données des clés pour chaque différents médicaments. Chaque clé sera associée à différents mots comme « mal de crâne » pour un doliprane.

Après récupération de cette clé , on pourra facilement afficher le mot « doliprane » si « mal de crâne » est écrit dans le « TextEdit » prévu pour.

J'aurais bien aimé pouvoir réaliser la fenêtre pour importer un dossier , dans laquelle la création aurait été maître.

J'imagine qu'on aurait pu utiliser une fonctionnalité où nous pouvons saisir un répertoire de notre ordinateur , récupérer les informations du fichier souhaiter , et pouvoir afficher directement la fenêtre de la fiche du patient.

Lien GitHub

