



RAPPORT PROJET WS

BINOME:

AISSI MED SALIM* *NOUAR SOFIA



TABLE DES MATIERES :

INTRODUCTION.....

PARTIE 1 : Ontologie

1.Définition des classes

2.Définition des relations

PARTIE 2 : Interface

Introduction

1.Cas d'utilisation

2.Requêtes sparql

PARTIE 3 : Annexe

1.Annexes

2.Graphe

Introduction :

Dans le cadre du projet de web sémantique qui s'intitule « l'open data sur les données d'évolution de la maladie du covid 19 », nous avons mis en place une application « **iKOVID** » développé en python qui se compose essentiellement de deux grands titres : ontologie et interface.



Figure 0.0

PARTIE 1 : Ontologie

Définition des classes :

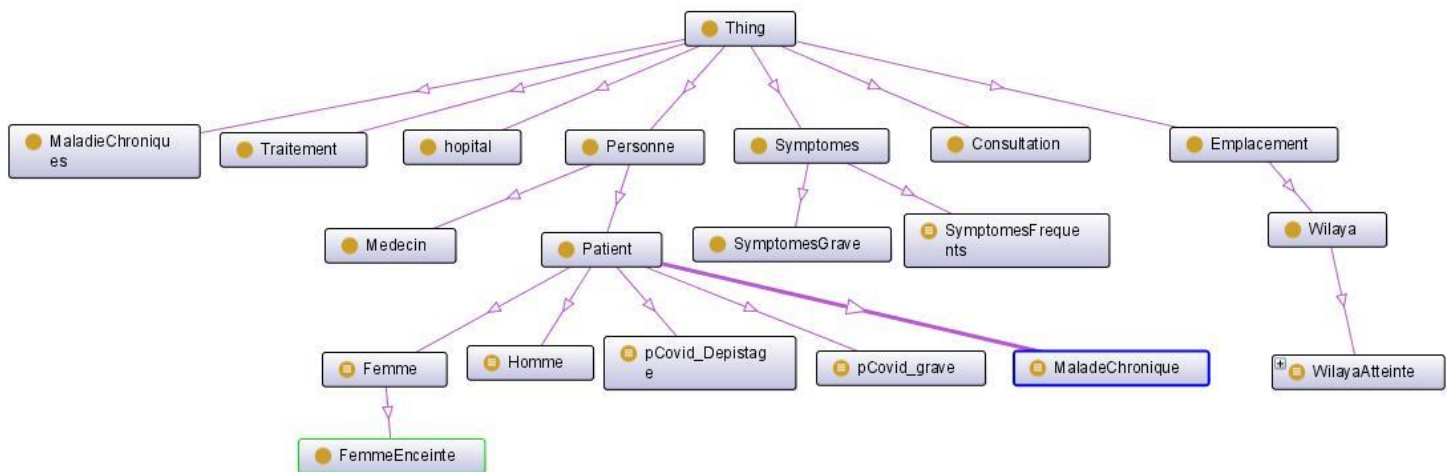


Figure I.1

Notre ontologie se compose de 7 grandes classes qui héritent de Thing :

- 1- **MaladieChroniques** : contient les individus maladies présent dans notre ontologie.
- 2- **Traitement** : contient les individus traitement présent dans notre ontologie.
- 3- **Hôpital** : contient des individus hôpitaux crée à partir du fichier H.xlsx (voir annexe 1).
- 4- **Personne** : se divise en deux sous-classes :
 - 4.1. **Medecin** : contient les individus médecin crée à partir du fichier medecin.xlsx (voir annexe 2).
 - 4.2. **Patient** : contient les individus qui utilisent l'application et qui sont enregistrées dans le fichier de donnée test.xlsx (voir annexe 3).

4.2.1. Homme : individus homme (selon le sexe saisi dans le fichier test).

4.2.2. Femme : individus femme.

4.2.2.1. FemmeEnceinte : individus qui ont 'oui' dans la case
« Femme enceinte » du fichier test.xlsx.

4.2.3. MaladeChronique : les individus patients qui ont au moins une maladie de type MaladieChroniques, selon l'équivalence :

$\text{MaladeChronique} \sqsubseteq \text{Patient} \sqcap \exists a_pour_maladie. \text{MaladieChronique}$

4.2.4. pCovid_grave : classer par le reasoner, cette classe contient les patients qui sont , selon nos recherches , atteint du sars-cov-2, selon l'une des équivalences suivantes :

- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap (\geq 2 a_Symptomes.SymptomesFrequents) \sqcap (\geq 57 \text{ Age})$
- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap (\geq 3 a_Symptomes.SymptomesFrequents) \sqcap \text{MaladeChronique}$
- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap \exists a_Symptomes.SymptomesGraves)$
- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap (\geq 4 a_Symptomes.SymptomesFrequents) \sqcap (" +8\text{jrs}" \text{jrsSympt})$
- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap (\geq 4 a_Symptomes.SymptomesFrequents) \sqcap ("oui" \text{obesite})$
- $\text{pCovid_grave} \sqsubseteq \text{Patient} \sqcap (\geq 4 a_Symptomes.SymptomesFrequents) \sqcap \text{FemmeEnceinte}$

4.2.5. pCovid_Depistage : classer par le reasoner, cette classe contient les patients qui sont susceptible d'avoir le sars-cov-2, et a qui on conseille de faire un dépistage (PCR) , selon cet équivalence :

- $\text{pCovid_Depistage} \sqsubseteq \text{Patient} \sqcap (\geq 3 a_Symptomes.symptomesFrequent) \sqcap (" +8\text{jrs} \mid \mid 4-8 \text{jrs}" \text{jrsSympt})$

5- Symptomes :

La classe symptôme se divise en deux sous-classes :

5.1. SymptomesGrave : cette classe contient les individus qui sont, selon nos recherches, des symptômes Grave, ces individus sont :

- difficultés_à_respirer_ou_essoufflement
- sensation_d_oppresion_ou_douleur_au_niveau_de_la_poitrine

5.2. SymptomesFrequents : classer par le reasoner, cette classe contient les symptômes qui sont présents chez 40% de nos patients (nombre de personne dans le fichier test.xlsx)

6- Consultation : les individus de cette classe serrent de jointure entre les individus de la classe médecins et patients pour avoir une consultation a une date donnée.

7- Emplacement : contient la sous classe :

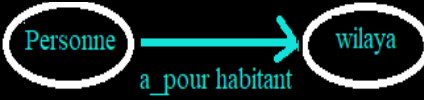
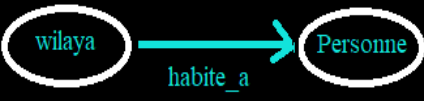
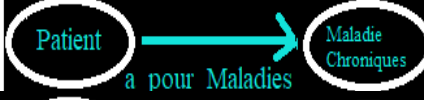

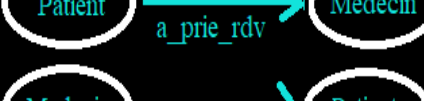
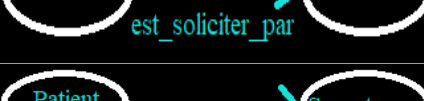
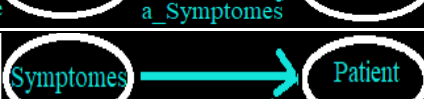

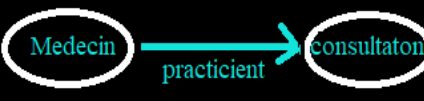
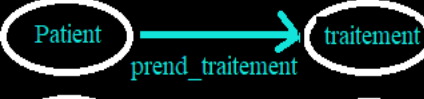
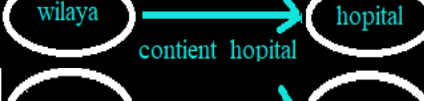
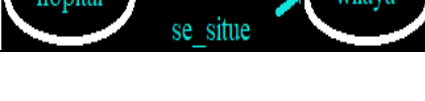

7.1. wilaya : représente les individus qui sont les 58 wilaya de notre pays ,et contient la sous classe :

7.1.1 WilayaAtteinte : contient les individus qui sont considérer par le reasoner comme de wilaya possédant au moins une personne atteinte du sars-cov-2, selon l'équivalence :

- $WilayaAtteinte \sqsubseteq Wilaya \sqcap \exists a_pour_habitant.pCovid_Grave$

Définition des relations :

Le tableau suivant regroupe les différentes relations existantes entre nos classe.

Relation	domaine	Range	Inverse	figure
A_pour_habitant	wilaya	Personne	Habite_a	
Habite_a	Personne	Wilaya		
A_pour_maladie	Patient	Maladie Chronique	Affecte	
Affecte	Maladie Chronique	Personne		
A_pris_rdv	Patient	Medecin	Est_soliciter_par	
Est_soliciter_par	Medecin	Patient		
A_symptome	Patient	Symptome	Personne_atteinte	
Personne_atteinte	symptome	Patient		
Consultant	Patient	Consultation		
Praticien	Medecin	Consultation		
Prend_traitement	Patient	Traitement		
Contient_hopital	Wilaya	Hopital	Se situe	
Se situe	Hopital	Wilaya		

PARTIE 2 : Interface

Introduction :

Après avoir définis les classes de base et les relations nécessaires de notre projet, nous introduisons dans cette partie les différentes interactions entre cette dite interface (voir figure II.0), des fichiers Excel et l'ontologie.

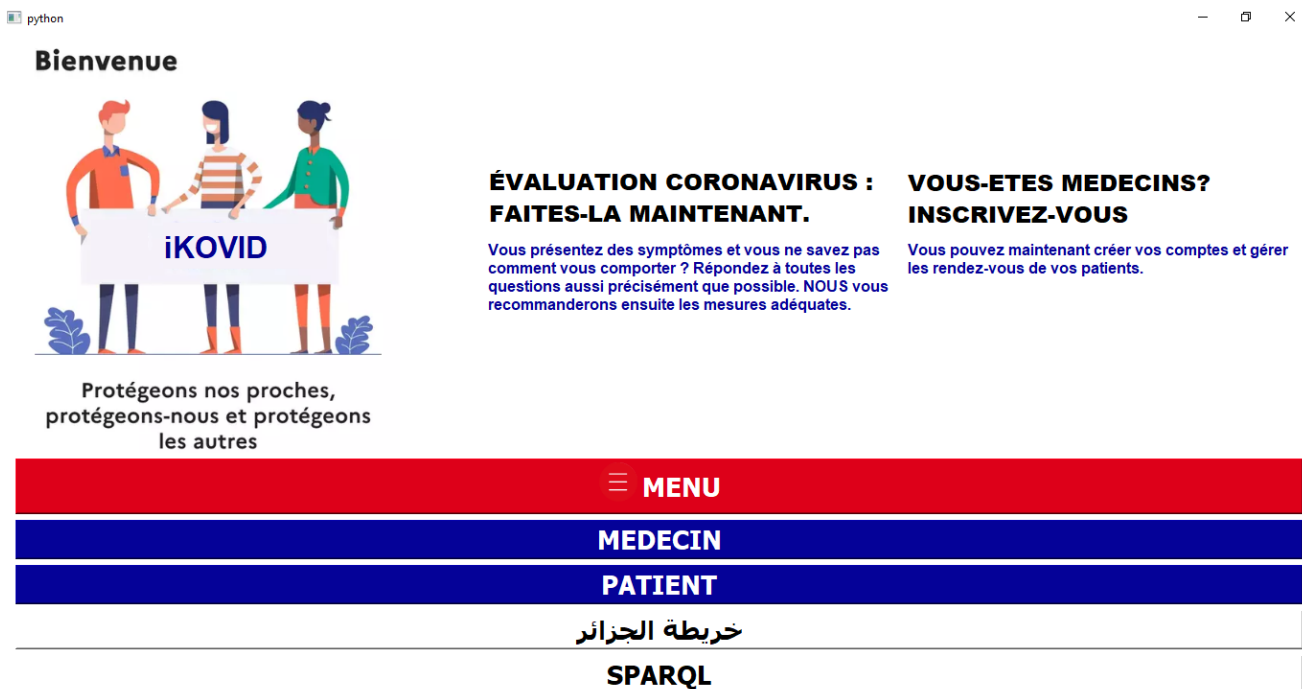


Figure II.0

1. Cas d'utilisation :

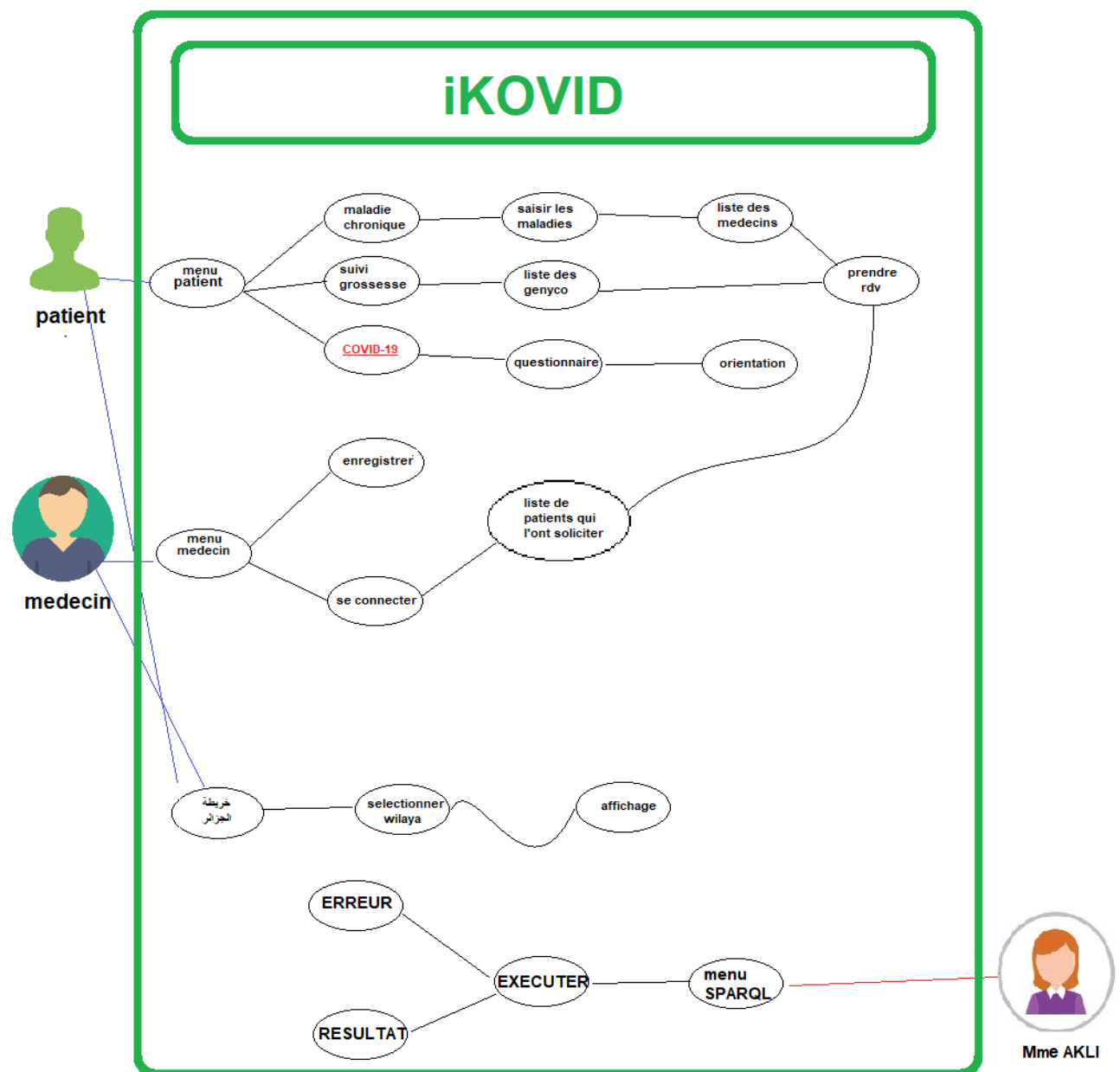


Figure II.1

La figure II.1 représente le diagramme de cas d'utilisation de notre programme, on peut distinguer 3 acteurs principaux :

1.1. Patient :

Avant que le patient commence à remplir le formulaire, une fenêtre informative s'affiche (voir figure II.2) lui indiquant les gestes à adopter en

cette période de confinement ainsi que les consignes d'utilisation.



Figure II.2

Ensuite, le patient clique sur commencer pour se diriger vers une page qui contient un formulaire pour remplir ses informations personnelles et choisir une des 3 options.

Figure II.3

1.1.1. Maladies chroniques : un menu contenant les différents types de maladies va s'afficher. Le patient devra choisir les types de ses maladies et saisir manuellement leurs noms ainsi que les traitements s'ils existent. Pour faire le lien avec l'ontologie, les maladies (resp. Traitement) saisies seront classées comme des individus de type « Maladies »

(resp. « Traitement ») et le patient sera classé suivant le reasoner comme « MaladeChronique ».

Après exécution d'une requête SPARQL

```
SELECT DISTINCT ?nom ?adr ?num ?dm ?x ?y WHERE{
  ?x a ns2:Medecin ."
  ?y ns2:Nom " + "'" + nom+ "'^xsd:string
  ?y ns2:Prenom " + "'" + prenom+ "'^xsd:string .
  ?x ns2:habite_a ?z."
  ?y a ns2:MaladeChronique.
  ?y ns2:habite_a ?z.
  ?y ns2:a_Pour_Maladies ?t.?t ns2:typemaladie ?v.?x ns2:specialité ?v.
  ?x ns2:domaine ?dm."
  ?x ns2:NomMed ?nom.?x ns2:adressesmed ?adr.?x ns2:numtelmed ?num}
ORDER BY DESC(?dm)
```

(Cette requête sparql cherche l'individu patient courant "celui qui vient de se s'enregistrer" dans l'ontologie et sélectionne selon la wilaya du patient les médecins spécialistes de sa/ses maladie(s).)

, une fenêtre s'affiche avec les noms des médecins spécialistes qui s'occupe des maladies que le patient a cité, avec possibilités de prise de rendez-vous.

Si le patient clique sur le bouton « prendre rendez-vous » alors 2 relations se crée : la première « a_pris_rdv » et le reasoner crée en parallèle la relation inverse (voir figure II.4).

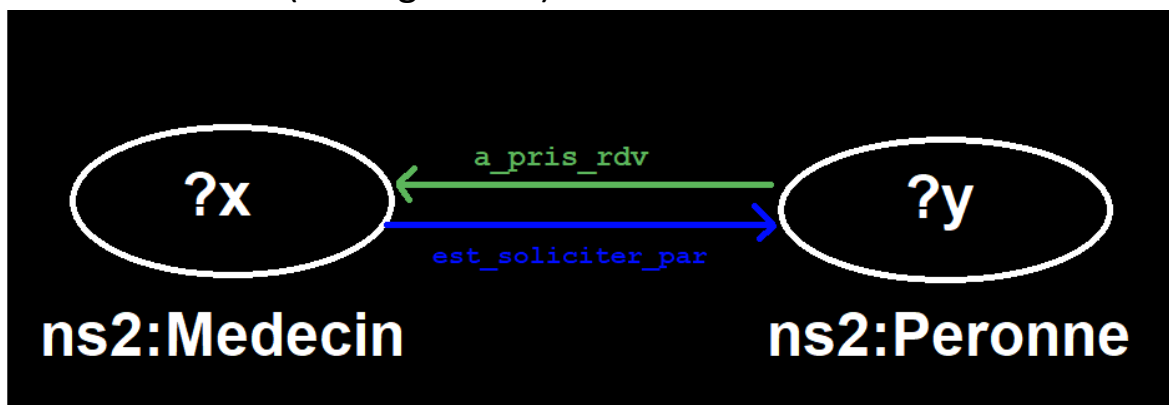


Figure II.4

1.1.2. Suivi grossesse :

Qui s'active une fois le sexe « Femme » sélectionner. Cette personne fera partis donc des individus de la classe « FemmeEnceinte » Et suggère une liste des médecins gynécologues obtenus grâce à la requête SPARQL suivante :

```
SELECT DISTINCT ?nom ?adr ?num ?x ?y WHERE{
  ?x a ns2:Medecin .
  ?y ns2:Nom " + "'" + str(self.liste[0].nom.text()) + "'^xsd:string.
  ?y ns2:Prenom " + "'" + prenom + "'^xsd:string .
  ?x ns2:habite_a ?z.
  ?y a ns2:FemmeEnceinte.
  ?y ns2:habite_a ?z.
  ?x ns2:domaine "Gyneco Obstetrique"^xsd:string.'
  ?x ns2:NomMed ?nom.?x ns2:adressesmed ?adr.?x ns2:numtelmed ?num}
```

Une même relation sera créée comme dans la figure II.4.

1.1.3. Test covid-19 :

Le patient répondra successivement à ces questions :

- 'Avez-vous une forte fièvre ?'
- 'Ces derniers jours, avez-vous une toux ou une augmentation de votre toux habituelle ?'
- 'Ces derniers jours, avez-vous noté une forte diminution ou perte de votre goût ou de votre odorat ?'
- 'Ces derniers jours, avez-vous eu un mal de gorge ?'
- 'Ces derniers jours, avez-vous des douleurs musculaires et/ou des courbatures inhabituelles ?'
- 'Avez-vous de la diarrhée ? '
- 'Ces derniers jours, avez-vous une fatigue inhabituelle ?'
- 'Avez-vous eu un mal de tête ?'
- 'Avez-vous noté un manque de souffle inhabituel lorsque vous parlez ou faites un petit effort ?'
- 'Avez-vous des douleurs au niveau de la poitrine ?'
- 'Etes-vous enceinte ? ' (si le sexe « Femme » est sélectionner, le patient sera considéré aussi comme « FemmeEnceinte ».)

- "Avez-vous une obésité de classe III (morbide, IMC ≥ 40 kg/m²) ?"

En plus d'indiquer les maladies chroniques qu'il a.

Selon les réponses obtenues, le reasoner classera cette personne comme étant une « pCovid_grave », « pCovid_Depistage » ou personne normale, et un message (orientation) sera affiché avec des indications à prendre en compte et ce, après l'exécution de la requête SPARQL suivante :

```
SELECT DISTINCT ?diag ?y WHERE{
```

```
?x a ?diag .
```

```
?x ns2:habite_a ?y.
```

```
?x ns2:Nom " + "'" + str(self.liste[0].nom.text()) + "'^xsd:string .
```

```
?x ns2:Prenom " + "'" + str(self.liste[0].prenom.text()) + "'^xsd:string}
```

Dans la liste résultante, qui sont les différents types de notre individu, on vérifie l'existence du type « pCovid_grave », sinon on cherche le type « pCovid_Depistage ».

Si aucun des 2 types n'est présent, alors l'individu sera considéré comme une personne normale.

1.2. Médecin :

Si le médecin s'est déjà enregistré, il pourra accéder à son compte en donnant son nom et son mot de passe qu'il a créé pendant l'enregistrement (voir figure II.5).

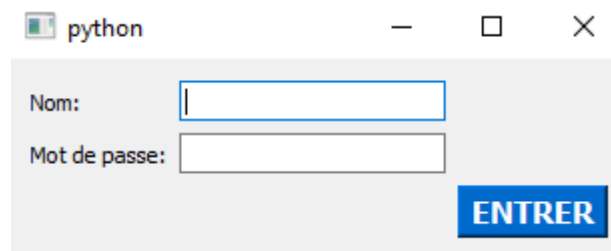


Figure II.5

La vérification des identifiants est faite grâce à la requête SPARQL suivante :

```
SELECT DISTINCT ?name ?prenom ?tel ?age ?sexe ?nmed ?adrmed  
?telmed ?y ?x WHERE{
```

```
?x a ns2:Medecin.
```

```
?x ns2:NomMed " + "'" + str(self.nametexte.text()) + "'^xsd:string.
```

```
?x ns2:mdp " + "'" + str(self.passwordtexte.text()) + "'^xsd:string.
```

?x ns2:NomMed ?nmed. ?x ns2:adressedmed ?adrmed. ?x ns2:numtelmed ?telmed.

OPTIONAL {?y ns2:a_pris_rdv ?x.

?y ns2:Nom ?name.

?y ns2:Prenom ?prenom.

?y ns2:NumTel ?tel.

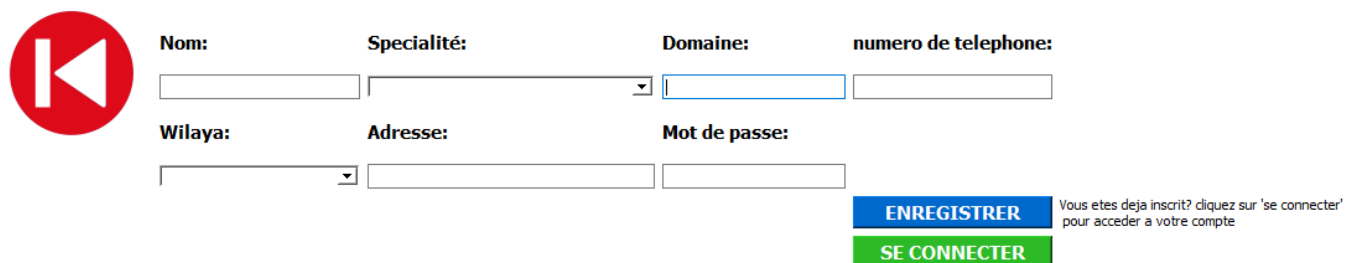
?y ns2:Age ?age.

?y ns2:Sexe ?sexe.}

OPTIONAL{?y ns2:a_Pour_Maladies ?maladies}}

Sinon, un message d'erreur sera affiché.

Pour créer son compte la première fois, un médecin doit remplir les champs suivants :



The registration form is located on a white background with a red header bar. On the left, there is a red circular icon containing a white left-pointing arrow. The form consists of two rows of input fields. The first row has four fields: 'Nom:' (text input), 'Specialité:' (dropdown menu), 'Domaine:' (text input), and 'numero de telephone:' (text input). The second row has three fields: 'Wilaya:' (dropdown menu), 'Adresse:' (text input), and 'Mot de passe:' (text input). To the right of the 'Mot de passe:' field, there are two buttons: a blue 'ENREGISTRER' button and a green 'SE CONNECTER' button. To the right of the 'SE CONNECTER' button, there is a small text link: 'Vous etes deja inscrit? cliquez sur 'se connecter' pour acceder a votre compte'.

Figure II.6

Si la connexion réussie, le compte du médecin s'affiche contenant ses informations ainsi que les patients l'ayant sollicité. Il pourra ainsi valider les rendez-vous en donnant une date précise. De ce fait, un individu consultation sera créé, ceci est expliqué par la figure II.7 suivantes :

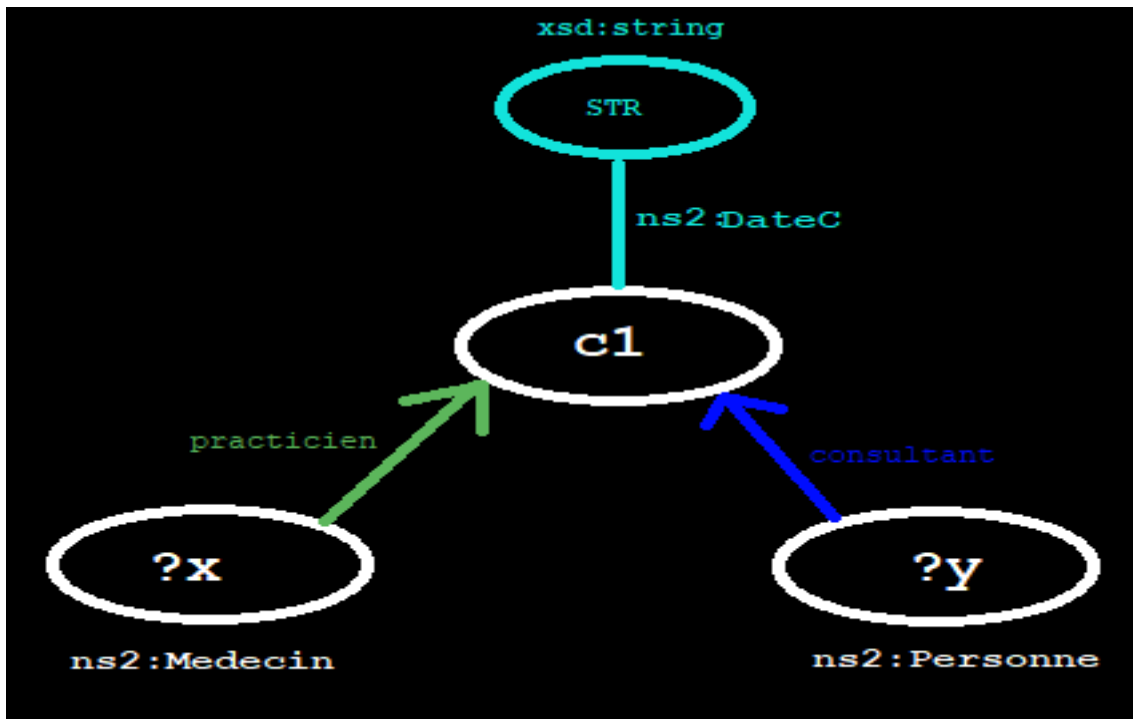


Figure II.7

1.3. Administrateur (Mme AKLI) :

Le menu sparql se présente comme dans le figure II.8.

Et se compose de 3 grandes parties : 1# pour l'écriture de la requête, 2# outil d'aide qui contient les relations ainsi que les classes de l'ontologie et enfin 3# pour l'affichage des résultats ou des erreurs.

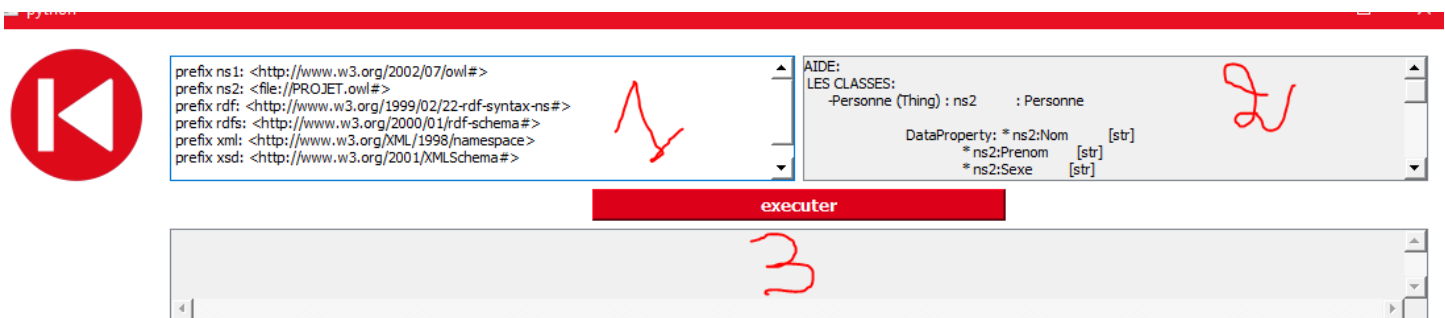


Figure II.8

1.4. خريطة الجزائر :

Cette page affiche la carte géographique de notre pays avec un bouton cliquable sur chaque wilaya (voir la figure II.9).

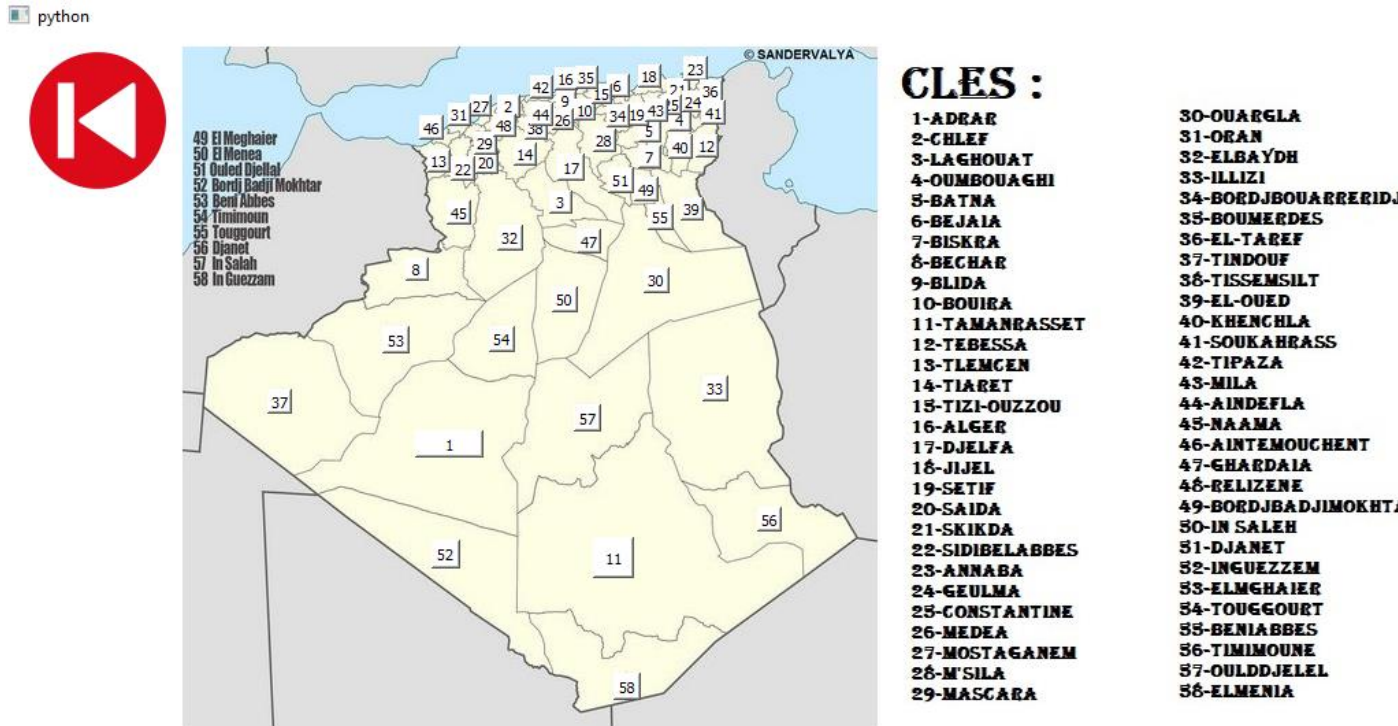


Figure II.9

2.Requêtes sparql :

Chaque bouton affiche une mini fenêtre contenant le nombre d'habitant présent dans notre ontologie, le nombre des personnes atteintes du covid, le nombre de médecins et d'hôpitaux, et ce en exécutant les 4 requêtes SPARQL suivantes :

POUR LE NOMBRE DE PERSONNE ATTEINTE DU COVID DE CETTE WILAYA :

```
SELECT (COUNT( ?x) AS ?nbpatteints) WHERE{
OPTIONAL{?x a ns2:pCovid_grave.
?x ns2:habite_a <file://PROJET.owl#WILAYASELECTIONNER>}}
```

POUR LE NOMBRE D'HABITANTS DE CETTE WILAYA DANS NOTRE ONTOLOGIE :


```
SELECT (COUNT( ?z) AS ?nbhabitants) WHERE{
OPTIONAL{<file://PROJET.owl#WILAYASELECTIONNER>
ns2:a_pour_habitant ?z.}}
```

POUR LE NOMBRE D'HOPITAUX :

```
SELECT (COUNT( ?w) AS ?nbhop) \n WHERE{
OPTIONAL{ <file://PROJET.owl#WILAYASELECTIONNER>
ns2:contient_hopital ?w}}
```

POUR LE NOMBRE DE MEDECINS :

```
SELECT (COUNT(?e) AS ?nbmed ) WHERE{
OPTIONAL{ ?e a ns2:Medecin.
?e ns2:habite_a <file://PROJET.owl#WILAYASELECTIONNER>.}}
```

Nous avons pensé à d'autres requêtes SPARQL que ne nous avons pas intégrer dans notre application et qui reste intéressante à voir dans ce qui suit :

-sélectionner wilayaAtteinte :

```
SELECT ?w
WHERE {
    ?w rdf:type ns2:WilayaAtteinte
}
```

-sélectionner FemmeEnceinte :

```
SELECT ?f
WHERE {
    ?f rdf:type ns2:FemmeEnceinte
}
```

-demander si un patient prend un traitement :

```
ASK WHERE {  
  ?x ns2:Nom #nom ^^xsd:string.  
  ?x ns2:prend_traitement ?traitement  
}
```

-demander si un médecin existe dans notre ontologie:

```
ASK WHERE {  
  ?x ns2:NomMed #nommedecin ^^xsd:string  
}
```

-décrire les individus d'une classe par exemple la classe Medecin :

```
DESCRIBE ?p WHERE {  
  ?p rdf:type ns2:Medecin}
```

**NB: la clause DESCRIBE doit être utiliser dans protégé car elle est par
encore implémente dans python**

PARTIE 3 : Annexe

1.ANNEXES :

1.H.xlsx : ce fichier contient les hôpitaux des 48 wilaya de notre pays crée à partir du fichier PDF :

« http://www.santemaghreb.com/algerie/documentations_pdf/liste_tel_chu.pdf »

2.medecin.xlsx : ce fichier est créé au préalable à partir du script praticien.py qui est un

Web scrapper qui extrait à partir du site « <https://www.algerie-pratique.com/Medecins> » les différents médecins selon les spécialités décrites et rempli notre fichier Excel.

3.test.xlsx : vu la sensibilité des données (information personnelle des patients) aucune base de données n'est divulguée sur le web, donc les données présentes dans ce fichier sont des données abstraites créer afin de montrer les multiples facettes de notre ontologie.

2.graphe

Vu le nombre conséquent d'individus (environ 600) présent dans notre ontologie, le graphe généré sera énorme et distinguer les différentes classes et les relations entre les individus sera compliquer, nous avons donc mis en place un pseudo-graphe contenant quelques individus ainsi que les relations entre eux (voir figure II.10).

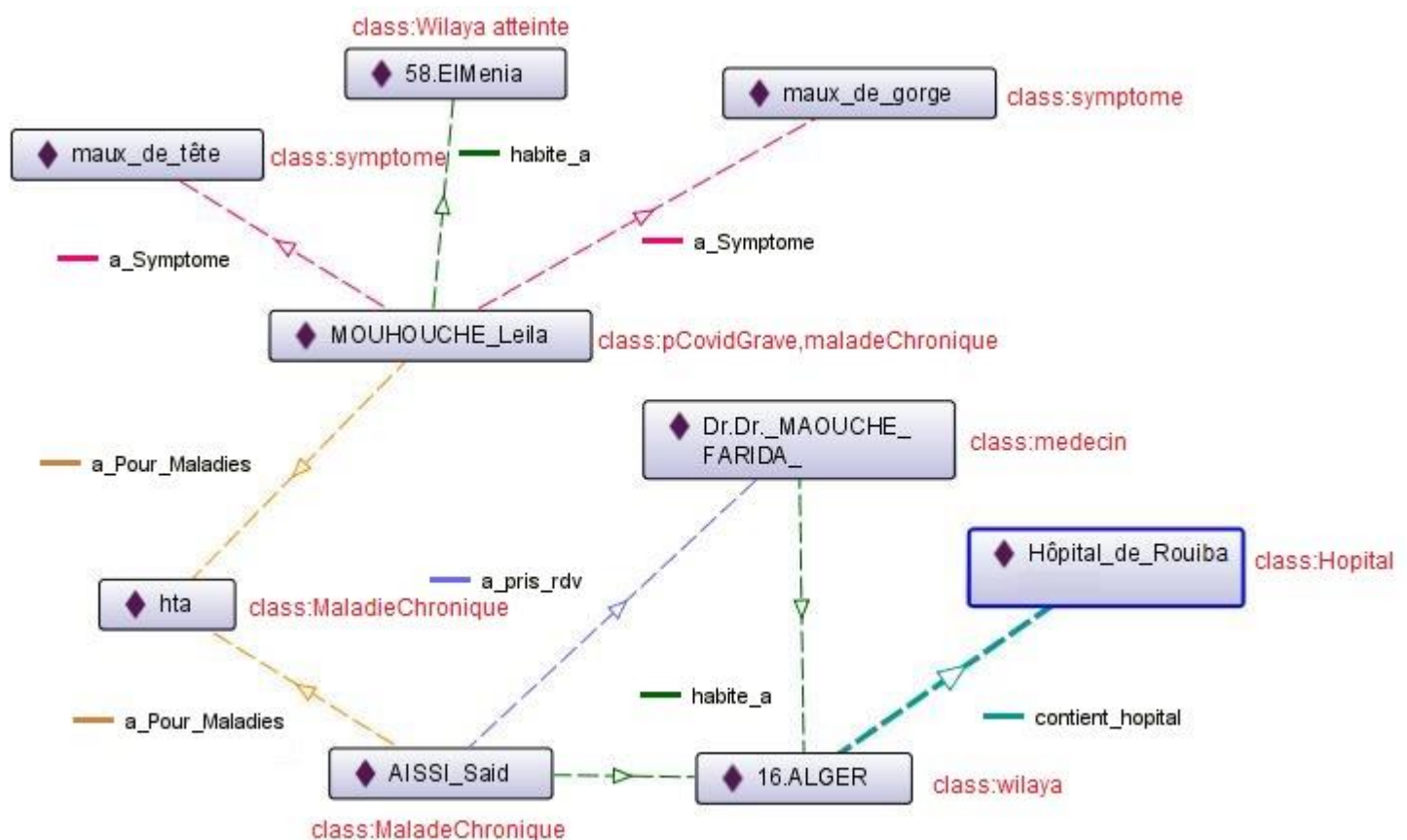


Figure II.10