**Report on**

**Hospital Management**

**INDEX**

1. Introduction 2
2. Concept and Aim 2
3. Design 3
   1. Semantic Data Technology 3
   2. Ontology for Hospital management 5
4. Implementation 10
   1. Environment Setup 10
   2. Packages and Classes for hospital management 12

4.2.1 Queries for front page 13

* 1. Application User Interface 15

1. Evaluation and use 17
   1. Queries for age 17
   2. Queries for id 18
   3. Queries for disease 19
   4. Query for name 21
2. Critical Reflection 22
3. Conclusion 23
4. Appendix 24
5. References 28
6. Bibliography 29
7. **Introduction**

The basic aim of the project is to provide an effective communication between various agents available in the hospital system. The ontology is modeled in such a way that the classes are able to communicate and possibly answer the questions on their own.

This report’s main aim is to present, evaluate and critically analyze the hospital System. Semantic Data Technologies concepts are applied to the Hospital management System, which provides us with the information for the treatment, symptoms, medicines and disease. This ontology can be used by the data analyst and also with the hospitals who wants to use a semantic web for their applications.

The basic idea of the project will be explained in the Concept and Aim section of the report. There are various results of existing system explained in detail. Various terminologies used in the semantic Data technology is defined in the Design section. The motivation behind ontology of the hospital system is explained in detail. The tools and applications used for the development are discussed and explained in the Implementation section. In the Evaluation and Use section SPARQL queries used by the application are explained and the results are shown. In the critical reflection section, the whole ontology will be critically analyzed and will be compared with the similar application that are available, what are they missing and what necessary development is required in the current system.

1. **Concept, Aim**

This report is about the semantic application which uses the treatment methods available for the various type of disease that are previously available. As there are not much development in the medical science field in the previous decades, most of the hospitals are trying to go on the internet so that to reduce the cost of maintenance, minimize records of the patient and even make the patients work easier as even they can search and understand their cause and treatment of the of sickness they are facing, and in all these scenario the semantic web can help. As the data are linked across the web so the doctors will find it easier to check the previous history of the patients, at the same time the patients can understand the type and treatment of the disease if the doctors are charging something different from the requirement.

This application open a whole new way of understanding the management of doctor, patient and disease cure procedures. The main purpose of this application is to make the data readable and understandable to doctor, patient and machines. The patients may or may not be understanding the problems previously and could not clearly explain the problem being unaware of the medical terminology. Another use of the application can be to analyze the type of disease happening in a particular region so that to emphasize on that particular disease treatment and possible reduce the number of cases from that area. This application can even be used to analyze the most vulnerable age group due to a particular disease. It can even be used for analysis of several diseases in a particular area which will help to understand the cause of the disease more intuitively.

The main use of this ontology is to make websites more reliable and if a particular website is having false information then the user will not use those sites which in turn makes the owners to redesign to a new website or to even close the system as a whole. Another idea here is to reach the doctor who is specialist of a particular disease, which will make treatment very smooth for patients. With the help of this ontology we can see that which medicines have a side effect, and use of those medicines can be decreased.

1. **Design**

**3.1** **Semantic Data Technology**

The basic aim of semantic data technology is to fill the gap between machine and human. It basically consists of two main parts (1) a model definition to represent and define relationship between resources which is possible by using RDF and (2) vocabulary which is used to define the meaning of the resources. It is possible by using RDF Schema and “web ontology language” (OWL).

As the knowledge base of medical science is increasing the demand for big data analytics is increasing as well. More companies are trying to invest in medical science, and hence there is requirement for some medium to share the data to each other and the information should have to be consistent throughout the web. With the help of semantic web technology we can achieve interoperability which opens a new door for big data analytics as web can also understand the language. Semantic data technology uses a schema language to show the relationship between the objects which is RDF, it uses graph model to show the relationship between the models. A query language is used to query on the given model here SPARQL comes handy, and a schema to represent and query the hospital management ontology is achieved by RDFS or OWL.

Now linked web has changed the way people used to interact with various data bases. Unlike the relational databases like SQL, where to update even a single instance we had to update the entire databases, the linked data can be used and just by updating a single instance we update the whole set of databases at once. For instance BBC which is the first company to use semantic web for commercial use now can manage its all websites with just one structured semantic web data. Linking a collection of distributed datasets forms knowledge graphs. Popular knowledge graphs are Cyc, Freebase, Wikidata, DBpedia, YAGO, NELL, Google’s Knowledge Graph, Yahoo!’s Knowledge Graph, Microsoft’s Satori, and Facebook’s Social Graph (based on RDFa13) (H. Paulheim, 2016). These companies have interacted to make the web pages linked and even user friendly at the same time.

RDF schema is use to show the relation between various objects in the form of subject predicate and object. For instance in my ontology

Patient treated\_by doctor

This set of information is called triple. And this tripe is having combination of subject, predicate and object. Now, here my subject is patient and predicate is treated\_by and object is doctor. Triples are directed nodes of each graph and each node is labelled.

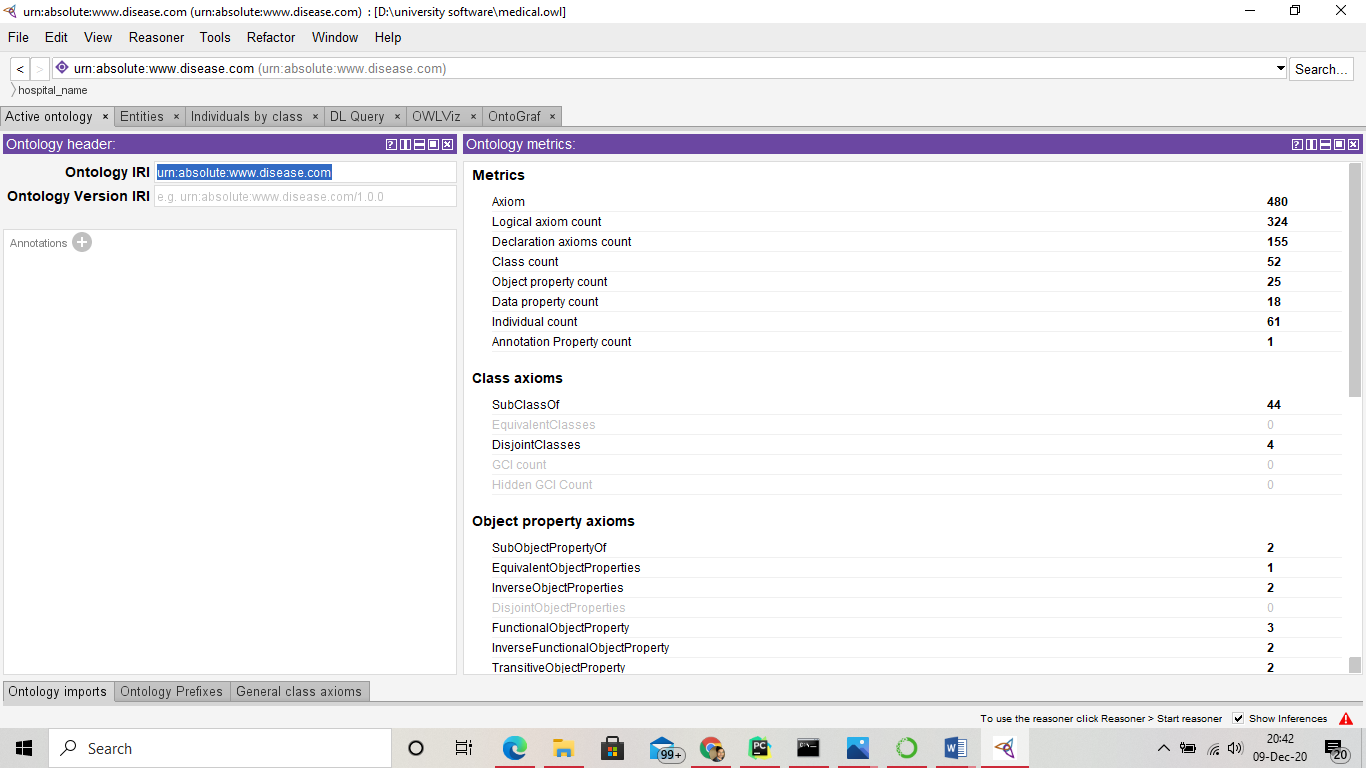
RDF schema is the extension of RDF which is used to describe the relationship between the classes.

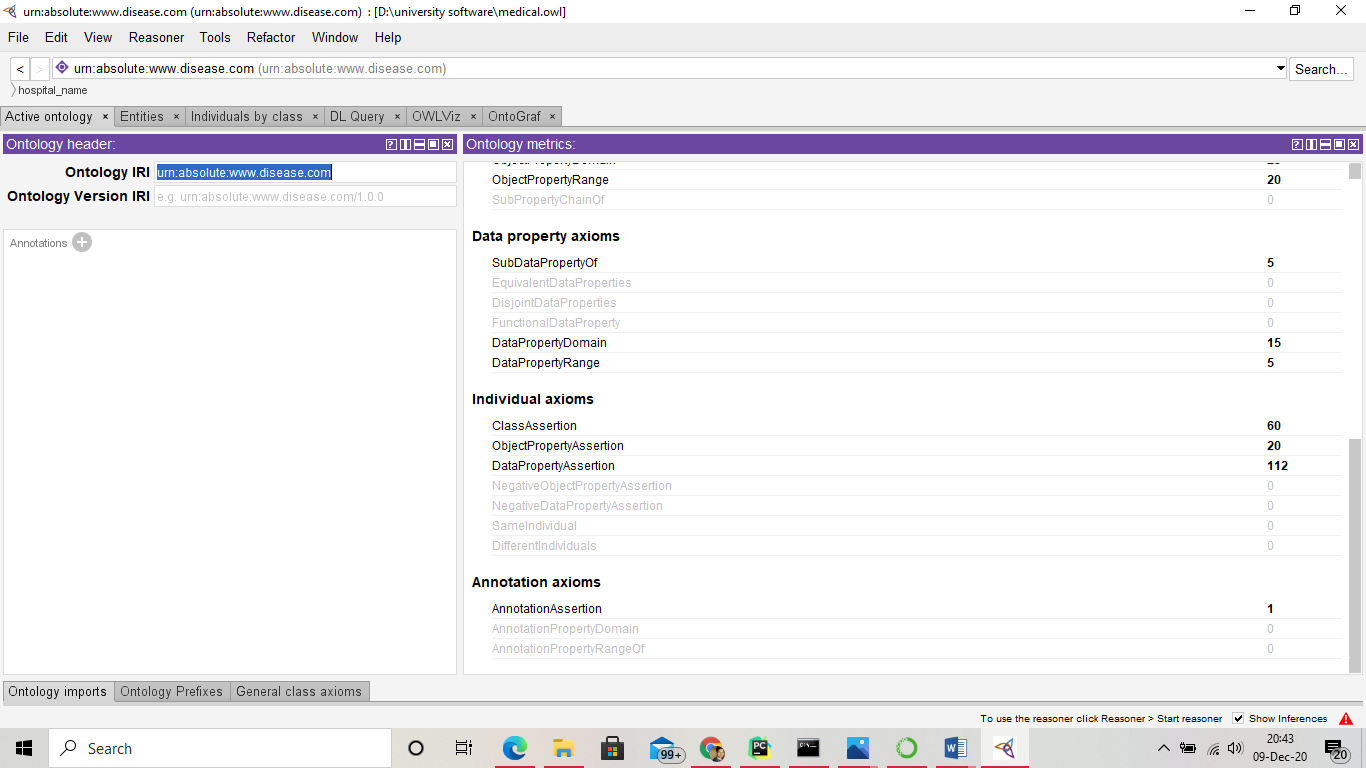
Web Ontology Language (OWL) is the extension of RDFS which provides a deeper meaning to web by making the data machine readable while previously the information was just used to present the data in form of text to users.

SPARQL is the pattern matching query language for RDF. So this is considered backbone of semantic web. It is used to query the various end points of our schema. This provides a better requirement analysis and fulfillment to the user end. These queries are run on triples of the graph. SPARQL have various build in function to achieve the desired result from the ontology and ontology manipulation. Some of the functions include SELECT, it selects the name of result or row that we need to access from our ontology. CONSTRUCT query creates a RDF triple as a result. There are function to modify the data like GROUP BY, ORDER BY and LIMIT. The GROUP BY is called when we need to aggregate the data. The ORDER BY is called when we need or data in ascending or descending order. LIMIT function limits the number of results to be accessed from the ontology. Query operations are provided in WHERE clause.

Triples are the small piece of the entire schema which provides a single information from the entire knowledge base. These triples are known as the end point of sparql query.

**3.2** **ontology for hospital management**





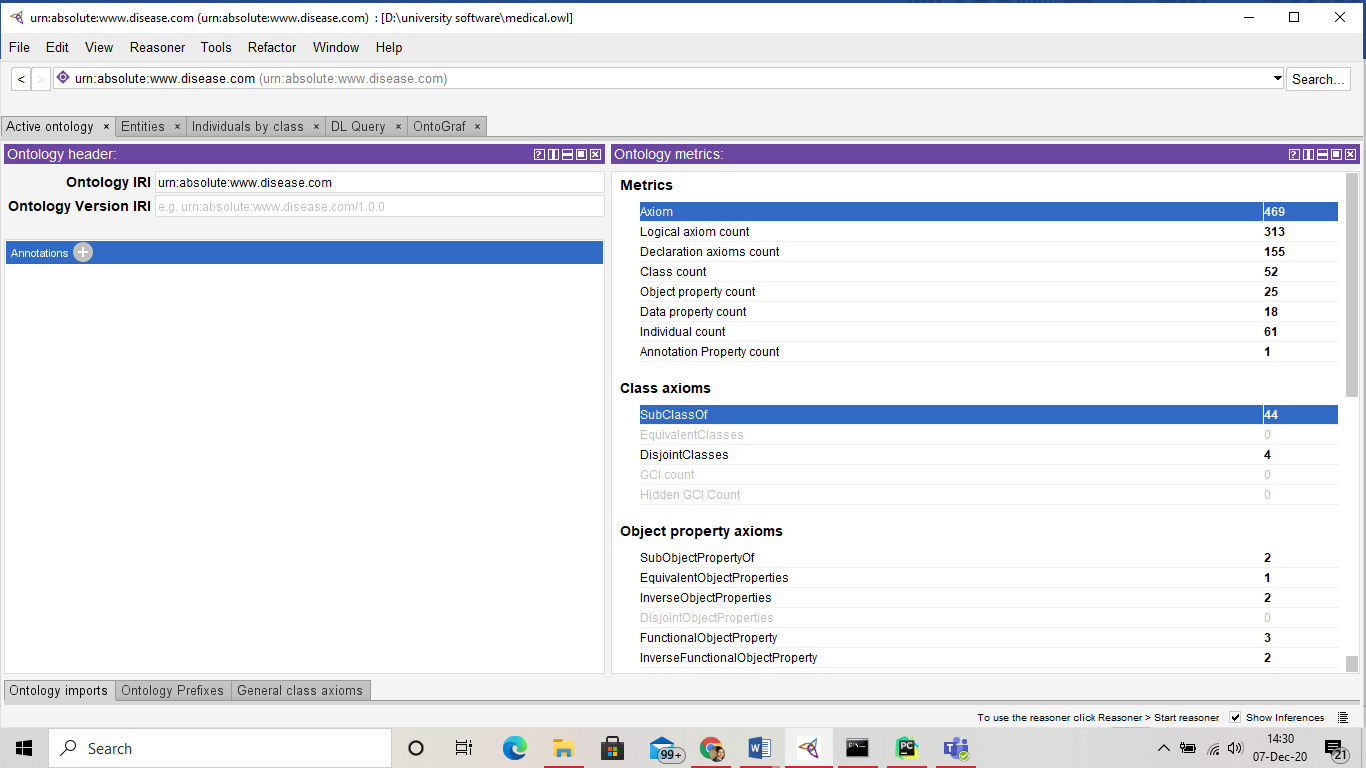
The hospital management ontology is designed using protégé which is an open source software coded using java. The main purpose of this software is to provide a graphic user interface for ontology development.

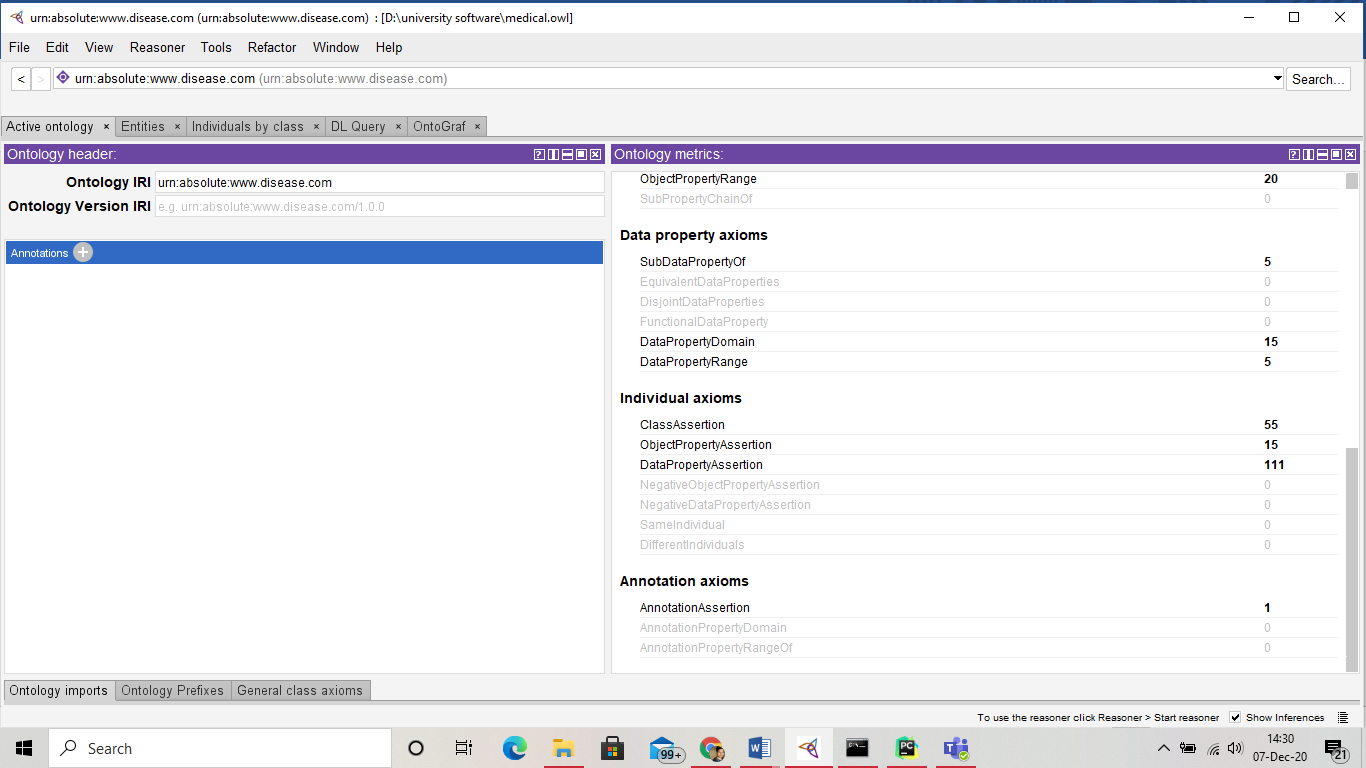
The given hospital management ontology have 480 axioms, 52 classes, and 61 individuals.

Axioms: Assertions (including rules) in logical form that together comprise the overall theory that the ontology describes in its domain of application. (Maureen Donnelly; Giancarlo Guizzardi, 2012).

Classes are the set of collection of various objects. It can further be divided into other subclasses. Class in ontology itself is the subclass of thing.

Individuals are the instances of the objects. These are the last end object which cannot be further divided into other classes.



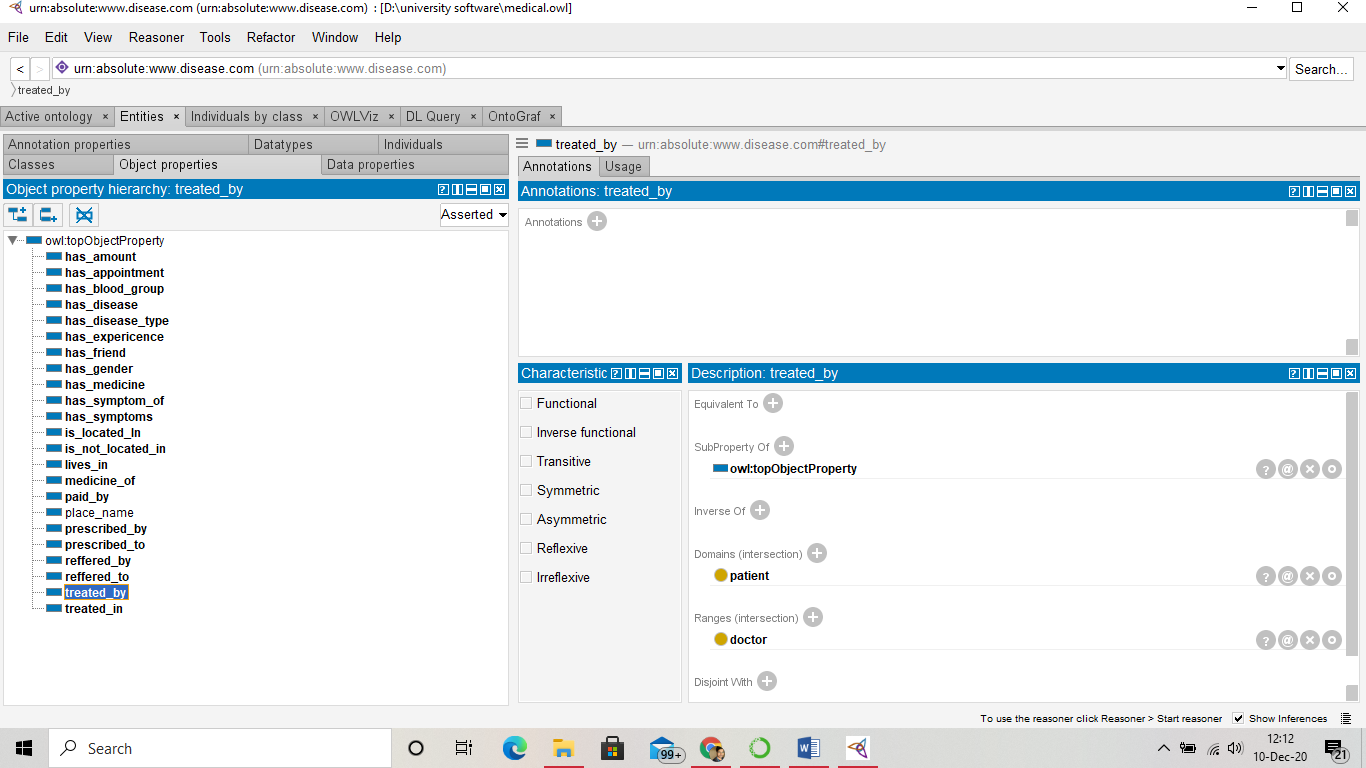


The main classes in my ontology comprises of hospital\_name, injuries, external, close, open, internal, disease, disorder, medicine, payments, amount, person, Family, friend, patient, worker, place, prescription, symptoms. These classes are further divided into other subclasses. Each of these classes are used to define the hospital system in a more semantic way.

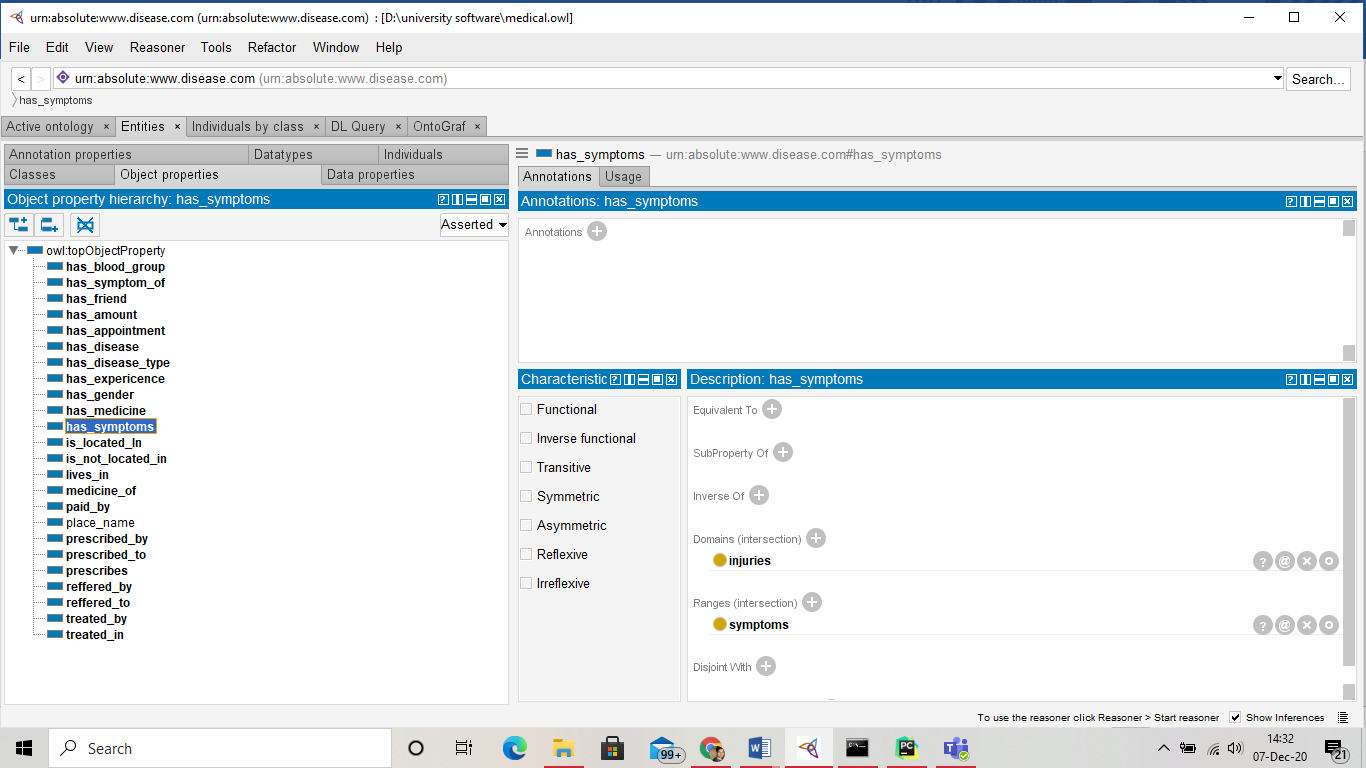
Hospital name is used to define the name of hospital where every bit of work is being performed. The injuries section defines the injuries that happens to a normal human. It has further been divided into two sections named internal and external injuries. The external diseases further classifies into two parts as open and close. These are basically the diseases which a human can see directly. And in the internal injuries I have divided into diseases and disorders. These are such diseases which can are not visible sometimes but harms the body internally.

Person class defines the type of people that are into the hospital the person can be a patient, friend, family, worker etc. place defines the number place where the hospital is situated or patient belongs to. Prescription is the medication given to the patient during the treatment. Symptoms sections defines the number of various small diseases that ends up being a heavy disease.

The other section is object properties. Object properties are used to show the relation between various classes. In this ontology we have defined various object properties for instance has\_amount, has\_appointment, has\_disease, has\_experience, has\_friend, has\_gender, has\_symptoms, medicines etc, and these all have various domain and ranges.



As it can be seen in the figure the treated\_by object property have domain patient and a range is doctor. These three nodes of the graphs are making a tuple to form a sentence. In other words it views the asserted characteristics for a selected objects in a particular class. The object property that are available can be of 7 types in protégé. In protégé there are 7 check boxes with the names of every class property written on it. If the checkbox is ticked then the mentioned property is activated if it is not checked then it means that the characteristics is not asserted at all. All these property types possess a special meaning to define a particular class type



Functional: In this property individuals can have at most one value and not more than that. So there can be at most one outgoing relationship along the property for that individual. In hospital management ontology has\_experience is a functional property since one person can have only one experience label.

Inverse Functional: Inverse property of a selected property is functional whether its defined explicitly or not. In other words it can be said that there can be at most one incoming relationship with that property. If multiple individuals are specified as incoming values then it will be considered to be as a same object. Here is\_located\_in can have just one incoming relations and hence is defined as inverse functional.

Transitive : this is defined by the fact that if A is related to B and B is related to C then individual A is related to C. for instance patient\_1 has\_friend patient\_2 and patient\_2 has\_friend patient\_3 then patient\_1 is relate to patient\_3 with a triple patient\_1 has\_friend patient\_3

Symmetric: this implies that if individual A is related to B then individual B is also related to A along the same property. Here disease\_1 has\_symptom\_of disease\_2 then disease\_2 also has symptom of disease\_1.

Asymmetric: this asserts opposite to symmetric, it is defined as if individual A is related to B then B is not related to A. here diasease\_1 has\_symptom symptom\_1 is a Asymmetric property as disease has symptom but a symptom can occur in various other diseases and is not only to only one disease.

Reflexive: it causes every individual to be related to itself by the given property.

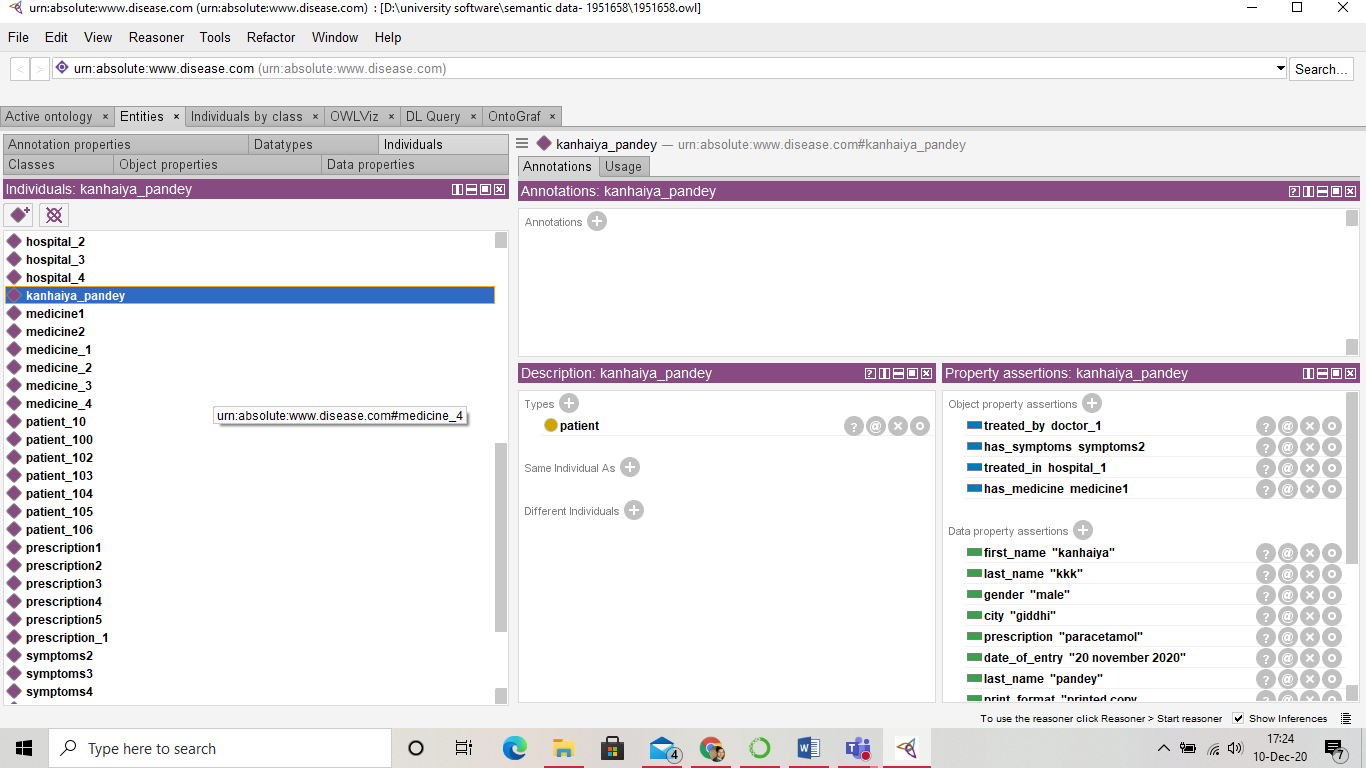
Irreflexive: this property states that a given individual cannot be related to itself via that poperty.

(protégé)

Based on these property itself we have made our ontology.

The ontology comprises of various data properties. Data properties is used to assign every tuple a value. Various types of data properties are available to specifically assign the datatype to be used but if we don’t assign any range to our data property then it is automatically considered as literal. The literal can be anything an integer, date, string etc. It can be seen that my ontology have name as a data property which can be assigned to any person or a medicine. As medicines also have a name. Data property is the values that are assigned to classes and objects and are stored in ontology.

The last part to be considered in ontology is the individuals. They are the real world objects which adds to make an entire knowledge base of the ontology. In hospital management system there are individuals such as medicine names, patient names, doctor name, city names etc. these values are allowed to repeat as a single class can have multiple individuals with the same value but the individual id have to be unique. Data stored in the individual can be same to others as well or even can have multiple data in itself. So here my individuals are patient\_names, doctor\_names,city\_names, friends, disease\_name, family, workers,mecine, symptoms etc.,



As per Noy and Mcggunise framework the basic requirement of an ontology is

* “To share a common understanding of structure of information among people or software agents.
* To enable reuse of domain knowledge
* To make domain assumptions explicit
* To separate domain knowledge from the operational knowledge
* To analyze domain knowledge “ (noy, McGuinness, 2020)

Here hospital management uses all these concepts after development. So, if an individual uses my ontology the most of the domain specific details are already provided for instance INJURIES, HOSPITAL NAME, PEOPLE ,etc. and these can be used as it is which will aggregate the information to answer the user questions.

Here data is not hard coded and hence it can be imported by the new users and can make specific changes as per the requirements.

The hospital management is made up of various classes and all these are having their specific works for instance the person class have other works compared to that of injuries class and these combine to make a whole working hospital management ontology.

Here all the words used are from a formal understanding of the domain and hence if someone imports r uses the ontology will understand it very easily and hence extending or reusing the ontology will be easy.

1. **Implementation**

**4.1 Environment set-up**

To access the information from the ontology and make it user friendly, here python language is used. The interface of the application is made with the help of html, css, javascript. The query language used here is SPARQL. The python frameworks used is the Django and SparqlWrapper2. Full application is designed with the help of pycharm IDE. Django framework helps in rapid development of the database application.

Sparqlwrapper helps to easily run the sparql queries.

“Apache Jena Fuseki is a SPARQL server. It can run as an operating system service, as a Java web application (WAR), and as a standalone server.

Fuseki comes in two forms: a single “web app” system combined with admin and query UI. And a main server suitable running as a part of large deployment like Docker or embedded running. The same core protocol engine and same configuration file format are used for both types.

Fuseki provides query and upgrade protocols for SPARQL 1.1 as well as the SPARQL Graph Store protocol.

To provide a stable, transactional, persistent storage layer, Fuseki is tightly integrated with TDB, and integrates Jena text query.”(Apache software foundation, 2020)

* 1. **package and class for hospital management**

To make a full working interface and connect to the backend of the ontology various packages have been provided by the developers. Here the packages used are Django and sparqlWrapper2. Python uses MVC (Model View Controller) concept to make an application. In the model section the application queries or logics are written, which is unseen from the user. In the view section the representation of the application is written. This section takes input from the user and converts these inputs to command for the use of model and view. In order to work the all the three components of the application should interact very precisely. Now to gather a good understanding some commands is given

Sparql=SPARQLWrapper(‘http://localhost:3030/disease/sparql’)

This is the section where server address is given it can be saved in any variable here Sparql is used as a variable name. With the help of this method we build a connection between our application and database.

Sparql.setquery(“”” “””) in this method we pass the query that needs to be performed on the ontology.

This method is used to run the query on the ontology.

return sparql.query().bindings

This command return the query output to the method name.

The class names are defined as per the convenience. For instance to fetch the id, we have defined the class name as ‘id’ and to fetch the result from the previous id the class name is used as ‘idr’. In python the classes are defined with the help of ‘def’ keyword, and various input parameters can be passed as an argument operator in the class.

* + 1. **Queries for front page**

The queries are first used to access an end point and then with the help of those endpoints the results are fetched.

The query used to get the id of a person is

PREFIX di: <urn:absolute:www.disease.com#>  
 SELECT ?id  
 WHERE {  
 ?name di:id ?id  
}

In the above mentioned query all the id’s available is accessed.

These are then used to for further evaluation

PREFIX di: <urn:absolute:www.disease.com#>  
 SELECT ?age  
 WHERE {   
 ?name di:age ?age.  
  
}

This fetches the age of all the person in the ontology.

PREFIX di: <urn:absolute:www.disease.com#>  
 SELECT ?place  
 WHERE {  
 ?x di:city ?place.  
  
}

This query returns all the city names from the ontology.

PREFIX di: <urn:absolute:www.disease.com#>  
 SELECT ?last  
 WHERE {  
   
 ?name di:last\_name ?last  
  
}

This query returns the last name of the every person in the ontology.

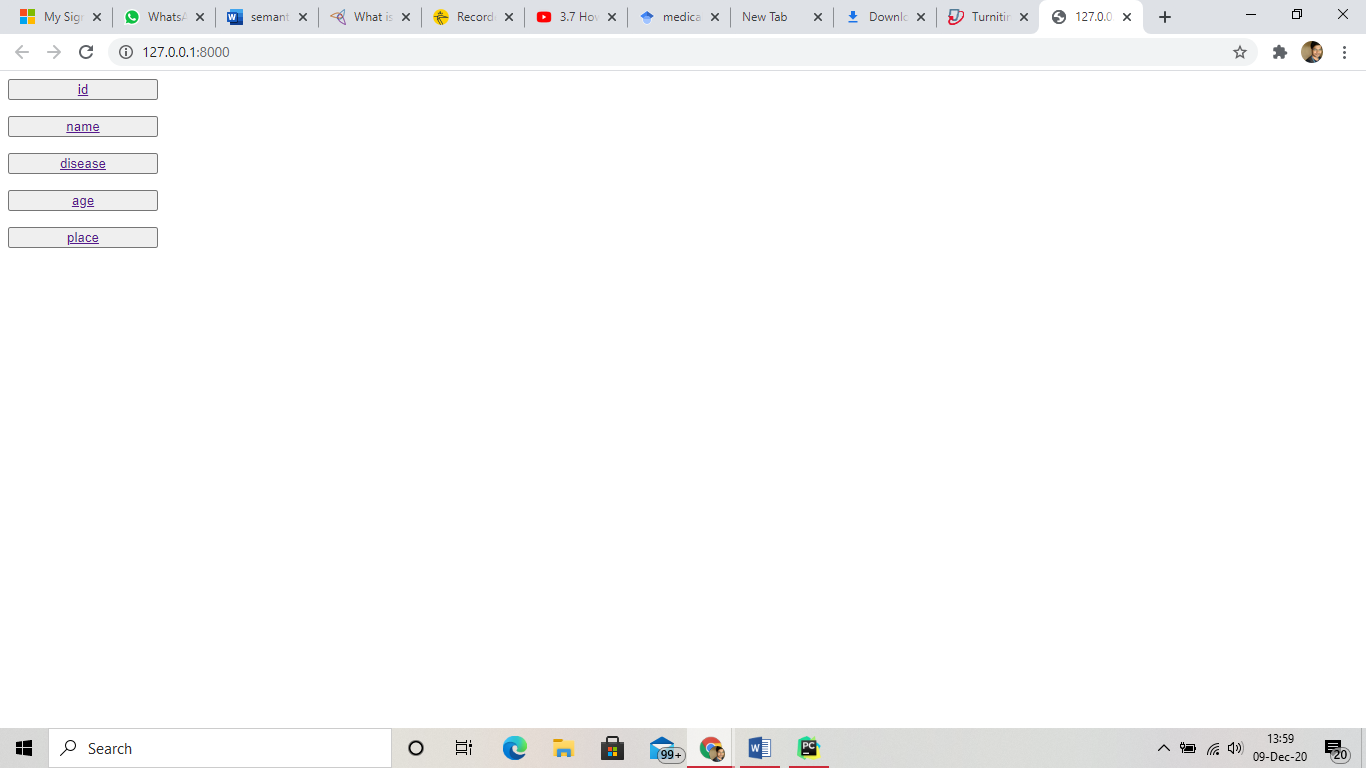
SELECT ?disease  
WHERE {  
  
?x di:disease\_name ?diseas.  
BIND(REPLACE(STR(?diseas),"\_"," ") AS ?disease)   
  
}order by asc(?diseas)

This query helps to access the disease of the patients and with the help of bind operator all the “\_” values are replaced with a space value. Since most of the disease names were given with the underscore in it. With the use of order by function the diseases are ordered in ascending order of their names.

All these queries are used to just access the single values from the ontology, and further analysis will be done on them with the help of another queries.

* 1. **Application user interface**

The user interface of the system is made fairly simple so as to make it more understandable to the end users. The direction for next page can be found at every page.

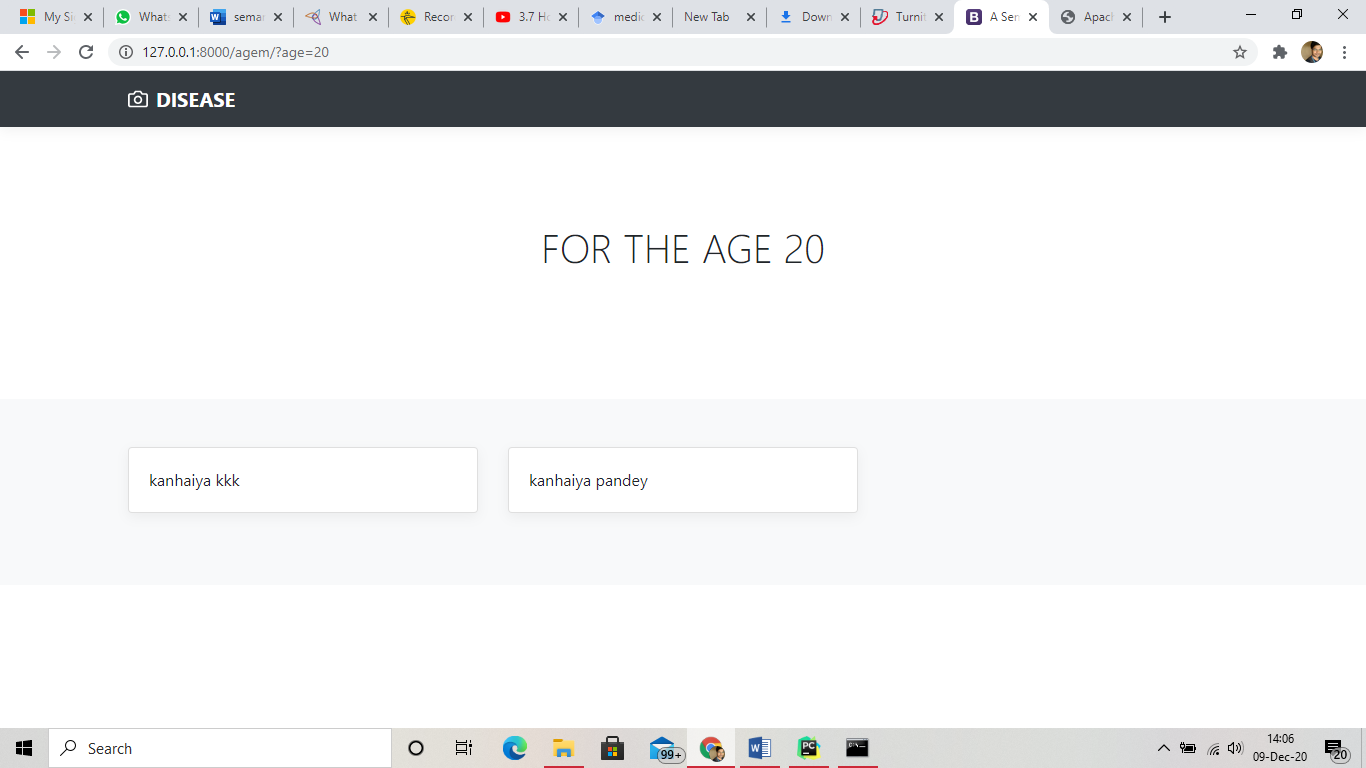


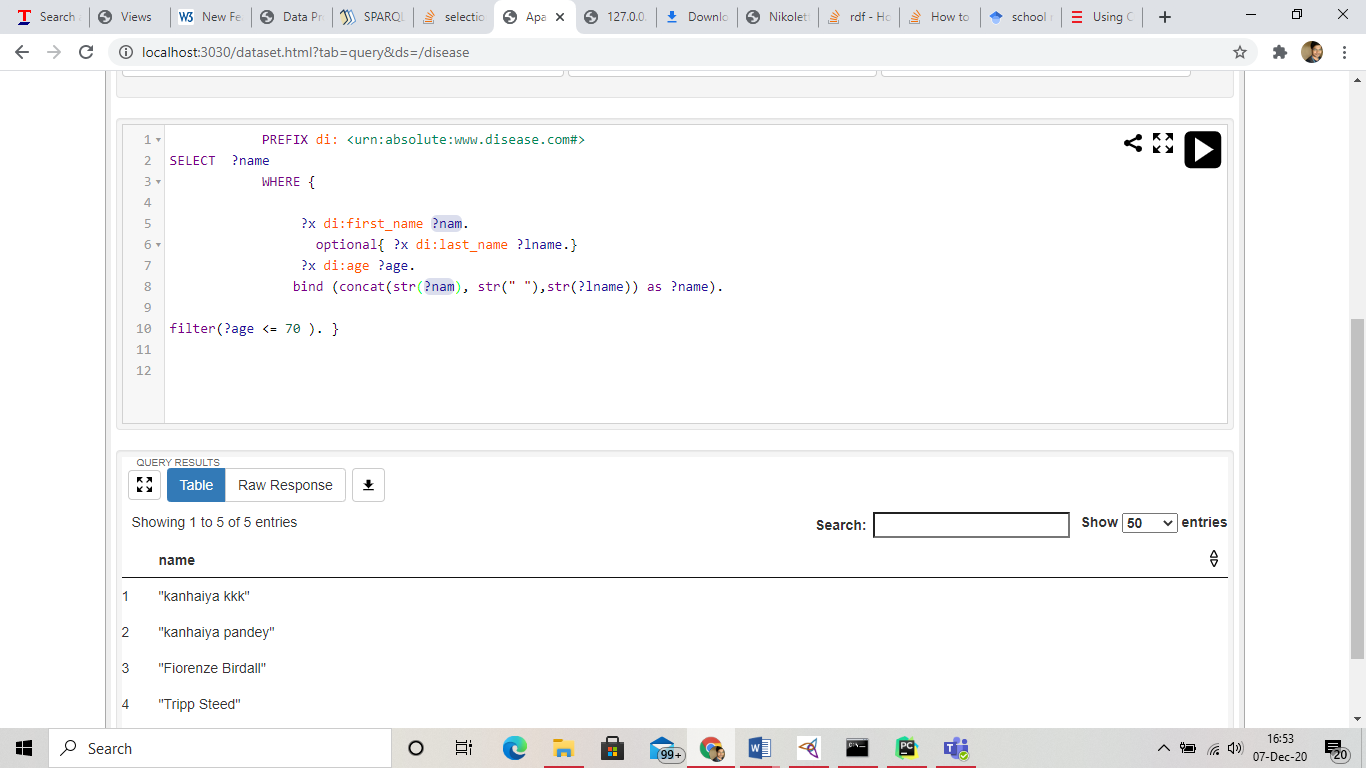
The home page of the application is composed of 5 buttons which points to the specific type of results that a user would like to access. The user is allowed to search by name, id, disease, age, and place. As a user selects the type of information he needs from the ontology, the pages takes to the next search box page where all the specific details from the ontology is already provided in the dropdown menu and user can select whichever data point he needs to select. All these data in the drop down menu is made available with from the ontology itself. If the user needs to access the information outside the ontology or which are not already available then this options will not be available as the information is not a part of ontology. Then after selection of the data point user needs to click the search button just below the dropdown tab.

The selected endpoint will then be queried with the help of python and SPARQL in the backend and on the next page user can see the results of the selected data point. The upper section of the result shows the selected data point value and the lower part of the page shows the results obtained from the queries. Either the result obtained will be in sentence form or the result can be a simple output obtained from running the query on the database. The sentences are made as simple as possible to make understandable to the end users. Once the query is finished the user have an option to stay at the page until the requirement is completed as there is no time limit provided at any page. If the user needs he can perform another search. On the top left corner of every page the “Disease ontology” is written which is clickable. If a user needs to perform another search then a clickable logo is available at top left corner of each page and it will take back to the home page where again one can choose the type of search needed or if the user needs the query to be of same endpoint then the back button from the browser can be used which will again take him to the previous page where selection can again be made.

1. **Evaluation and use**

**5.1 query for age results**





**SELECT ?name  
 WHERE {  
  
 ?x di:first\_name ?nam.  
 optional{ ?x di:last\_name ?lname.}  
 ?x di:age ?age.  
 bind (concat(str(?nam), str(" "),str(?lname)) as ?name).  
   
filter(?age <=** age **). }**

This query is used to access the details of the person whose age is less than or equal to the age selected from the previous menu.

Select statement selects the number of rows that are to be fetched from the ontology. Here I have given the variable name ?name.

Where clause is used to write the queries on the triple pattern. The variable or the values are accessed with the help of those triple pattern.

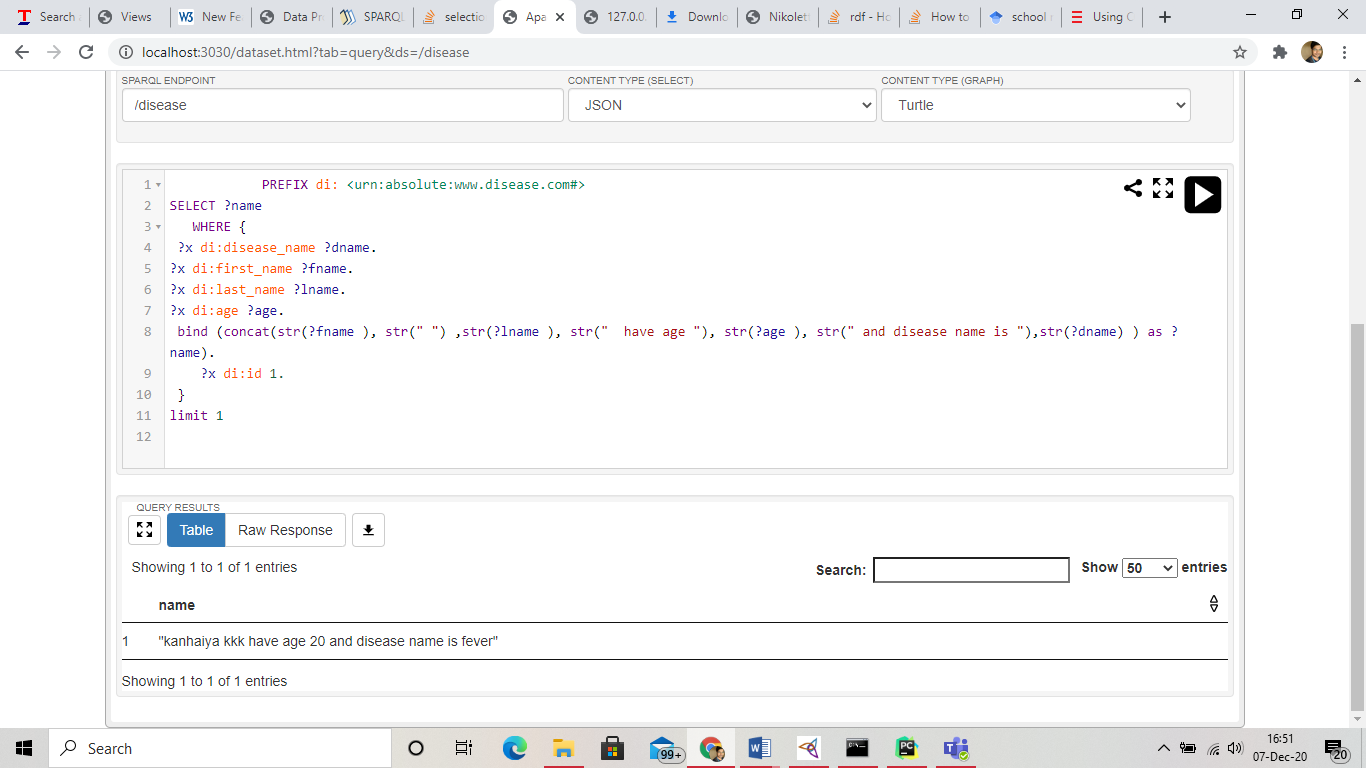
Here prefix used is

**PREFIX di:<urn:absolute:www.disease.com#>**

This prefixing says that wherever “di” is used it points to the whole URI.

In the first line of where clause, first\_name of all the triples are being fetched. With the help of optional function we are saying that few persons may not be having their last names, this fetches even that results as well. Now the value whose first name and last names are available we are matching their age as well. So, now I have first name, last name and age of every triple in the ontology. With the help of bind command we concatenate the string to and get the whole name of the person. Since we are interested in the names of person whose age is less than the selected value we have used filter function and removed those people names whose age is more than the required selected or specified value.

* 1. **Query for id**



With the use of ‘id’ we need to access the first name, last name, disease name, age.

**PREFIX di: <urn:absolute:www.disease.com#>  
  
 SELECT ?name   
 WHERE {  
 ?x di:disease\_name ?dname.  
?x di:first\_name ?fname.  
?x di:last\_name ?lname.  
?x di:age ?age.  
 bind (concat(str(?fname ), str(" ") ,str(?lname ), str(" have age "), str(?age ), str(" and disease name is "),str(?dname) ) as ?name).   
 ?x di:id** id **.**

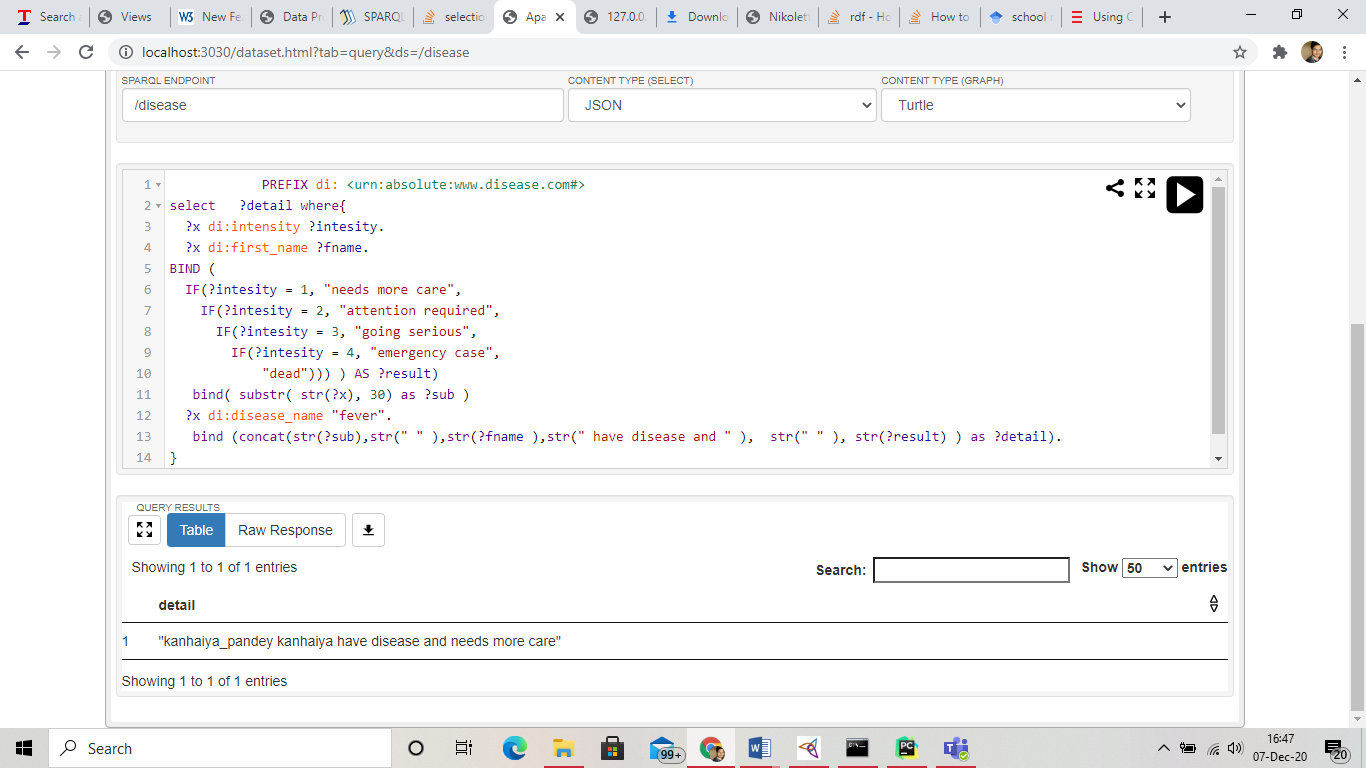
**}**

**limit 1**

here we are accessing the disease name first name, last name and age of the person this all the values are saved in the specific variables. With the help of bind statement we have created a sentence that “first name” +”last name” + have age+ “age” + and disease name is +“ disease name” with help of id of a person.

We have used the limit function to access the top result of the id.

* 1. **Query on disease**



As we have already accessed the disease name from the previous queries we now need to use this data to access some useful insight from the ontology. To achieve the results the query used is:

**PREFIX di: <urn:absolute:www.disease.com#>  
 select ?city where{  
 ?x di:intensity ?intesity.  
 ?x di:first\_name ?fname.  
  
BIND (  
 IF(?intesity = 1, "needs more care",  
 IF(?intesity = 2, "attention required",  
 IF(?intesity = 3, "going serious",  
 IF(?intesity = 4, "emergency case",  
   
 "dead")  
 )  
 )  
 ) AS ?result  
 )  
   
   
 "?x di:disease\_name**  “di”**.**

**bind (concat(str(?fname ), str(**" "**), str(?result) )**

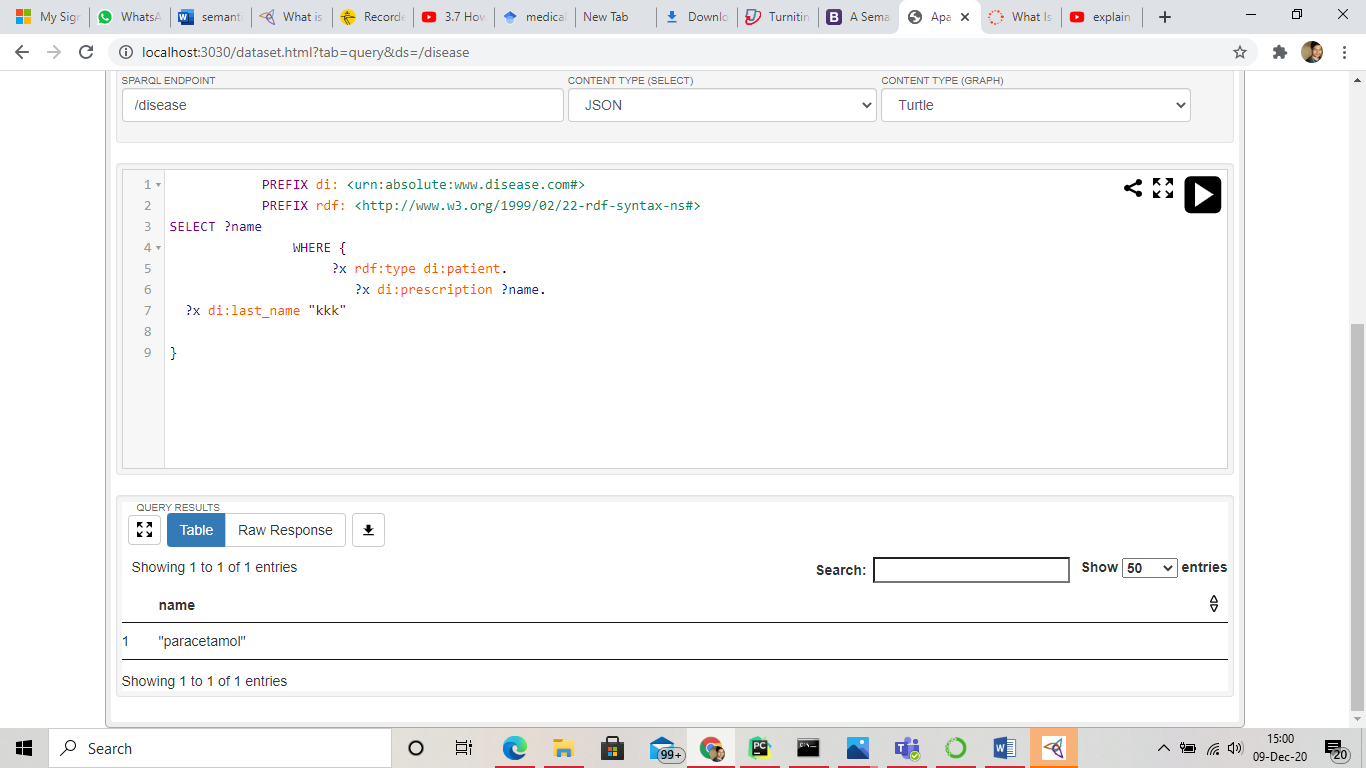
**as?city). }**

Here ‘di’ can be any disease name.

The query is used to access the intensity of disease which can be anything from 1 to 4. Based on the intensity the doctors are decided to react. Anything which is not among these values are considered dead person. The patient with intensity 1 is assigned the value of low impact of disease and hence needs more care with intensity 2 the patient requires attention. If the intensity is 3 then he is on the verge of going serious and proper treatment is required. If the same person have an intensity of 4 then it is an emergency case and immediate action should be taken. With the bind function we have made a sentence where

First name + the intensity conclusion is combined in a string format.

* 1. **Query for name**



**PREFIX di: <urn:absolute:www.disease.com#>  
 PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>  
 SELECT ?name   
 WHERE {  
 ?x rdf:type di:patient.  
 ?x di:prescription ?name.  
   
  
?x di:last\_name** + last\_name +**. }**

In the given query rdf function is used and hence the prefix is already added here. This query fetches every instances of class patient. Now from that information here we have accessed the prescription name of the patients whose last names were accessed from previous page query on the application page.

1. **Critical reflection**

Since there is not much data available on the internet for healthcare systems which are using semantic web, and those which are available as an ontology are still under development and hence the results are analyzed based on the my ontology.

The ontology is helpful only if the patients are sure of their disease or if the doctors are sure of what the probability of the patient is for being right. There are various terms which are different in other parts of the world or which are not even used. Hence, this ontology can help the people who have knowledge of English while those countries where English is not spoken will not understand the ontology at all which makes it difficult to deliver the ontology. Another issue that arises is lack of knowledge bases, and even though we organize some knowledge base can it be used in other ontologies. This is another big issue with this system. The guidelines given by the government should be followed as it might be harmful to the people who does not have any prior knowledge of the medicines and diseases. In the excitement there may be some mishaps. There are no useful system as of now which are creating a realistic virtual hospitals which makes it very difficult to understand the overall requirement for hospital management and even no benchmark is available for this where improvements are required. And hence to make a better ontology for hospital management firstly small sets of realist hospital management approaches should be used as a part of experiment and then only a full working hospital management system can be possible.

There is another chances where a person can access the ontology and just use it to misguide the users for their benefits in that case the same ontology plays a different role for new users. So to use this system some prior checks should have to be made such that no illegal or deceit activity can be made with the help of ontology.

The ontologies which are in development may have different names compared to what is being used in my ontology in such a case the removal of duplicates from web is another point. As those who don’t have knowledge of medicines might get confused about the names.

If a patient is not specific about the type of disease and tries various diseases with various probabilities of outcomes in that case also to show a specific result would not be possible.

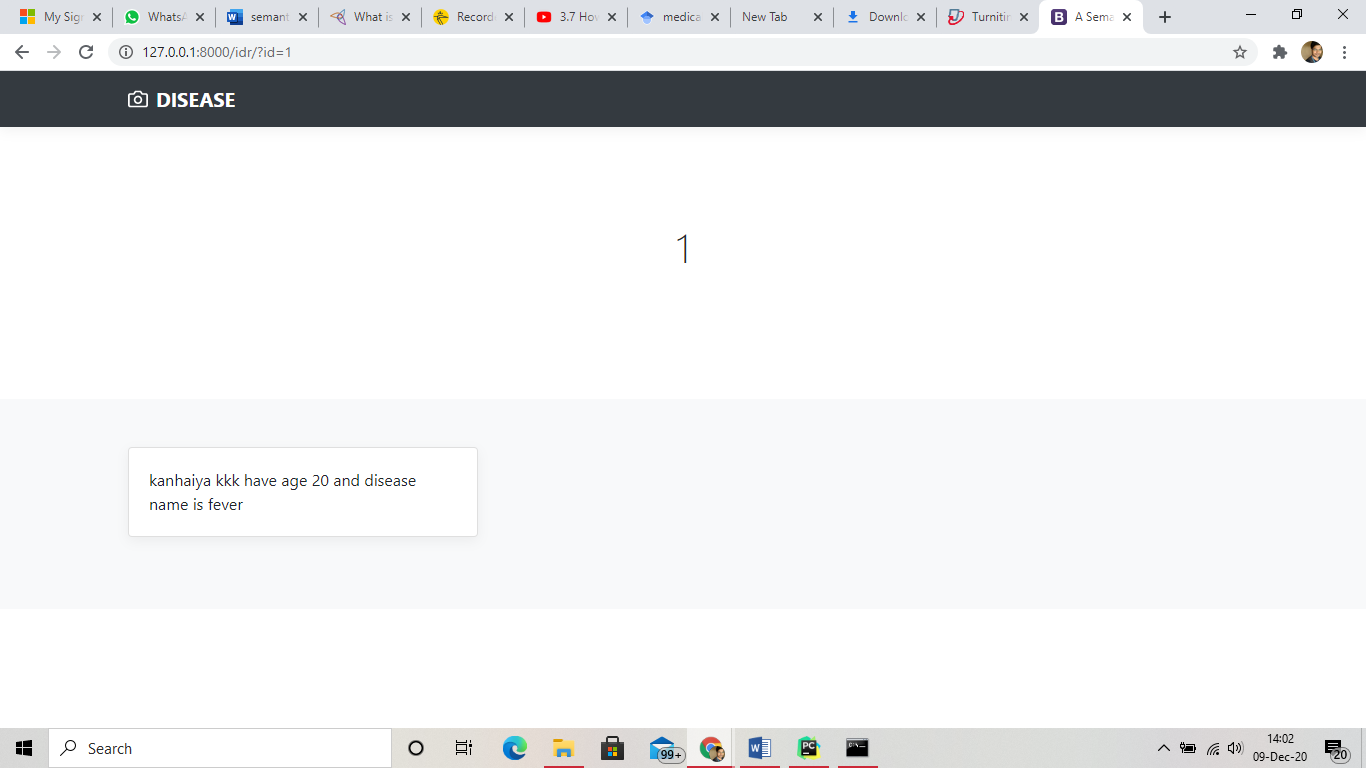
1. **Conclusion**

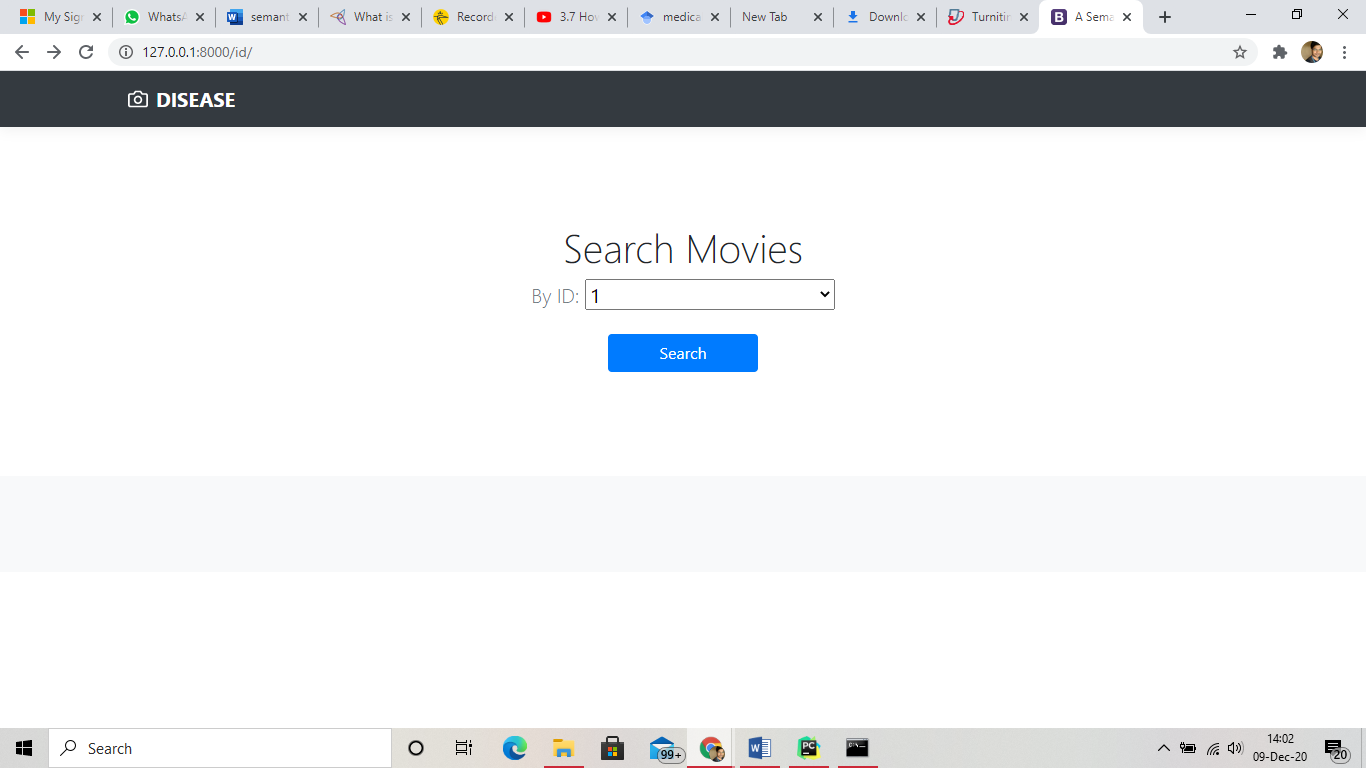
In the hospital management ontology the main goal was to reach a point where all of the hospital systems are integrated and make it understandable even to the new users. One of the widely used software protégé is used for creating ontology. Various queries are tried on the ontology to get specific results using SPARQL and python.

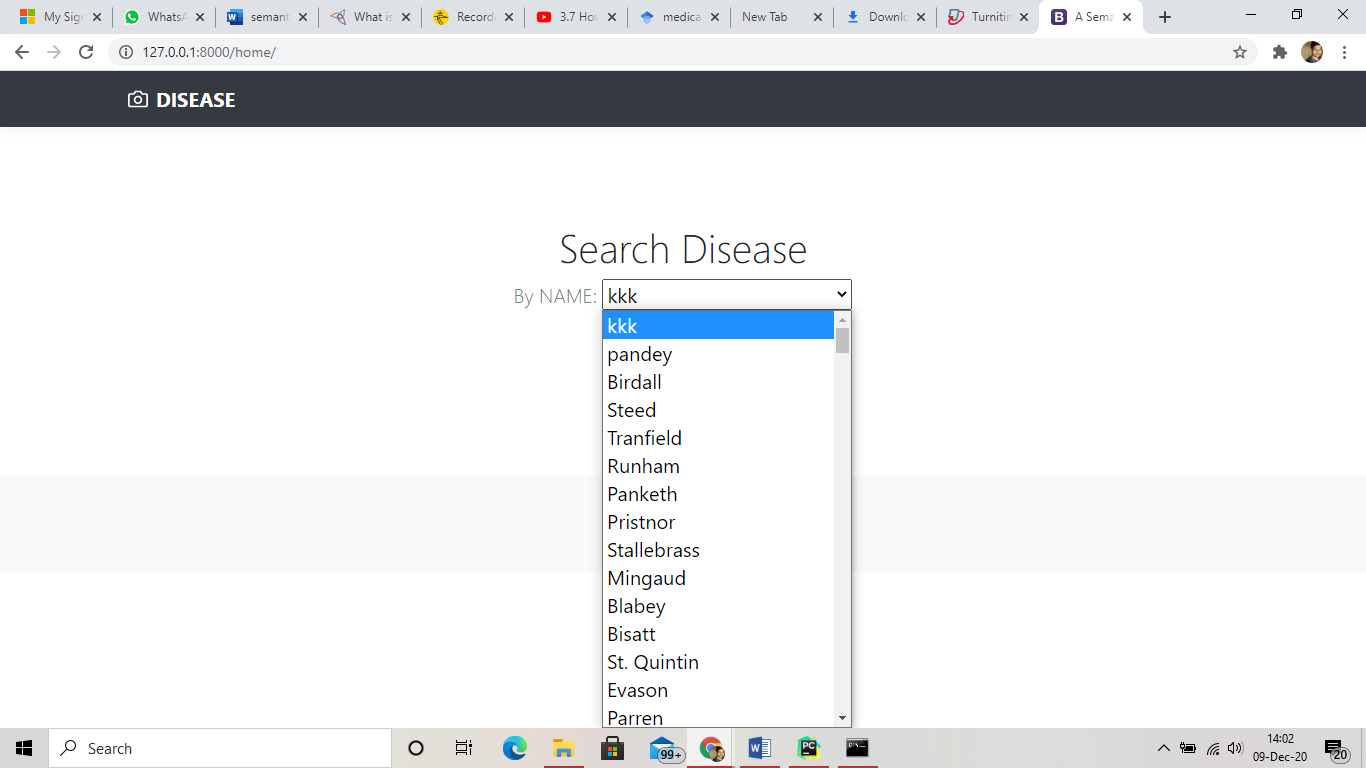
This all benefitted to understand the development of a new hospital system and what the system is lacking in with the current system. Various approaches are tried to make it a less complex system, still it have been noticed that there are chances that we do not reach up to requirements, still there are various points where ontology needs further development. Flaws in the ontologies are discussed when the new system will be used

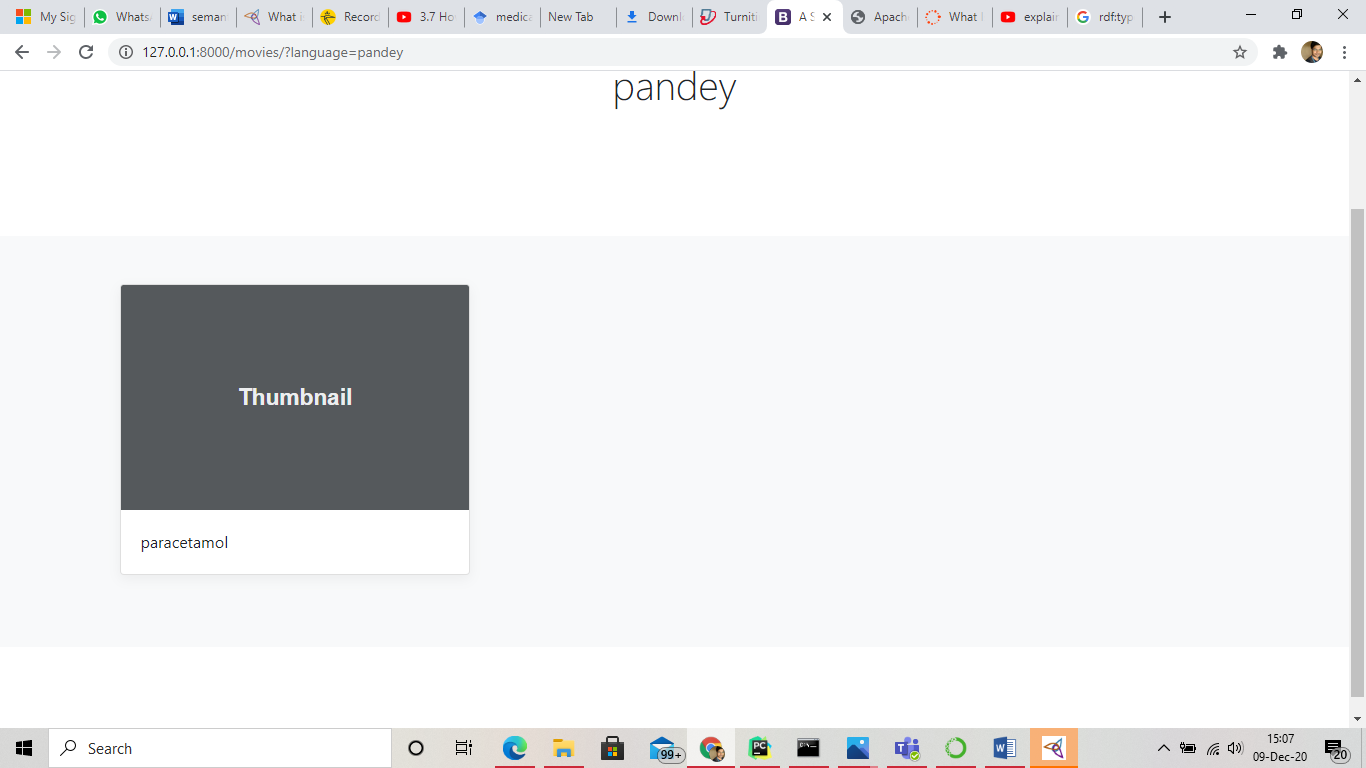
The most important achievement of using the ontology is that it gives an intuition of how the semantic web will look like once the results have been achieved. It can even be concluded that to reach a better management system we need to develop some small but similar system to check if we can achieve the requirements in virtual world.

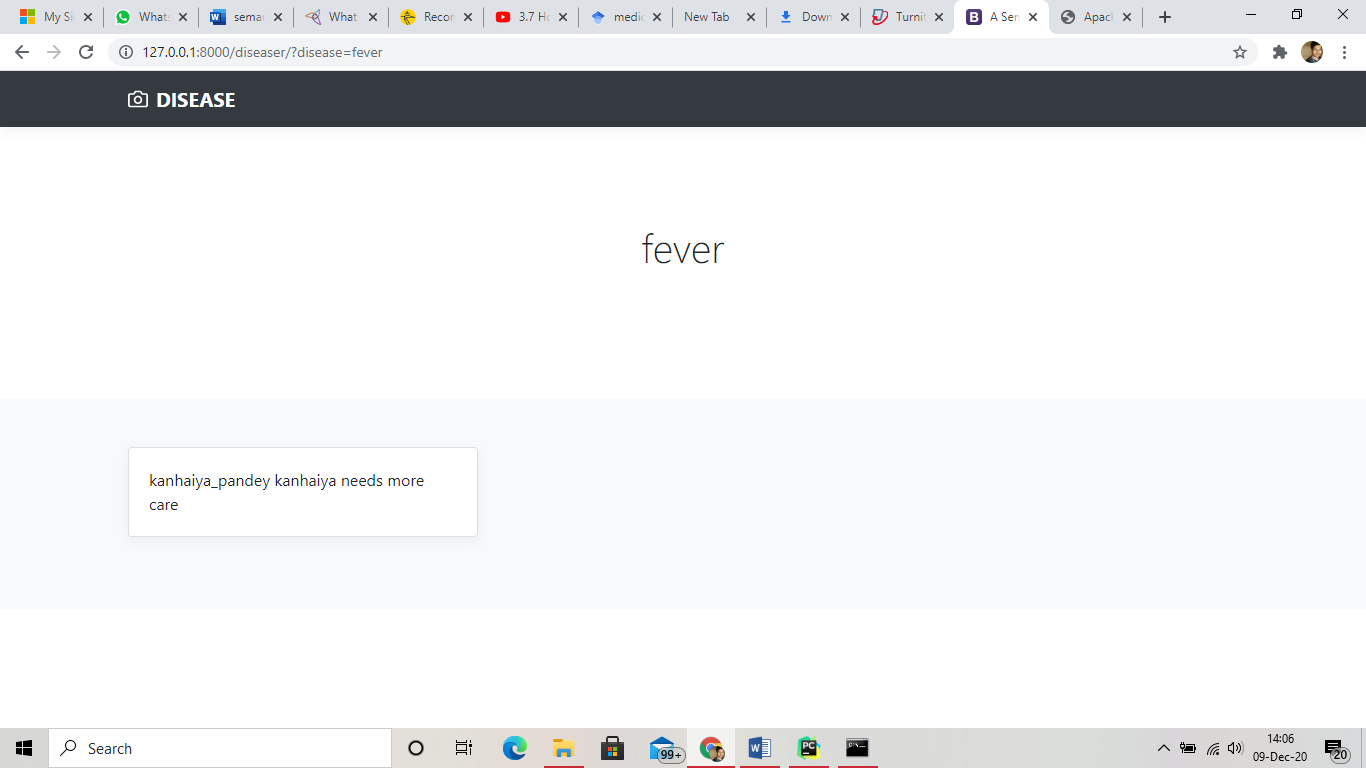
1. **Appendix**

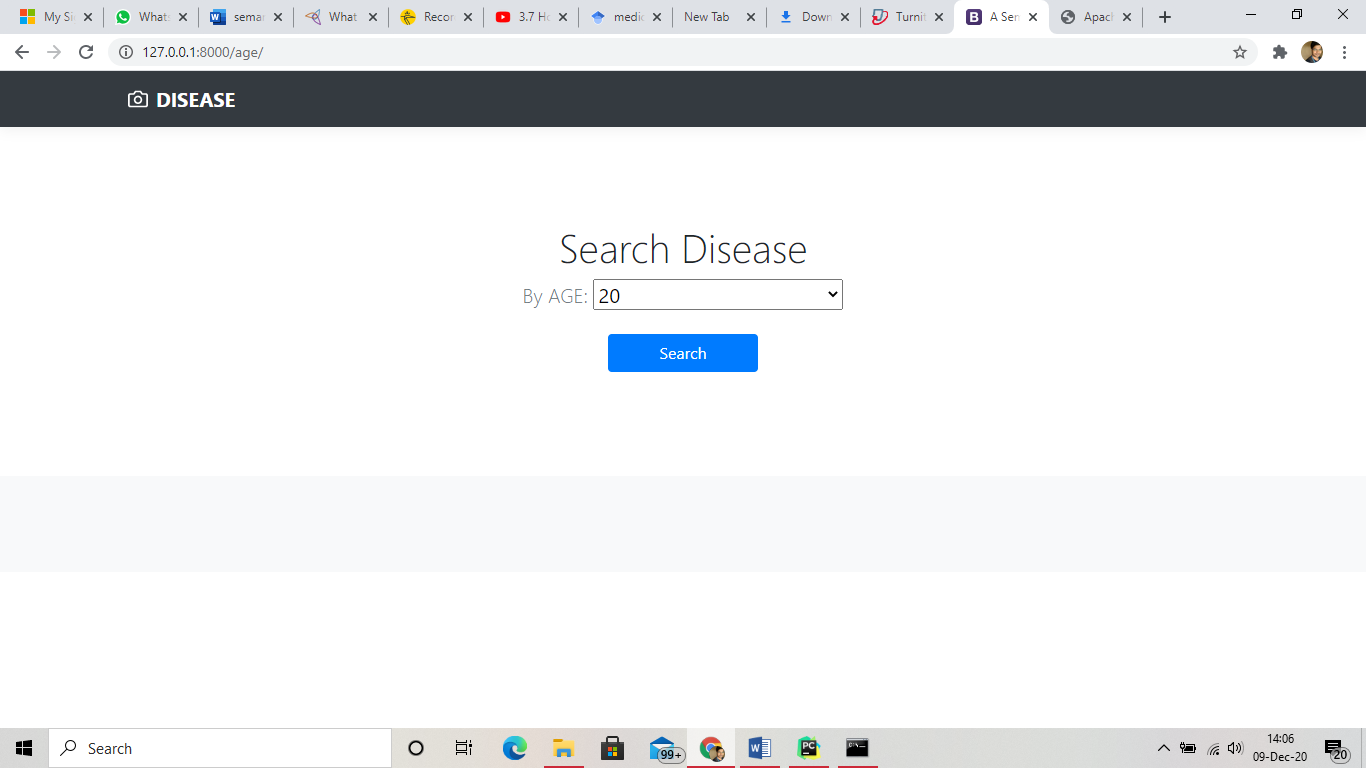
****

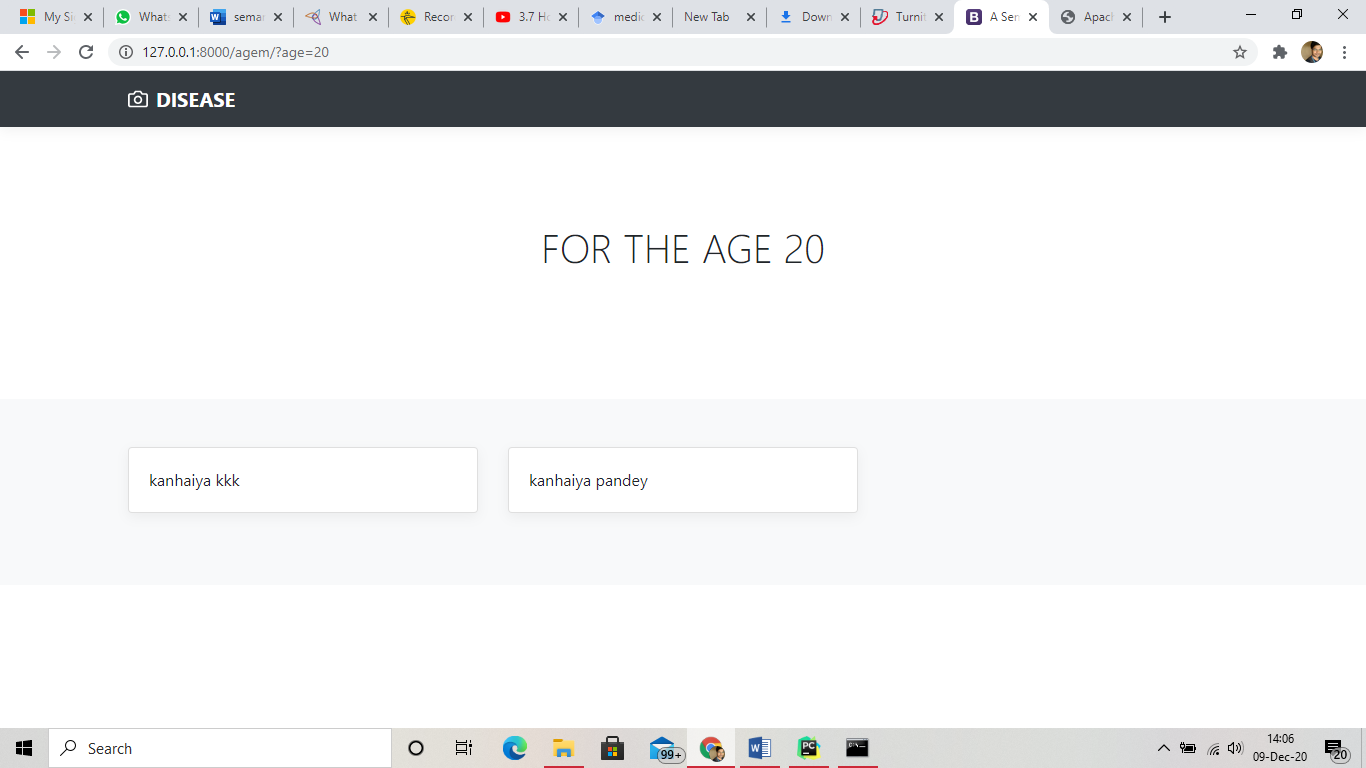
****

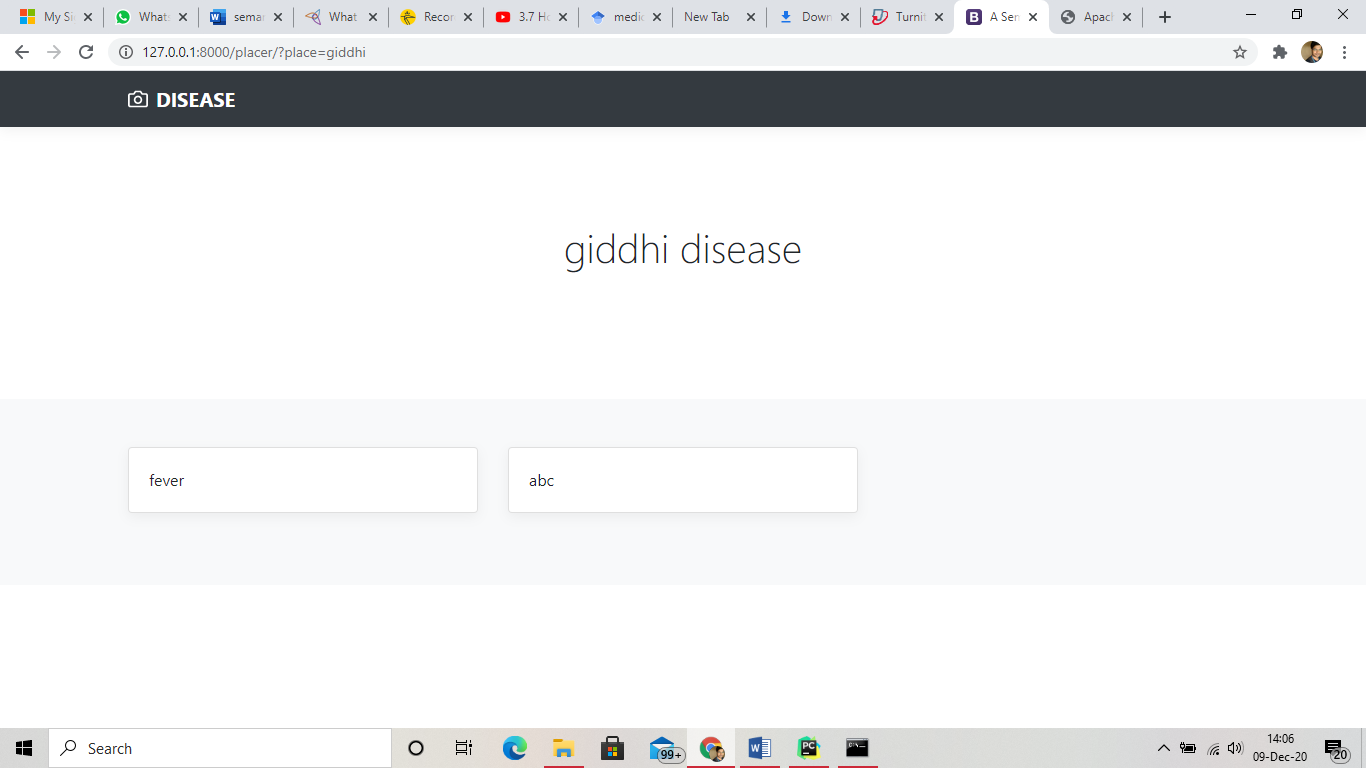
****

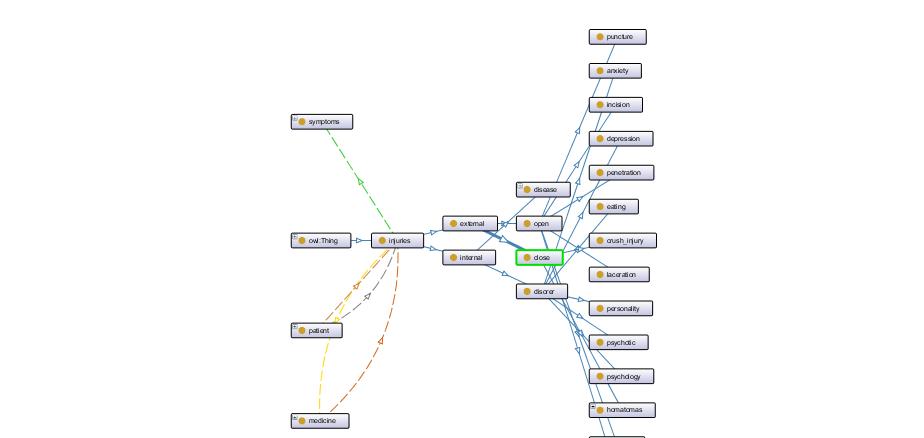


****

****

****

****

****

1. **REFERENCES**

H. Paulheim, “Knowledge graph refinement: A survey of approaches and evaluation methods,” Semantic Web, no. Preprint, pp. 1–20, 2016.

Maureen Donnelly; Giancarlo Guizzardi (2012). Formal Ontology in Information Systems: Proceedings of the Seventh International Conference (FOIS 2012). IOS Press. ISBN 978-1-61499-083-3.

Development 101: A Guide to Creating Your First OntologyNatalya F. Noy  and Deborah L. McGuinnessStanford University

OntHoS – an Ontology for Hospital Scenarios\* Marc Becker1 , Christian Heine2 , Rainer Herrler3, Karl-Heinz Krempels4

1. **Bibliography**

World Wide Web Consortium (W3C), "RDF/XML Syntax Specification (Revised)", 10 Feb. 2004

Uncertainty Reasoning for the World Wide Web". *W3.org*. Retrieved 20 December2018

wikipedia.org/wiki/Semantic\_Web

w3.org/standards/semanticweb

semantic-web.com

<https://www.ontotext.com/knowledgehub/fundamentals/what-is-the-semantic-web>

Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by its inventor, p. 185