Recipe Finder with Semantic Data Technologies

Module Code: MOD004979 SID: 1826585

Academic Year: 2019/2020 Trimester: 2

Table of Contents

1. Introduction

Everybody loves Mouth-watering food, but what about the process that goes into it? Cooking is an art that can make someone else's day memorable by the quality of the food served. There's a lot of people out there who take it as a hobby or profession. But one can't learn to cook on the go, whether the user is a seasoned cook looking for inspiration, or maybe less confident in the kitchen. One needs to have a guide like a cooking mobile app or web app that can make a person walk through the learning process without much fuss. Hence, to help users interested in cooking, an application has been built called “Recipe Finder” using semantic data technologies. The application aims to get customers cooking and guides the cook in detail with carefully written recipes that suit them. The web application provides flexibility to the user to search for top-rated and a variety of recipes. This app is a time saver. By combining title search and ingredients filters, Recipe Finder saves time providing recipes in few clicks and helps users deciding what to choose easily. The user gets step by step instructions to make the recipe with their specified ingredients along with nutritional values, chef name, cuisines, duration, ratings and difficulty level. The application also helps the users to filter recipes according to their allergies.

This report aims at providing an insight into how the application competes with the existing systems. It also contains a design and implementation of how the application is built using semantic data technologies. Finally, the whole idea will be critically analyzed and compared to recipe websites.

2. Concept & Aim

Everyone loves to eat and finding a new recipe online something deeply satisfying. Several websites have been developed to help users to find their favourite recipes to cook and enjoy their meal. Within the website, you can scan and compare meals by class, so it is highly convenient to make a final choice as to which meal to prepare for a given occasion. Every recipe has ingredients to prepare, chef name and step by step instructions that are easy to understand. One such website is BBC Good Food. It has healthy recipes and guides, meal plans, foodie travel inspirations, and the best feature is that the user can search recipes with ingredients. Another popular recipe search website is Yummly. It has a vast content of trending recipes, shows nutrition facts and user reviews for the recipes. Although Recipe Finder is similar to these two websites, the use of semantics for their searches is not implemented where Yummly is trying to implement semantic data technologies in the near future. Recipe finder uses semantic search filters to search meaningful results along with many new features.

The objective of the "Recipe Finder" application is to create a platform for recipes from different cuisines and from top chefs, which in turn creating a huge sized Linked Data database. The main concept is to display recipe results from a different perspective than the usual recipe cooking websites using Semantic data technologies. This enables the machines and also the people to add, change and to implement new relationships between the recipe attributes easily.

One of the main aims of the application is to help users with their health being the main focus. The application allows the users to know the nutritional content of the recipe they find interesting. The users are notified with all the nutritional breakdown as per the ingredients used. This helps the users to understand how the food fits into their daily dietary goals. The application also helps people having food allergies. Many people suffer from certain food allergies and food intolerance. Consuming food which one is allergic to can cause a condition called food hypersensitivity. So to prevent this, the application has allergies filters, where the users can select the allergies and get the non-allergic recipe results.

The application uses the same structure for all the recipes from different chefs, making it easier and more readable for the user. This would help the users to access different varieties of recipes and create a better healthier view for the reader. By applying semantics on a recipe search, it can be used in different areas. One possible usage of the application could be useful for a recipe based website developers to build a similar website and budding chefs who want to make a research on a specific recipe.

3. Design

3.1 Semantic Data Technology

Semantic Technology, together with Linked Data Technology, as proposed by the founder of the World Wide Web, Sir Tim Berners-Lee, creates relationships between data in different formats and sources, from one string to the next, helping to create meaning and links from these relationships. Semantic technology's ultimate aim is to get the machine to understand the information. It links and defines data on the web by developing suitable languages to express rich interrelations of data in a form that machines can process. Hence, machines can not only index huge data and process long strings of characters but can store, manage and retrieve information based on meaning and logical relationships. semantics adds a different dimension to the Internet and can show related facts instead of just matching words. Such embedded semantics with the data offer substantial advantages such as information reasoning and dealing with heterogeneous sources of data.

The core difference between Semantic Technology and other relational databases is that it deals with the meaning rather than the structure of the data. For relational databases, it can be complicated and time-consuming to modify the data model. Each time a new attribute is added to it, it involves a redesign of the entire database. The ever-changing internet, however, needs a more versatile data storage solution. To solve this issue, semantic data technology used a concept called 'triple' to represent a relationship between the subject and an object. The third attribute called predicate is used to link the subject and the object. An example of a triple from the Recipe Finder looks like this:

Recipe1 is_allergic_to Allergies

In this example, Recipe1 is the subject, is_allergic_to is the predicate and Allergies is the object. Subjects and objects are called classes that are equivalent to entities. Predicates are the properties of entities. There are different types of properties such as object property, data property, and annotation property.

To create a dataset from these triples, some concepts need to be understood. Unicode is a standard for encoding international character sets and allows to use (written and read) all human languages on the internet using one uniform type. Resource Identifier (URI) is a structured type string allowing resource recognition to be unique. Resource Description Framework (RDF) is a central information representation format for the semantic web. RDF itself serves as a definition of a triple-formed graph. The standard RDF serialization format is XML in the form of RDF / XML and RDF schema (RDFS) allows a standardized description of ontological constructs. The Web Ontology Language (OWL) is a language derived from the logic of description, offering more constructs through RDFS. OWL has three versions: 1. OWL Lite 2. OWL DL 3.OWL Full. Recipe finder application's ontology uses OWL Full.

A Simple Protocol and RDF Query Language (SPARQL) is available to query RDF data as well as RDFS and OWL ontologies with knowledge bases. SPARQL is a SQL-like language but uses RDF triples and resources to match the search as well as return query results. Since both RDFS and OWL are based on RDF, it is also possible to use SPARQL directly to query ontologies and knowledge bases. Every SPARQL Query has one (or head) Solution Modifier and one Query Body. Similar to other querying languages, SPARQL also has similar kind of query types. It has Read oriented query type like SELECT, CONSTRUCT, DESCRIBE and ASK. The SELECT query will project a query solution in tabular form. The CONSTRUCT query takes the form of an RDF sentence/statement graph specification. The DESCRIBE query akes the form of an RDF sentence/statement graph specification that describes a selection of entities. The ASK query returns a simple boolean (Yes or No) query solution. SPARQL also has Write-oriented Query Types like CREATE, INSERT, ADD, MOVE AND DELETE. Some query modifiers like ORDER BY, GROUP BY and LIMIT are also present. The ORDER BY makes a result sorting based on the parameter given to the result. The GROUP BY clause allows grouping on the properties of the product to be used. The value of the presented result can be limited by the LIMIT.

To make all this information usable and understandable for the computer, it is necessary to describe the items used in the application and make an order. Ontologies should be clearly defined so that the RDF triples are expressed well. Regulations can be extended to objects in ontology, and computers can understand the information presented. The detailed semantic web architecture is shown below.

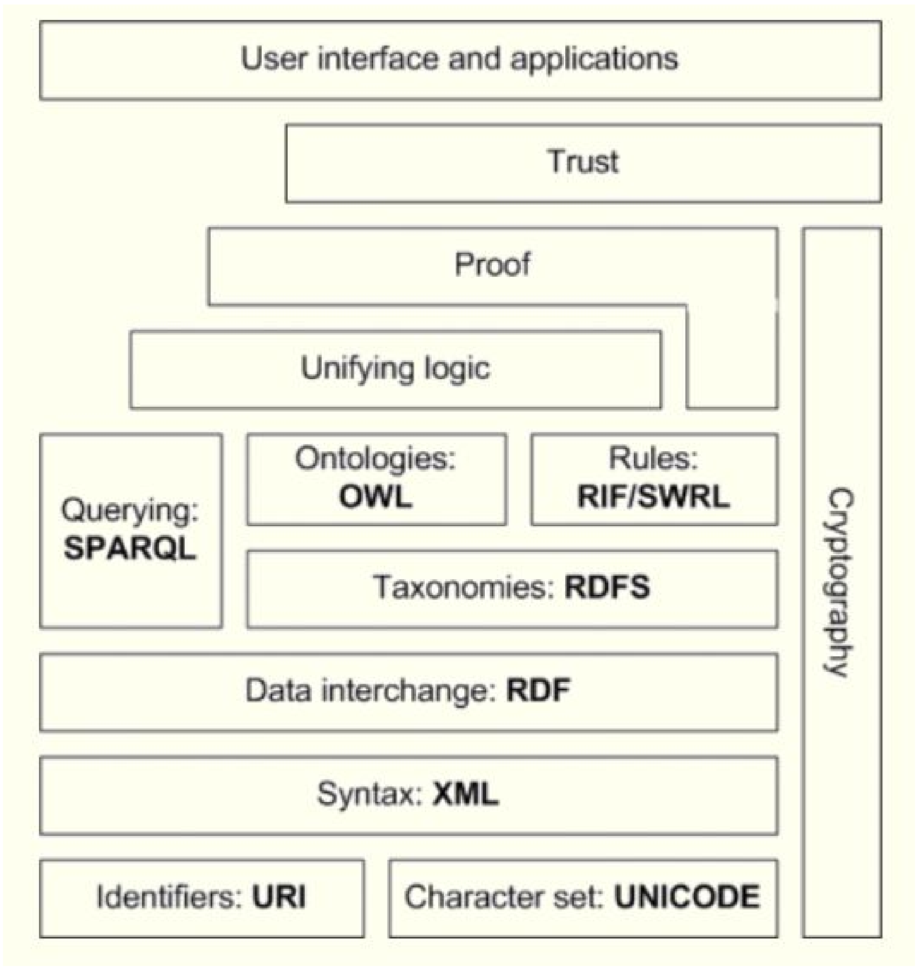
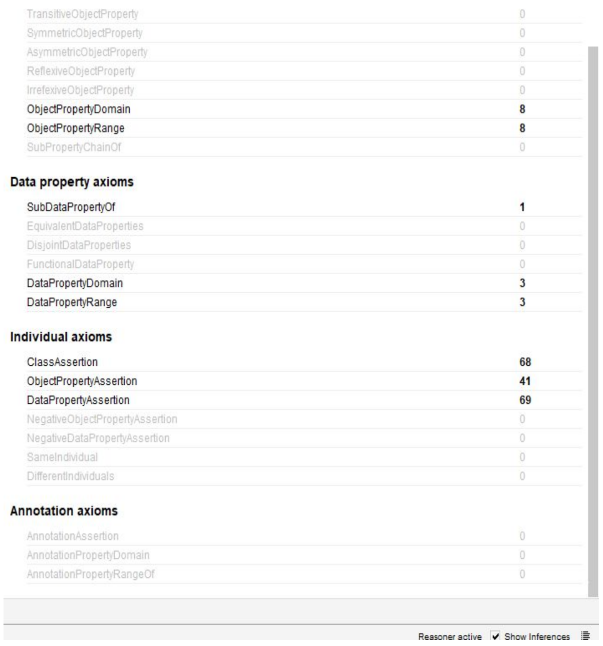


Fig 1. Semantic web architecture in layers

3.2 Ontology for Recipe Finder

Recipe Finder’s ontology is built on Protege. Protege is a free, open-source ontology editor and an intelligent system building platform. It is used to construct domain models and knowledge-based applications with ontologies. It also involves deductive classifiers to test appropriate models and to infer new information based on an ontology analysis. The software was written on java which provides a graphical user interface to develop ontologies. Protégé is being developed at Stanford University and is made available for Linux, Mac OS x and Windows operating systems.



TransitiveObjectProperty	0
SymmetricObjectProperty	0
AsymmetricObjectProperty	0
ReflexiveObjectProperty	0
IrrefexiveObjectProperty	0
ObjectPropertyDomain	8
ObjectPropertyRange	8
SubPropertyChainOf	0

Data property axioms

SubDataPropertyOf	1
EquivalentDataProperties	0
DisjointDataProperties	0
FunctionalDataProperty	0
DataPropertyDomain	3
DataPropertyRange	3

Individual axioms

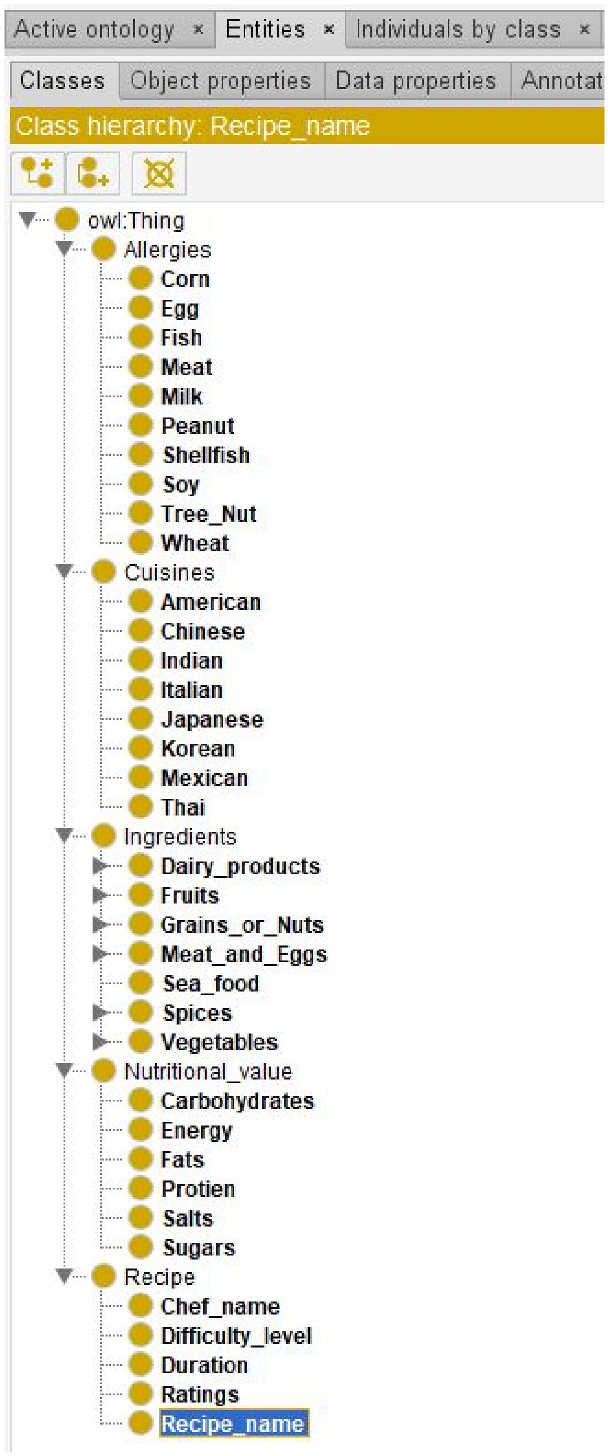
ClassAssertion	68
ObjectPropertyAssertion	41
DataPropertyAssertion	69
NegativeObjectPropertyAssertion	0
NegativeDataPropertyAssertion	0
SameIndividual	0
DifferentIndividuals	0

Annotation axioms

AnnotationAssertion	0
AnnotationPropertyDomain	0
AnnotationPropertyRangeOf	0

Fig 2. Ontology metrics

Figure 2 shows the ontology metrics of Recipe Finder from Protege software. There are 403 axioms, 65 classes, 8 object properties, 4 data properties, and 68 individuals.



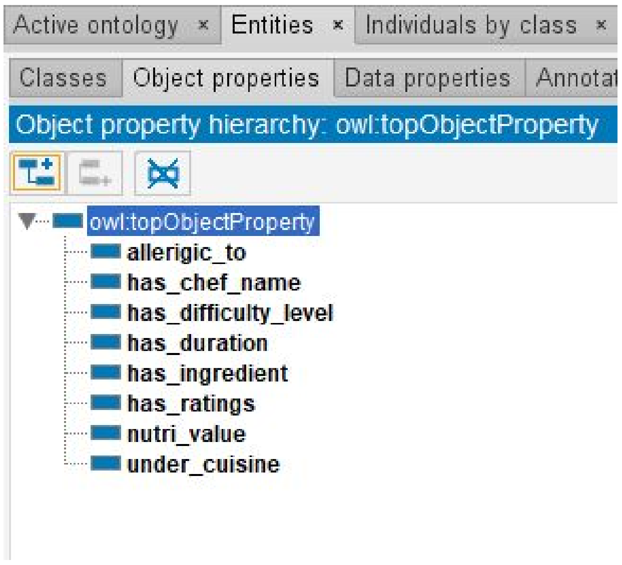
- owl:Thing
  - Allergies
    - Corn
    - Egg
    - Fish
    - Meat
    - Milk
    - Peanut
    - Shellfish
    - Soy
    - Tree_Nut
    - Wheat
  - Cuisines
    - American
    - Chinese
    - Indian
    - Italian
    - Japanese
    - Korean
    - Mexican
    - Thai
  - Ingredients
    - Dairy_products
    - Fruits
    - Grains_or_Nuts
    - Meat_and_Eggs
    - Sea_food
    - Spices
    - Vegetables
  - Nutritional_value
    - Carbohydrates
    - Energy
    - Fats
    - Protien
    - Salts
    - Sugars
  - Recipe
    - Chef_name
    - Difficulty_level
    - Duration
    - Ratings
    - Recipe_name

Fig 3. Class hierarchy

Figure 3 shows the class hierarchy of Recipe Finder which has its superclass and respective subclasses. The main classes are *Allergies*, *Cuisines*, *Ingredients*, *Nutritional_value and Recipe*. *Allergies* represent food allergies such as meat allergies, milk allergies that are caused by the recipe ingredients. *Cuisines* represent the different categories of recipes based on geographical locations all over the world like Indian, Chinese, American, etc. The *Ingredients* represent the items that are used in the preparation of the recipe such as chicken, salts, lime, etc. *Nutritional_value* indicates the dietary information of the recipe in quantities such as energy in kcal, protein in grams. The *Recipe* contains recipe information like the recipe names, who cooked the food, how long would it take to cook the recipe, how difficult is it to prepare it, the ratings of the recipe.

The subclasses of *Allergies* are *Corn, Egg, Fish, Meat, Milk, Peanut, Shellfish, Soy, Tree_Nut, and Wheat*. The subclasses of *Cuisines* are *American, Chinese, Indian, Italian, Japanese, Mexican and Thai*. The subclasses of *Ingredients* are *Dairy_products, Fruits, Grains_and_Nuts, Meat_and_Eggs, Sea_food, Spices, Vegetables*. The subclasses of *Nutritional_value* are *Carbohydrates, Energy, Fats, Protein, Salts, Sugars*. The subclasses of *Recipe* are *Chef_name, Difficulty_level, Duration, Ratings, and Recipe_name*.

The next important thing in the ontology is the object properties. Classes are related to each other by object properties. Each of the object property must have a domain and ranges. This relationship will form a triple in the ontology.



- owl:topObjectProperty
  - allerigic_to
  - has_chef_name
  - has_difficulty_level
  - has_duration
  - has_ingredient
  - has_ratings
  - nutri_value
  - under_cuisine

Fig 4. Object Properties

The Recipe Finder's ontology has 8 object properties as shown in the figure. The object properties are *allerigic_to*, *has_chef_name*, *has_difficulty_level*, *has_duration*, *has_ingredient*, *has_ratings*, *nutri_value* and *under_cuisine*. All the object properties of Recipe Finder has the domain *Recipe_name*. Each of the object property has different ranges such as *allerigic_to* has a domain *Recipe_name* and a range *Allergies*, *has_chef_name* has a domain *Recipe_name* and a range *Chef_name*, *has_difficulty_level* has a domain *Recipe_name* and a range *Difficulty_level*, *has_duration* has a domain *Recipe_name* and a range *Duration*, *has_ingredient* has a domain *Recipe_name* and a range *Ingredients*, *has_ratings* has a domain *Recipe_name* and a range *Ratings*, *nutri_value* has a range *recipe_name* and a range *Nutritional_value* and *under_cuisine* has a range *Recipe_name* and a range *Cuisines*.

Another important thing in an ontology is data properties. Data properties can have asserted mode or the inferred mode where one can create in an asserted mode for default. There are 3 data properties in the Recipe Finder application as shown in figure 5. The data properties are *name*, *recipe_name*, and *recipe_procedure*. The name data property is the name of every individual. It is a sub-property of owl:topdataproperty. It has the domain owl:Thing and a range xsd:string. The recipe_name represents the name od the recipes. It has a range Recipe_name and a range xsd:string. The last data property is the recipe_procedure which represents a step by step instructions on how to make a recipe. It has a range Recipe_name and a range xsd:string.

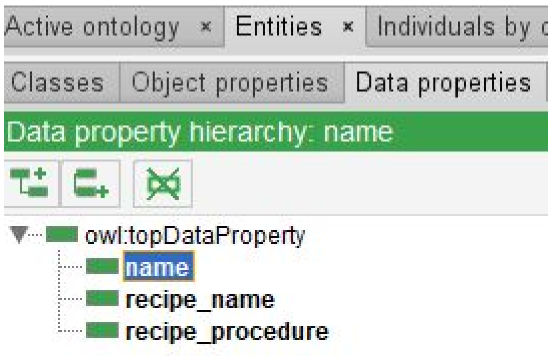
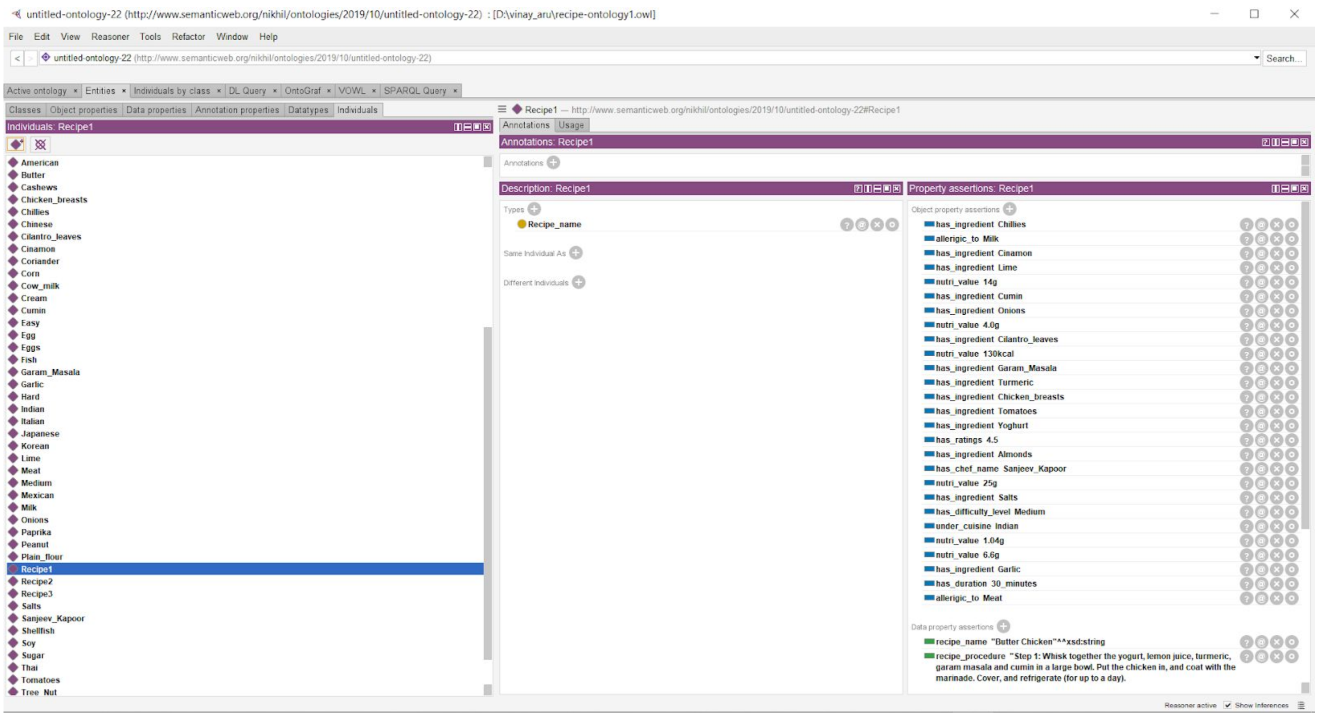


Fig 5. Data Properties

One of the main attributes in creating an ontology is individuals. Figure 6 shows the different kinds of individuals used for building the ontology.



untitled-ontology-22 (http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22) : [D:\vinay_aru\recipe-ontology1.owl]

untitled-ontology-22 (http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22)

Individuals: Recipe1

American
Butter
Cashews
Chicken_breasts
Chillies
Chinese
Cilantro_leaves
Cinamon
Coriander
Corn
Cow_milk
Cream
Cumin
Easy
Egg
Eggs
Fish
Garam_Masala
Garlic
Hard
Indian
Italian
Japanese
Korean
Lime
Meat
Medium
Mexican
Milk
Onions
Paprika
Peanut
Plain_flour
Recipe1
Recipe2
Recipe3
Salts
Sanjeev_Kapoor
Shellfish
Soy
Sugar
Thai
Tomatoes
Tree_Nut

Recipe1 — http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#Recipe1

Annotations: Recipe1

Description: Recipe1

Types: Recipe_name

Property assertions: Recipe1

Object property assertions

has_ingredient Chillies
allerigic_to Milk
has_ingredient Cinamon
has_ingredient Lime
nutri_value 14g
has_ingredient Cumin
has_ingredient Onions
nutri_value 4.0g
has_ingredient Cilantro_leaves
nutri_value 130kcal
has_ingredient Garam_Masala
has_ingredient Turmeric
has_ingredient Chicken_breasts
has_ingredient Tomatoes
has_ingredient Yoghurt
has_ratings 4.5
has_ingredient Almonds
has_chef_name Sanjeev_Kapoor
nutri_value 25g
has_ingredient Salts
has_difficulty_level Medium
under_cuisine Indian
nutri_value 1.04g
nutri_value 6.6g
has_ingredient Garlic
has_duration 30_minutes
allerigic_to Meat

Data property assertions

recipe_name "Butter Chicken"^^xsd:string
recipe_procedure "Step 1: Whisk together the yogurt, lemon juice, turmeric, garam masala and cumin in a large bowl. Put the chicken in, and coat with the marinade. Cover, and refrigerate (for up to a day).

Fig. 6 Individuals

Every subclass may have instances. The instances can have one or more instances or individuals. The selected Recipe1 has multiple object properties. The individual has different values under the same object property. It also has data properties. The below-given figure represents the Recipe1 individual's attributes. It clearly shows which object properties and data properties the individual is having. The figure also shows how the Recipe1 individual is related to the classes.

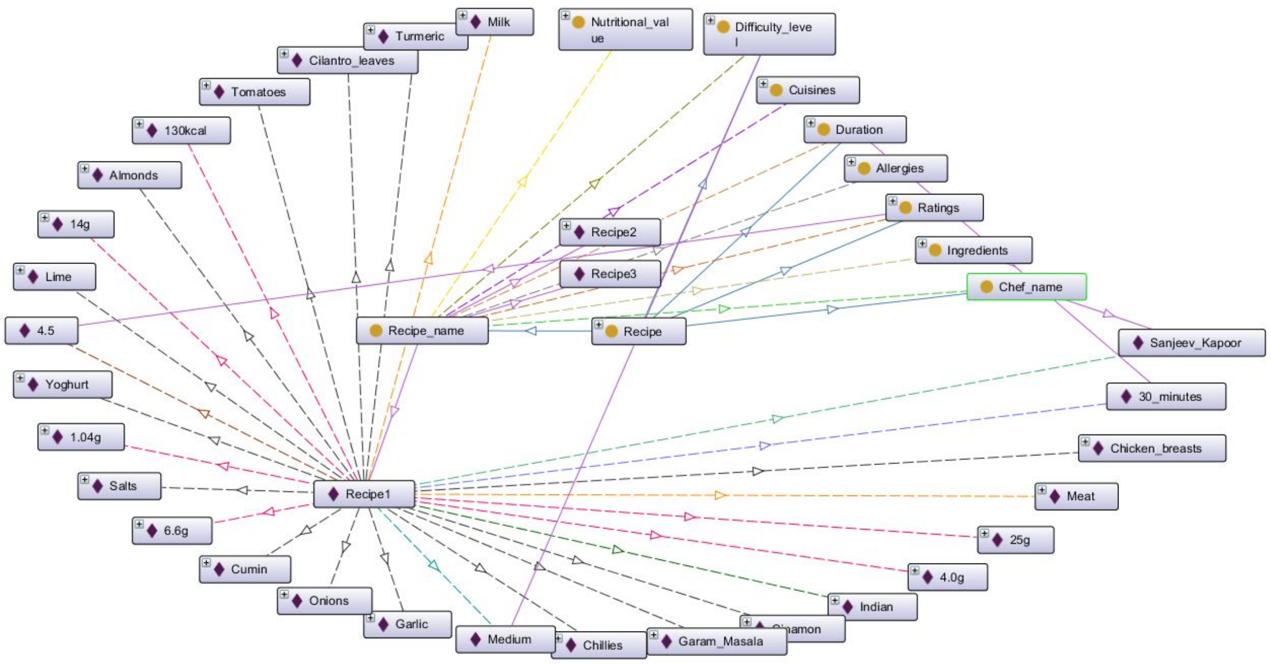


Fig. 7 Recipe Individual

4. Implementation

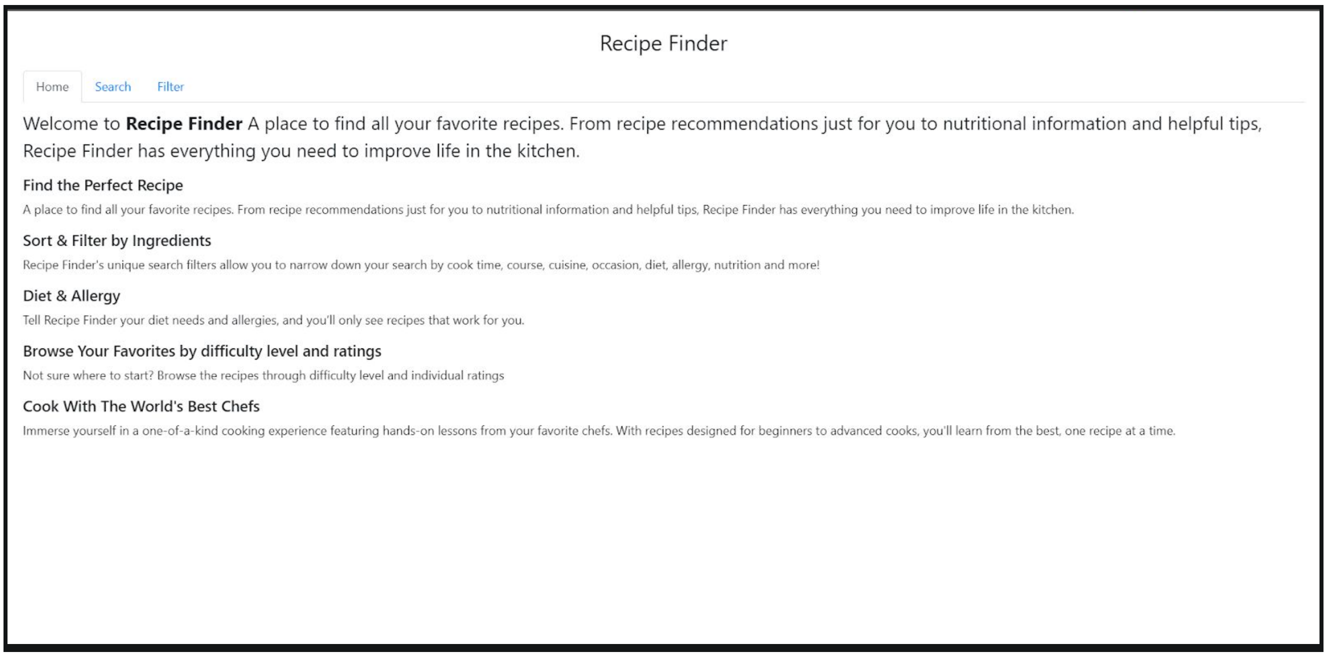
This section of the report introduces the techniques used in the implementation of the web application. It gives a detailed description of how semantic data technologies and other technologies are used for easy implementation.

4.1 Environment Setup

The programming languages utilized in the development of the application are PHP and HTML. XAMPP was used to interact with the Apache server which is basically a localhost or a local server. The SPARQL queries are requested and fetched from the SPARQL endpoint. Fuseki, a SPARQL server is utilized in web app development.

4.2 End-User Features

4.2.1 Home Tab



Welcome to **Recipe Finder** A place to find all your favorite recipes. From recipe recommendations just for you to nutritional information and helpful tips, Recipe Finder has everything you need to improve life in the kitchen.

Find the Perfect Recipe
A place to find all your favorite recipes. From recipe recommendations just for you to nutritional information and helpful tips, Recipe Finder has everything you need to improve life in the kitchen.

Sort & Filter by Ingredients
Recipe Finder's unique search filters allow you to narrow down your search by cook time, course, cuisine, occasion, diet, allergy, nutrition and more!

Diet & Allergy
Tell Recipe Finder your diet needs and allergies, and you'll only see recipes that work for you.

Browse Your Favorites by difficulty level and ratings
Not sure where to start? Browse the recipes through difficulty level and individual ratings

Cook With The World's Best Chefs
Immerse yourself in a one-of-a-kind cooking experience featuring hands-on lessons from your favorite chefs. With recipes designed for beginners to advanced cooks, you'll learn from the best, one recipe at a time.

Fig 8. Home Tab

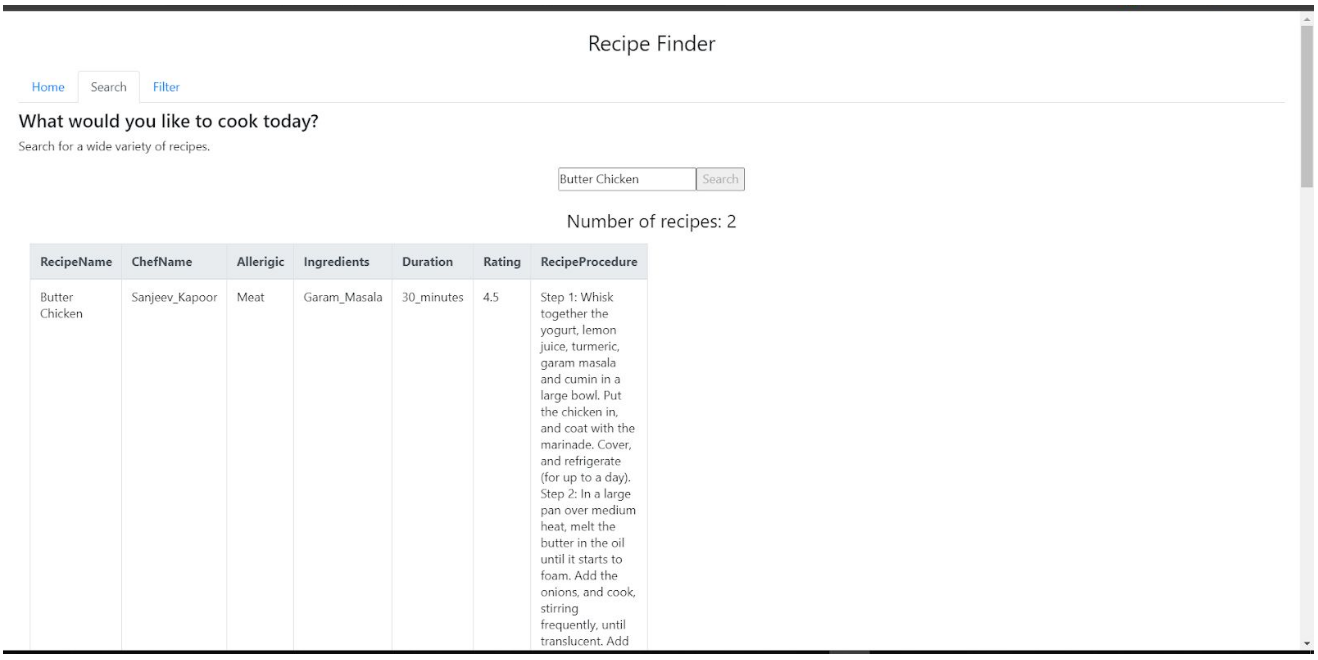
Recipe Finder application consists of three main tabs. The first tab is the Home tab. The home tab shows a welcoming message to the user. The tab gives a brief introduction to the application and its features. The features include finding the perfect recipe, sort, and filter by ingredients, diet, and allergies, browse recipes by difficulty level and ratings and search for favourite chef recipes.

From this, the user gets an overall idea of the application and excites them to explore the website. The code snippet for the home tab is given below:

```
<div class="bs-example">
    <ul class="nav nav-tabs">
        <li class="nav-item">
            <a href="#home" class="nav-link active"
data-toggle="tab">Home</a>
        </li>
        <li class="nav-item">
            <a href="#profile" class="nav-link" data-toggle="tab">Search</a>
        </li>
        <li class="nav-item">
            <a href="#messages" class="nav-link"
data-toggle="tab">Filter</a>
        </li>
    </ul>
      <div class="tab-content">
        <div class="tab-pane fade show active  " id="home">
            <h5 class="mt-2"></h5>
            <p><font size="5">Welcome to <b>Recipe Finder</b> A place to
find all your favorite recipes. From recipe recommendations just for you to
nutritional information and helpful tips, Recipe Finder has everything you
need to improve life in the kitchen.</font></p>
                 <h5>Find the Perfect Recipe</h5>
                 <p>A place to find all your favorite recipes. From recipe
recommendations just for you to nutritional information and helpful tips,
Recipe Finder has everything you need to improve life in the kitchen.</p>
                 <h5>Sort & Filter by Ingredients</h5>
                 <p>Recipe Finder's unique search filters allow you to
narrow down your search by cook time, course, cuisine, occasion, diet,
allergy, nutrition and more!</p>
                 <h5>Diet & Allergy</h5>
                 <p>Tell Recipe Finder your diet needs and allergies, and
you’ll only see recipes that work for you.</p>
                 <h5>Browse Your Favorites by difficulty level and
ratings</h5>
                 <p>Not sure where to start? Browse the recipes through
difficulty level and individual ratings</p>
                 <h5>Cook With The World's Best Chefs</h5>
                 <p>Immerse yourself in a one-of-a-kind cooking experience
featuring hands-on lessons from your favorite chefs. With recipes designed
for beginners to advanced cooks, you'll learn from the best, one recipe at a
time.</p>
        </div>
```

Code snippet for Home tab

4.2.2 Search Tab:



What would you like to cook today?

Search for a wide variety of recipes.

Number of recipes: 2

RecipeName	ChefName	Allerigic	Ingredients	Duration	Rating	RecipeProcedure
Butter Chicken	Sanjeev_Kapoor	Meat	Garam_Masala	30_minutes	4.5	Step 1: Whisk together the yogurt, lemon juice, turmeric, garam masala and cumin in a large bowl. Put the chicken in, and coat with the marinade. Cover, and refrigerate (for up to a day). Step 2: In a large pan over medium heat, melt the butter in the oil until it starts to foam. Add the onions, and cook, stirring frequently, until translucent. Add

Fig 9. Search Tab

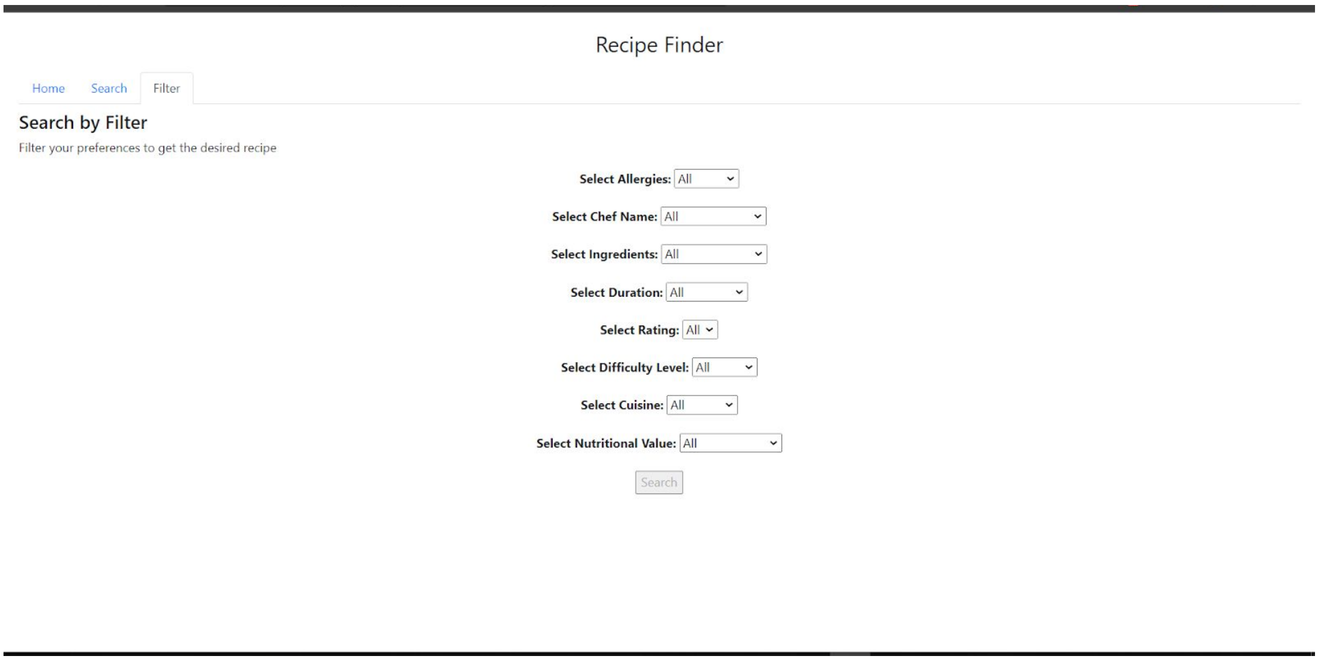
The next tab in the application is the Search tab. The search tab consists of a search bar where you can search for a wide variety of recipes. The search function is intelligent as it fetches you the results precisely as per your request. One can search for a recipe with just an ingredient name, a chef name, cuisines, duration, difficulty level, allergies, and nutritional content. The results of the search query are displayed under the search bar itself in a tabular format. The tabular format of displaying all the content pertaining to the recipe is chosen as it would be appealing to the user to watch every detail of the recipe at one glance. The code snippet for the Search tab is given below:

```
<div class="tab-pane fade" id="profile">
            <h4 class="mt-2">What would you like to cook today?</h4>
            <p>Search for a wide variety of recipes.</p>
                 <div class='col-6 offset-3 text-center'><form
method="GET"><input type="text" name="x" placeholder="Recipe Name" /><input
type="submit" value="Search" /></form></div>
                      <h4 class='h4 mb-3 mt-4 font-weight-normal
```

```
text-center'>Number of recipes: <?php echo sparql_num_rows( $result );
?></h4>
                              <div class='col-4 '><table class='table
table-bordered table-hover'></div>
                              <thead class='thead-light'><tr></tr>


        <?php
                foreach( $fields as $field )
                {
                        print "<th>$field</th>";
                }
                print "</tr></thead><tbody>";
                while( $row = sparql_fetch_array( $result ) )
                {
                        print "<tr>";
                        foreach( $fields as $field )
                        {
                                print "<td>".str_replace($fa, "", $row[$field]
)."</td>";
                        }
                        print "</tr>";
                }
                print "";
        ?>
```

Code snippet for Search Tab

4.2.3 Filter Tab:


Recipe Finder

Home Search Filter

Search by Filter

Filter your preferences to get the desired recipe

Select Allergies: All

Select Chef Name: All

Select Ingredients: All

Select Duration: All

Select Rating: All

Select Difficulty Level: All

Select Cuisine: All

Select Nutritional Value: All

Search

Fig 10. Filter Tab

The next tab in the application is the Filter tab. The filter tab is just like a search tab but the user is able to filter his requests based on their choices. The filters can range from a variety of recipe details. The user can filter recipes using allergies, chef names, ingredients, duration of recipe making, ratings, difficulty level, cuisines, and nutritional content. The search button sends the request in the form of the SPARQL query and fetches back the values. The results are displayed on the search tab in a tabular format. The code for the Search tab is given below; along with the query request sending from an allergies filter:

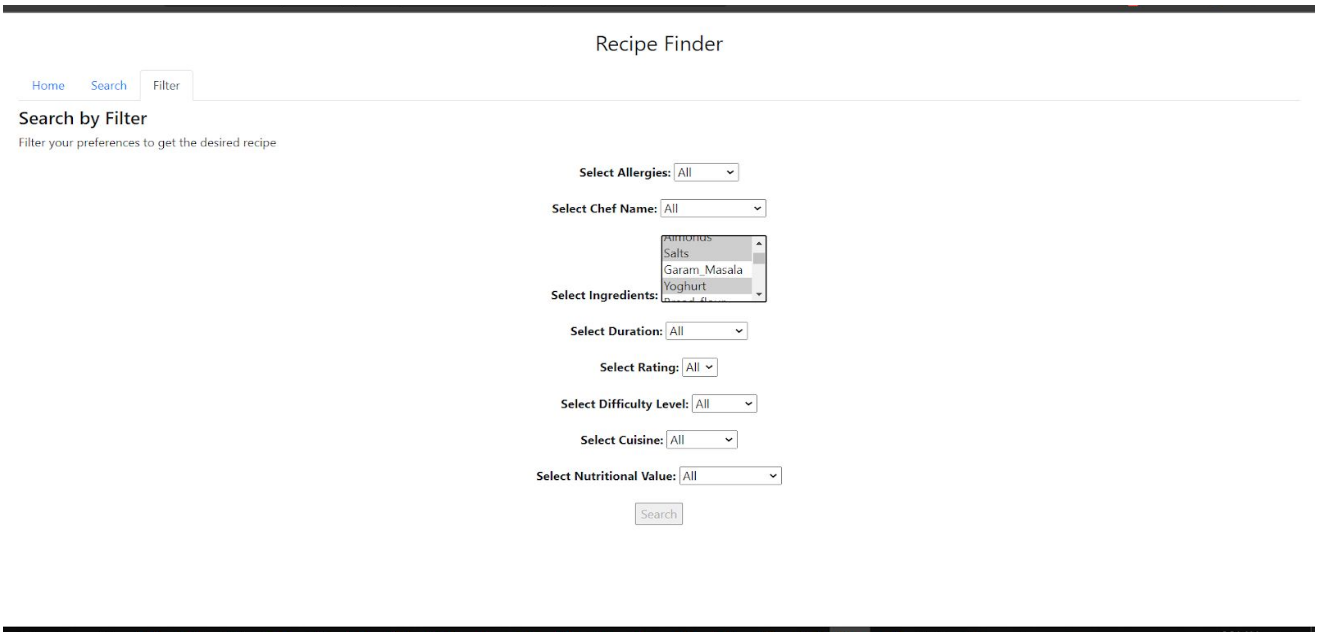
```
<div class="tab-pane fade" id="messages">
            <h4 class="mt-2">Search by Filter</h4>
            <p>Filter your preferences to get the desired recipe</p>
                 <div class='col-6 offset-3 text-center'><form method="GET">
<?php

$sparql1 = "SELECT  ?Allerigic  WHERE {     ?recipe foaf:allerigic_to
?Allerigic .  }  GROUP BY ?Allerigic ";

//echo $sparql;
$result1 = sparql_query( $sparql1 );
if( !$result1 ) { print sparql_errno() . ": " . sparql_error(). "\n"; exit;
}

$fields1 = sparql_field_array( $result1 );
?>
<b>Select Allergies:</b>
<select name="Allergies" id="a">
</div>
<?php
foreach( $fields1 as $field1 )
{

}
print "<option selected='selected' value='ALL'>All</option>";
while( $row1 = sparql_fetch_array( $result1 ) )
{

      foreach( $fields1 as $field1 )
      {
           print "<option  value='".str_replace($fa, "", $row1[$field1]
)."'>".str_replace($fa, "", $row1[$field1] )."</option>";
      }

}

?>
</select>
```

```
<br>
<br>
```

Code snippet for Filter Tab

The query request for other filters like chef name, duration, ingredients, ratings, difficulty level, cuisines, and nutritional value will be in a similar manner.

The ingredient values in the ingredients filter can be selected multiple times. This helps the user to choose a recipe with the ingredients they have. The multiple selections can be done by holding the Ctrl button and left-clicking on the ingredients. The below image shows how it is reflected on the front end:


Recipe Finder

Home Search Filter

Search by Filter

Filter your preferences to get the desired recipe

Select Allergies: All

Select Chef Name: All

Select Ingredients: Almonds, Salts, Garam_Masala, Yoghurt

Select Duration: All

Select Rating: All

Select Difficulty Level: All

Select Cuisine: All

Select Nutritional Value: All

Search

Fig 11. Ingredients filter with multiple selections

The code snippet for ingredients filter with multiple selections is given below:

```
<b>Select Ingredients:</b>
<select name="Ingredient" id="i" multiple>
```

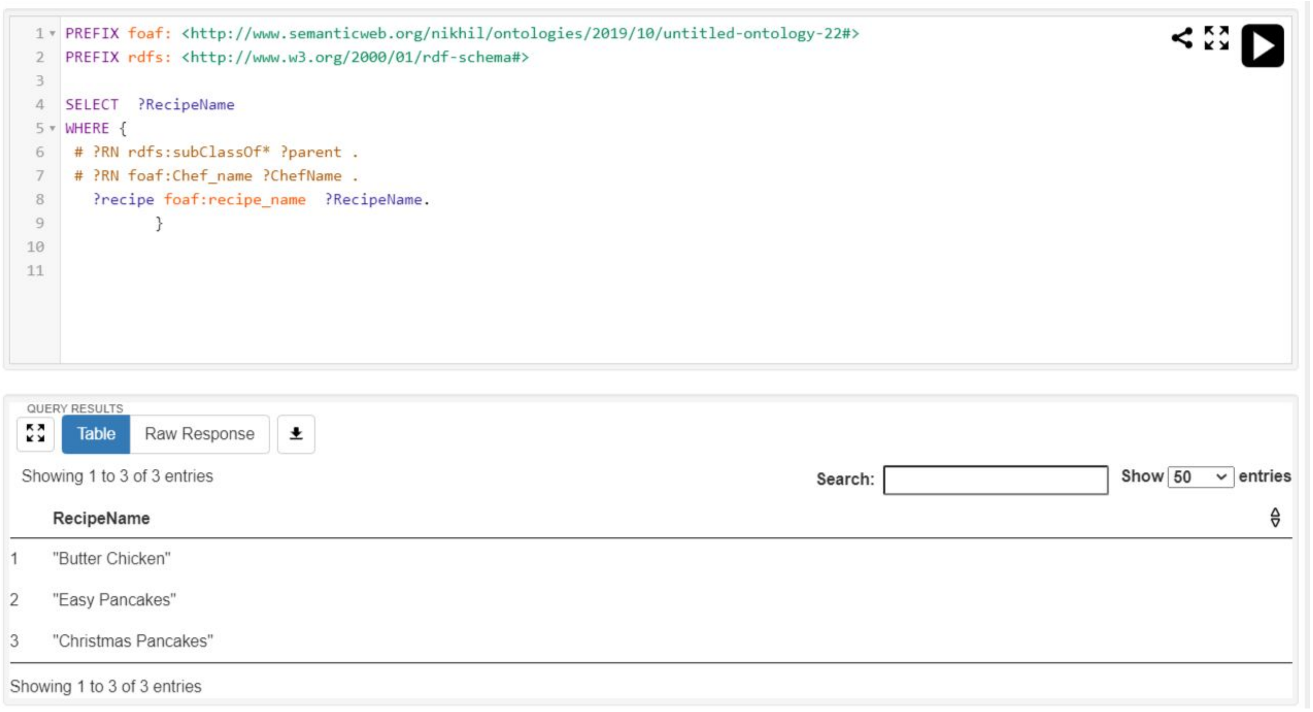
Code for multiple selections in Ingredients

5. Evaluation and Use

The SPARQL queries used in the application are discussed in this section:

5.1 Query for all recipes:

The query to list all the names of the recipes present in the ontology.



```
PREFIX foaf: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?RecipeName
WHERE {
 # ?RN rdfs:subClassOf* ?parent .
 # ?RN foaf:Chef_name ?ChefName .
   ?recipe foaf:recipe_name  ?RecipeName.
         }
```

QUERY RESULTS

Table Raw Response

Showing 1 to 3 of 3 entries

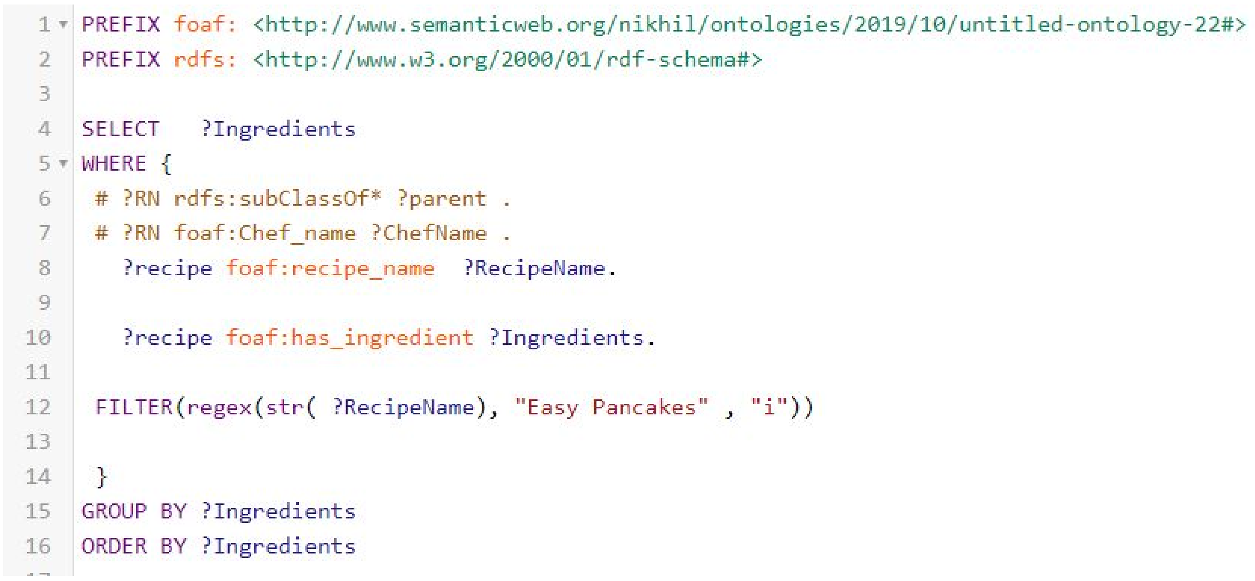
Search: Show 50 entries

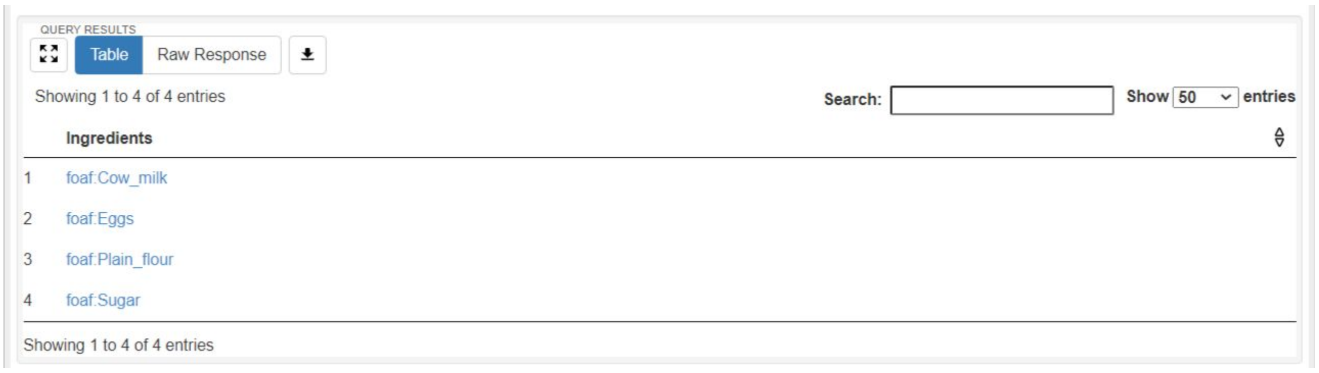
	RecipeName
1	"Butter Chicken"
2	"Easy Pancakes"
3	"Christmas Pancakes"

Showing 1 to 3 of 3 entries

5.2 Query for ingredients:

The query to show all the ingredients used in the making of "Easy Pancakes" recipe.


```
PREFIX foaf: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT   ?Ingredients
WHERE {
 # ?RN rdfs:subClassOf* ?parent .
 # ?RN foaf:Chef_name ?ChefName .
   ?recipe foaf:recipe_name  ?RecipeName.

   ?recipe foaf:has_ingredient ?Ingredients.

 FILTER(regex(str( ?RecipeName), "Easy Pancakes" , "i"))

 }
GROUP BY ?Ingredients
ORDER BY ?Ingredients
```


QUERY RESULTS

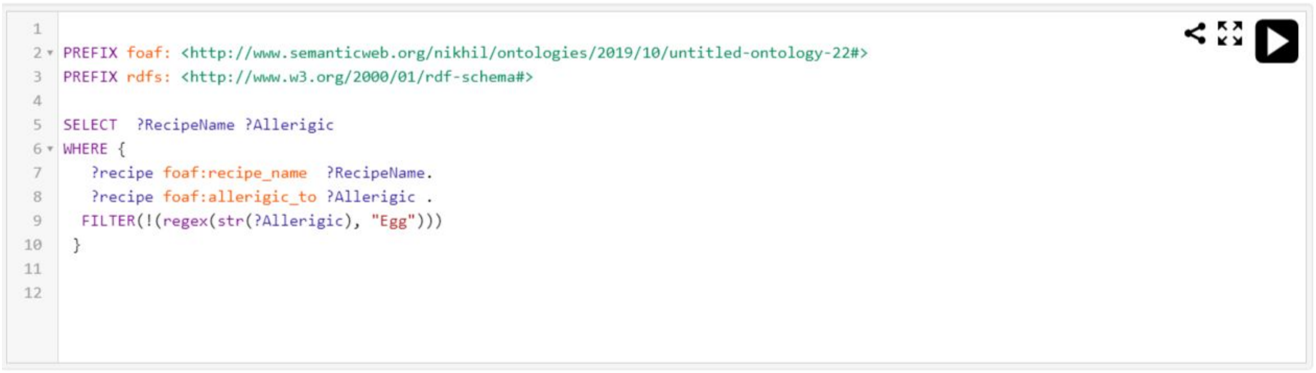
Showing 1 to 4 of 4 entries

	Ingredients
1	foaf:Cow_milk
2	foaf:Eggs
3	foaf:Plain_flour
4	foaf:Sugar

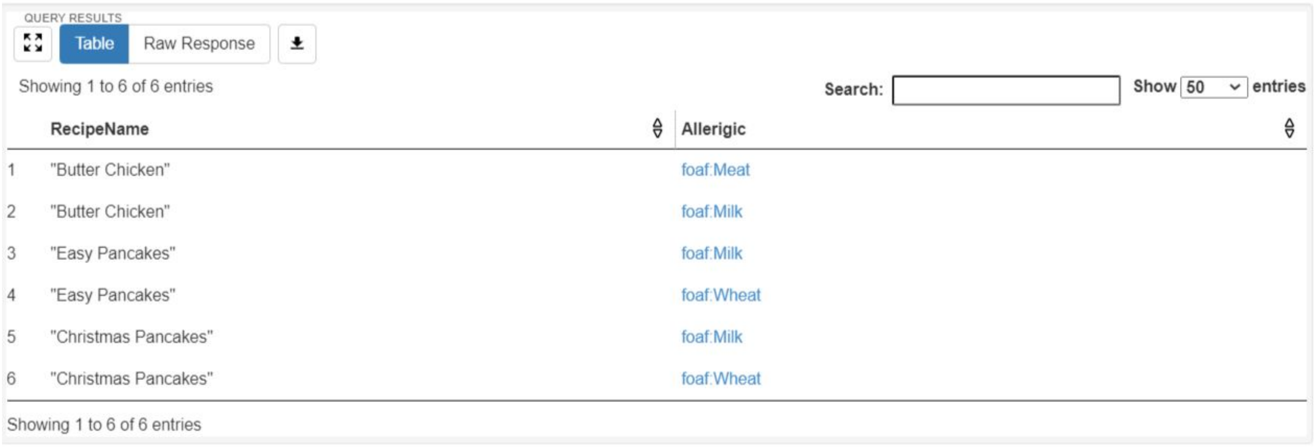
Showing 1 to 4 of 4 entries

5.3 Query for allergies:

The query to show all the allergies that are present in every recipe. Note that when a user chooses a recipe with an allergy, the application should fetch them the recipes without that allergy. This can be done by using a negate function in the FILTER statement. The "!" symbol converts whatever the FILTER statement result is into a False result. Thus giving us the allergies without the name specified. This helps the users to browse through the recipes with allergy-free.



```
PREFIX foaf: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?RecipeName ?Allerigic
WHERE {
   ?recipe foaf:recipe_name  ?RecipeName.
   ?recipe foaf:allerigic_to ?Allerigic .
  FILTER(!(regex(str(?Allerigic), "Egg")))
 }
```



QUERY RESULTS

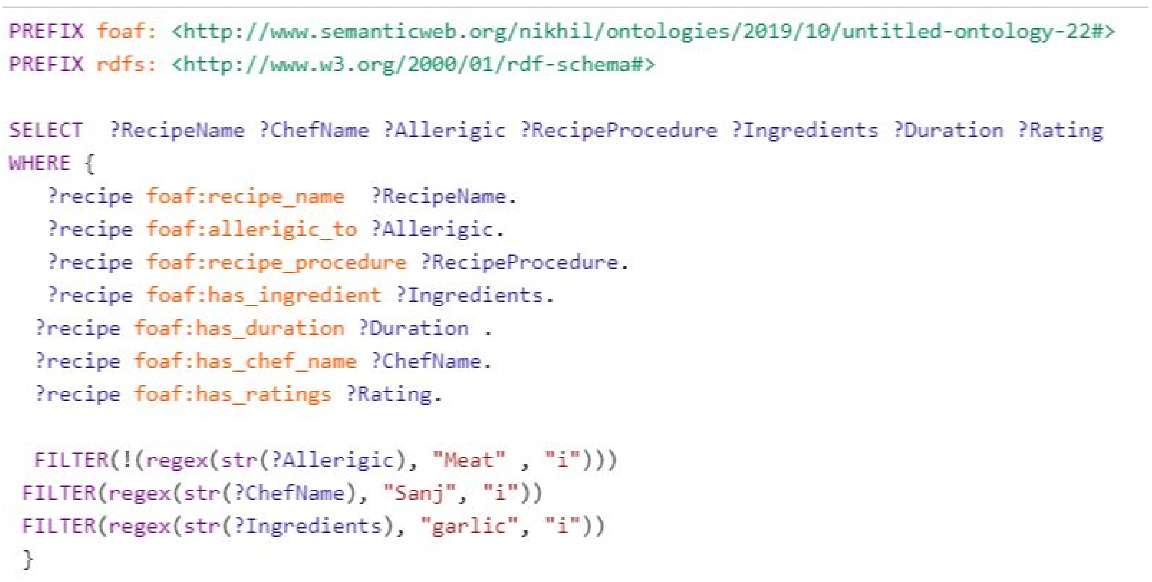
Showing 1 to 6 of 6 entries

	RecipeName	Allerigic
1	"Butter Chicken"	foaf:Meat
2	"Butter Chicken"	foaf:Milk
3	"Easy Pancakes"	foaf:Milk
4	"Easy Pancakes"	foaf:Wheat
5	"Christmas Pancakes"	foaf:Milk
6	"Christmas Pancakes"	foaf:Wheat

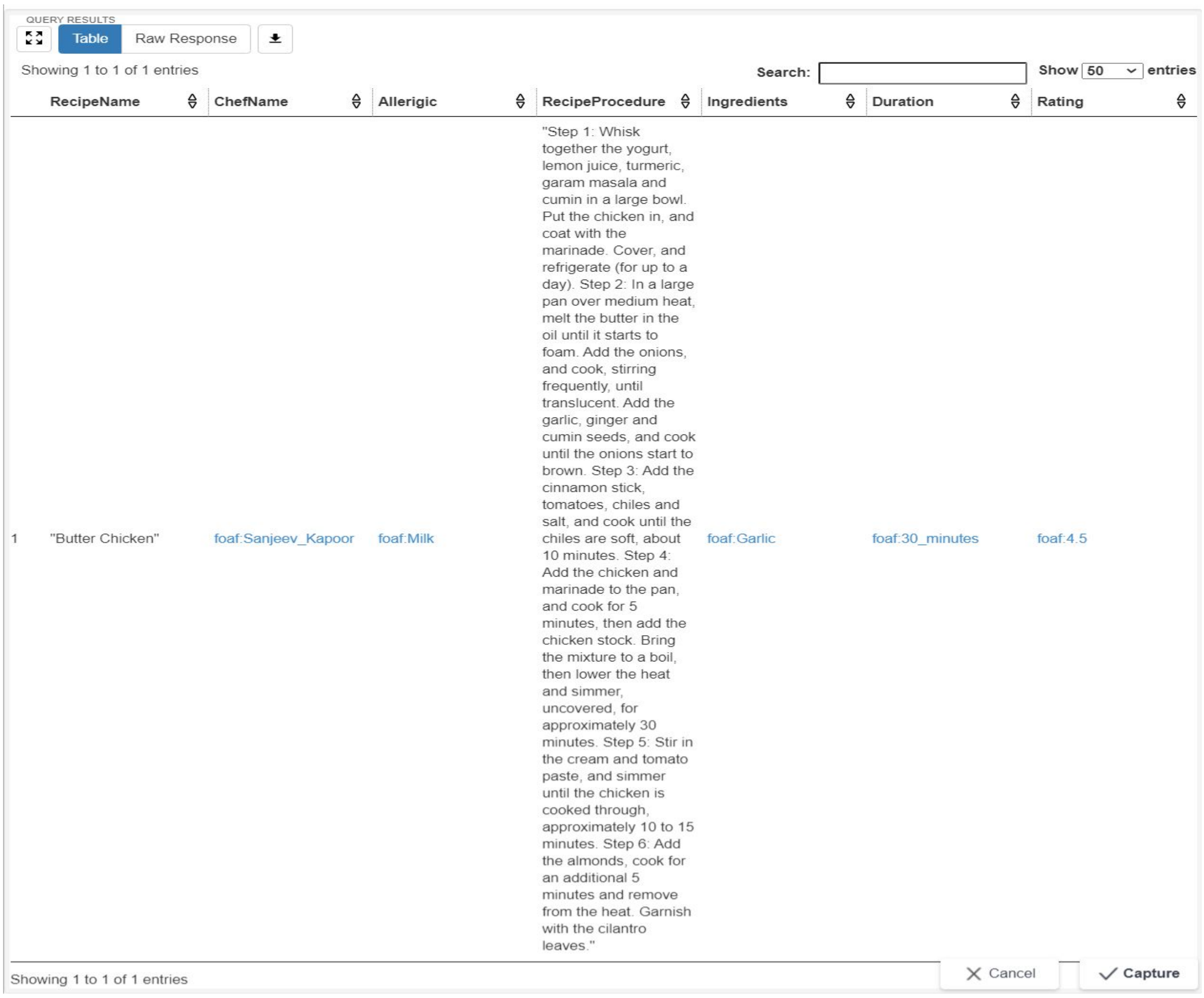
Showing 1 to 6 of 6 entries

5.4 Query for filters:

The query to show how the filters work in the filter tab of the application. Here, filtering is performed to get a recipe result based on allergies, chef name and ingredients. Note that when one filters with an allergy it should fetch the result with recipes without that specified allergies. Also, note that the chef name is not completely given, but the expected result should fetch results with a similar name. The ingredient is against case-sensitive but expected to fetch the result with no issues. This is because using “i” in the FILTER statements which prevents keywords from case-sensitivity. The result obtained is given below:



```
PREFIX foaf: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?RecipeName ?ChefName ?Allerigic ?RecipeProcedure ?Ingredients ?Duration ?Rating
WHERE {
   ?recipe foaf:recipe_name  ?RecipeName.
   ?recipe foaf:allerigic_to ?Allerigic.
   ?recipe foaf:recipe_procedure ?RecipeProcedure.
   ?recipe foaf:has_ingredient ?Ingredients.
  ?recipe foaf:has_duration ?Duration .
  ?recipe foaf:has_chef_name ?ChefName.
  ?recipe foaf:has_ratings ?Rating.

  FILTER(!(regex(str(?Allerigic), "Meat" , "i")))
 FILTER(regex(str(?ChefName), "Sanj", "i"))
 FILTER(regex(str(?Ingredients), "garlic", "i"))
 }
```



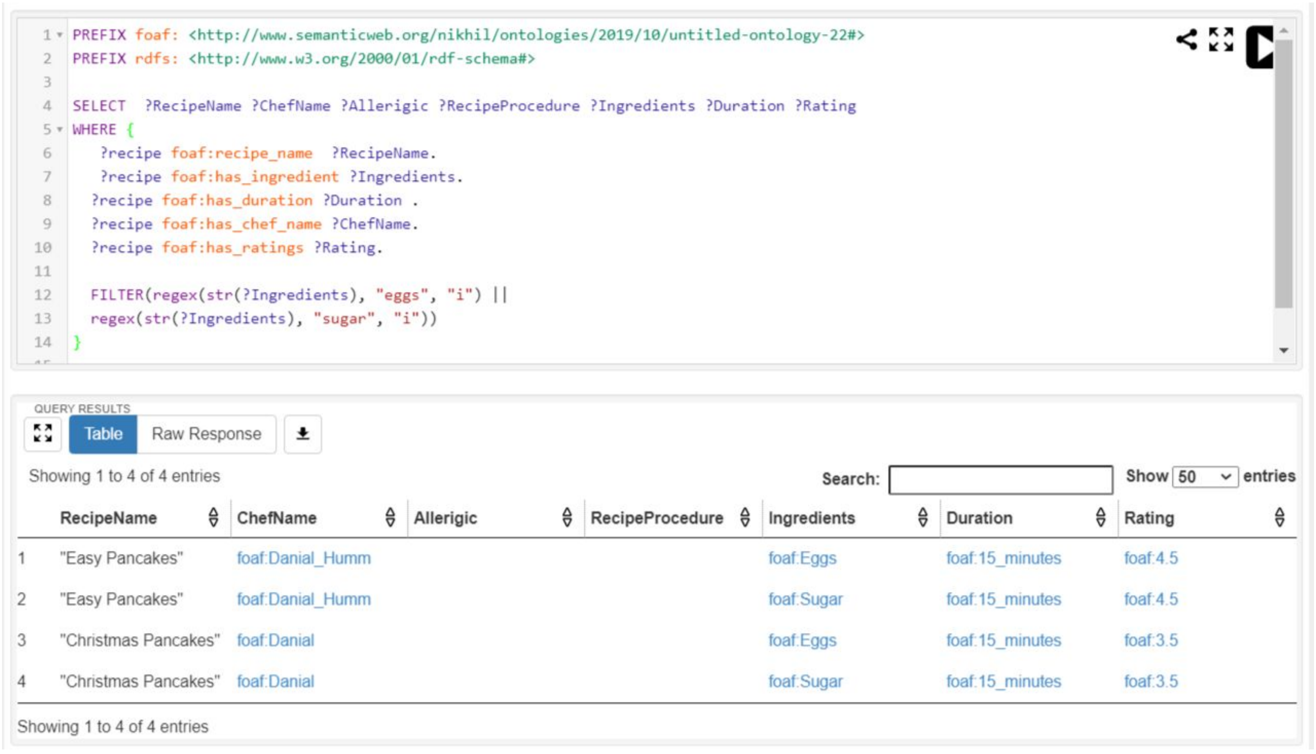
Showing 1 to 1 of 1 entries

	RecipeName	ChefName	Allerigic	RecipeProcedure	Ingredients	Duration	Rating
1	"Butter Chicken"	foaf:Sanjeev_Kapoor	foaf:Milk	"Step 1: Whisk together the yogurt, lemon juice, turmeric, garam masala and cumin in a large bowl. Put the chicken in, and coat with the marinade. Cover, and refrigerate (for up to a day). Step 2: In a large pan over medium heat, melt the butter in the oil until it starts to foam. Add the onions, and cook, stirring frequently, until translucent. Add the garlic, ginger and cumin seeds, and cook until the onions start to brown. Step 3: Add the cinnamon stick, tomatoes, chiles and salt, and cook until the chiles are soft, about 10 minutes. Step 4: Add the chicken and marinade to the pan, and cook for 5 minutes, then add the chicken stock. Bring the mixture to a boil, then lower the heat and simmer, uncovered, for approximately 30 minutes. Step 5: Stir in the cream and tomato paste, and simmer until the chicken is cooked through, approximately 10 to 15 minutes. Step 6: Add the almonds, cook for an additional 5 minutes and remove from the heat. Garnish with the cilantro leaves."	foaf:Garlic	foaf:30_minutes	foaf:4.5

Showing 1 to 1 of 1 entries

5.5 Query for multiple Ingredients:

The query to show how multiple ingredient filter works in the filter tab of the application. The query describes that there are two ingredients selected by the user, i.e eggs and sugar. The query fetches the list of recipes which contains eggs and sugar in the preparation process. The result obtained is given below. The user can select any number of ingredients from the ingredient filter dropdown menu. The query fetches them the recipes which contain the ingredients specified by the user.



```
PREFIX foaf: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT  ?RecipeName ?ChefName ?Allerigic ?RecipeProcedure ?Ingredients ?Duration ?Rating
WHERE {
   ?recipe foaf:recipe_name  ?RecipeName.
   ?recipe foaf:has_ingredient ?Ingredients.
  ?recipe foaf:has_duration ?Duration .
  ?recipe foaf:has_chef_name ?ChefName.
  ?recipe foaf:has_ratings ?Rating.

  FILTER(regex(str(?Ingredients), "eggs", "i") ||
  regex(str(?Ingredients), "sugar", "i"))
}
```

QUERY RESULTS

Table Raw Response

Showing 1 to 4 of 4 entries

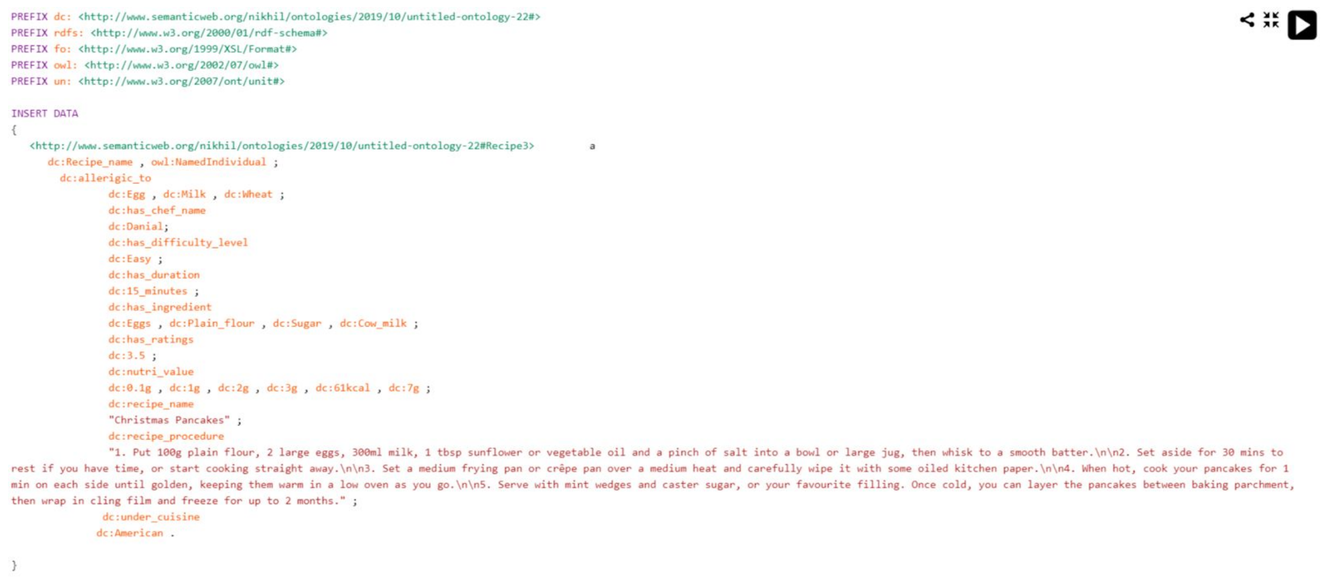
Search: Show 50 entries

	RecipeName	ChefName	Allerigic	RecipeProcedure	Ingredients	Duration	Rating
1	"Easy Pancakes"	foaf:Danial_Humm			foaf:Eggs	foaf:15_minutes	foaf:4.5
2	"Easy Pancakes"	foaf:Danial_Humm			foaf:Sugar	foaf:15_minutes	foaf:4.5
3	"Christmas Pancakes"	foaf:Danial			foaf:Eggs	foaf:15_minutes	foaf:3.5
4	"Christmas Pancakes"	foaf:Danial			foaf:Sugar	foaf:15_minutes	foaf:3.5

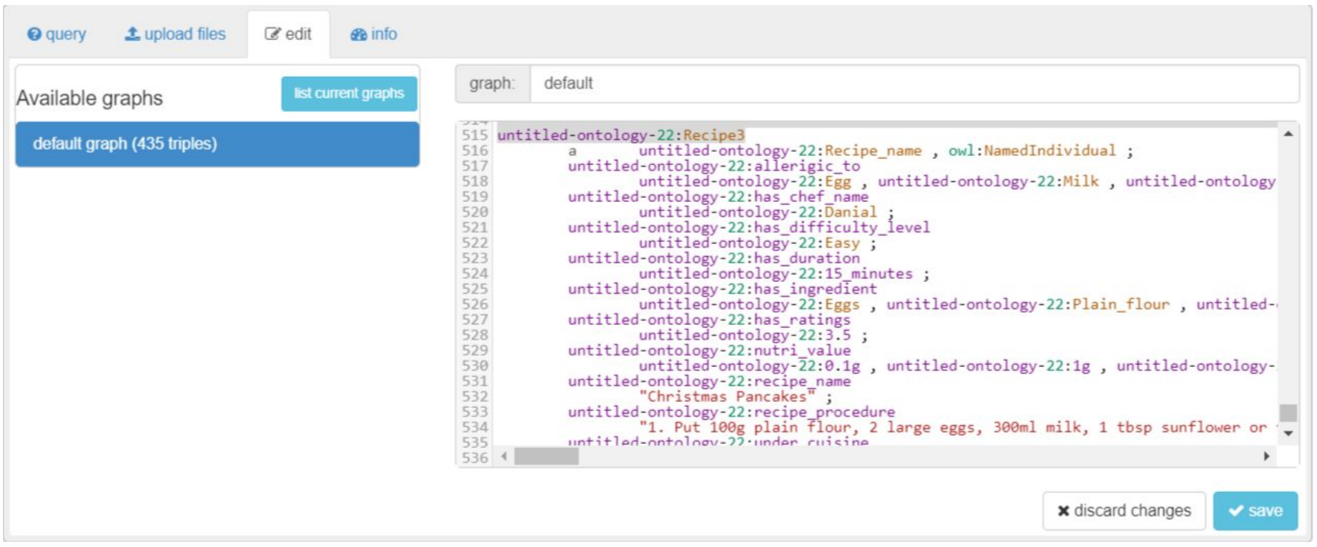
Showing 1 to 4 of 4 entries

5.6 Query for Insert:

The query to show how to insert a recipe and its respective attributes. One can insert any amount of data pertaining to that recipe from recipe name to the ratings. The below-given screenshot shows how the INSERT statement works. Note that the SPARQL endpoint address should always be changed from /query to /update whenever performing INSERT operation. After successful insertion of the defined data, the query results displays update succeeded. To check if the data is added, clicking the Edit tab and checking in the default graph is required. Once the graph is shown, scrolling down to the bottom, one should find the recipe details which are added by them.



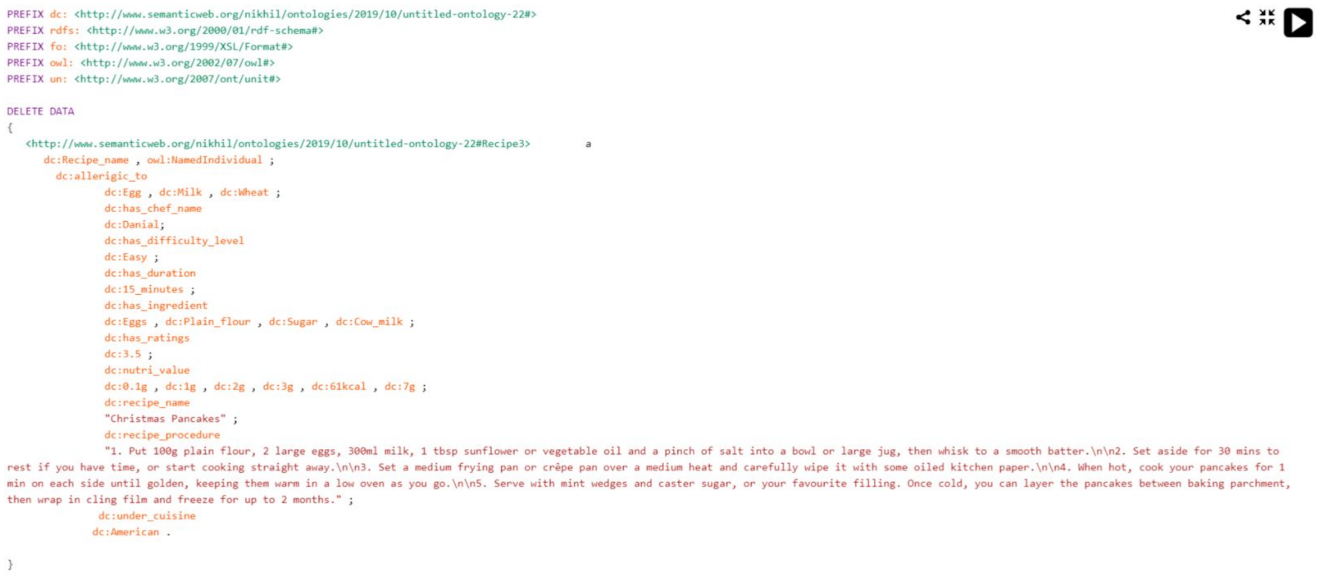
```
PREFIX dc: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX fo: <http://www.w3.org/1999/XSL/Format#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX un: <http://www.w3.org/2007/ont/unit#>

INSERT DATA
{
   <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#Recipe3>        a
      dc:Recipe_name , owl:NamedIndividual ;
        dc:allerigic_to
                dc:Egg , dc:Milk , dc:Wheat ;
                dc:has_chef_name
                dc:Danial;
                dc:has_difficulty_level
                dc:Easy ;
                dc:has_duration
                dc:15_minutes ;
                dc:has_ingredient
                dc:Eggs , dc:Plain_flour , dc:Sugar , dc:Cow_milk ;
                dc:has_ratings
                dc:3.5 ;
                dc:nutri_value
                dc:0.1g , dc:1g , dc:2g , dc:3g , dc:61kcal , dc:7g ;
                dc:recipe_name
                "Christmas Pancakes" ;
                dc:recipe_procedure
                "1. Put 100g plain flour, 2 large eggs, 300ml milk, 1 tbsp sunflower or vegetable oil and a pinch of salt into a bowl or large jug, then whisk to a smooth batter.\n\n2. Set aside for 30 mins to
rest if you have time, or start cooking straight away.\n\n3. Set a medium frying pan or crêpe pan over a medium heat and carefully wipe it with some oiled kitchen paper.\n\n4. When hot, cook your pancakes for 1
min on each side until golden, keeping them warm in a low oven as you go.\n\n5. Serve with mint wedges and caster sugar, or your favourite filling. Once cold, you can layer the pancakes between baking parchment,
then wrap in cling film and freeze for up to 2 months." ;
               dc:under_cuisine
              dc:American .

}
```



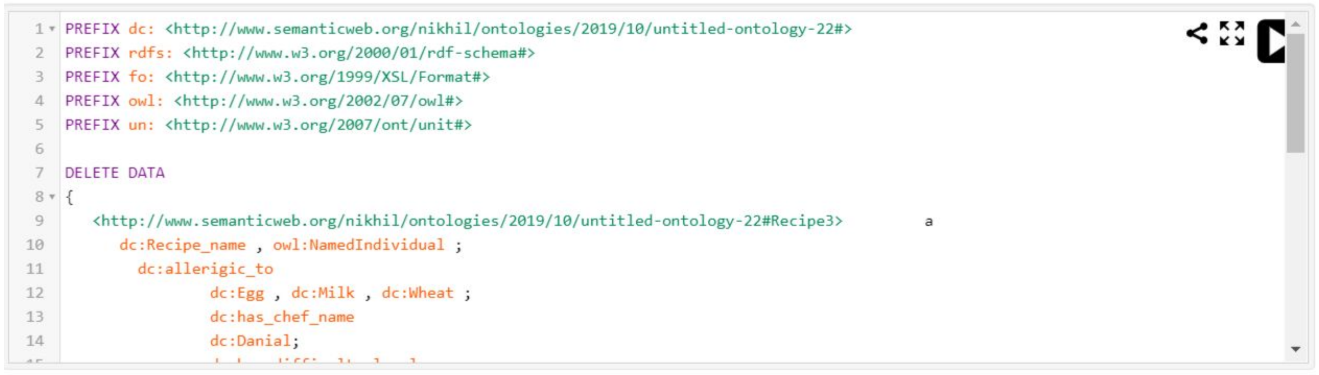
```
untitled-ontology-22:Recipe3
        a       untitled-ontology-22:Recipe_name , owl:NamedIndividual ;
        untitled-ontology-22:allerigic_to
                untitled-ontology-22:Egg , untitled-ontology-22:Milk , untitled-ontology
        untitled-ontology-22:has_chef_name
                untitled-ontology-22:Danial ;
        untitled-ontology-22:has_difficulty_level
                untitled-ontology-22:Easy ;
        untitled-ontology-22:has_duration
                untitled-ontology-22:15_minutes ;
        untitled-ontology-22:has_ingredient
                untitled-ontology-22:Eggs , untitled-ontology-22:Plain_flour , untitled-
        untitled-ontology-22:has_ratings
                untitled-ontology-22:3.5 ;
        untitled-ontology-22:nutri_value
                untitled-ontology-22:0.1g , untitled-ontology-22:1g , untitled-ontology-
        untitled-ontology-22:recipe_name
                "Christmas Pancakes" ;
        untitled-ontology-22:recipe_procedure
                "1. Put 100g plain flour, 2 large eggs, 300ml milk, 1 tbsp sunflower or
        untitled-ontology-22:under_cuisine
```

5.7 Query for Delete:

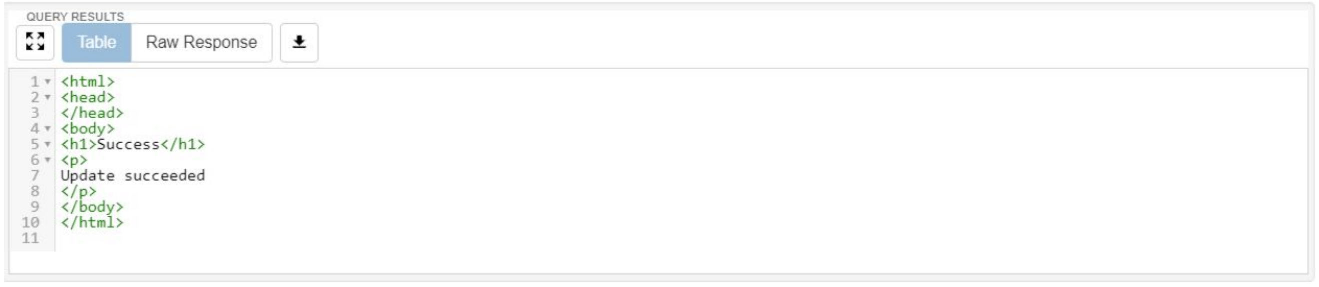
The query to show how to delete a recipe and its respective attributes. One can delete the entire recipe and its defined attributes with the following query as shown in the screenshot below. After a successful deletion, the query results section will display as update succeeded. The deleted data cannot be checked in the edit tab as the details have been deleted already.



```
PREFIX dc: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX fo: <http://www.w3.org/1999/XSL/Format#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX un: <http://www.w3.org/2007/ont/unit#>

DELETE DATA
{
   <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#Recipe3>         a
      dc:Recipe_name , owl:NamedIndividual ;
        dc:allerigic_to
                dc:Egg , dc:Milk , dc:Wheat ;
                dc:has_chef_name
                dc:Danial;
                dc:has_difficulty_level
                dc:Easy ;
                dc:has_duration
                dc:15_minutes ;
                dc:has_ingredient
                dc:Eggs , dc:Plain_flour , dc:Sugar , dc:Cow_milk ;
                dc:has_ratings
                dc:3.5 ;
                dc:nutri_value
                dc:0.1g , dc:1g , dc:2g , dc:3g , dc:61kcal , dc:7g ;
                dc:recipe_name
                "Christmas Pancakes" ;
                dc:recipe_procedure
                "1. Put 100g plain flour, 2 large eggs, 300ml milk, 1 tbsp sunflower or vegetable oil and a pinch of salt into a bowl or large jug, then whisk to a smooth batter.\n\n2. Set aside for 30 mins to
rest if you have time, or start cooking straight away.\n\n3. Set a medium frying pan or crêpe pan over a medium heat and carefully wipe it with some oiled kitchen paper.\n\n4. When hot, cook your pancakes for 1
min on each side until golden, keeping them warm in a low oven as you go.\n\n5. Serve with mint wedges and caster sugar, or your favourite filling. Once cold, you can layer the pancakes between baking parchment,
then wrap in cling film and freeze for up to 2 months." ;
               dc:under_cuisine
              dc:American .

}
```



```
PREFIX dc: <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX fo: <http://www.w3.org/1999/XSL/Format#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX un: <http://www.w3.org/2007/ont/unit#>

DELETE DATA
{
   <http://www.semanticweb.org/nikhil/ontologies/2019/10/untitled-ontology-22#Recipe3>         a
      dc:Recipe_name , owl:NamedIndividual ;
        dc:allerigic_to
                dc:Egg , dc:Milk , dc:Wheat ;
                dc:has_chef_name
                dc:Danial;
```



```
<html>
<head>
</head>
<body>
<h1>Success</h1>
<p>
Update succeeded
</p>
</body>
</html>
```

6. Critical Reflection

Some of the cooking web applications have been reviewed to gain more knowledge on how recipes are organized and work. Besides, thorough research has been made to understand how a fruitful search of recipes is made over the world wide web. Earlier this year Google has launched recipe structure to help the web developers to enhance, the visibility of their recipes including Google Home devices, in both Google Search and Google Assistant. This proves that increasing numbers of people use the internet to help them decide what to cook. To help customers with cooking resources and recipes online, several websites have been developed with new ideas.

Several websites and blogs have been developed to help users choose their favourite recipes. Many of these websites aggregate chef's recipes from various parts of the world based on different cuisines, thus helping the users to decide what to cook. Some websites also allow users to upload their own recipes and offer shopping lists that will contain everything the user will need to create whatever recipe/s they have chosen. Some of the famous recipe search websites are represented in the figure below which were used for requirement analysis.

Search for recipes based on the ingredients the users have is a good idea to solve the most cooking selection issues which were implemented by one of the popular websites called BBC Good Food.

Another popular website called Yummly, which search for recipes that contain the ingredients the user want, find tips and guides on how to do things like separating egg yolks from the whites, or how to roast a chicken has announced its users to implement semantic data technologies for its recipe search soon.

All these websites are special in its own features but are basic recipe search applications that show ingredients, instructions to cook, duration, chef name etc of the recipe. Coming to the implementation side, all of these websites use relational databases and uses keyword search which is less efficient in fetching the accurate results.

So, the main aim of the recipe finder application is to search for recipes that use semantic data technologies with users health being the main focus. The application not only fetches accurate recipe search results but also provides dietary and nutritional information for the users based on their requirements. The application also helps the users choose allergic free recipes to keep them safe from allergic ingredients.

To summarize, Recipe Finder is an improvement for the existing several recipe search websites. The proposed application used structured data and is built with rich relationships. It also uses textual searching, which provides accurate results when compared to the keyword search in many existing popular websites. In the future, machine learning techniques could be used to fetch search results based on users interaction with the application and based on their interests.

7. Conclusion

As a result, the web application has accomplished its objectives of developing a useful customer-centric recipe search system using semantic data technologies, which is meant to be used by the chefs to try new recipes in their daily activities.

One main advantage of this application is its simplicity and availability to general users. The user interface is so simple to understand that when used by all types of users like technical and non-technical background, can easily fulfil their needs.

Moreover, the application’s main characteristic is to provide dietary and nutritional content to the users along with allergic free recipes. The web application also provides tons of information about the recipe to cook which are even rated genuinely. The ratings of the recipe serve as the general review by the customers who have tried those recipes. This will help current customers to easily choose their favourite chef’s recipes.

Nonetheless, there is still room for improvement. As an example, the application can still be improved in terms of the user interface like improving the front end looks by adding attractive colours to the overall template, adding beautiful recipe pictures along with chefs profiles and adding good animations. The dataset in the ontology is limited to a small set of recipes, which can be increased to make it a professional website. The web app can further be developed to provide quantity based searching for ingredients. The application has a major scope for machine learning purposes, where the system recommends the recipes, dietary or nutritional suggestions and allergic free recipes based on the user’s preferences and his cooking habits. This will save much more time for the users and the application may get good popularity.

The choice of consuming PHP, SPARQL and using Web Ontology Language has provided an opportunity to learn many things in terms of building structured data with rich relationships and textual search engine development. The use for combining different technologies, not only allows to further use this knowledge of building creative websites and applications, but reusing them in multiple platforms, and produce results with faster deployment.

8. References

1. Addlesee, A. (2019). *Constructing SPARQL Queries*. [online] Medium. Available at: https://medium.com/wallscope/constructing-sparql-queries-ca63b8b9ac02 [Accessed 12 Nov. 2019].
2. Apache Jena. (2019). *SPARQL Tutorial - A First SPARQL Query*. [online] Available at: https://jena.apache.org/tutorials/sparql_query1.html [Accessed 12 Nov. 2019].
3. Arooj Fatima 2019, Module Notes(University site) [Online] Available at: <https://canvas.anglia.ac.uk/courses/11828/modules> [Accessed 9 Nov. 2019]
4. Belavkin, R. (2019). *Lecture 8: Ontologies*. [online] Eis.mdx.ac.uk. Available at: http://www.eis.mdx.ac.uk/staffpages/rvb/teaching/BIS4410/hand08.pdf [Accessed 24 Nov. 2019].
5. Bermejo, J. (2019). *A Simplified Guide to Create an Ontology*. [online] Tierra.aslab.upm.es. Available at: http://tierra.aslab.upm.es/documents/controlled/ASLAB-R-2007-004.pdf [Accessed 6 Nov. 2019].
6. Brickley, D. (2019). *SPARQL Negation: All foaf:Agents which aren't foaf:Persons*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/1615472/sparql-negation-all-foafagents-which-arent-foafpersons [Accessed 9 Dec. 2019].
7. En.wikipedia.org. (2019). *Ontology components*. [online] Available at: https://en.wikipedia.org/wiki/Ontology_components [Accessed 24 Nov. 2019].
8. Go-protege-tutorial.readthedocs.io. (2019). *Object properties — GO Ontology 0.5 documentation*. [online] Available at:

https://go-protege-tutorial.readthedocs.io/en/latest/ObjectProperties.html [Accessed 22 Nov. 2019].

9. Hegde, P. (2019). *How to filter an html table based on drop down selected value?*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/51515778/how-to-filter-an-html-table-based-on-drop-down-selected-value/51517342 [Accessed 4 Dec. 2019].
10. Horridge, M., Jupp, S., Moulton, G., Rector, A., Stevens, R. and Wroe, C. (2019). [online] Mowl-power.cs.man.ac.uk. Available at: http://mowl-power.cs.man.ac.uk/protegeowltutorial/resources/ProtegeOWLTutorialP4_v1_1.pdf [Accessed 22 Nov. 2019].
11. HTML.com. (2019). *Attribute for MULTIPLE*. [online] Available at: https://html.com/attributes/select-multiple/ [Accessed 2 Dec. 2019].
12. Jayawardana, V. (2019). *Ontology Generation and Visualization with Protégé*. [online] Medium. Available at: https://medium.com/@vindulajayawardana/ontology-generation-and-visualization-with-prot%C3%A9g%C3%A9-6df0af9955e0 [Accessed 6 Nov. 2019].
13. Kim, A. (2019). *What is Semantic Search?*. [online] Medium. Available at: https://blog.graphiq.com/what-is-semantic-search-71808207cfcc [Accessed 6 Dec. 2019].
14. Linkeddatatools.com. (2019). *What Are Classes And Individuals?*. [online] Available at: http://www.linkeddatatools.com/help/classes [Accessed 24 Nov. 2019].
15. Noy, N. and McGuinness, D. (2019). *Ontology Development 101: A Guide to Creating Your First Ontology*. [online] Protege.stanford.edu. Available at: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf [Accessed 6 Nov. 2019].

16. OCLC. (2019). *Manipulating output with FILTER, OPTIONAL and UNION | OCLC Developer Network*. [online] Available at: https://www.oclc.org/developer/news/2016/manipulating-output-with-filter-optional-union.en.html [Accessed 20 Nov. 2019].
17. Ontotext. (2019). *What are Ontologies and What are the Benefits of Using Ontologies*. [online] Available at: https://www.ontotext.com/knowledgehub/fundamentals/what-are-ontologies/ [Accessed 6 Nov. 2019].
18. Ontotext. (2019). *What is SPARQL - Semantic Search Query Language - Ontotext*. [online] Available at: https://www.ontotext.com/knowledgehub/fundamentals/what-is-sparql/ [Accessed 6 Nov. 2019].
19. Protegeproject.github.io. (2019). *Object Property Characteristics*. [online] Available at: https://protegeproject.github.io/protege/views/object-property-characteristics/ [Accessed 24 Nov. 2019].
20. Protegewiki.stanford.edu. (2019). *Protege Wiki*. [online] Available at: https://protegewiki.stanford.edu/wiki/Main_Page [Accessed 4 Nov. 2019].
21. Rector, A. (2019). *Protege4Pizzas10Minutes - Protege Wiki*. [online] Protegewiki.stanford.edu. Available at: https://protegewiki.stanford.edu/wiki/Protege4Pizzas10Minutes [Accessed 12 Nov. 2019].
22. RightsDirect. (2019). *The Limitations of Keyword Research vs Semantic Search | RightsDirect*. [online] Available at: https://www.rightsdirect.com/limitations-keyword-search/ [Accessed 6 Dec. 2019].
23. *SPARQL in 11 minutes*. (2015). [video] Youtube: bobdc.

24. Taylor, J. (2019). *SPARQL Querying multiple ORs in the same filter.* [online] Stack Overflow. Available at: https://stackoverflow.com/questions/23561351/sparql-querying-multiple-ors-in-the-same-filter [Accessed 24 Nov. 2019].
25. Verdemato, P. (2019). *Semantic Search vs. Keyword Search - Copyright Clearance Center.* [online] Copyright Clearance Center. Available at: https://www.copyright.com/blog/semantic-search-vs-keyword-search/ [Accessed 5 Dec. 2019].
26. W3.org. (2019). *Design:Negation - SPARQL Working Group.* [online] Available at: https://www.w3.org/2009/sparql/wiki/Design:Negation [Accessed 9 Dec. 2019].
27. W3schools.com. (2019). *How To Search for Items in a Dropdown.* [online] Available at: https://www.w3schools.com/howto/howto_js_filter_dropdown.asp [Accessed 2 Dec. 2019].
28. W3schools.com. (2019). *HTML select multiple Attribute.* [online] Available at: https://www.w3schools.com/tags/att_select_multiple.asp [Accessed 2 Dec. 2019].
29. Wikidata.org. (2019). *Wikidata:SPARQL tutorial - Wikidata.* [online] Available at: https://www.wikidata.org/wiki/Wikidata:SPARQL_tutorial#SPARQL_basics [Accessed 20 Nov. 2019].
30. Zimmermann, A. and Kralin, S. (2019). *Ontology Modelling: Object Property or Data Property?.* [online] Stack Overflow. Available at: https://stackoverflow.com/questions/51617500/ontology-modelling-object-property-or-data-property [Accessed 24 Nov. 2019].

9. Appendix

The code for the recipe finder application and the ontology used for the same is made into a GitHub repository which is given as a link below:

https://github.com/nikhilmarathi/Recipe-Finder.git

Installation:

1. XAMPP is required to make the application work. Start the apache server on the XAMPP window.
2. Install Fuseki and run in on the localhost.
3. Place the index.php and sparqllib.php in the recipe folder which is inside htdocs(xampp installation folder).
4. Upload the owl file into the fuseki server and perform basic queries to make sure it is working.
5. In the browser, go to localhost:80/ to access the web application.