AUTOMATING SPARQL QUERY CREATION: AN LLM-POWERED APPROACH FOR SEMANTIC KNOWLEDGE GRAPHS

FACULTY OF ENGINEERING
AIN SHAMS UNIVERSITY
International Credit Hours Engineering Programs (i.CHEP)

Project Document

Due Date:

8/5/2024

Submitted by:

Omar Ashraf Mabrouk

Osama Ayman Mokhtar Amin

Omar Ayman Ayoub Abdelshafi

Course Information

ASU Course Code CSE 488	***ASU Course Name*** Ontologies and the Semantic Web
Semester Spring 2024	**Due Date** 8-5-2024

Participants Information

Full Name	ASU ID
Omar Ashraf Mabrouk	19P8102
Omar Ayman Ayoub Abdelshafi	1900804
Osama Ayman Mokhtar Amin	19P1609

Contents

Introduction 5
Overview 5
The Objective of The Report 5
Preliminary Information 6
SPARQL 6
Large Language Models 7
Merging SPARQL and LLM 7
Problem Domain 9
Proposed Architecture 10
User Interface (UI): 11
Query Preprocessing Module: 11
Large Language Model (LLM): 11
Ontology Knowledge Base: 11
Output: 11
Dataflow Diagram 12
Code Explanation: 12
Ontology Visualization 16
Sample Outputs (Screenshots) 16
Get 5 athletes, their teams and coaches 16
Get athletes who are in Arsenal, Liverlpool, ManCity with their teams and heights. 17
Get team names which participated in ChampionsLeague. 17
Appendix 18
Java Code 18
GITHUB: 48

Introduction

Overview

In the hastily evolving panorama of statistics technology, the powerful control and usage of good sized quantities of statistics have come to be imperative. The emergence of ontologies and the semantic net has supplied a dependent framework for representing and interlinking expertise in a machine-readable format. However, harnessing the total ability of semantic statistics calls for green approach of querying and extracting applicable statistics.

The SPARQL (SPARQL Protocol and RDF Query Language) serves as a effective device for querying RDF (Resource Description Framework) statistics, permitting customers to retrieve unique statistics from ontologies and semantic net sources. However, the guide production of SPARQL queries may be complicated and time-consuming, regularly requiring information in each area expertise and question language syntax.

In this context, the combination of Large Language Models (LLMs) affords a promising street for automating SPARQL question era. LLMs, along with GPT (Generative Pre-educated Transformer) fashions, have established extremely good competencies in herbal language information and era tasks. Leveraging those fashions for SPARQL question era holds the ability to streamline the querying process, lessen the barrier to access for customers, and facilitate expertise discovery from semantic statistics sources.

This file explores the usage of LLMs for SPARQL question era withinside the area of ontologies and the semantic net. It investigates the feasibility, effectiveness, and implications of using LLMs to automate the era of SPARQL queries, aiming to beautify the accessibility and application of semantic statistics for diverse applications, together with expertise control, statistics integration, and semantic search.

The Objective of The Report

The objective of this report is to present a conceptual framework and proposed architecture for leveraging Large Language Models (LLMs) in automating SPARQL query generation within the domain of ontologies and the semantic web. Specifically, the report aims to:

1. Provide an overview of SPARQL and Large Language Models (LLMs), outlining their respective roles in querying RDF data and natural language processing tasks.
2. Propose an architectural framework for integrating LLMs into the SPARQL query generation process, emphasizing the potential benefits for knowledge discovery and semantic data analysis.
3. Discuss the current project status, including preliminary research, methodology development, and future directions for implementation and experimentation.
4. Explore the challenges and considerations associated with implementing LLM-based approaches for SPARQL query generation, including data preprocessing, model adaptation, and scalability concerns.

By addressing these objectives, the report seeks to lay the foundation for future research and development efforts aimed at harnessing the capabilities of LLMs to enhance query generation and semantic data exploration within ontologies and the semantic web.

Preliminary Information

SPARQL

SPARQL (SPARQL Protocol and RDF Query Language) stands as a pivotal component in the realm of Semantic Web technologies, providing a standardized query language and protocol for interrogating RDF (Resource Description Framework) data. Its primary purpose is to facilitate the retrieval and manipulation of data stored in RDF format, enabling users to perform sophisticated queries across distributed, heterogeneous datasets on the web.

SPARQL was conceptualized to address the growing need for a standardized query language capable of extracting knowledge from the Semantic Web. Its intended purpose revolves around enabling efficient and expressive querying of RDF data, thereby supporting tasks such as data integration, knowledge discovery, semantic search, and ontology-driven applications.

The development of SPARQL traces back to the early 2000s, amid the emergence of the Semantic Web initiative spearheaded by the World Wide Web Consortium (W3C). The need for a standardized query language to query RDF data became increasingly apparent as the Semantic Web gained traction. Initial efforts led to the development of query languages like RDQL (RDF Data Query Language) and Versa, which laid the groundwork for SPARQL.

In 2006, the W3C formed the RDF Data Access Working Group to develop a standardized query language for RDF. After several iterations and refinements, the W3C published the SPARQL specification as a recommendation in January 2008, establishing it as a foundational technology for the Semantic Web.

SPARQL operates based on a pattern matching approach, where queries are formulated using a combination of triple patterns, filters, and optional patterns to specify the desired data patterns and constraints. The basic structure of a SPARQL query consists of a SELECT clause for specifying the variables to be returned in the query results, and a WHERE clause for defining the patterns to match against the RDF graph.

- **Triple Patterns:** At the core of SPARQL queries are triple patterns, which represent simple subject-predicate-object statements in RDF. These patterns define the criteria for matching RDF triples in the queried dataset.
- **Filters:** SPARQL allows the use of filters to apply additional constraints on query variables or expressions, enabling more precise control over query results. Filters can be used to enforce conditions such as equality, comparison, and pattern matching.
- **Optional Patterns**: Optional patterns in SPARQL queries specify optional parts of the query graph, allowing for flexible querying of RDF data. Optional patterns are useful for querying data with varying levels of completeness or for specifying alternative data paths.

SPARQL queries can also incorporate features such as graph patterns, aggregation functions, and federated querying to enable more complex and powerful querying capabilities. Moreover, SPARQL supports various result formats, including XML, JSON, and RDF, facilitating interoperability and integration with existing web technologies.

Large Language Models

Large Language Models (LLMs) stand as a monumental achievement in the realm of natural language processing (NLP), characterized by their colossal size, capacity for learning, and proficiency in generating coherent and contextually relevant text. These models have redefined various NLP tasks, spanning from language translation and summarization to question answering and text generation. At the core of LLMs lies the transformative potential of deep learning, particularly the neural network architecture known as transformers.

The defining characteristics of Large Language Models (LLMs) underscore their remarkable capabilities and versatility in processing natural language data. Central to LLMs is their unprecedented scale, boasting hundreds of millions to billions of parameters that enable them to capture intricate linguistic patterns and nuances. Additionally, LLMs undergo a two-step process of pre-training and fine-tuning, wherein they glean insights from vast corpora of text data through unsupervised learning before adapting to specific tasks via supervised learning. This dual approach empowers LLMs to generalize across diverse linguistic contexts and excel in a wide array of NLP applications.

The applications of Large Language Models (LLMs) span a broad spectrum of domains, reflecting their versatility and effectiveness in addressing diverse linguistic challenges. From generating human-like text to facilitating language translation and sentiment analysis, LLMs have revolutionized how we interact with and harness the power of language. These models serve as the backbone of chatbots, virtual assistants, and content creation platforms, providing seamless and contextually relevant interactions in real-time. Moreover, LLMs contribute to breaking down language barriers, enabling cross-lingual communication and knowledge dissemination on a global scale.

While Large Language Models (LLMs) offer unprecedented capabilities in natural language processing, they also pose significant challenges and considerations that warrant careful attention. Chief among these challenges is the substantial computational resources required for training and fine-tuning large-scale LLMs, necessitating high-performance computing infrastructure and energy-intensive processes. Furthermore, issues related to data bias and fairness, ethical implications, and environmental impact underscore the need for responsible development and deployment of LLMs. Addressing these challenges is essential for ensuring the equitable, ethical, and sustainable advancement of LLM technology in the realm of natural language processing.

Merging SPARQL and LLM

In the realm of semantic web technologies, the ability to seamlessly bridge the gap between natural language queries and structured RDF data holds immense promise for enabling more intuitive and efficient knowledge discovery. This overview delves into the application of generating SPARQL queries from text-based search queries and subsequently fetching results

from RDF (Resource Description Framework) datasets, highlighting its significance, challenges, and potential implications.

The process of generating SPARQL queries from text-based search queries represents a crucial step towards democratizing access to semantic data and facilitating its utilization across diverse domains. By enabling users to articulate their information needs in natural language, rather than requiring expertise in SPARQL query language, this approach lowers the barrier to entry for querying RDF datasets. Moreover, it enhances the accessibility and usability of semantic web resources, empowering a broader range of stakeholders to leverage the wealth of knowledge encoded within RDF data.

Despite its potential benefits, the task of generating SPARQL queries from text-based search queries poses several challenges that must be addressed. One key challenge lies in the ambiguity and variability of natural language expressions, which can lead to ambiguity in query interpretation and generation. Resolving this ambiguity requires sophisticated natural language understanding techniques capable of discerning the user's intent and translating it into precise SPARQL queries. Additionally, ensuring the correctness and efficiency of generated queries, as well as handling complex semantic constructs and ontological relationships, presents further challenges that require careful consideration.

Several approaches have been proposed to tackle the task of generating SPARQL queries from text-based search queries, ranging from rule-based systems to machine learning-based methods. These approaches include:

Rule-based Systems:

- Utilize predefined templates or grammars to map natural language expressions to corresponding SPARQL queries.
- Specify rules that encode syntactic and semantic patterns in the input text and generate corresponding SPARQL query structures.
- Can be effective for simple query patterns and well-defined domains but may lack flexibility and scalability.

Machine Learning-based Methods:

- Leverage techniques such as deep learning and natural language processing to learn mappings between textual inputs and SPARQL query structures.
- Involve training neural network models on annotated training data to predict SPARQL query components based on input text.
- Enable automatic generation of SPARQL queries from text by leveraging learned associations between linguistic features and query structures.
- Offer greater flexibility and adaptability compared to rule-based systems, allowing for more nuanced query generation in diverse contexts.
- May require substantial amounts of annotated training data and computational resources for training and inference.

Hybrid Approaches:

- Combine elements of rule-based and machine learning-based methods to leverage the strengths of both approaches.
- Use rule-based components for initial query structure generation and refinement, followed by machine learning models for fine-tuning and optimization.
- Provide a balance between the expressiveness of rule-based systems and the adaptability of machine learning-based methods.
- Can offer improved performance and robustness by integrating complementary techniques for query generation.

Semantic Parsing Techniques:

- Apply semantic parsing techniques to transform natural language queries into executable semantic representations, such as logical forms or semantic graphs.
- Utilize semantic parsers to map natural language expressions to intermediate representations that can be directly translated into SPARQL queries.
- Benefit from the formal semantics provided by semantic parsing, which facilitates precise query generation and interpretation.
- Require robust semantic parsing models capable of handling the variability and ambiguity inherent in natural language expressions.

The successful application of generating SPARQL queries from text-based search queries and fetching results from RDF datasets holds profound implications for various domains and applications. From enhancing search engines and knowledge graphs to facilitating data integration and semantic search, this approach opens up new avenues for harnessing the power of semantic web technologies. Furthermore, it has the potential to revolutionize information retrieval and knowledge discovery processes, enabling users to interact with RDF data in a more natural and intuitive manner.

The application of generating SPARQL queries from text-based search queries and fetching results from RDF datasets represents a significant advancement in the field of semantic web technologies. By bridging the gap between natural language expressions and structured RDF data, this approach facilitates more intuitive and efficient querying of semantic web resources. While challenges remain in terms of ambiguity resolution, query correctness, and efficiency, ongoing research and development efforts continue to drive progress in this area, paving the way for the widespread adoption of semantic search and knowledge discovery solutions.

Problem Domain

The ontology project addresses the domain of sports, with a focus on soccer (football). It aims to model various aspects related to athletes, teams, coaches, competitions, and venues within the context of soccer. By providing a structured representation of these entities and their relationships, the ontology facilitates better understanding, organization, and retrieval of information related to the sports domain.

Number of Entities: The ontology defines several classes or entities:

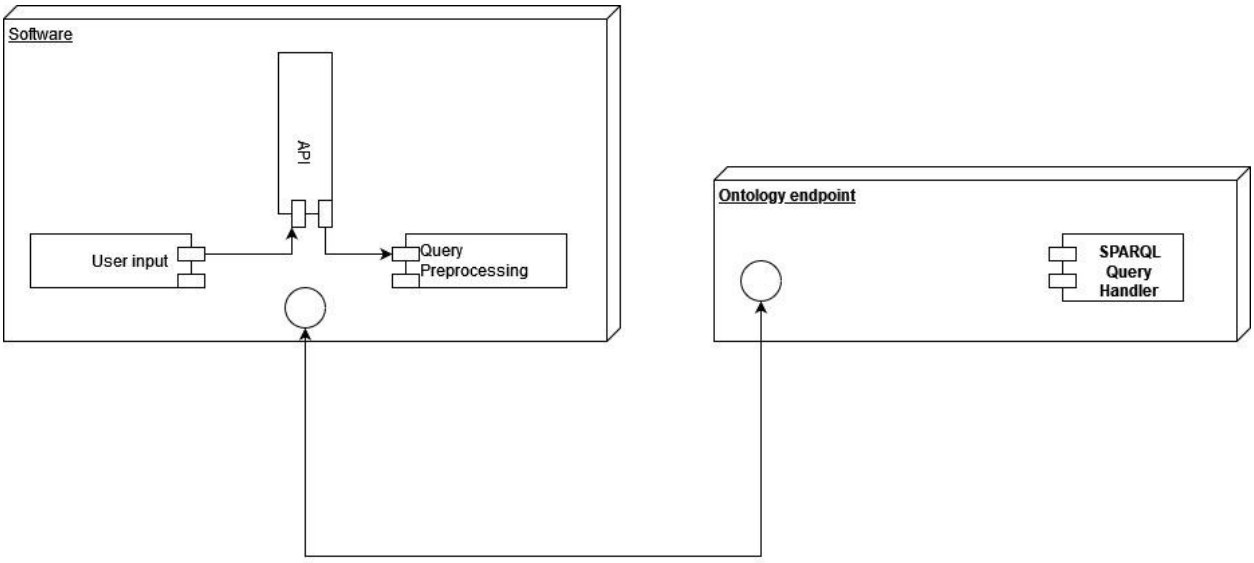
1. Athletes: Represent individual players participating in soccer.

2. Coach: Represents coaches associated with soccer teams.
3. Competitions: Represents soccer competitions such as leagues and tournaments.
4. Sports: Represents the overarching concept of sports.
5. Teams: Represents soccer teams participating in various competitions.
6. Venues: Represents stadiums or venues where soccer matches take place.

Number of Relations: The ontology defines several relationships (properties) between entities:

1. belongs_to: Relates athletes to the teams they belong to.
2. coached_by: Associates teams with their coaches.
3. defeated_by: Indicates which teams were defeated by other teams.
4. has_home_venue: Links teams to their home venues.
5. participates_in: Specifies which competitions teams participate in.
6. takes_place_at: Establishes the venue where competitions take place.

Proposed Architecture



The high-level architecture for LLM-based SPARQL query generation is designed to enable users to effortlessly formulate SPARQL queries using natural language descriptions. This system comprises several interconnected components that work together to translate user queries into formal SPARQL queries compatible with ontologies. Below is a breakdown of each component and its role within the architecture:

User Interface (UI):

- The User Interface serves as the front-end of the system, providing a platform for users to interact with the query generation process.
- It typically includes a text box or another interface where users can input their natural language queries about the ontology.

Query Preprocessing Module:

- The Query Preprocessing Module serves as the initial processing stage for user queries, preparing them for input into the Large Language Model (LLM).
- Its primary function is to transform the user's natural language query into a suitable format that can be understood by the LLM.
- This module performs tasks such as tokenization and stemming/lemmatization to structure the query appropriately.
- Additionally, the Query Preprocessing Module parses the response from the LLM to ensure it can be displayed to the user for review and potential editing.
- By handling both query preparation and response parsing, this module ensures seamless communication between the user interface and the LLM, facilitating an intuitive user experience.

Large Language Model (LLM):

- The LLM module leverages the processed query from the NLP module to generate a formal SPARQL query.
- It utilizes its vast knowledge and understanding of language encoded within its parameters to accurately translate the user's natural language query into SPARQL syntax.

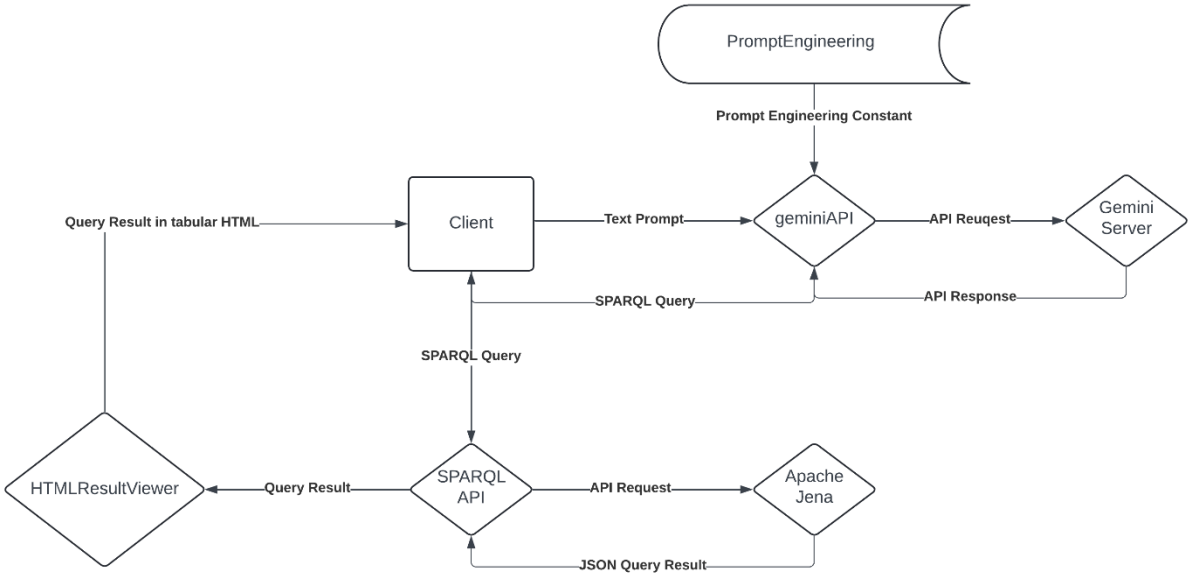
Ontology Knowledge Base:

- The Ontology Knowledge Base stores the ontology that the user's query pertains to, providing a structured representation of the domain's entities and relationships.
- Both the LLM and the SPARQL generation module may access this knowledge base to understand the ontology's structure and incorporate relevant information into the generated SPARQL query.
- Depending on the specific implementation, this knowledge base might be integrated within the LLM itself or accessed externally by the modules.

Output:

- The final output of the system is the generated SPARQL query, which is ready to be executed against the target ontology.
- The UI may display the generated SPARQL query for the user to review, modify if necessary, or directly execute against the ontology to retrieve relevant information.

Dataflow Diagram



Code Explanation:

GeminiApiRequest.java

generateContent(String queryText)

This method facilitates the generation of content using Google's Gemini API. It takes a SPARQL query as input, combines it with predefined prompts and constants related to engineering, and sends a request to the Gemini API. The generated content is then returned.

Parameters:

- queryText (String): The SPARQL query to be used in content generation.

Returns:

- String: The generated content if the request is successful. If the request fails, it returns the error message as a string.

Functionality:

1. OkHttpClient Creation:
    - An OkHttpClient instance is created for making HTTP requests to the Gemini API.
2. API Key and URL Construction:
    - A placeholder API key (apiKey) is specified. Developers should replace it with their actual Google API key.

- The URL for the Gemini API endpoint is constructed using the apiKey.

3. Prompt and Constants Configuration:
    - A promptEngineringConstants instance is created to access predefined prompts and constants related to engineering.
    - The predefined prompts and constants are combined with the input queryText to form the complete query.
4. JSON Request Body Construction:
    - The combined query string is wrapped in a JSON structure representing the request body (jsonReq).

refineContent(String queryText)

This method is responsible for refining a SPARQL query using Google's Gemini API. It takes a SPARQL query as input and returns the refined SPARQL query on success. In case of a failed request, it returns the error message as a string.

Parameters:

- queryText (String): The SPARQL query to be refined.

Returns:

- String: The refined SPARQL query on success. If the request fails, it returns the error message as a string.

Functionality:

1. OkHttpClient Creation:
    - An OkHttpClient instance is created for making HTTP requests to the Gemini API.
2. API Key and URL Construction:
    - A placeholder API key (apiKey) is specified. Developers should replace it with their actual Google API key.
    - The URL for the Gemini API endpoint is constructed using the apiKey.
3. Query Construction:
    - A query header (queryHeader) is defined to provide context to the Gemini API.
    - The queryText is prepended with the query header.
4. JSON Request Body Construction:
    - The combined query string is wrapped in a JSON structure representing the request body (jsonReq).
5. Request Building:

- A RequestBody object is created from the JSON request body.
- A Request object is built using the URL, the HTTP POST method, the request body, and the content type header.

Class Documentation: StringReplacement

This class provides functionality to replace specific patterns within a given input string.

Method: main(String[] args)

This method serves as the entry point of the program. It demonstrates the replacement of certain patterns within a sample input string and outputs the replaced string.

Sample Input String:

String inputString = "sparql\nSELECT ?player\nWHERE {\n ?player rdf:type dbpedia-owl:Person .\n…….

Sample output String:

String inputString = "sparql SELECT ?player WHERE { ?player rdf:type dbpedia-owl:Person . …….

Replacements:

1. Pattern to Replace: "`|\\n|\\""
    - This pattern targets occurrences of triple backticks, escape sequences for newline characters, and escape sequences for double quotes.
    - Replacement: Space (" ")
2. Pattern to Replace: "sparql"
    - This pattern targets the term "sparql".
    - Replacement: Empty string ("")

Functionality:

1. String Replacement:
    - The input string is processed using the replaceAll() method to replace occurrences of triple backticks, escape sequences for newline characters, and escape sequences for double quotes with spaces.
    - Additionally, occurrences of the term "sparql" are replaced with an empty string.

DBpediaApiReq.java

This class facilitates sending queries to a local DBpedia SPARQL endpoint and retrieving the results in HTML format.

Constants:

- **BASE_URL**: The base URL of the local DBpedia SPARQL endpoint.

- **client**: An OkHttpClient instance for making HTTP requests.
- **httpBuilder**: A builder for constructing HTTP URLs.
- **queryParams**: A map containing query parameters such as output format and the actual query.

Methods:

Method: sendQuery(String query, Map<String,String> extraParams)

This method sends a SPARQL query to the local DBpedia SPARQL endpoint and retrieves the results in HTML format.

Parameters:

- **query** (String): The SPARQL query to be sent to the DBpedia endpoint.
- **extraParams** (Map<String,String>): Additional query parameters to be included in the request, if any.

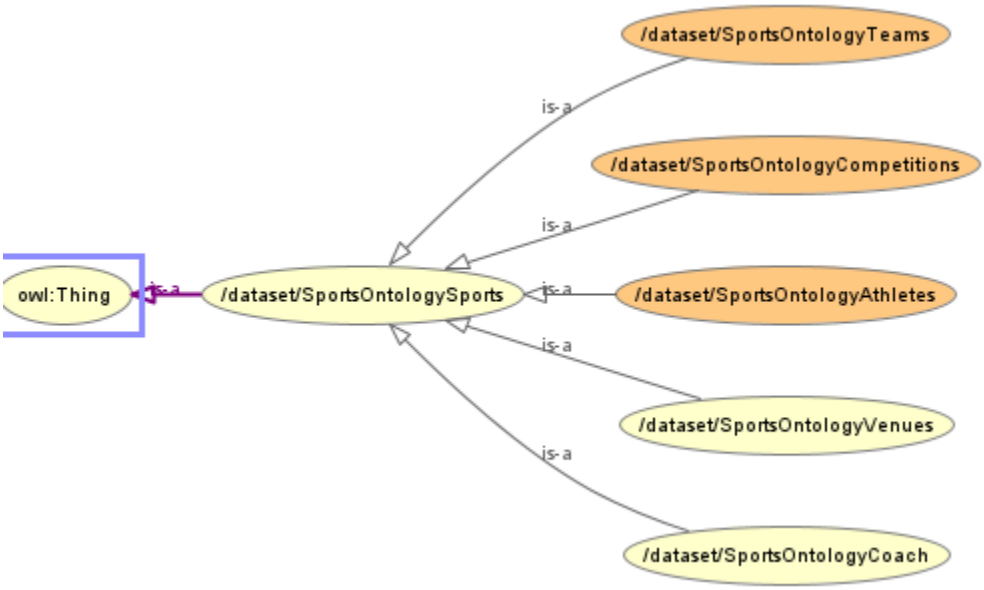
Returns:

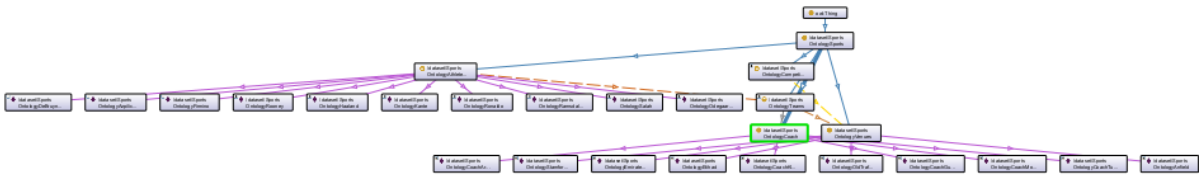
- String: The HTML result of the query execution.

Functionality:

1. Query Parameter Setup:
    - If additional query parameters are provided (extraParams), they are added to the request URL.
    - The SPARQL query is added to the request URL.
2. Request Building and Execution:
    - A Request object is built using the constructed URL.
    - The request is executed using OkHttpClient.
    - If the response is successful:
        - The JSON response is parsed into a Map using Gson.
        - An HTMLResultViewBuilder is initialized for constructing an HTML view of the query results.
        - The header of the HTML view is set based on the variables in the query results.
        - Each row of the query results is added to the HTML view.
        - The HTML result is built using the HTMLResultViewBuilder.
    - If the request fails, "bad request" is printed to the console.

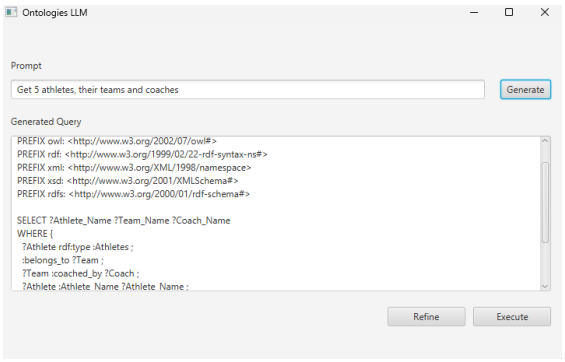
Ontology Visualization

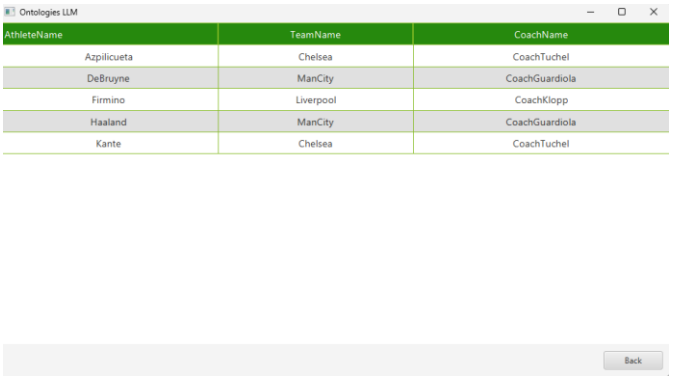




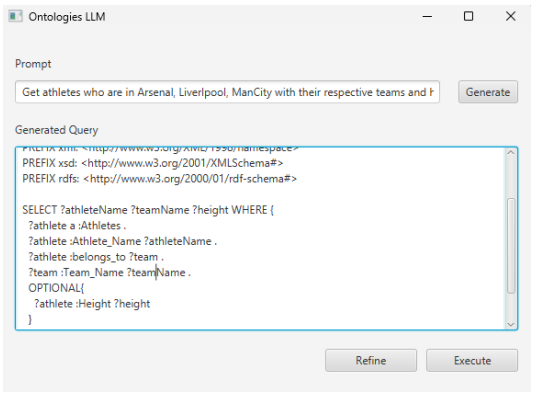
Sample Outputs (Screenshots)

Get 5 athletes, their teams and coaches

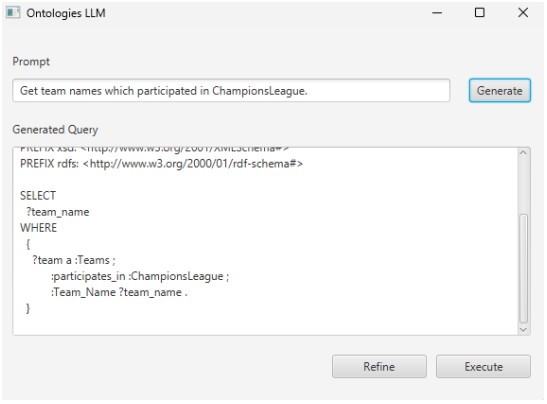


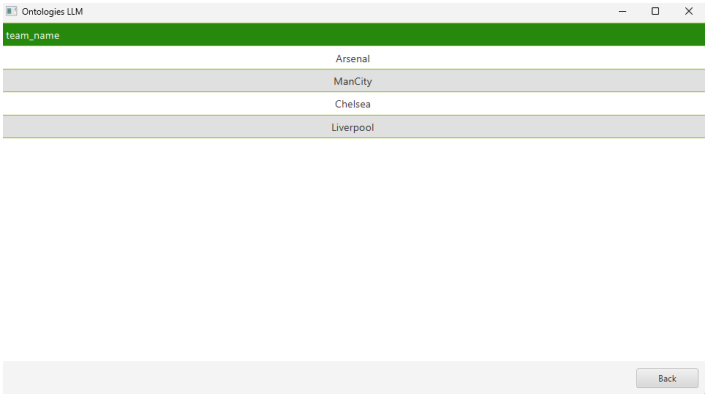


Get athletes who are in Arsenal, Liverlpool, ManCity with their teams and heights.



Get team names which participated in ChampionsLeague.





Appendix

Java Code

DbpediaApiRequest.java

```
package org.example.demo;
package org.example.demo;
import com.google.gson.Gson;
import okhttp3.*;

import java.io.IOException;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;

public class DbpediaApiRequest {

    static final private String BASE_URL =
"http://localhost:3030/SportsOntology/sparql";
    static final private OkHttpClient client = new OkHttpClient();
    static private HttpUrl.Builder httpBuilder =
Objects.requireNonNull(HttpUrl.parse(BASE_URL)).newBuilder();

    private DbpediaApiRequest(){}


    public static String sendQuery(String query,
Map<String,String>extraParams) {
        String jsonRes = "";
        String htmlRes = "";

        httpBuilder = httpBuilder.removeAllQueryParameters("query");
        httpBuilder = httpBuilder.removeAllQueryParameters("output");
        final HashMap<String,String> queryParams = new HashMap<>() {{
            put("output", "json");
        }};
        if (extraParams != null) {
            for(Map.Entry<String, String> param : extraParams.entrySet()) {

httpBuilder.addQueryParameter(param.getKey(),param.getValue());
            }
        }
        if(query != null && !query.isEmpty()){
            queryParams.put("query", query);
            for(Map.Entry<String, String> param : queryParams.entrySet()) {
                httpBuilder.addQueryParameter(param.getKey(),
param.getValue());
            }
            Request request = new
Request.Builder().url(httpBuilder.build()).build();
```

```
            try{
                Response response = client.newCall(request).execute();
                if(response.isSuccessful()){
                    jsonRes = response.body().string();
                    Map res = new Gson().fromJson(jsonRes, Map.class);
                    HTMLResultViewBuilder viewBuilder =
HTMLResultViewBuilder.newBuilder();
                    viewBuilder = viewBuilder.setHeader((List)((Map)
res.get("head")).get("vars"));
                    for(Map row:
(List<Map>)((Map)res.get("results")).get("bindings")){
                        viewBuilder = viewBuilder.addRow(row);
                    }
                    htmlRes = viewBuilder.build();
                    System.out.println(htmlRes);
                } else {
                    System.out.println("bad request");
                    System.out.println(response.body().string());
                }
            } catch (IOException e){
                e.printStackTrace();
            }

        }
        return htmlRes;
    }
}
```

GeminiApiRequest.java

```
package org.example.demo;

import com.google.gson.Gson;
import com.google.gson.JsonArray;
import com.google.gson.JsonObject;
import okhttp3.*;

import java.io.IOException;

public class GeminiApiRequest {

    public static String generateContent(String queryText) {
        OkHttpClient client = new OkHttpClient();

        // Replace "YOUR_API_KEY" with your actual Google API key
        String apiKey = "AIzaSyAXdsMLYA8_g95nBiPuQ35-wX4TSXbGea4";
        String url =
"https://generativelanguage.googleapis.com/v1beta/models/gemini-
pro:generateContent?key=" + apiKey;
        promptEngineringConstants prompts = new promptEngineringConstants();
```

```
        String query = prompts.queryHeader
+prompts.breaker+prompts.turtleOfOntology2 +prompts.breaker + "prompt:\n" +
queryText;
        String jsonReq = "{\"contents\":[{\"parts\":[{\"text\":\"" + query +
"\"}]}]}";
        System.out.println(jsonReq);
        MediaType mediaType = MediaType.parse("application/json");
        RequestBody body = RequestBody.create(jsonReq, mediaType);
        // Create a Gson instance
        Request request = new Request.Builder()
                .url(url)
                .post(body)
                .addHeader("Content-Type", "application/json")
                .build();

        // Execute the request and handle the response
        try {
            Response response = client.newCall(request).execute();
            if (response.isSuccessful()) {
                Gson gson = new Gson();
                JsonObject jsonObject =
gson.fromJson(response.body().string(), JsonObject.class);
                System.out.println(jsonObject);

                // Extract the "candidates" array directly
                JsonArray candidatesArray =
jsonObject.getAsJsonArray("candidates");
                System.out.println(candidatesArray);

                // Extract the "text" field from the first element of the
"parts" array within the "content" object
                String text =
candidatesArray.get(0).getAsJsonObject().getAsJsonObject("content").getAsJson
Array("parts").get(0).getAsJsonObject().get("text").getAsString();
                System.out.println("OMAR ::: " + text);
                text = text.replaceAll("```", "");
                text = text.replaceAll("sparql", "");

                return text;
            } else {
                System.out.println("Request failed: " + response);
                return response.body().string();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return apiKey;
    }


    public static String refineContent(String queryText) {
        OkHttpClient client = new OkHttpClient();
```

```
        // Replace "YOUR_API_KEY" with your actual Google API key
        String apiKey = "AIzaSyAXdsMLYA8_g95nBiPuQ35-wX4TSXbGea4";
        String url =
"https://generativelanguage.googleapis.com/v1beta/models/gemini-
pro:generateContent?key=" + apiKey;
        String queryHeader = "fix this sparql query and assume that prefixes
are defined and return it without any extra characters: ";
        String query = queryHeader + queryText;
        // JSON request body
        String jsonReq = "{\"contents\":[{\"parts\":[{\"text\":\"" + query +
"\"}]}]}";

        MediaType mediaType = MediaType.parse("application/json");
        RequestBody body = RequestBody.create(jsonReq, mediaType);
        // Create a Gson instance

        Request request = new Request.Builder()
                .url(url)
                .post(body)
                .addHeader("Content-Type", "application/json")
                .build();

        // Execute the request and handle the response
        try {
            Response response = client.newCall(request).execute();
            if (response.isSuccessful()) {
                Gson gson = new Gson();
                JsonObject jsonObject =
gson.fromJson(response.body().string(), JsonObject.class);
                System.out.println(jsonObject);

                // Extract the "candidates" array directly
                JsonArray candidatesArray =
jsonObject.getAsJsonArray("candidates");
                System.out.println(candidatesArray);

                // Extract the "text" field from the first element of the
"parts" array within the "content" object
                String text =
candidatesArray.get(0).getAsJsonObject().getAsJsonObject("content").getAsJson
Array("parts").get(0).getAsJsonObject().get("text").getAsString();
                System.out.println("OMAR ::: " + text);
                text = text.replaceAll("```", "");
                text = text.replaceAll("sparql", "");

                return text;
            } else {
                System.out.println("Request failed: " + response);
                return response.body().string();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
```

```
        return apiKey;
    }
}
```

HTMLResultViewBuilder.java

```
package org.example.demo;

import java.util.List;
import java.util.Map;

public class HTMLResultViewBuilder {


    private final String HEADER = """
         <!DOCTYPE html>
         <html lang="en">
           <head>
             <meta charset="utf-8">
             <meta http-equiv="X-UA-Compatible" content="IE=edge">
             <meta name="viewport" content="width=device-width, initial-
scale=1, shrink-to-fit=no">
             <link rel="icon" href="/favicon.ico">
             <head>
               <meta charset="UTF-8">
               <link rel="apple-touch-icon" type="image/png"
href="https://cpwebassets.codepen.io/assets/favicon/apple-touch-icon-
5ae1a0698dcc2402e9712f7d01ed509a57814f994c660df9f7a952f3060705ee.png">
               <meta name="apple-mobile-web-app-title" content="CodePen">
               <link rel="shortcut icon" type="image/x-icon"
href="https://cpwebassets.codepen.io/assets/favicon/favicon-
aec34940fbc1a6e787974dcd360f2c6b63348d4b1f4e06c77743096d55480f33.ico">
               <link rel="mask-icon" type="image/x-icon"
href="https://cpwebassets.codepen.io/assets/favicon/logo-pin-
b4b4269c16397ad2f0f7a01bcdf513a1994f4c94b8af2f191c09eb0d601762b1.svg"
color="#111">
               <link rel="canonical"
href="https://codepen.io/pixelchar/pen/rNaKLM">
               <script
src="https://cdnjs.cloudflare.com/ajax/libs/modernizr/2.8.3/modernizr.min.js"
type="text/javascript"></script>
               <meta name="viewport" content="width=device-width">
               <style media="" data-
href="https://cdnjs.cloudflare.com/ajax/libs/normalize/5.0.0/normalize.min.cs
s">
                 button,
                 hr,
                 input {
                   overflow: visible
                 }

                 audio,
                 canvas,
```

```
progress,
video {
  display: inline-block
}

progress,
sub,
sup {
  vertical-align: baseline
}

html {
  font-family: sans-serif;
  line-height: 1.15;
  -ms-text-size-adjust: 100%;
  -webkit-text-size-adjust: 100%
}

body {
  margin: 0
}

menu,
article,
aside,
details,
footer,
header,
nav,
section {
  display: block
}

h1 {
  font-size: 2em;
  margin: .67em 0
}

figcaption,
figure,
main {
  display: block
}

figure {
  margin: 1em 40px
}

hr {
  box-sizing: content-box;
  height: 0
}
```

```
code,
kbd,
pre,
samp {
  font-family: monospace, monospace;
  font-size: 1em
}

a {
  background-color: transparent;
  -webkit-text-decoration-skip: objects
}

a:active,
a:hover {
  outline-width: 0
}

abbr[title] {
  border-bottom: none;
  text-decoration: underline;
  text-decoration: underline dotted
}

b,
strong {
  font-weight: bolder
}

dfn {
  font-style: italic
}

mark {
  background-color: #ff0;
  color: #000
}

small {
  font-size: 80%
}

sub,
sup {
  font-size: 75%;
  line-height: 0;
  position: relative
}

sub {
  bottom: -.25em
}
```

```
sup {
  top: -.5em
}

audio:not([controls]) {
  display: none;
  height: 0
}

img {
  border-style: none
}

svg:not(:root) {
  overflow: hidden
}

button,
input,
optgroup,
select,
textarea {
  font-family: sans-serif;
  font-size: 100%;
  line-height: 1.15;
  margin: 0
}

button,
input {}

button,
select {
  text-transform: none
}

[type=submit],
[type=reset],
button,
html [type=button] {
  -webkit-appearance: button
}

[type=button]::-moz-focus-inner,
[type=reset]::-moz-focus-inner,
[type=submit]::-moz-focus-inner,
button::-moz-focus-inner {
  border-style: none;
  padding: 0
}

[type=button]:-moz-focusring,
[type=reset]:-moz-focusring,
```

```
[type=submit]:-moz-focusring,
button:-moz-focusring {
  outline: ButtonText dotted 1px
}

fieldset {
  border: 1px solid silver;
  margin: 0 2px;
  padding: .35em .625em .75em
}

legend {
  box-sizing: border-box;
  color: inherit;
  display: table;
  max-width: 100%;
  padding: 0;
  white-space: normal
}

progress {}

textarea {
  overflow: auto
}

[type=checkbox],
[type=radio] {
  box-sizing: border-box;
  padding: 0
}

[type=number]::-webkit-inner-spin-button,
[type=number]::-webkit-outer-spin-button {
  height: auto
}

[type=search] {
  -webkit-appearance: textfield;
  outline-offset: -2px
}

[type=search]::-webkit-search-cancel-button,
[type=search]::-webkit-search-decoration {
  -webkit-appearance: none
}

::-webkit-file-upload-button {
  -webkit-appearance: button;
  font: inherit
}

summary {
```

```
      display: list-item
    }

    [hidden],
    template {
      display: none
    }

    /*# sourceMappingURL=normalize.min.css.map */
  </style>
  <style>
    html {
      box-sizing: border-box;
    }

    *,
    *:before,
    *:after {
      box-sizing: inherit;
    }

    body {
      font-family: system-ui, -apple-system, BlinkMacSystemFont,
"Avenir Next", "Avenir", "Segoe UI", "Lucida Grande", "Helvetica Neue",
"Helvetica", "Fira Sans", "Roboto", "Noto", "Droid Sans", "Cantarell",
"Oxygen", "Ubuntu", "Franklin Gothic Medium", "Century Gothic", "Liberation
Sans", sans-serif;
      color: rgba(0, 0, 0, 0.87);
    }

    a {
      color: #26890d;
    }

    a:hover,
    a:focus {
      color: #046a38;
    }

    .container {
      margin: 5% 3%;
    }

    @media (min-width: 48em) {
      .container {
        margin: 2%;
      }
    }

    @media (min-width: 75em) {
      .container {
        margin: 2em auto;
        max-width: 75em;
```

```
  }
}

.responsive-table {
  width: 100%;
  margin-bottom: 1.5em;
  border-spacing: 0;
}

@media (min-width: 48em) {
  .responsive-table {
    font-size: 0.9em;
  }
}

@media (min-width: 62em) {
  .responsive-table {
    font-size: 1em;
  }
}

.responsive-table thead {
  position: absolute;
  clip: rect(1px 1px 1px 1px);
  /* IE6, IE7 */
  padding: 0;
  border: 0;
  height: 1px;
  width: 1px;
  overflow: hidden;
}

@media (min-width: 48em) {
  .responsive-table thead {
    position: relative;
    clip: auto;
    height: auto;
    width: auto;
    overflow: auto;
  }
}

.responsive-table thead th {
  background-color: #26890d;
  border: 1px solid #86bc25;
  font-weight: normal;
  text-align: center;
  color: white;
}

.responsive-table thead th:first-of-type {
  text-align: left;
}
```

```
.responsive-table tbody,
.responsive-table tr,
.responsive-table th,
.responsive-table td {
  display: block;
  padding: 0;
  text-align: left;
  white-space: normal;
}

@media (min-width: 48em) {
  .responsive-table tr {
    display: table-row;
  }
}

.responsive-table th,
.responsive-table td {
  padding: 0.5em;
  vertical-align: middle;
}

@media (min-width: 30em) {

  .responsive-table th,
  .responsive-table td {
    padding: 0.75em 0.5em;
  }
}

@media (min-width: 48em) {

  .responsive-table th,
  .responsive-table td {
    display: table-cell;
    padding: 0.5em;
  }
}

@media (min-width: 62em) {

  .responsive-table th,
  .responsive-table td {
    padding: 0.75em 0.5em;
  }
}

@media (min-width: 75em) {

  .responsive-table th,
  .responsive-table td {
    padding: 0.75em;
```

```
  }
}

.responsive-table caption {
  margin-bottom: 1em;
  font-size: 1em;
  font-weight: bold;
  text-align: center;
}

@media (min-width: 48em) {
  .responsive-table caption {
    font-size: 1.5em;
  }
}

.responsive-table tfoot {
  font-size: 0.8em;
  font-style: italic;
}

@media (min-width: 62em) {
  .responsive-table tfoot {
    font-size: 0.9em;
  }
}

@media (min-width: 48em) {
  .responsive-table tbody {
    display: table-row-group;
  }
}

.responsive-table tbody tr {
  margin-bottom: 1em;
}

@media (min-width: 48em) {
  .responsive-table tbody tr {
    display: table-row;
    border-width: 1px;
  }
}

.responsive-table tbody tr:last-of-type {
  margin-bottom: 0;
}

@media (min-width: 48em) {
  .responsive-table tbody tr:nth-of-type(even) {
    background-color: rgba(0, 0, 0, 0.12);
  }
}
```

```
.responsive-table tbody th[scope=row] {
  background-color: #26890d;
  color: white;
}

@media (min-width: 30em) {
  .responsive-table tbody th[scope=row] {
    border-left: 1px solid #86bc25;
    border-bottom: 1px solid #86bc25;
  }
}

@media (min-width: 48em) {
  .responsive-table tbody th[scope=row] {
    background-color: transparent;
    color: #000001;
    text-align: left;
  }
}

.responsive-table tbody td {
  text-align: right;
}

@media (min-width: 48em) {
  .responsive-table tbody td {
    border-left: 1px solid #86bc25;
    border-bottom: 1px solid #86bc25;
    text-align: center;
  }
}

@media (min-width: 48em) {
  .responsive-table tbody td:last-of-type {
    border-right: 1px solid #86bc25;
  }
}

.responsive-table tbody td[data-type=currency] {
  text-align: right;
}

.responsive-table tbody td[data-title]:before {
  content: attr(data-title);
  float: left;
  font-size: 0.8em;
  color: rgba(0, 0, 0, 0.54);
}

@media (min-width: 30em) {
  .responsive-table tbody td[data-title]:before {
    font-size: 0.9em;
```

```
                  }
                }

                @media (min-width: 48em) {
                  .responsive-table tbody td[data-title]:before {
                    content: none;
                  }
                }
              </style>
              <script>
                window.console = window.console || function(t) {};
              </script>
              <script
src="https://cdnjs.cloudflare.com/ajax/libs/prefixfree/1.0.7/prefixfree.min.j
s"></script>
            </head>
          <body>
            <table class="responsive-table">
        """;

    private String tableBody = "";

    private final String FOOTER = """
                <table>
                </body>
                </html>
            """;

    private String tableHeader = "";

    private HTMLResultViewBuilder() {

    }

   public static HTMLResultViewBuilder newBuilder(){
        return new HTMLResultViewBuilder();
   }

   public HTMLResultViewBuilder setHeader(List<Object> headers){
        String tableHeaderH = """
                <thead>
                    <tr>
                """;

        String tableHeaderF = """
                    </tr>
                </thead>
                """;
        StringBuilder tableRow = new StringBuilder();
       for (Object header : headers) {
           tableRow.append("<th scope='col'>").append((String)
header).append("</th>");
       }
```

```
        this.tableHeader = tableHeaderH + tableRow + tableHeaderF;
        return this;
    }

    public HTMLResultViewBuilder addRow(Map<Object, Object> row){
        StringBuilder body = new StringBuilder("<tr>");

         for(Map.Entry<Object, Object> entry : row.entrySet()){
             body.append("<td>").append(((Map<?, ?>)
(entry.getValue())).get("value")).append("</td>");
         }
        body.append("</tr>");

         tableBody += body;
         return this;
    }

    public String build(){
         return HEADER + tableHeader + tableBody + FOOTER;
    }

    public void reset(){
        tableHeader = "";
        tableBody = "";
    }

}
```

promptEngineringConstants.java

```
package org.example.demo;

public class promptEngineringConstants {
    String breaker = "\n";
    String queryHeader = "write sparql query for the given prompt and define
all the prefixes needed accordingly and return only the sparql query without
any extra characters and make sure it's valid and will return answer from the
defined ontology: ";

    String turtleOfOntology2 = """
            Ontology:
            @prefix : <http://localhost:3030/#/dataset/SportsOntology> .
            @prefix owl: <http://www.w3.org/2002/07/owl#> .
            @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
            @prefix xml: <http://www.w3.org/XML/1998/namespace> .
            @prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
            @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
            @base <http://localhost:3030/#/dataset/SportsOntology> .

            <http://localhost:3030/#/dataset/SportsOntology> rdf:type
owl:Ontology .
```

```
            ###############################################################
            #    Object Properties
            ###############################################################

            ###  http://localhost:3030/#/dataset/SportsOntology#belongs_to
            :belongs_to rdf:type owl:ObjectProperty ;
                        rdfs:subPropertyOf owl:topObjectProperty ;
                        rdfs:domain :Athletes ;
                        rdfs:range :Teams .


            ###  http://localhost:3030/#/dataset/SportsOntology#coached_by
            :coached_by rdf:type owl:ObjectProperty ;
                        rdfs:subPropertyOf owl:topObjectProperty ;
                        rdfs:domain :Teams ;
                        rdfs:range :Coach .


            ###  http://localhost:3030/#/dataset/SportsOntology#defeated_by
            :defeated_by rdf:type owl:ObjectProperty ;
                         rdfs:subPropertyOf owl:topObjectProperty ;
                         rdfs:domain :Teams ;
                         rdfs:range :Teams .


            ###
http://localhost:3030/#/dataset/SportsOntology#has_home_venue
            :has_home_venue rdf:type owl:ObjectProperty ;
                            rdfs:subPropertyOf owl:topObjectProperty ;
                            rdfs:domain :Teams ;
                            rdfs:range :Venues .


            ###
http://localhost:3030/#/dataset/SportsOntology#participates_in
            :participates_in rdf:type owl:ObjectProperty ;
                             rdfs:subPropertyOf owl:topObjectProperty ;
                             rdfs:domain :Teams ;
                             rdfs:range :Competitions .


            ###
http://localhost:3030/#/dataset/SportsOntology#takes_place_at
            :takes_place_at rdf:type owl:ObjectProperty ;
                            rdfs:subPropertyOf owl:topObjectProperty ;
                            rdfs:domain :Competitions ;
                            rdfs:range :Venues .


            ###############################################################
            #    Data properties
            ###############################################################
```

```
            ###  http://localhost:3030/#/dataset/SportsOntology#Address
            :Address rdf:type owl:DatatypeProperty ;
                     rdfs:subPropertyOf owl:topDataProperty ;
                     rdfs:domain :Venues ;
                     rdfs:range xsd:string .


            ###  http://localhost:3030/#/dataset/SportsOntology#Athlete_Age
            :Athlete_Age rdf:type owl:DatatypeProperty ;
                         rdfs:subPropertyOf owl:topDataProperty ;
                         rdfs:domain :Athletes ,
                                     [ rdf:type owl:Restriction ;
                                       owl:onProperty :Athlete_Age ;
                                       owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
                                                            owl:onDatatype
xsd:integer ;

owl:withRestrictions ( [ xsd:minExclusive 15

]

[ xsd:maxInclusive 100

]

)
                                                          ]
                                     ] ;
                         rdfs:range xsd:integer .


            ###  http://localhost:3030/#/dataset/SportsOntology#Athlete_Name
            :Athlete_Name rdf:type owl:DatatypeProperty ;
                          rdfs:subPropertyOf owl:topDataProperty ;
                          rdfs:domain :Athletes ;
                          rdfs:range xsd:string .


            ###
http://localhost:3030/#/dataset/SportsOntology#Athlete_Nationality
            :Athlete_Nationality rdf:type owl:DatatypeProperty ;
                                 rdfs:subPropertyOf owl:topDataProperty ;
                                 rdfs:domain :Athletes ;
                                 rdfs:range xsd:string .


            ###
http://localhost:3030/#/dataset/SportsOntology#Athlete_Salary
            :Athlete_Salary rdf:type owl:DatatypeProperty ;
                            rdfs:subPropertyOf owl:topDataProperty ;
                            rdfs:domain :Athletes ,
```

```
                                        [ rdf:type owl:Restriction ;
                                          owl:onProperty :Athlete_Salary ;
                                          owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
                                                               owl:onDatatype
xsd:float ;

owl:withRestrictions ( [ xsd:minExclusive '10000.0'^^xsd:float

]

[ xsd:maxInclusive '5.0E8'^^xsd:float

]

)
                                                           ]
                                    ] ;
                            rdfs:range xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#Budget
            :Budget rdf:type owl:DatatypeProperty ;
                    rdfs:subPropertyOf owl:topDataProperty ;
                    rdfs:domain :Teams ,
                                [ rdf:type owl:Restriction ;
                                  owl:onProperty :Budget ;
                                  owl:someValuesFrom [ rdf:type rdfs:Datatype
;
                                                       owl:onDatatype
xsd:float ;
                                                       owl:withRestrictions (
[ xsd:minExclusive '1000000.0'^^xsd:float

]

[ xsd:maxInclusive '2.0E9'^^xsd:float

]
                                                                            )
                                                   ]
                                ] ;
                    rdfs:range xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#Capacity
            :Capacity rdf:type owl:DatatypeProperty ;
                      rdfs:subPropertyOf owl:topDataProperty ;
                      rdfs:domain :Venues ,
                                  [ rdf:type owl:Restriction ;
                                    owl:onProperty :Capacity ;
                                    owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
```

```
                                                        owl:onDatatype
xsd:integer ;
                                                        owl:withRestrictions
( [ xsd:minExclusive 5000

]

[ xsd:maxInclusive 150000

]

)
                                                    ]
                                ] ;
                        rdfs:range xsd:integer .


            ###  http://localhost:3030/#/dataset/SportsOntology#Coach_Age
            :Coach_Age rdf:type owl:DatatypeProperty ;
                       rdfs:subPropertyOf owl:topDataProperty ;
                       rdfs:domain :Coach ,
                                   [ rdf:type owl:Restriction ;
                                     owl:onProperty :Coach_Age ;
                                     owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
                                                          owl:onDatatype
xsd:integer ;

owl:withRestrictions ( [ xsd:minExclusive 15

]

[ xsd:maxInclusive 100

]

)
                                                    ]
                                ] ;
                        rdfs:range xsd:integer .


            ###  http://localhost:3030/#/dataset/SportsOntology#Coach_Name
            :Coach_Name rdf:type owl:DatatypeProperty ;
                        rdfs:subPropertyOf owl:topDataProperty ;
                        rdfs:domain :Coach ;
                        rdfs:range xsd:string .


            ###
http://localhost:3030/#/dataset/SportsOntology#Coach_Nationality
            :Coach_Nationality rdf:type owl:DatatypeProperty ;
                               rdfs:subPropertyOf owl:topDataProperty ;
```

```
                              rdfs:domain :Coach ;
                              rdfs:range xsd:string .


            ###  http://localhost:3030/#/dataset/SportsOntology#Coach_Salary
            :Coach_Salary rdf:type owl:DatatypeProperty ;
                          rdfs:subPropertyOf owl:topDataProperty ;
                          rdfs:domain :Coach ,
                                      [ rdf:type owl:Restriction ;
                                        owl:onProperty :Coach_Salary ;
                                        owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
                                                             owl:onDatatype
xsd:float ;

owl:withRestrictions ( [ xsd:minExclusive '10000.0'^^xsd:float

]

[ xsd:maxInclusive '5.0E8'^^xsd:float

]

)
                                                           ]
                                      ] ;
                          rdfs:range xsd:float .


            ###
http://localhost:3030/#/dataset/SportsOntology#Competition_Name
            :Competition_Name rdf:type owl:DatatypeProperty ;
                              rdfs:subPropertyOf owl:topDataProperty ;
                              rdfs:domain :Competitions ;
                              rdfs:range xsd:string .


            ###
http://localhost:3030/#/dataset/SportsOntology#Governing_body
            :Governing_body rdf:type owl:DatatypeProperty ;
                            rdfs:subPropertyOf owl:topDataProperty ;
                            rdfs:domain :Competitions ,
                                        [ rdf:type owl:Class ;
                                          owl:unionOf ( [ rdf:type
owl:Restriction ;
                                                          owl:onProperty
:Governing_body ;
                                                          owl:hasValue 'AFC'
                                                        ]
                                                        [ rdf:type
owl:Restriction ;
                                                          owl:onProperty
:Governing_body ;
```

```
                                                        owl:hasValue 'CAF'
                                                      ]
                                                      [ rdf:type
owl:Restriction ;
                                                        owl:onProperty
:Governing_body ;
                                                        owl:hasValue
'CONMEBOL'
                                                      ]
                                                      [ rdf:type
owl:Restriction ;
                                                        owl:onProperty
:Governing_body ;
                                                        owl:hasValue
'LIBERTADORES'
                                                      ]
                                                      [ rdf:type
owl:Restriction ;
                                                        owl:onProperty
:Governing_body ;
                                                        owl:hasValue 'UEFA'
                                                      ]
                                                    )
                                    ] ;
                            rdfs:range xsd:string .


            ###  http://localhost:3030/#/dataset/SportsOntology#Height
            :Height rdf:type owl:DatatypeProperty ;
                    rdfs:subPropertyOf owl:topDataProperty ;
                    rdfs:domain :Athletes ;
                    rdfs:range xsd:float .


            ###
http://localhost:3030/#/dataset/SportsOntology#Number_of_participants
            :Number_of_participants rdf:type owl:DatatypeProperty ;
                                    rdfs:subPropertyOf owl:topDataProperty ;
                                    rdfs:domain :Competitions ,
                                                [ rdf:type owl:Restriction ;
                                                  owl:onProperty
:Number_of_participants ;
                                                  owl:someValuesFrom [
rdf:type rdfs:Datatype ;

owl:onDatatype xsd:integer ;

owl:withRestrictions ( [ xsd:minInclusive 2

]

[ xsd:maxInclusive 256
```

```
]

)
                                                                    ]
                                                ] ;
                                    rdfs:range xsd:integer .


            ###
http://localhost:3030/#/dataset/SportsOntology#Number_of_players
            :Number_of_players rdf:type owl:DatatypeProperty ;
                               rdfs:subPropertyOf owl:topDataProperty ;
                               rdfs:domain :Teams ,
                                           [ rdf:type owl:Restriction ;
                                             owl:onProperty
:Number_of_players ;
                                             owl:someValuesFrom [ rdf:type
rdfs:Datatype ;

owl:onDatatype xsd:integer ;

owl:withRestrictions ( [ xsd:minInclusive 1

]

[ xsd:maxInclusive 100

]

)
                                                                  ]
                                        ] ;
                               rdfs:range xsd:integer .


            ###  http://localhost:3030/#/dataset/SportsOntology#Prize_money
            :Prize_money rdf:type owl:DatatypeProperty ;
                         rdfs:subPropertyOf owl:topDataProperty ;
                         rdfs:domain :Competitions ,
                                     [ rdf:type owl:Restriction ;
                                       owl:onProperty :Prize_money ;
                                       owl:someValuesFrom [ rdf:type
rdfs:Datatype ;
                                                            owl:onDatatype
xsd:float ;

owl:withRestrictions ( [ xsd:minExclusive '100000.0'^^xsd:float

]

[ xsd:maxInclusive '1.0E9'^^xsd:float

]
```

```
)
                                                        ]
                                    ] ;
                          rdfs:range xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#Team_Name
            :Team_Name rdf:type owl:DatatypeProperty ;
                       rdfs:subPropertyOf owl:topDataProperty ;
                       rdfs:domain :Teams ;
                       rdfs:range xsd:string .


            ###  http://localhost:3030/#/dataset/SportsOntology#Venue_Name
            :Venue_Name rdf:type owl:DatatypeProperty ;
                        rdfs:subPropertyOf owl:topDataProperty ;
                        rdfs:domain :Venues ;
                        rdfs:range xsd:string .


            #################################################################
            #    Classes
            #################################################################

            ###  http://localhost:3030/#/dataset/SportsOntology#Athletes
            :Athletes rdf:type owl:Class ;
                      owl:equivalentClass [ rdf:type owl:Restriction ;
                                            owl:onProperty :belongs_to ;
                                            owl:qualifiedCardinality
'1'^^xsd:nonNegativeInteger ;
                                            owl:onClass :Teams
                                          ] ;
                      rdfs:subClassOf :Sports .


            ###  http://localhost:3030/#/dataset/SportsOntology#Coach
            :Coach rdf:type owl:Class ;
                   rdfs:subClassOf :Sports .


            ###  http://localhost:3030/#/dataset/SportsOntology#Competitions
            :Competitions rdf:type owl:Class ;
                          owl:equivalentClass [ rdf:type owl:Restriction ;
                                                owl:onProperty
:takes_place_at ;
                                                owl:minQualifiedCardinality
'1'^^xsd:nonNegativeInteger ;
                                                owl:onClass :Venues
                                              ] ;
                          rdfs:subClassOf :Sports .
```

```
            ###  http://localhost:3030/#/dataset/SportsOntology#Sports
            :Sports rdf:type owl:Class .


            ###  http://localhost:3030/#/dataset/SportsOntology#Teams
            :Teams rdf:type owl:Class ;
                   owl:equivalentClass [ rdf:type owl:Restriction ;
                                         owl:onProperty :participates_in ;
                                         owl:minQualifiedCardinality
'1'^^xsd:nonNegativeInteger ;
                                         owl:onClass :Competitions
                                       ] ,
                                       [ rdf:type owl:Restriction ;
                                         owl:onProperty :coached_by ;
                                         owl:qualifiedCardinality
'1'^^xsd:nonNegativeInteger ;
                                         owl:onClass :Coach
                                       ] ,
                                       [ rdf:type owl:Restriction ;
                                         owl:onProperty :has_home_venue ;
                                         owl:qualifiedCardinality
'1'^^xsd:nonNegativeInteger ;
                                         owl:onClass :Venues
                                       ] ,
                                       [ rdf:type owl:Restriction ;
                                         owl:onProperty :defeated_by ;
                                         owl:maxQualifiedCardinality
'20'^^xsd:nonNegativeInteger ;
                                         owl:onClass :Teams
                                       ] ,
                                       [ rdf:type owl:Restriction ;
                                         owl:onProperty :participates_in ;
                                         owl:maxQualifiedCardinality
'7'^^xsd:nonNegativeInteger ;
                                         owl:onClass :Competitions
                                       ] ;
                   rdfs:subClassOf :Sports .


            ###  http://localhost:3030/#/dataset/SportsOntology#Venues
            :Venues rdf:type owl:Class ;
                    rdfs:subClassOf :Sports .


            ###################################################################
            #    Individuals
            ###################################################################

            ###  http://localhost:3030/#/dataset/SportsOntology#Anfield
            :Anfield rdf:type owl:NamedIndividual ,
                              :Venues ;
                     :Address 'Liverpool' ;
                     :Capacity 50000 ;
```

```
                :Venue_Name 'Anfield' .


        ###  http://localhost:3030/#/dataset/SportsOntology#Arsenal
        :Arsenal rdf:type owl:NamedIndividual ,
                          :Teams ;
                 :coached_by :CoachArteta ;
                 :defeated_by :ManCity ;
                 :has_home_venue :Emirates ;
                 :participates_in :ChampionsLeague ,
                                  :PremierLeague ;
                 :Budget '3.0E8'^^xsd:float ;
                 :Number_of_players 36 ;
                 :Team_Name 'Arsenal' .


        ###  http://localhost:3030/#/dataset/SportsOntology#Azpilicueta
        :Azpilicueta rdf:type owl:NamedIndividual ,
                              :Athletes ;
                     :belongs_to :Chelsea ;
                     :Athlete_Age 32 ;
                     :Athlete_Name 'Azpilicueta' ;
                     :Athlete_Nationality 'Spanish' ;
                     :Athlete_Salary '1.2E7'^^xsd:float .


        ###
http://localhost:3030/#/dataset/SportsOntology#ChampionsLeague
        :ChampionsLeague rdf:type owl:NamedIndividual ,
                                  :Competitions ;
                         :takes_place_at :Anfield ;
                         :Competition_Name 'ChampionsLeague' ;
                         :Governing_body 'UEFA' ;
                         :Number_of_participants 32 ;
                         :Prize_money '1.0E9'^^xsd:float .


        ###  http://localhost:3030/#/dataset/SportsOntology#Chelsea
        :Chelsea rdf:type owl:NamedIndividual ,
                          :Teams ;
                 :coached_by :CoachTuchel ;
                 :defeated_by :Arsenal ,
                              :Liverpool ,
                              :ManCity ;
                 :has_home_venue :StamfordBridge ;
                 :participates_in :ChampionsLeague ,
                                  :PremierLeague ;
                 :Budget '6.0E8'^^xsd:float ;
                 :Number_of_players 55 ;
                 :Team_Name 'Chelsea' .


        ###  http://localhost:3030/#/dataset/SportsOntology#CoachArteta
```

```
            :CoachArteta rdf:type owl:NamedIndividual ,
                                 :Coach ;
                         :Coach_Age 45 ;
                         :Coach_Name 'CoachArteta' ;
                         :Coach_Nationality 'Spanish' ;
                         :Coach_Salary '6000000.0'^^xsd:float .


            ###
http://localhost:3030/#/dataset/SportsOntology#CoachGuardiola
            :CoachGuardiola rdf:type owl:NamedIndividual ,
                                    :Coach ;
                           :Coach_Age 55 ;
                           :Coach_Name 'CoachGuardiola' ;
                           :Coach_Nationality 'Spanish' ;
                           :Coach_Salary '1.5E7'^^xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#CoachKlopp
            :CoachKlopp rdf:type owl:NamedIndividual ,
                                 :Coach ;
                        :Coach_Age 53 ;
                        :Coach_Name 'CoachKlopp' ;
                        :Coach_Nationality 'German' ;
                        :Coach_Salary '1.0E7'^^xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#CoachMourinho
            :CoachMourinho rdf:type owl:NamedIndividual ,
                                   :Coach ;
                          :Coach_Age 60 ;
                          :Coach_Name 'CoachMourinho' ;
                          :Coach_Nationality 'Portuguese' ;
                          :Coach_Salary '1.8E7'^^xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#CoachTuchel
            :CoachTuchel rdf:type owl:NamedIndividual ,
                                 :Coach ;
                         :Coach_Age 42 ;
                         :Coach_Name 'CoachTuchel' ;
                         :Coach_Nationality 'German' ;
                         :Coach_Salary '1.6E7'^^xsd:float .


            ###  http://localhost:3030/#/dataset/SportsOntology#DeBruyne
            :DeBruyne rdf:type owl:NamedIndividual ,
                              :Athletes ;
                     :belongs_to :ManCity ;
                     :Athlete_Age 31 ;
                     :Athlete_Name 'DeBruyne' ;
                     :Athlete_Nationality 'Belgian' ;
                     :Athlete_Salary '3.0E7'^^xsd:float .
```

```
###  http://localhost:3030/#/dataset/SportsOntology#Emirates
:Emirates rdf:type owl:NamedIndividual ,
                   :Venues ;
          :Address 'London' ;
          :Capacity 60000 ;
          :Venue_Name 'Emirates' .


###  http://localhost:3030/#/dataset/SportsOntology#Etihad
:Etihad rdf:type owl:NamedIndividual ,
                  :Venues ;
        :Address 'Manchester' ;
        :Capacity 55000 ;
        :Venue_Name 'Etihad' .


###  http://localhost:3030/#/dataset/SportsOntology#Firmino
:Firmino rdf:type owl:NamedIndividual ,
                  :Athletes ;
         :belongs_to :Liverpool ;
         :Athlete_Age 33 ;
         :Athlete_Name 'Firmino' ;
         :Athlete_Nationality 'Brazilian' ;
         :Athlete_Salary '1.5E7'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Haaland
:Haaland rdf:type owl:NamedIndividual ,
                  :Athletes ;
         :belongs_to :ManCity ;
         :Athlete_Age 21 ;
         :Athlete_Name 'Haaland' ;
         :Athlete_Nationality 'Norwegian' ;
         :Athlete_Salary '4.0E7'^^xsd:float ;
         :Height '1.94'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Kante
:Kante rdf:type owl:NamedIndividual ,
                 :Athletes ;
       :belongs_to :Chelsea ;
       :Athlete_Age 32 ;
       :Athlete_Name 'Kante' ;
       :Athlete_Nationality 'French' ;
       :Athlete_Salary '2.5E7'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Liverpool
:Liverpool rdf:type owl:NamedIndividual ,
                    :Teams ;
           :coached_by :CoachKlopp ;
```

```
            :defeated_by :Arsenal ,
                         :ManUtd ;
            :has_home_venue :Anfield ;
            :participates_in :ChampionsLeague ,
                             :PremierLeague ;
            :Budget '2.5E8'^^xsd:float ;
            :Number_of_players 30 ;
            :Team_Name 'Liverpool' .


###  http://localhost:3030/#/dataset/SportsOntology#ManCity
:ManCity rdf:type owl:NamedIndividual ,
                  :Teams ;
          :coached_by :CoachGuardiola ;
          :defeated_by :Liverpool ;
          :has_home_venue :Etihad ;
          :participates_in :ChampionsLeague ,
                           :PremierLeague ;
          :Budget '7.0E8'^^xsd:float ;
          :Number_of_players 40 ;
          :Team_Name 'ManCity' .


###  http://localhost:3030/#/dataset/SportsOntology#ManUtd
:ManUtd rdf:type owl:NamedIndividual ,
                 :Teams ;
         :coached_by :CoachMourinho ;
         :defeated_by :Arsenal ,
                      :Liverpool ,
                      :ManCity ;
         :has_home_venue :OldTrafford ;
         :participates_in :PremierLeague ;
         :Budget '4.6E8'^^xsd:float ;
         :Number_of_players 46 ;
         :Team_Name 'ManUtd' .


###  http://localhost:3030/#/dataset/SportsOntology#Odegaard
:Odegaard rdf:type owl:NamedIndividual ,
                   :Athletes ;
           :belongs_to :Arsenal ;
           :Athlete_Age 23 ;
           :Athlete_Name 'Odegaard' ;
           :Athlete_Nationality 'Norwegian' ;
           :Athlete_Salary '1.3E7'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#OldTrafford
:OldTrafford rdf:type owl:NamedIndividual ,
                      :Venues ;
              :Address 'Manchester' ;
              :Capacity 65000 ;
              :Venue_Name 'OldTrafford' .
```

```
###  http://localhost:3030/#/dataset/SportsOntology#PremierLeague
:PremierLeague rdf:type owl:NamedIndividual ,
                        :Competitions ;
               :takes_place_at :Anfield ,
                               :Emirates ,
                               :Etihad ,
                               :OldTrafford ,
                               :StamfordBridge ;
               :Competition_Name 'PremierLeague' ;
               :Governing_body 'UEFA' ;
               :Number_of_participants 20 ;
               :Prize_money '9.0E8'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Ramsdale
:Ramsdale rdf:type owl:NamedIndividual ,
                   :Athletes ;
          :belongs_to :Arsenal ;
          :Athlete_Age 24 ;
          :Athlete_Name 'Ramsdale' ;
          :Athlete_Nationality 'British' ;
          :Athlete_Salary '5000000.0'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Ronaldo
:Ronaldo rdf:type owl:NamedIndividual ,
                  :Athletes ;
         :belongs_to :ManUtd ;
         :Athlete_Age 38 ;
         :Athlete_Name 'Ronaldo' ;
         :Athlete_Nationality 'Portuguese' ;
         :Athlete_Salary '1.0E8'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Rooney
:Rooney rdf:type owl:NamedIndividual ,
                 :Athletes ;
        :belongs_to :ManUtd ;
        :Athlete_Age 40 ;
        :Athlete_Name 'Rooney' ;
        :Athlete_Nationality 'British' ;
        :Athlete_Salary '1.7E7'^^xsd:float .


###  http://localhost:3030/#/dataset/SportsOntology#Salah
:Salah rdf:type owl:NamedIndividual ,
                :Athletes ;
       :belongs_to :Liverpool ;
       :Athlete_Age 31 ;
       :Athlete_Name 'Salah' ;
       :Athlete_Nationality 'Egyptian' ;
```

```
                    :Athlete_Salary '3.1E7'^^xsd:float ;
                    :Height '1.75'^^xsd:float .


            ###
http://localhost:3030/#/dataset/SportsOntology#StamfordBridge
            :StamfordBridge rdf:type owl:NamedIndividual ,
                                        :Venues ;
                            :Address 'London' ;
                            :Capacity 45000 ;
                            :Venue_Name 'StamfordBridge' .


            ###  Generated by the OWL API (version 4.5.26.2023-07-
17T20:34:13Z) https://github.com/owlcs/owlapi



            """;
}
```

StringReplacement.java

```
package org.example.demo;

public class StringReplacement {
        public static void main(String[] args) {
            // Sample string containing the patterns to replace
            String inputString = "sparql\nSELECT ?player\nWHERE {\n  ?player
rdf:type dbpedia-owl:Person .\n  ?player foaf:name ?name .\n  FILTER
regex(?name, \"Cristiano\")\n}\n";

            // Replace occurrences of ``` or \n or \" with a space
            String replacedString = inputString.replaceAll("`|\\\\n|\\\\\"",
" ");
            replacedString = inputString.replaceAll("sparql", "");

            // Output the replaced string
            System.out.println("Replaced String:");
            System.out.println(replacedString);
        }
}
```

GITHUB:

https://github.com/oaayoub/ontologies-proj