

## VOCABULARY

In our Knowledge Graph Vocabulary, we have 11 classes, in which we have 3 classes that we have reused from existing vocabularies and rest are custom classes of our RoboProf, chatbot:

### **Reused Classes:**

1. University – we have used University from the DBpedia ontology, we deemed it appropriate as the concept of university is well defined as an existing resource in DBpedia.
2. Student – This class is reused from the Vivo ontology. Vivo provides well defined semantics and properties associated with students.
3. Courses – This class is reused from the Vivo ontology.

### **Custom Classes:**

1. Lecture
2. Lecture Content
3. Slides
4. Worksheets
5. Reading Material
6. Other Materials
7. Topics
8. Grade

We created custom classes to address specific needs and requirements of the RoboProf chatbot. Each custom class has a different property, in a way that there is proper inference in the vocabulary. It was necessary to create Lecture Content subclass hierarchy with Slides, Worksheets, Reading Material, Other Materials to ensure a comprehensive approach when querying about content material. This enhances the chatbot's capabilities in understanding and responding to user queries.

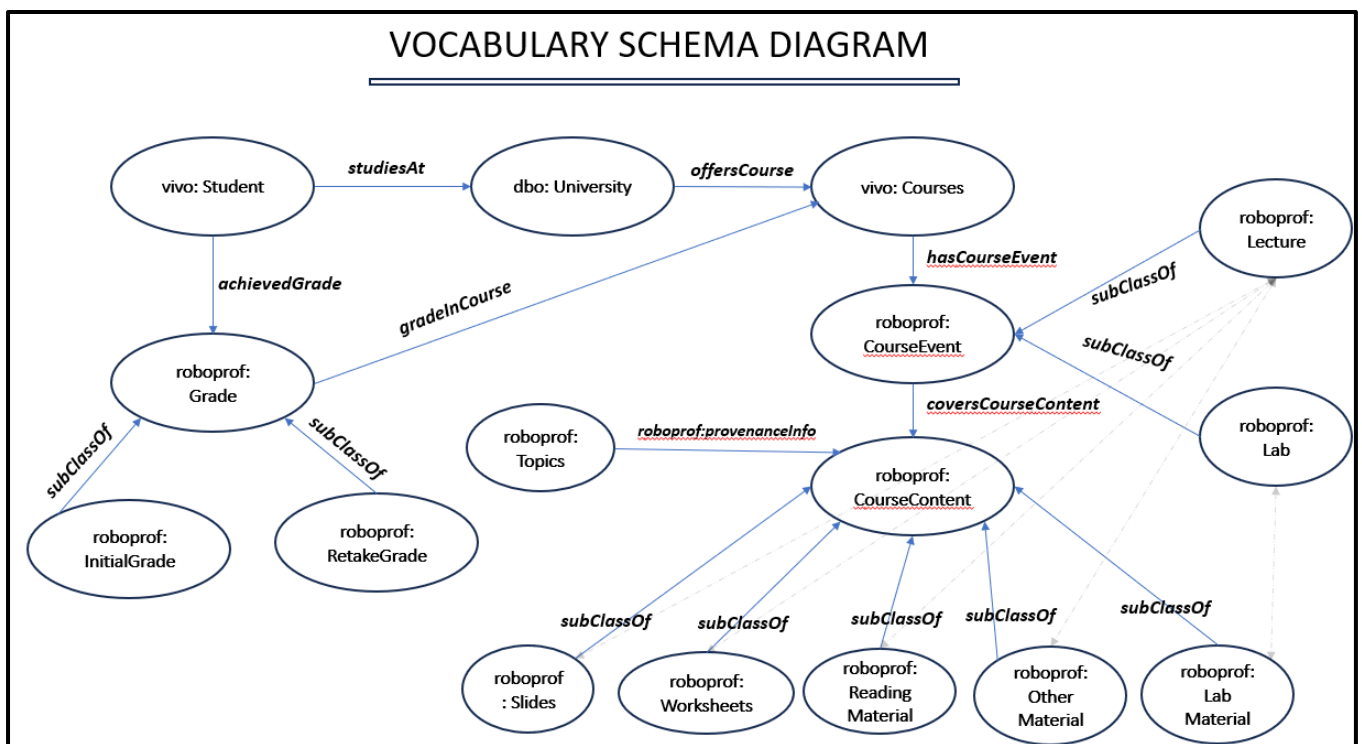
In our Knowledge Graph Vocabulary, we have the following properties:

### **Reused Properties:**

1. RDF : Type
2. Dbo : name
3. RDFS : seeAlso
4. Vivo : hasSubjectArea
5. Dbo : number
6. Vivo : courseCredits
7. Dcterms : description
8. FOAF : firstName
9. FOAF : lastName
10. Vcard : hasEmail

### Custom Properties:

1. courseOutline : specifies the link to outline of a Course
2. offersCourse : relation between university and a course
3. hasCourseEvent : relation between a course and a CourseEvent
4. coversCourseContent : relation between a course and a content type
5. coversTopic : relation between a lecture and a topic
6. provenanceInfo : specifies the content where the topic was found
7. studiesAt : relation between a Student and University
8. completedCourse : relation between a Student and Course
9. gradeValue : specifies the grade
10. achievedGrade : specifies the grade achieved by a Student.
11. gradeInCourse : specifies the grade that can be achieved in a particular Course.



## REPORT UPDATES:

### Fixes

We didn't have any changes from the last report, so we kept the base of the structure the same for part 2.

### Changes

As the second part of the assignment required restructuring of our schema(to incorporate courseEvents in part2 which was not part of the design in part1), we had to make some changes which included the following:

1. Earlier we were picking up topics from an excel file so we were using a property coversTopic to keep a track of it, we removed it as we didn't need it anymore, we covered this information using the provenanceInfo property of Topic.
2. We changed the range of the provenance info to lecture content so that the source of the topic could be identified.
3. In part 2, we needed to add Labs , in addition to Lectures so we created a new CourseEvent class. We created another class Lab. Now, Lecture and Lab are subclasses of CourseEvent.

We mapped the properties of Lecture from Part 1 to CourseEvent, properties of Lecture Content to Courses Content. We added new properties for Lab , called Lab Material.

4. According to the changed schema, we restructured all our SPARQL Queries and added new ones according to requirements.
5. As we had added Topics again, dynamically recognised by Dbpedia spotlight, we generated the triples again.
6. For our requirement of Lab Material, we added new material downloaded pdfs and converted them to text files for lab content, same we did for Reading, OtherMaterials.

## KNOWLEDGE BASE POPULATION

### Dataset

For creating the schema vocabulary, we manually designed and created the schema rdf file, to define classes and properties we will need to customise it, the file is called "vocabulary.rdf". It is written in turtle format.

Our dataset for populating data fields in our graph is an excel file which consists of multiple sheets representing classes from our vocabulary, called "dataset.xlsx". The file consists of the following sheets, each for a purpose:

- **Courses:**

This sheet consists of data from the given link: <https://opendata.concordia.ca/datasets/>, we used the file CATALOG.csv, to form courses, we chose this file as it reflected the most with the attributes required. We added COMP 6741, COMP 6461 as the two courses, we added course outline to the data for adding another attribute.

- **Lectures:**

\_This sheet contains mocked data for lecture of courses COMP 6741, COMP 6461. For COMP 6741, we are using the information available on moodle for this course. COMP 6461 is data

from existing resources and materials we had during our time doing the course. Lecture Content related to this field is present in the folder called Lecture Content.

- **Labs:**

The second assignment required us to add Labs as a parallel to Lecture as a subclass to CourseContent, so we created a class called Labs, and mocked data for it for two focussed subjects.

- **Students:**

We have mocked data for 3 students, namely Nandini Bibra, Jatish Bhatia and Shruti Bibra, we have added the course they have completed, grades achieved, and personal details.

### CONVERT PDF/PPT TO TEXT FILES

To convert pdf files to text files, we have used PyPDF2, a python library. Our files take the folder path from where we have to pick the pdf files, it starts reading pdf page by page, and after extracting each page, we write the content of the file to a text file of the same name saved at the same location in a folder called text\_files.

To convert pdf files to text files, we have used win32com.client, a python library. Our files take the folder path from where we have to pick the ppt files, it starts reading ppt slide by slide, and extracts text from each slide, this text is then we written into a text file of the same name saved at the same location in a folder called text\_files.

### CONVERT TOPIC TRIPLES CREATION

To generate topic triples, we created a separate file called entityTag.py. When graph is being generated, we pass the course material files (Lectures, Lab, Reading etc.) in text format to the function. We use the small English model from spacy and parse the document. In this model, we have added Dbpedia spotlight in the pipe, so model is trained on it as well. The first step is to recognise the concepts from the text file, which has a dbpedia url. Then on that particular entity, we apply a rule based pattern, if the token in the topic is a PROP, we consider it as a topic, the second rule is to consider topics with VERB followed by PROP. To filter out any weak links, we only keep topics, which have a similarity score of 100%. This gives us a good well filtered list of topics for the graph.

### THE PROCESS

To create an automated process, we used Python. We have created the following python files:

- main.py: main entry point to run the pipeline
- rdfGraph.py: this is the file which has the code to generate the Graph.ttl
- dataCleaning.py: this is the file that preprocesses the courses data as it is from open-source data, we need to be selective of the data we choose.
- entityTag.py: file to identify Dbpedia uris of topics from text files.
- query.py: functions to generate SPARQL queries.
- actions.py: rasa actions file to define the actions taken for intent.
- pdftotxt.py: file to convert pdf to text files.

- ppttotxt.py: file to convert ppt to text files.
- **Working of rdfGraph.py:**

We first start off by parsing the vocabulary.rdf file to define the custom properties, classes and constraints of the graph. Next, we need to populate data into the graphs, so we have created a function to populate each – University, Courses, Students, Lectures into the Graph. To do this each function starts off by reading the excel in a dataframe, we then iterate over the dataframe and start loading the data we find in each row into a form of a triple, this is done in each function. To add topics, when we encounter a course material with a location uri we pick the text file from the location, pass it to entitytag.py extract Dbpedia urls and add it to the graph. Once done the function returns an overwritten graph, in the end we serialize that graph into a Graph.ttl and Graph.nt file.

### STATISTICS

Field		Value
Number of Topics		12662
Number of Distinct Topics		385
Number of Topics Instances Per Course	COMP 6461	370
	COMP 6741	273

### CHATBOT DESIGN

To implement RASA as a Natural Language Understanding interface, we had to create, intents, actions, stories, slots in rasa specific files. We defined all the actions in actions.py, declared them in domain.yml, we also added slots and entities in our domain.yml file. To define stories with their specific intents and entities, we added them to nlu.yml file. We validated the files using a yml validation checker. The contents of each could be explained below :

### INTENTS

Following is the list of intents we have created for each of the query from the problem statement-

1. ***get\_course\_by\_university***  
List all courses offered by [university]
2. ***get\_courses\_discussed\_topic***  
In which courses is [topic] discussed?
3. ***get\_topics\_in\_course\_lecture***  
Which [topics] are covered in [course] during [lecture number]?
4. ***get\_courses\_by\_university\_subject***  
List all [courses] offered by [university] within the [subject] (e.g., "COMP", "SOEN").
5. ***get\_material\_topic\_in\_course***  
What [materials] (slides, readings) are recommended for [topic] in [course] [number]?
6. ***get\_credits\_of\_course***  
How many credits is [course] [number] worth?
7. ***get\_links\_course\_number***  
For [course] [number], what additional resources (links to web pages) are available?
8. ***get\_content\_of\_lecture\_in\_course***

Detail the content (slides, worksheets, readings) available for [lecture number] in [course] [number].

**9. *get\_reading\_for\_topic\_course***

What reading materials are recommended for studying [topic] in [course]?

**10. *get\_competency\_by\_course***

What competencies [topics] does a student gain after completing [course] [number]?

**11. *get\_student\_grade\_in\_course***

What grades did [student] achieve in [course] [number]?

**12. *student\_completed\_courses***

Which [students] have completed [course] [number]?

**13. *print\_transcript***

Print a transcript for a [student], listing all the course taken with their grades.

**14. *total\_triples***

**15. *num\_of\_courses***

**16. *course\_number***

What is the about?

**17. *topics\_course\_event\_course***

Which topics are covered in?

**18. *topics\_in\_course\_events***

Which course events cover?

## STORY

Below is a list of stories we have created one for each query for chatbot.

1. Get courses by university
2. Get courses by topic
3. Get topics discussed in a course lecture
4. Get courses by university and subject
5. Get material for a topic in a course
6. Get credits of a course
7. Get links for a course
8. Get content of a lecture in a course
9. Get readings for a topic in a course
10. Get competencies for a course
11. Get grade of a student in a course
12. Get completed courses of a student
13. Print transcript of a student
14. Get total triples in the KB
15. Get course number
16. Get course event covers topic
17. Get topics covered in course events
18. Get number of courses in the KB

## ACTIONS

Each registered action in actions.py, accesses it corresponding SPARQL query from the query.py.

Following is the list of actions we have created for each of the query from the problem statement-

1. ***get\_course\_by\_university***  
List all courses offered by [university]
2. ***get\_courses\_discussed\_topic***  
In which courses is [topic] discussed?
3. ***get\_topics\_in\_course\_lecture***  
Which [topics] are covered in [course] during [lecture number]?
4. ***get\_courses\_by\_university\_subject***  
List all [courses] offered by [university] within the [subject] (e.g., "COMP", "SOEN").
5. ***get\_material\_topic\_in\_course***  
What [materials] (slides, readings) are recommended for [topic] in [course] [number]?
6. ***get\_credits\_of\_course***  
How many credits is [course] [number] worth?
7. ***get\_links\_course\_number***  
For [course] [number], what additional resources (links to web pages) are available?
8. ***get\_content\_of\_lecture\_in\_course***  
Detail the content (slides, worksheets, readings) available for [lecture number] in [course] [number].
9. ***get\_reading\_for\_topic\_course***  
What reading materials are recommended for studying [topic] in [course]?
10. ***get\_competency\_by\_course***  
What competencies [topics] does a student gain after completing [course] [number]?
11. ***get\_student\_grade\_in\_course***  
What grades did [student] achieve in [course] [number]?
12. ***student\_completed\_courses***  
Which [students] have completed [course] [number]?
13. ***print\_transcript***  
Print a transcript for a [student], listing all the course taken with their grades.
14. ***total triples***
15. ***num\_of\_courses***
16. ***course\_number***  
What is the about?
17. ***topics\_course\_event\_course***  
Which topics are covered in?
18. ***topics\_in\_course\_events***  
Which course events cover?

## ENTITY

The below is a list of entities, we have created in our implementation of Rasa:

1. Uni: extract name of university from input
2. Fname: extract first name from input
3. Lname: extract last name from input
4. sub\_num: extract course number from input
5. subject: extract subject of course from input
6. lecture: extract lecture number from input
7. topic: extract topic name from input
8. course: extract course name from input

9. event: extract the type of event (Lab or Lecture) from input
10. event\_num: extract the number associated with event from input

### FUSEKI INTEGRATION & SPARQL QUERY CREATION

To integrate our chatbot with Fuseki, we used Requests, a python library to do a post request to our local SPARQL engine. We ensured the server was on while querying, by populating it with the Graph generated by rdfGraph.py. Our actions.py file had imported Query.py, which is our python file where we have a function declared for each of the query, each functions returns a specific string formatted query to our actions.py file. When an action is triggered it takes the query from query.py, sends a post request to <http://localhost:3030/Graph/query> and once we get a response we process it, and convert it to a simple string for chatbot to return the result.

### OOV & ERROR HANDLING

Following are the techniques used to handle Out Of Vocabulary words and a Fallback strategy -

Added the following algorithms/models to the config file –

- WhitespaceTokenizer
- RegexFeaturizer
- LexicalSyntacticFeaturizer
- CountVectorsFeaturizer
- DIETClassifier
- EntitySynonymMapper
- ResponseSelector
- FallbackClassifier

#### CountVectorsFeaturizer

Declared a OOV\_token as \_\_oov\_\_ and added it to the examples with the intents. The analyzer was set to “word” and the min\_ngram set to 1 and max\_ngram set to 2.

Considered the use of shared vocab and set the respective parameter to True.

#### FallbackClassifier

Set the threshold and ambiguity threshold to 0.5 so that if the similarity score of a user input with the intent is less than 0.5 then would resort to calling the “action\_default\_fallback” which is basically defining our own implementation of the FallbackAction and printing the relevant response in the natural language understood by user.

In addition to this,

A rule “out-of-scope” is created to define the flow of the intent followed by the relevant action for the cases where the user inputs are the ones that can be predicted and are known to be such that cannot be handled by the model. Example, “Order a pizza”



Defined the intent “out\_of\_scope” and action “action\_default\_fallback” for the same purpose.

### STATIC TEMPLATE FOR NATURAL LANGUAGE ANSWERING

We used a static template approach for natural language answering by our chatbot. For each specific query if the result from SPARQL was a success, we proceeded to analyse the result, if its not an empty json table, we combined the result to form a sense making string and the chatbot would utter it to the user. In case, the table is empty, we tell the user information is not in database. If we get a error for server, we just say failed to connect to the server. We followed this approach for each query, we also developed a fallback response to answer issues in case chatbot fails to understand a query.

### EXAMPLES

Given below are the knowledge graph questions and a corresponding SPARQL query to retrieve the results. For each query, there is also a screenshot of the sample output obtained when run on the knowledge base on the Fuseki server.

Given below is one concrete example from each query type (the 13 queries from Part #1 plus the 3 new ones) that showcases the chatbot's workflow –

#### Query 1

##### **1. User Input**

List all courses offered by Concordia University

##### **2. Intent Recognition and Entity Extraction**

Intent –

get\_course\_by\_university

Entity –

uni [Concordia University]

##### **3. SPARQL Query**

```
SELECT ("Concordia University" AS ?University) ?Course_Name ?Course_URI
WHERE {
    ?uni dbo:name "Concordia University" .
    ?uni roboprof:offersCourse ?Course_URI .
    ?Course_URI dbo:name ?Course_Name .
}
```

##### **4. Output on Fuseki**

Table Response 1542 results in 0.1 seconds

	University	Course_Name	Course_URI
1	Concordia University	Organic Chemistry III: Organic Reactions	<http://roboprof.ca/data/CHEM_324>
2	Concordia University	Capstone Aerospace Engineering Design Project	<http://roboprof.ca/data/AERO_490>
3	Concordia University	Performance Creation Mainstage	<http://roboprof.ca/data/PERC_490>
4	Concordia University	Special Topics in Drawing	<http://roboprof.ca/data/DRAW_399>
5	Concordia University	Religions of the West	<http://roboprof.ca/data/REL_214>
6	Concordia University	Executive Communication	<http://roboprof.ca/data/INSE_6240>
7	Concordia University	Experimental Physics II	<http://roboprof.ca/data/PHYS_330>
8	Concordia University	Plant Ecology	<http://roboprof.ca/data/GEOG_374>
9	Concordia University	Leisure Services Leadership	<http://roboprof.ca/data/AHSC_361>

## 5. Natural Language Response

```

Your input -> List all courses offered by Concordia University
Here are the courses offered by Concordia University:
Organic Chemistry III: Organic Reactions
Capstone Aerospace Engineering Design Project
Performance Creation Mainstage
Special Topics in Drawing
Religions of the West
Executive Communication
Experimental Physics II

```

### Query 2

#### 1. User Input

In which courses is Deep Learning discussed?

#### 2. Intent Recognition and Entity Extraction

Intent –

get\_courses\_discussed\_topic

Entity –

topic [Deep Learning]

#### 3. SPARQL Query

```

SELECT DISTINCT ?Course_URI ?Course_Name ("Deep Learning" AS ?Topic_Name)
WHERE {
    ?Topic_URI dbo:name "Deep Learning" .
    ?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
    ?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
    ?Course_URI roboprof:hasCourseEvent ?CourseEvent_URI .
    ?Course_URI dbo:name ?Course_Name .
}

```

#### 4. Output on Fuseki

[Table](#)
[Response](#)
1 result in 0.013 seconds

	Course_URI	Course_Name	Topic_Name
1	< <a href="http://roboprof.ca/data/COMP_6741">http://roboprof.ca/data/COMP_6741</a> >	Intelligent Systems	Deep Learning

#### 5. Natural Language Response

```

Your input -> In which courses is Deep Learning discussed?
Here are the courses which discuss Deep Learning:
Intelligent Systems

```

#### Query 3

##### 1. User Input

Which topics are covered in Intelligent Systems during lecture 4?

##### 2. Intent Recognition and Entity Extraction

Intent –  
 get\_topics\_in\_course\_lecture

Entity –  
 course [Intelligent Systems]  
 lecture [1]

##### 3. SPARQL Query

```

SELECT ?Topic_URI ?Topic_Name ("Intelligent Systems" AS ?Course_Name) (STR(4) AS
?Lecture_Number)
WHERE {
    ?course dbo:name "Intelligent Systems" .
    ?course roboprof:hasCourseEvent ?CourseEvent_URI .
}

```

```

?CourseEvent_URI rdf:type ?CourseEvent_Type .
?CourseEvent_Type rdfs:label "Lecture"@en .
?CourseEvent_URI dbo:number 4 .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
?Topic_URI dbo:name ?Topic_Name .
}

```

#### 4. Output on Fuseki

Table Response 73 results in 0.016 seconds

	Topic_URI	Topic_Name	Course_Name	Lecture_Number
1	<http://dbpedia.org/resource/Time-division_multiplexing>	TDM	Intelligent Systems	4
2	<http://dbpedia.org/resource/Informa>	informa	Intelligent Systems	4
3	<http://dbpedia.org/resource/Coreference>	Coreference	Intelligent Systems	4
4	<http://dbpedia.org/resource/Natural_language_processing>	Natural Language Processing	Intelligent Systems	4
5	<http://dbpedia.org/resource/Natural_language_processing>	NLP	Intelligent Systems	4
6	<http://dbpedia.org/resource/Wikipedia>	Wikipedia	Intelligent Systems	4
7	<http://dbpedia.org/resource/Israel_Defense_Forces>	idf	Intelligent Systems	4
8	<http://dbpedia.org/resource/Recommender_system>	Recommender Systems	Intelligent Systems	4
9	<http://dbpedia.org/resource/SPARQL>	SPARQL	Intelligent Systems	4
10	<http://dbpedia.org/resource/Resource_Description_Framework...>	rdf	Intelligent Systems	4
11	<http://dbpedia.org/resource/Resource_Description_Framework...>	RDF	Intelligent Systems	4
12	<http://dbpedia.org/resource/Resource_Description_Framework...>	Resource Description Framework	Intelligent Systems	4
13	<http://dbpedia.org/resource/RDF_query_language>	RDF Query Language	Intelligent Systems	4

#### 5. Natural Language Response

```

Your input -> Which topics are covered in Intelligent Systems during lecture 4?
Here are the topics discussed in lecture 4 in Intelligent Systems:
TDM
informa
Coreference
Natural Language Processing
NLP
Wikipedia
idf
Recommender Systems
SPARQL
rdf

```

#### Query 4

##### 1. User Input

List all courses offered by Concordia University within the COMP

## 2. Intent Recognition and Entity Extraction

Intent –

get\_courses\_by\_university\_subject

Entity –

uni [Concordia University]

subject [COMP]

## 3. SPARQL Query

```
SELECT ?Course_URI ?Course_Name ("Concordia University" AS ?University_Name) ("COMP" AS
?Subject)
WHERE {
    ?uni dbo:name "Concordia University" .
    ?Course_URI dbo:name ?Course_Name .
    ?uni roboprof:offersCourse ?Course_URI .
    ?Course_URI vivo:hasSubjectArea "COMP" .
}
```

## 4. Output on Fuseki

Table Response 49 results in 0.014 seconds				
	Course_URI	Course_Name	University_Name	Subject
1	<http://roboprof.ca/data/COMP_6311>	Animation for Computer Games	Concordia University	COMP
2	<http://roboprof.ca/data/COMP_346>	Operating Systems	Concordia University	COMP
3	<http://roboprof.ca/data/COMP_6791>	Information Retrieval and Web Search	Concordia University	COMP
4	<http://roboprof.ca/data/COMP_476>	Advanced Game Development	Concordia University	COMP
5	<http://roboprof.ca/data/COMP_6771>	Image Processing	Concordia University	COMP
6	<http://roboprof.ca/data/COMP_6411>	Comparative Study of Programming Languages	Concordia University	COMP
7	<http://roboprof.ca/data/COMP_472>	Artificial Intelligence	Concordia University	COMP
8	<http://roboprof.ca/data/COMP_492>	Computer Science Project II	Concordia University	COMP

## 5. Natural Language Response

```
Your input -> List all courses offered by Concordia University within the COMP
Here are the courses covered by Concordia University in subject COMP :
Animation for Computer Games
Operating Systems
Information Retrieval and Web Search
Advanced Game Development
Image Processing
Comparative Study of Programming Languages
Artificial Intelligence
Computer Science Project II
Computer Science Industrial Experience Work Term II
Databases
Discrete Structures and Formal Languages
```

### Query 5

#### 1. User Input

What materials are recommended for Aloha in COMP 6461?

#### 2. Intent Recognition and Entity Extraction

Intent –  
get material topic in course

Entity –  
topic [Aloha]  
subject [COMP]  
sub\_num [6461]

#### 3. SPARQL Query

```
SELECT ?Course_URI ?Course_Name ("Concordia University" AS ?University_Name) ("COMP" AS
?Subject)
WHERE {
    ?uni dbo:name "Concordia University" .
    ?Course_URI dbo:name ?Course_Name .
    ?uni roboprof:offersCourse ?Course_URI .
    ?Course_URI vivo:hasSubjectArea "COMP" .
}
```

#### 4. Output on Fuseki

Table Response 18 results in 0.013 seconds Simple view

	Course_Name	Course_Num...	Topic_Na...	Course_Content_URI	Material_Type	Material_Name
1	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/Lectures/5.ppt>	"Slide"@en	5.ppt
2	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/TCP.pdf>	"Other Material"@en	TCP.pdf
3	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA3.pdf>	"Worksheet"@en	TA3.pdf
4	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/Readings/SelectiveRepea...	"Reading"@en	SelectiveRepeat.pdf
5	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/Lectures/3.ppt>	"Slide"@en	3.ppt
6	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA2.pdf>	"Worksheet"@en	TA2.pdf
7	Computer Networks and Protocols	COMP 6461	Aloha	<file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/Application...	"Other Material"@en	ApplicationLayer.pdf

## 5. Natural Language Response

```

Your input -> What materials are recommended for Aloha in COMP 6461?
Here are the materials covered in COMP 6461 :
1. 5.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/5.ppt
2. TCP.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/TCP.pdf
3. TA3.pdf (Worksheet), file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA3.pdf
4. SelectiveRepeat.pdf (Reading), file:///dataset/Lecture%20Content/COMP%206461/Readings/SelectiveRepeat.pdf
5. 3.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/3.ppt
6. TA2.pdf (Worksheet), file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA2.pdf
7. ApplicationLayer.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/ApplicationLayer.pdf
8. GoBackN.pdf (Reading), file:///dataset/Lecture%20Content/COMP%206461/Readings/GoBackN.pdf
9. TA1.pdf (Worksheet), file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA1.pdf
10. 2.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/2.ppt
11. InternetProtocolSuite.pdf (Reading), file:///dataset/Lecture%20Content/COMP%206461/Readings/InternetProtocolSuite.pdf
12. 1.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/1.ppt
13. ComputerNetwork.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/ComputerNetwork.pdf
14. UDP.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/UDP.pdf
15. NetworkCongestion.pdf (Reading), file:///dataset/Lecture%20Content/COMP%206461/Readings/NetworkCongestion.pdf
16. 6.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/6.ppt
17. NetworkLayer.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/NetworkLayer.pdf
18. 4.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/4.ppt

```

## Query 6

### 1. User Input

How many credits is COMP 6741 worth?

### 2. Intent Recognition and Entity Extraction

Intent –

get\_credits\_of\_course

Entity –

subject [COMP]

sub\_num [6741]

### 3. SPARQL Query

```

SELECT ?Course_Name (CONCAT("COMP", " ", STR(6741)) AS ?Course_Number) ?Credits
WHERE {

```

```

?course dbo:number 6741 .
?course vivo:hasSubjectArea "COMP" .
?course dbo:name ?Course_Name .
?course vivo:courseCredits ?Credits .
}

```

#### 4. Output on Fuseki

Table Response 1 result in 0.013 seconds

	Course_Name	Course_Number	Credits
1	Intelligent Systems	COMP 6741	"4"^^<http://www.w3.org/2001/XMLSchema#integer>

#### 5. Natural Language Response

```

Your input -> How many credits is COMP 6741 worth?
COMP 6741 is a 4 credit course.

```

### Query 7

#### 1. User Input

For COMP 6741, what additional resources are available?

#### 2. Intent Recognition and Entity Extraction

Intent –  
get\_links\_course\_number

Entity –  
subject [COMP]  
sub\_num [6741]

#### 3. SPARQL Query

```

SELECT ?Course_Name (CONCAT("COMP", " ", STR(6741)) AS ?Course_Number) ?Resource
WHERE {
  ?course dbo:number 6741 .
  ?course vivo:hasSubjectArea "COMP" .
  ?course dbo:name ?Course_Name .
  ?course rdfs:seeAlso ?Resource .
}

```



#### 4. Output on Fuseki

Table Response 1 result in 0.012 seconds

	Course_Name	Course_Number	Resource
1	Intelligent Systems	COMP 6741	< <a href="https://www.concordia.ca/academic">https://www.concordia.ca/academic</a>

#### 5. Natural Language Response

```
Your input -> For COMP 6741, what additional resources are available?
You can refer to https://www.concordia.ca/academics/graduate/calendar/current/gina-cody-school-of-engineering-and-computer-science/courses/computer-science-and-software-engineering-master-s-and-phd-courses.html for more information on COMP 6741.
```

#### Query 8

##### 1. User Input

Detail the content available for lecture 3 in COMP 6461

##### 2. Intent Recognition and Entity Extraction

Intent –  
get\_content\_of\_lecture\_in\_course

Entity –  
subject [COMP]  
sub\_num [6461]  
lecture [1]

##### 3. SPARQL Query

```
SELECT ?Course_Name (CONCAT("COMP", " ", STR(6461)) AS ?Course_Number) (STR(3) AS
?Lecture_Number) ?Course_Content_URI ?Course_Content_Type_Label
?Course_Content_Name
WHERE {
    ?course dbo:number 6461 .
    ?course vivo:hasSubjectArea "COMP" .
    ?course dbo:name ?Course_Name .
    ?course roboprof:hasCourseEvent ?CourseEvent_URI .
    ?CourseEvent_URI rdf:type ?CourseEvent_Type .
    ?CourseEvent_Type rdfs:label "Lecture"@en .
```

```

?CourseEvent_URI dbo:number 3 .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?Course_Content_URI rdf:type ?Course_Content_Type .
?Course_Content_URI dbo:name ?Course_Content_Name .
?Course_Content_Type rdfs:label ?Course_Content_Type_Label .
}

```

#### 4. Output on Fuseki

Table Response 4 results in 0.017 seconds Simple view Ellip

	Course_Name	Course_Number	Lecture_Num...	Course_Content_URI	Course_Content_Type_L...	Course_Content_Name
1	Computer Networks and Protocols	COMP 6461	3	<file:///dataset/Lecture%20Content/COMP%206461/Lectures/3.ppt>	"Slide"@en	3.ppt
2	Computer Networks and Protocols	COMP 6461	3	<file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA2...	"Worksheet"@en	TA2.pdf
3	Computer Networks and Protocols	COMP 6461	3	<file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/A...	"Other Material"@en	ApplicationLayer.pdf
4	Computer Networks and Protocols	COMP 6461	3	<file:///dataset/Lecture%20Content/COMP%206461/Readings/GoBac...	"Reading"@en	GoBackN.pdf

#### 5. Natural Language Response

```

Your input -> Detail the content available for lecture 3 in COMP 6461
Here is a list of content for COMP 6461 lecture 3:
1. 3.ppt (Slide), file:///dataset/Lecture%20Content/COMP%206461/Lectures/3.ppt
2. TA2.pdf (Worksheet), file:///dataset/Lecture%20Content/COMP%206461/Worksheets/TA2.pdf
3. ApplicationLayer.pdf (Other Material), file:///dataset/Lecture%20Content/COMP%206461/OtherMaterial/ApplicationLayer.pdf
4. GoBackN.pdf (Reading), file:///dataset/Lecture%20Content/COMP%206461/Readings/GoBackN.pdf

```

#### Query 9

##### 1. User Input

What reading materials are recommended for studying ARQ in Computer Networks and Protocols?

##### 2. Intent Recognition and Entity Extraction

Intent –  
get\_reading\_for\_topic\_course

Entity –  
topic [ARQ]  
course [Computer Networks and Protocols]

##### 3. SPARQL Query

```

SELECT DISTINCT ("Computer Networks and Protocols" AS ?Course_Name) ("ARQ" AS
?Topic_Name) ?Course_Content_URI ?Course_Content_Type_Label ?Course_Content_Name
WHERE {

```

```

?course dbo:name "Computer Networks and Protocols" .
?course roboprof:hasCourseEvent ?CourseEvent_URI .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?Course_Content_URI rdf:type roboprof:Reading .
?Topic_URI dbo:name "ARQ" .
?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
?Course_Content_URI rdf:type ?Course_Content_Type .
?Course_Content_URI dbo:name ?Course_Content_Name .
?Course_Content_Type rdfs:label ?Course_Content_Type_Label .
}

```

#### 4. Output on Fuseki

Table Response 2 results in 0.015 seconds Simple vi

	Course_Name	Topic_Name	Course_Content_URI	Course_Content_Type_Label	Course_Content_Name
1	Computer Networks and Protocols	ARQ	<file:///./dataset/Lecture%20Content/COMP%206461/Readings/SelectiveRepeat.pdf>	"Reading"@en	SelectiveRepeat.pdf
2	Computer Networks and Protocols	ARQ	<file:///./dataset/Lecture%20Content/COMP%206461/Readings/GoBackN.pdf>	"Reading"@en	GoBackN.pdf

#### 5. Natural Language Response

```

Your input -> What reading materials are recommended for studying ARQ in Computer Networks and Protocols?
Here is a list of readings for Computer Networks and Protocols :
SelectiveRepeat.pdf, file:///./dataset/Lecture%20Content/COMP%206461/Readings/SelectiveRepeat.pdf
GoBackN.pdf, file:///./dataset/Lecture%20Content/COMP%206461/Readings/GoBackN.pdf

```

#### Query 10

##### 1. User Input

What competencies does a student gain after completing COMP 6461?

##### 2. Intent Recognition and Entity Extraction

Intent –

get\_competency\_by\_course

Entity –

subject [COMP]

sub\_num [6461]

##### 3. SPARQL Query

```

SELECT DISTINCT ?Course_Name (CONCAT("COMP", " ", STR(6461)) AS ?Course_Number)
?Topic_Name ?Topic_URI
WHERE {
    ?course dbo:number 6461 .
    ?course vivo:hasSubjectArea "COMP" .
    ?course dbo:name ?Course_Name .
    ?course roboprof:hasCourseEvent ?CourseEvent_URI .
    ?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
    ?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
    ?Topic_URI dbo:name ?Topic_Name .
}

```

#### 4. Output on Fuseki

Table Response 175 results in 0.015 seconds

	Course_Name	Course_Number	Topic_Name	Topic_URI
1	Computer Networks and Protocols	COMP 6461	Mbps	<http://dbpedia.org/resource/Data-rate_units>
2	Computer Networks and Protocols	COMP 6461	Kbps	<http://dbpedia.org/resource/Data-rate_units>
3	Computer Networks and Protocols	COMP 6461	Gbps	<http://dbpedia.org/resource/Data-rate_units>
4	Computer Networks and Protocols	COMP 6461	kbps	<http://dbpedia.org/resource/Data-rate_units>
5	Computer Networks and Protocols	COMP 6461	packet	<http://dbpedia.org/resource/Network_packet>
6	Computer Networks and Protocols	COMP 6461	Shay	<http://dbpedia.org/resource/Shay_locomotive>
7	Computer Networks and Protocols	COMP 6461	Selective Repeat	<http://dbpedia.org/resource/Selective_Repeat_ARQ>

#### 5. Natural Language Response

Your input -> What competencies does a student gain after completing COMP 6461?

On completing COMP 6461 a student is competent in Mbps,Kbps,Gbps,kbps,packet,Shay,Selective Repeat,Selective Repeat ARQ,Finite State Machine,Finite State Machines,Aiman,Sliding Window Protocol,Software Engineering,algorithm,Computer Science,Concordia University,Jim Kurose,msec,ACK,Transmission Control Protocol,TCP,WWW,CYCLADES,Secure Shell,ARPANET,Transmission Control Program,RTTP,Vinton Cerf,Vint Cerf,Turing Award,Bob Kahn>User Datagram Protocol,UDP,Internet,Slotted Aloha,Aloha,Token Ring,Bibcode,IEEE,ARQ,New Jersey,Prentice Hall,STANAG,HF,Weldon,G.hn,RIPE,CNAME,U S DoD,DDoS,NASA,Application Layer,TTL,P2P,Alice,KanKan,Internet Engineering Task Force,IETF,ISP,ARL,Domain Name System,DNS,BitTorrent,Network Solutions,Educause,TLD,Canada,HTTP GET,HTTP,Hypertext Transfer Protocol,RFC2616,HTTPS,ICANN,CIDR,Due Date,Simple Mail Transfer Protocol,HELO,SMTP,telnet,Telnet,BOOTP,Remote Database Access,TFTP,Trivial File Transfer Protocol,FTAM,MMS,Hypertext,CMIP,Common Management Information Protocol,IRC,SNMP,Simple Network Management Protocol,Open Systems Interconnection,OSI,CASE,ACSE,Association Control Service Element,FTP,File Transfer Protocol,SFTP,DSL,ing,Traceroute,dedi,IPTV,OS,API,SSL,Ross,Apache,Apache HTTP Server,UA,POP,POP3,Post Office Protocol,gzip,POST,Skype,Hotmail,Sun,MIME,Montreal,lf,Uniform Resource Locator,URL,Netflix,Semantics,Thunderbird,YouTube,Internet Mail,server,Hulu,IMAPS,IMAP,ETag,Firefox,Safari,Paris,HTP,Webmail,cache,JPEG,Network Control Program,DARPA,Steve Crocker,Dartmouth Time Sharing System,Ethernet,10BASE2,Anatoly Kitov,BBN,ter,MTU,SABRE,SAGE,AT&T Corporation,Checksum,Precision Time Protocol,DHCP,IANA,SCTP,Stream Control Transmission Protocol,Routing Information Protocol,Network File System,Network Time Protocol,NTP,QUIC,Van Jacobson,NSFNET,FIN,SNA,Disconnect,ATM,TCP RENO,TCP Slow Start,NAKs

## Query 11

### 1. User Input

What grades did Jatish Bhatia achieve in COMP 6521?

### 2. Intent Recognition and Entity Extraction

Intent –

get\_student\_grade\_in\_course

Entity –

subject [COMP]

sub\_num [6521]

fname [Jatish]

lname [Bhatia]

### 3. SPARQL Query

```
SELECT (CONCAT("Jatish", " ", "Bhatia") AS ?Name) ?Course_Name (CONCAT("COMP", " ",  
STR(6521)) AS ?Course_Number) ?Grade ?Grade_URI  
WHERE {  
  ?student foaf:firstName "Jatish" .  
  ?student foaf:lastName "Bhatia" .  
  ?student roboprof:achievedGrade ?Grade_URI .  
  ?course dbo:number 6521 .  
  ?course vivo:hasSubjectArea "COMP" .  
  ?course dbo:name ?Course_Name .  
  ?Grade_URI roboprof:gradeInCourse ?course .  
  ?Grade_URI roboprof:gradeValue ?Grade .  
}
```

### 4. Output on Fuseki

Table		Response		1 result in 0.013 seconds	
Name	Course_Name	Course_Number	Grade	Grade_URI	
1 Jatish Bhatia	Advanced Database Techniques and Applications	COMP 6521	A	< <a href="http://roboprof.ca/data/COMP_6521_A">http://roboprof.ca/data/COMP_6521_A</a> >	

### 5. Natural Language Response

```
Your input -> What grades did Jatish Bhatia achieve in COMP 6521?  
Jatish Bhatia has achieved A in COMP 6521
```

## Query 12

### 1. User Input

Which students have completed COMP 6741?

### 2. Intent Recognition and Entity Extraction

Intent –  
student\_completed\_courses

Entity –  
subject [COMP]  
sub\_num [6741]

### 3. SPARQL Query

```
SELECT DISTINCT (CONCAT("COMP", " ", STR(6741)) AS ?Course_Number) (CONCAT(?first_name, " ", ?last_name) AS ?Student_Name) ?Student_URI
WHERE {
    ?course dbo:number 6741 .
    ?course vivo:hasSubjectArea "COMP" .
    ?Student_URI roboprof:completedCourse ?course .
    ?Student_URI foaf:firstName ?first_name .
    ?Student_URI foaf:lastName ?last_name .
}
```

### 4. Output on Fuseki

Table			
Response			
3 results in 0.012 seconds			
Course_Number	Student_N...	Student_URI	
1 COMP 6741	Shruti Bibra	< <a href="http://roboprof.ca/data/40226157">http://roboprof.ca/data/40226157</a> >	
2 COMP 6741	Nandini Bibra	< <a href="http://roboprof.ca/data/40227120">http://roboprof.ca/data/40227120</a> >	
3 COMP 6741	Jatish Bhatia	< <a href="http://roboprof.ca/data/40225156">http://roboprof.ca/data/40225156</a> >	

### 5. Natural Language Response

```
Your input -> Which students have completed COMP 6741?
This is the list of students who have completed COMP 6741 :
1. Shruti Bibra
2. Nandini Bibra
3. Jatish Bhatia
```

### Query 13

#### 1. User Input

Print a transcript for Nandini Bibra, listing all the course taken with their grades.

#### 2. Intent Recognition and Entity Extraction

Intent –  
print\_transcript

Entity –  
fname [Nandini]  
lname [Bibra]

#### 3. SPARQL Query

```
SELECT (CONCAT("Nandini", " ", "Bibra") AS ?Student) ?Course_Name ?Course_URI ?Grade
?Grade_URI ?Retake_Grade ?Retake_Grade_URI
WHERE {
    ?student foaf:firstName "Nandini" .
    ?student foaf:lastName "Bibra" .
    ?student roboprof:completedCourse ?Course_URI .
    ?Course_URI dbo:name ?Course_Name .
    ?student roboprof:achievedGrade ?Grade_URI .
    ?Grade_URI rdf:type roboprof:InitialGrade .
    ?Grade_URI roboprof:gradeInCourse ?Course_URI .
    ?Grade_URI roboprof:gradeValue ?Grade .
    OPTIONAL { ?student roboprof:achievedGrade ?Retake_Grade_URI .
        ?Retake_Grade_URI rdf:type roboprof:RetakeGrade .
        ?Retake_Grade_URI roboprof:gradeInCourse ?Course_URI .
        ?Retake_Grade_URI roboprof:gradeValue ?Retake_Grade . }
}
```

#### 4. Output on Fuseki

Table		Response		5 results in 0.015 seconds		Simple view		Ellipse	Filter query results
Student	Course_Name	Course_URI	Gr...	Grade_URI	Retake_Grade	Retake_Grade_URI			
1 Nandini Bibra	Programming and Problem Solving	<http://roboprof.ca/data/COMP_6481>	A-	<http://roboprof.ca/data/COMP_6481_A->					
2 Nandini Bibra	Advanced Database Techniques and Applications	<http://roboprof.ca/data/COMP_6521>	F	<http://roboprof.ca/data/COMP_6521_F>	B+	<http://roboprof.ca/data/Retake/COMP_6521_B+>			
3 Nandini Bibra	Computer Networks and Protocols	<http://roboprof.ca/data/COMP_6461>	A-	<http://roboprof.ca/data/COMP_6461_A->					
4 Nandini Bibra	Distributed Systems & Design	<http://roboprof.ca/data/COMP_6231>	F	<http://roboprof.ca/data/COMP_6231_F>	A	<http://roboprof.ca/data/Retake/COMP_6231_A>			
5 Nandini Bibra	Intelligent Systems	<http://roboprof.ca/data/COMP_6741>	A-	<http://roboprof.ca/data/COMP_6741_A->					

## 5. Natural Language Response

```

Your input -> Print a transcript for Nandini Bibra, listing all the course taken with their grades.
Here is Nandini Bibra's transcript:
| Student Name | Course Name | Grade | Retake Grade |
|:-----|:-----|:-----|:-----|
| Nandini Bibra | Programming and Problem Solving | A- | |
| Nandini Bibra | Advanced Database Techniques and Applications | F | B+ |
| Nandini Bibra | Computer Networks and Protocols | A- | |
| Nandini Bibra | Distributed Systems & Design | F | A |
| Nandini Bibra | Intelligent Systems | A- | |

```

## PART 2 QUERIES

### Query 14

#### 1. User Input

What is COMP 6741 about?

#### 2. Intent Recognition and Entity Extraction

Intent –  
course\_number

Entity –  
subject [COMP]  
sub\_num [6741]

#### 3. SPARQL Query

```

SELECT ?Course_Name (CONCAT("COMP", " ", STR(6741)) AS ?Course_Number)
?Course_Description
WHERE {
    ?course dbo:number 6741 .
    ?course vivo:hasSubjectArea "COMP" .
    ?course dbo:name ?Course_Name .
    ?course dcterms:description ?Course_Description .
}

```



#### 4. Output on Fuseki

[Table](#) [Response](#) 1 result in 0.013 seconds

	Course_Name	Course_Nu...	Course_Description
1	Intelligent Systems	COMP 6741	(COMP 474): Rule-based expert systems, blackboard architecture, and agent-based.

#### 5. Natural Language Response

```
Your input -> What is COMP 6741 about?
(COMP 474): Rule-based expert systems, blackboard architecture, and agent-based. Knowledge acquisition and representation. Uncertainty and conflict resolution. Reasoning and explanation. Design of intelligent systems. Project. (COMP 6741): Knowledge representation and reasoning. Uncertainty and conflict resolution. Design of intelligent systems. Grammar-based, rule-based, and b
```

#### Query 15

##### 1. User Input

Which topics are covered in Lab 2 of COMP 6741?

##### 2. Intent Recognition and Entity Extraction

Intent –

topics\_course\_event\_course

Entity –

subject [COMP]

sub\_num [6741]

event [Lecture]

event\_num [1]

##### 3. SPARQL Query

```
SELECT ?Topic_URI ?Topic_Name (CONCAT("COMP", " ", STR(6741)) AS ?Course_Number)
(CONCAT("Lab", " ", STR(2)) AS ?CourseEvent)
WHERE {
    ?course dbo:number 6741 .
    ?course vivo:hasSubjectArea "COMP" .
    ?course roboprof:hasCourseEvent ?CourseEvent_URI .
    ?CourseEvent_URI rdf:type ?CourseEvent_Type .
```

```

?CourseEvent_Type rdfs:label "Lab"@en .
?CourseEvent_URI dbo:number 2 .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
?Topic_URI dbo:name ?Topic_Name .
}

```

#### 4. Output on Fuseki

Table Response 18 results in 0.016 seconds

	Topic_URI	Topic_Name	Course_Num...	CourseEvent
1	<http://dbpedia.org/resource/Wikidata>	Wikidata	COMP 6741	Lab 2
2	<http://dbpedia.org/resource/Resource_Description_Framework>	rdf	COMP 6741	Lab 2
3	<http://dbpedia.org/resource/Resource_Description_Framework>	RDF	COMP 6741	Lab 2
4	<http://dbpedia.org/resource/Resource_Description_Framework>	Resource Description Framework	COMP 6741	Lab 2
5	<http://dbpedia.org/resource/Data_integration>	Data Integration	COMP 6741	Lab 2

#### 5. Natural Language Response

```

Your input -> Which topics are covered in Lab 2 of COMP 6741?
Wikidata,rdf,RDF,Resource Description Framework,Data Integration,Java,RDF schema,rdfs,RDFS,RDF
Schema,Machine Learning,ML,AI,Intelligent Systems,Artificial Intelligence,URI,Concordia Univers
ity,DBpedia are the topics covered in Lab 2 of COMP 6741

```

#### Query 16

##### 1. User Input

Which course events cover Deep Learning?

##### 2. Intent Recognition and Entity Extraction

Intent –  
topics\_in\_course\_events

Entity –  
topic [Deep Learning]

##### 3. SPARQL Query

```

SELECT DISTINCT ("Deep Learning" AS ?Topic_Name) (COUNT(?Topic_URI) AS ?Frequency)
?Course_Name ?Course_URI (CONCAT(?CourseEvent_Type_Label, " ", STR(?CourseEvent_Num))
AS ?Course_Event) ?CourseEvent_URI ?CourseEvent_Type
WHERE {
    ?Topic_URI dbo:name "Deep Learning" .

```

```

?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
  ?CourseEvent_URI rdf:type ?CourseEvent_Type .
?CourseEvent_Type rdfs:label ?CourseEvent_Type_Label .
  ?CourseEvent_URI dbo:number ?CourseEvent_Num .
?Course_URI roboprof:hasCourseEvent ?CourseEvent_URI .
?Course_URI dbo:name ?Course_Name .
}
GROUP BY ?Topic_Name ?Frequency ?Course_Name ?Course_URI ?CourseEvent_Type_Label
?CourseEvent_URI ?CourseEvent_Type ?CourseEvent_Num
ORDER BY DESC(?Frequency)

```

#### 4. Output on Fuseki

Table Response 2 results in 0.017 seconds Simple view Ellipsis

	Topic_Name	Frequency	Course_Name	Course_URI	Course_Ev...	CourseEvent_URI	CourseEvent_Type
1	Deep Learning	"1"^^<http://...	Intelligent Systems	<http://roboprof.ca/data/COMP_6741>	Lecture 1	<http://roboprof.ca/data/COMP_6741_Lecture_1>	<http://roboprof.ca/schema/Lecture>
2	Deep Learning	"1"^^<http://...	Intelligent Systems	<http://roboprof.ca/data/COMP_6741>	Lecture 6	<http://roboprof.ca/data/COMP_6741_Lecture_6>	<http://roboprof.ca/schema/Lecture>

#### 5. Natural Language Response

```

Your input -> Which course events cover Deep Learning?
The topic Deep Learning is covered by:
1. Lecture 1 of Intelligent Systems.
2. Lecture 6 of Intelligent Systems.

```

### ADDITIONAL QUERIES –

#### Additional Query 1

##### 1. User Input

Total number of triples?

##### 2. Intent Recognition and Entity Extraction

Intent –  
total\_triples

##### 3. SPARQL Query

```

SELECT (COUNT(?s) AS ?Total_Number_of_Triples)
WHERE {
  ?s ?p ?o .
}

```

#### 4. Output on Fuseki

Table		Response	1 result in 0.078 seconds
Total_Number_of_Triples			
1	"12662"	<sup>^^</sup> <http://www.w3.org/2001/XMLSchema#integer>	

#### 5. Natural Language Response

```
Your input -> Total number of triples
I am made up of 12662 triples.
```

#### Additional Query 2

##### 1. User Input

What is the total number of courses?

##### 2. Intent Recognition and Entity Extraction

Intent –  
num\_of\_courses

##### 3. SPARQL Query

```
SELECT (COUNT(?courses) AS ?Course_Count)
WHERE {
  ?courses rdf:type vivo:Course .
}
```

#### 4. Output on Fuseki

Table		Response	1 result in 0.017 seconds
Course_Count			
1	"1524"	<sup>^^</sup> <http://www.w3.org/2001/XMLSchema#integer>	

#### 5. Natural Language Response

```
Your input -> - What is the total number of courses?
I have 1524 courses in me.
```

### Additional Query 3

#### 1. User Input

For a course *c*, list all covered topics *t*, printing out their English labels and their DBpedia/Wikidata URI, together with the course event URI (e.g., 'lab3') and resource URI (e.g., 'slides10') where they appeared. Filter out duplicates.

#### 2. SPARQL Query

```
SELECT DISTINCT ?Topic_Name ?Topic_URI ?CourseEvent_URI ?Course_Content_URI
(CONCAT("COMP", " ", STR(6741)) AS ?Course_Number)
WHERE {
  ?course dbo:number 6741 .
  ?course vivo:hasSubjectArea "COMP" .
  ?course roboprof:hasCourseEvent ?CourseEvent_URI .
  ?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
  ?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
  ?Topic_URI dbo:name ?Topic_Name .
}
```

#### 3. Output on Fuseki

479 results

Table	Response	479 results in 0.039 seconds	Simple view	Ellipse	Filter query results
Topic_Name	Topic_URI	CourseEvent_URI	Course_Content_URI	Course_Number	
1 Wikidata	<http://dbpedia.org/resource/Wikidata>	<http://roboprof.ca/data/COMP_6741_Lecture_3>	<file:///dataset/Lecture%20Content/COMP%206741/Worksheets/worksheet03.pdf>	COMP 6741	
2 AI	<http://dbpedia.org/resource/Artificial_intelligence>	<http://roboprof.ca/data/COMP_6741_Lecture_3>	<file:///dataset/Lecture%20Content/COMP%206741/Worksheets/worksheet03.pdf>	COMP 6741	
3 Intelligent Systems	<http://dbpedia.org/resource/Artificial_intelligence>	<http://roboprof.ca/data/COMP_6741_Lecture_3>	<file:///dataset/Lecture%20Content/COMP%206741/Worksheets/worksheet03.pdf>	COMP 6741	
4 Artificial Intelligence	<http://dbpedia.org/resource/Artificial_intelligence>	<http://roboprof.ca/data/COMP_6741_Lecture_3>	<file:///dataset/Lecture%20Content/COMP%206741/Worksheets/worksheet03.pdf>	COMP 6741	
5 Linked Open Data	<http://dbpedia.org/resource/Linked_data>	<http://roboprof.ca/data/COMP_6741_Lecture_3>	<file:///dataset/Lecture%20Content/COMP%206741/Worksheets/worksheet03.pdf>	COMP 6741	

### Additional Query 4

#### 1. User Input

For a given topic *t* (DBpedia or Wikidata URI), list all courses *c* and their events *e* where the given topic *t* appears, along with the count of occurrences, with the results sorted by this count in descending order.

#### 2. SPARQL Query

```
SELECT ?Topic_URI (COUNT(?Topic_URI) AS ?Occurrence) ?Course_Name ?Course_URI
(CONCAT(?CourseEvent_Type_Label, " ", STR(?CourseEvent_Num)) AS ?Course_Event)
?CourseEvent_URI
WHERE {
  BIND(<http://dbpedia.org/resource/ALOHAnet> AS ?Topic_URI) .
  ?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
}
```

```

?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?CourseEvent_URI rdf:type ?CourseEvent_Type .
?CourseEvent_Type rdfs:label ?CourseEvent_Type_Label .
?CourseEvent_URI dbo:number ?CourseEvent_Num .
?Course_URI roboprof:hasCourseEvent ?CourseEvent_URI .
?Course_URI dbo:name ?Course_Name .
}
GROUP BY ?Topic_URI ?Course_Name ?Course_URI ?Course_Event ?CourseEvent_URI
?CourseEvent_Type_Label ?CourseEvent_Num
ORDER BY DESC(?Occurence)

```

### 3. Output on Fuseki

Table Response 2 results in 0.016 seconds Simple view Ellipse Filter query results

Topic_URI	Occurence	Course_Name	Course_URI	Course_Ev...	CourseEvent_URI
1 <http://dbpedia.org/resource/ALOHAnet>	"1"^^<http://www.w3.org/2001/XMLSchema#integer>	Computer Networks and Protocols	<http://roboprof.ca/data/COMP_6461>	Lecture 5	<http://roboprof.ca/data/COMP_6461_Lecture_5>
2 <http://dbpedia.org/resource/ALOHAnet>	"1"^^<http://www.w3.org/2001/XMLSchema#integer>	Computer Networks and Protocols	<http://roboprof.ca/data/COMP_6461>	Lecture 6	<http://roboprof.ca/data/COMP_6461_Lecture_6>

Showing 1 to 2 of 2 entries

## Additional Query 5

### 1. User Input

For a given topic *t*, list the precise course URI, course event URI and corresponding resource URI where the topic is covered (e.g., "NLP" is covered in COMP474 ! Lecture 10! Lab 10 Notes).

### 2. SPARQL Query

```

SELECT ?Course_URI ?CourseEvent_URI ?Course_Content_URI
WHERE {
?Topic_URI dbo:name "Deep Learning" .
?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
?Course_URI roboprof:hasCourseEvent ?CourseEvent_URI .
}

```

### 3. Output on Fuseki

Table Response 2 results in 0.026 seconds

Course_URI	CourseEvent_URI	Course_Content_URI
1 <http://roboprof.ca/data/COMP_6741>	<http://roboprof.ca/data/COMP_6741_Lecture_6>	<file:///dataset/Lecture%20Content/COMP%206741/Lectures/slides06.pdf>
2 <http://roboprof.ca/data/COMP_6741>	<http://roboprof.ca/data/COMP_6741_Lecture_1>	<file:///dataset/Lecture%20Content/COMP%206741/Lectures/slides01.pdf>

## Additional Query 6

### 4. User Input

Write a SPARQL query to identify any course events or resources within a specific course *c* that do not have any associated topic entities. This query helps in verifying that all relevant

educational materials have been adequately linked to topics in the knowledge base, ensuring completeness and providing insights into potential areas for improvement.

## 5. SPARQL Query

```
SELECT DISTINCT ?Course_Content_URI
WHERE {
  ?Course_URI dbo:name "Intelligent Systems" .
  ?Course_URI roboprof:hasCourseEvent ?CourseEvent_URI .
  ?CourseEvent_URI roboprof:coversCourseContent ?Course_Content_URI .
  FILTER NOT EXISTS {
    ?Topic_URI roboprof:provenanceInfo ?Course_Content_URI .
  }
}
```

## 6. Output on Fuseki

### Course\_Content\_URI

---

1	<a href="file:///./dataset/Lecture%20Content/COMP%206741/Readings/DeepLearning.pdf">&lt;file:///./dataset/Lecture%20Content/COMP%206741/Readings/DeepLearning.pdf&gt;</a>
---	---

---

One triple in the output, which means that all relevant educational materials have been adequately linked to topics in the knowledge base except for DeepLearning.pdf which means that there was no topic found in that pdf.

This is primarily because many pdfs were cut short to just 1 page so as to reduce the number of overall topics to avoid getting blocked from DBPEDIA endpoint.

## TRIPLESTORE AND SPARQL ENDPOINT SETUP

To setup the SPARQL endpoint to execute our queries, ours is a Windows System so we followed the following steps:

1. *We downloaded the [apache-jena-fuseki-5.0.0.zip](#) file.*

→ jena.apache.org/download/index.cgi

DOWNLOAD

## Apache Jena Releases

Apache Jena is packaged as downloads which contain the most commonly used portions of the systems:

- **apache-jena** – contains the APIs, SPARQL engine, the TDB native RDF database and command line tools
- **apache-jena-fuseki** – the Jena SPARQL server

Jena5 requires Java 17.

Jena jars are [available from Maven](#).

You may [verify the authenticity of artifacts](#) below by using the [PGP KEYS](#) file.

### Apache Jena Release

Source release: this forms the official release of Apache Jena. All binaries artifacts and maven binaries correspond to this source.

Apache Jena Release	SHA512	Signature
<a href="#">jena-5.0.0-source-release.zip</a>	<a href="#">SHA512</a>	<a href="#">PGP</a>

### Apache Jena Binary Distributions

The binary distribution of the Fuseki server:

Apache Jena Fuseki	SHA512	Signature
<a href="#">apache-jena-fuseki-5.0.0.tar.gz</a>	<a href="#">SHA512</a>	<a href="#">PGP</a>
<a href="#">apache-jena-fuseki-5.0.0.zip</a>	<a href="#">SHA512</a>	<a href="#">PGP</a>

**2. We extracted the zip file and copied the contents into a new folder called fuseki.**

× +

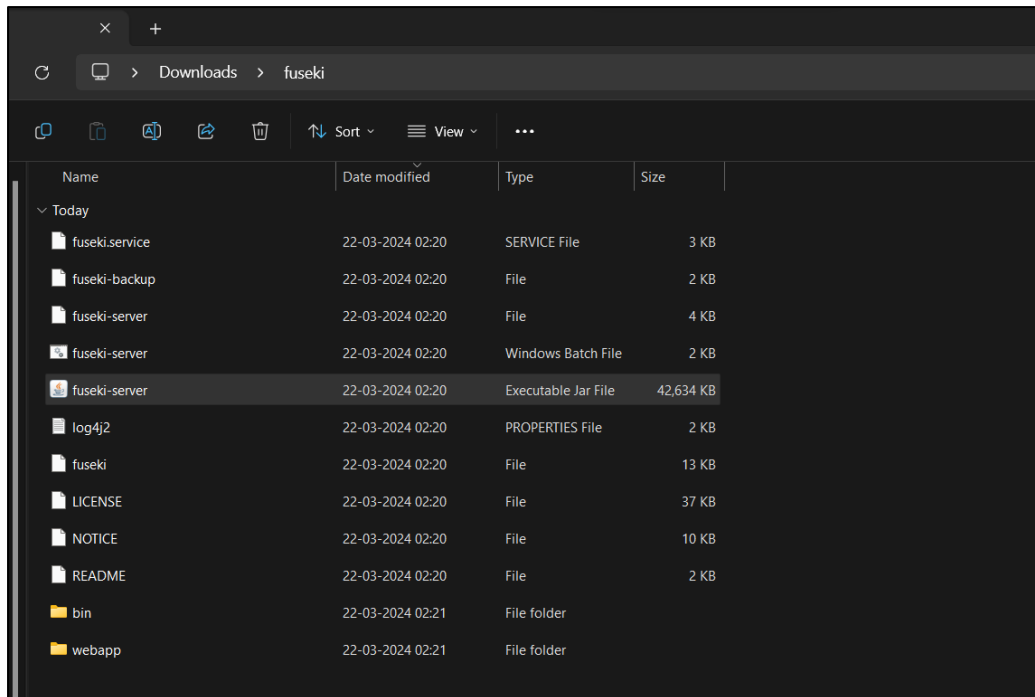
🔄 📁 > Downloads > fuseki

📁 📄 🗑️ 🔍 ⬆️ Sort View ⋮

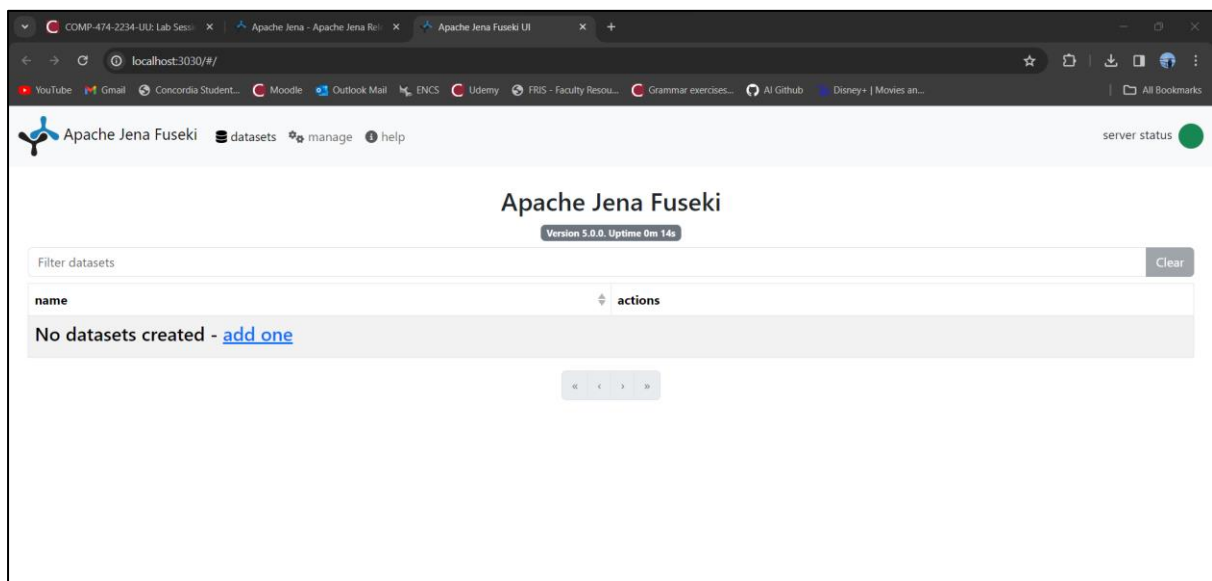
Name	Date modified	Type	Size
▼ Today			
📄 fuseki.service	22-03-2024 02:20	SERVICE File	3 KB
📄 fuseki-backup	22-03-2024 02:20	File	2 KB
📄 fuseki-server	22-03-2024 02:20	File	4 KB
📄 fuseki-server	22-03-2024 02:20	Windows Batch File	2 KB
📄 fuseki-server	22-03-2024 02:20	Executable Jar File	42,634 KB
📄 log4j2	22-03-2024 02:20	PROPERTIES File	2 KB
📄 fuseki	22-03-2024 02:20	File	13 KB
📄 LICENSE	22-03-2024 02:20	File	37 KB
📄 NOTICE	22-03-2024 02:20	File	10 KB
📄 README	22-03-2024 02:20	File	2 KB
📁 bin	22-03-2024 02:21	File folder	
📁 webapp	22-03-2024 02:21	File folder	

**3. We then selected the executable jar file and opened the file**

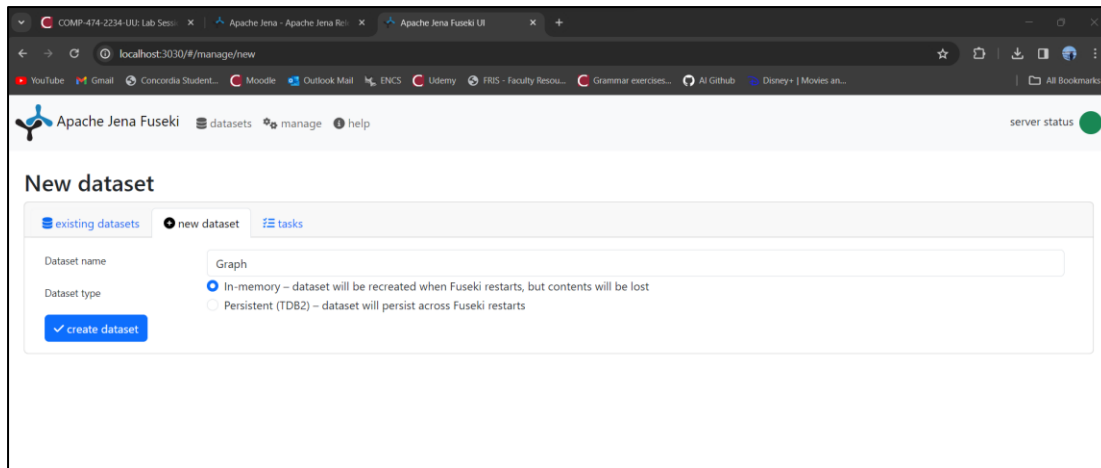




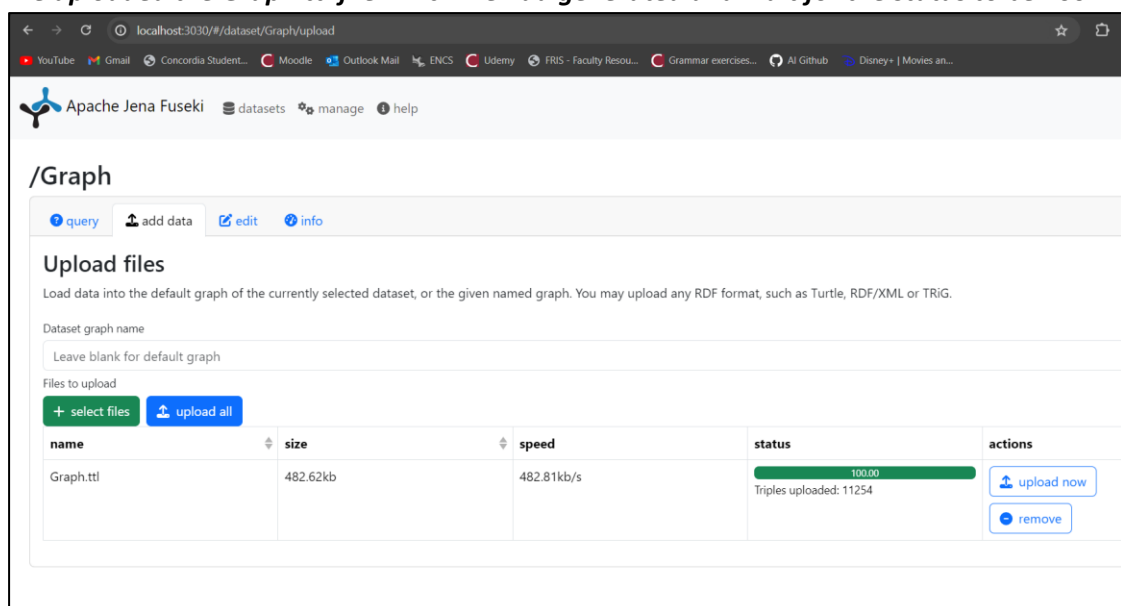
4. After the file is run, we go to any browser and go to <http://localhost:3030/>, if we open this, we were successful in running the server
5. Now we click on add one to add our Graph.ttl



6. We choose create dataset, after adding the dataset name, and choosing in memory.



**7. We uploaded the Graph.ttl file which we had generated and wait for the status to be 100.**



**8. After this we go to the query tab and put the query we have written and click on run, we are now able to see our query results**

