



Team 1

# Semantic Data Retrieval

By Sotheanith Sok, Lauro Cabral, Christopher Vargas, Abhinav Kacham, and Dinesh Reddy Kommera





# Datasets



# Arrest Data from 2020 to Present

- |                            |                           |
|----------------------------|---------------------------|
| ◇ Report ID                | ◇ Charge                  |
| ◇ Report Type              | ◇ Charge Description      |
| ◇ Arrest Date              | ◇ Disposition Description |
| ◇ Time                     | ◇ Address                 |
| ◇ Area ID                  | ◇ Cross Street            |
| ◇ Area Name                | ◇ LAT                     |
| ◇ Reporting District       | ◇ LON                     |
| ◇ Age                      | ◇ Location                |
| ◇ Sex Code                 | ◇ Booking Date            |
| ◇ Descent Code             | ◇ Booking Time            |
| ◇ Charge Group Code        | ◇ Booking Location        |
| ◇ Charge Group Description | ◇ Booking Location Code   |
| ◇ Arrest Type Code         |                           |





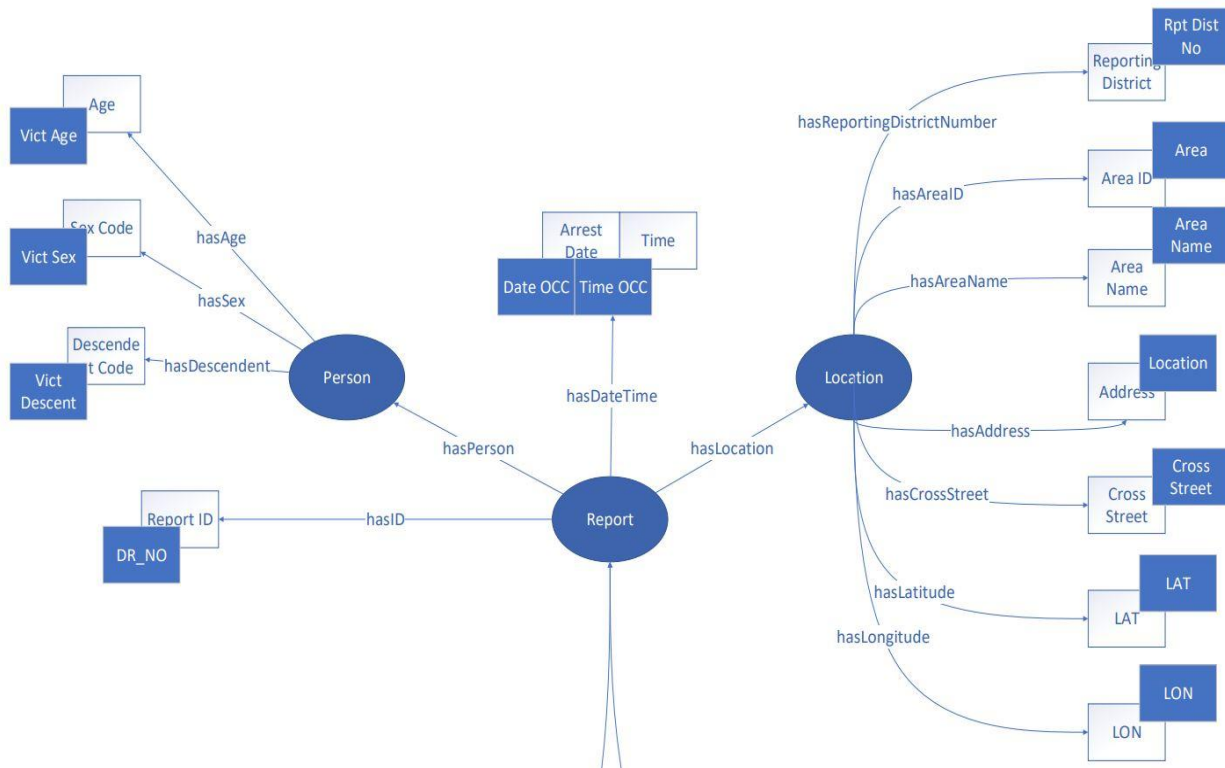
# Crime Data from 2020 to Present

- ◇ DR\_NO
- ◇ Date Rptd
- ◇ DATE OCC
- ◇ TIME OCC
- ◇ AREA
- ◇ AREA NAME
- ◇ Rpt Dist No
- ◇ Part 1-2
- ◇ Crm Cd
- ◇ Crm Cd Desc
- ◇ Mocodes
- ◇ Vict Age
- ◇ Vict Sex
- ◇ Vict Descent

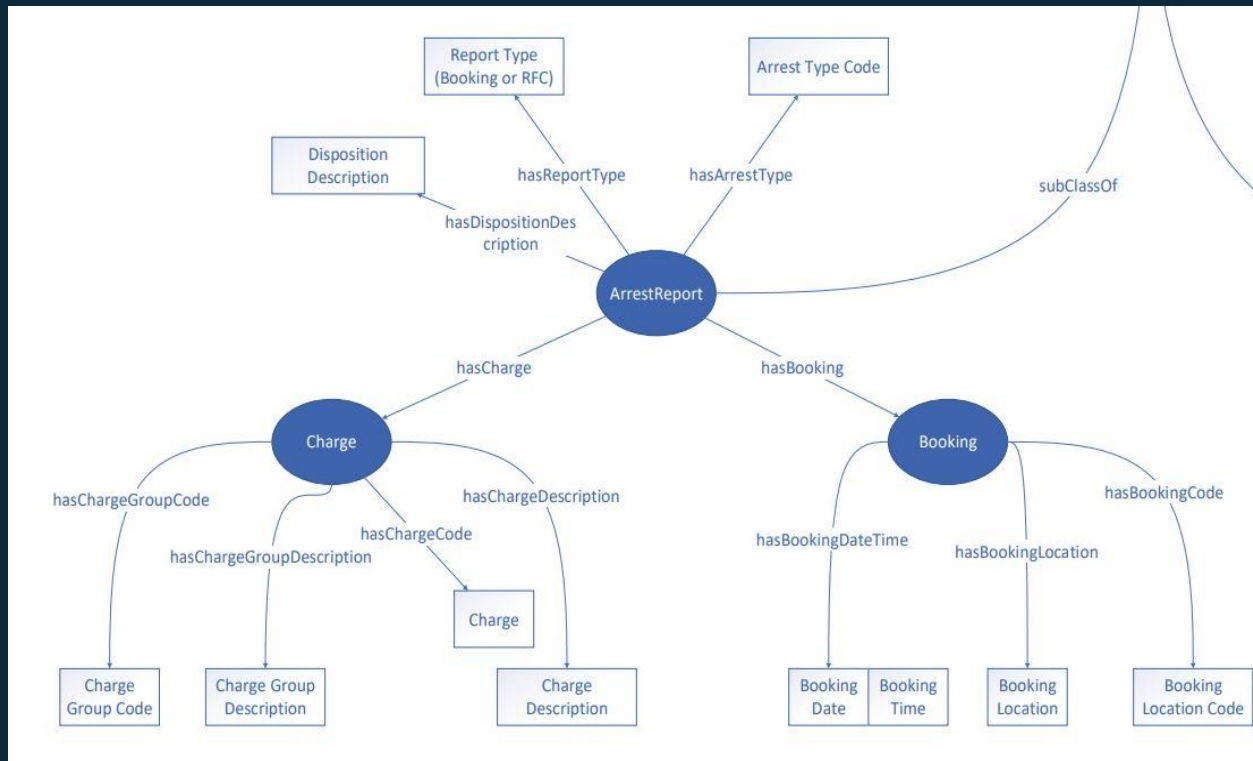
- ◇ Premis Cd
- ◇ Premis Desc
- ◇ Weapon Used Cd
- ◇ Weapon Desc
- ◇ Status
- ◇ Status Desc
- ◇ Crm Cd 1
- ◇ Crm Cd 2
- ◇ Crm Cd 3
- ◇ Crm Cd 4
- ◇ LOCATION
- ◇ Cross Street
- ◇ LAT
- ◇ LON



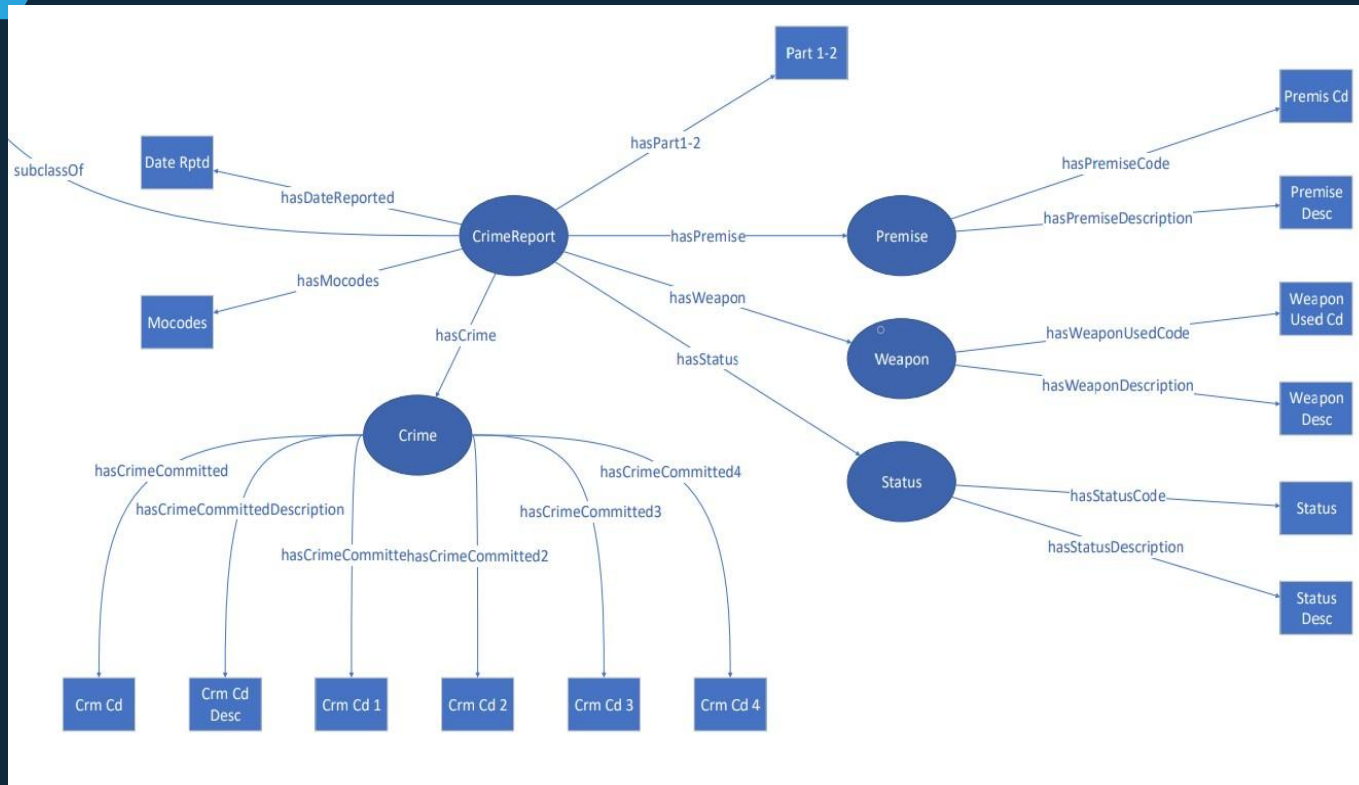
# Report Class



# ArrestReport Class



# CrimeReport Class





# Technical Debt

Issues with Projects 2





# Too Many Loops

## Problem:

- ◇ Most processes of our program for generating rdf files, such as data processing and adding data to rdf graph, rely on “for loops.”
- ◇ Everything has to be done linearly.
- ◇ It is very slow for our data size.

## Solution:

- ◇ Store all our data in Pandas dataframes.
- ◇ Convert as many “for loops” as possible to Pandas “map” function.
- ◇ “Map” function is faster than “for loop” because it vectorizes our data





# Naive Instance Checking of RDF Graph


## Problems:

- ◇ In project 2, the name of an instance of a class is formatted as “className - #s of instances of the class in the graph”
  - Ex: Report-0, Report-1,...etc.
- ◇ To check if an instance of a class already exists in the graph, we have to find all instances that have the property and then, we repeat the process for all properties.
- ◇ Finally, we intersect multiple lists of instances to check if there exist an instance with properties that we are looking for.
- ◇ It is really slow to do this.

## Solution:

- ◇ Change naming scheme to “className - hashValues”
- ◇  $\text{hasValues} = \text{md5}(\text{property}_1 + \text{property}_2 + \dots)$
- ◇ This approach means that two instances of a class with the same properties and properties' values will have the same name.
- ◇ This bypass the instance checking and any duplicate instances will simply override the existing one.



A decorative graphic on the left side of the slide consists of a large cyan hexagon in the center. Surrounding it are several smaller hexagons of varying shades of blue and cyan. Some of these hexagons contain white icons: a lightbulb, a thumbs-up, a smartphone, a magnifying glass, and a gear. There is also a network-like icon with a central node and three connecting lines, and a speech bubble icon.

# Search System using SPARQL

Integrating RDF Graph Manager with UI



# Requirements

## Old Libraries:

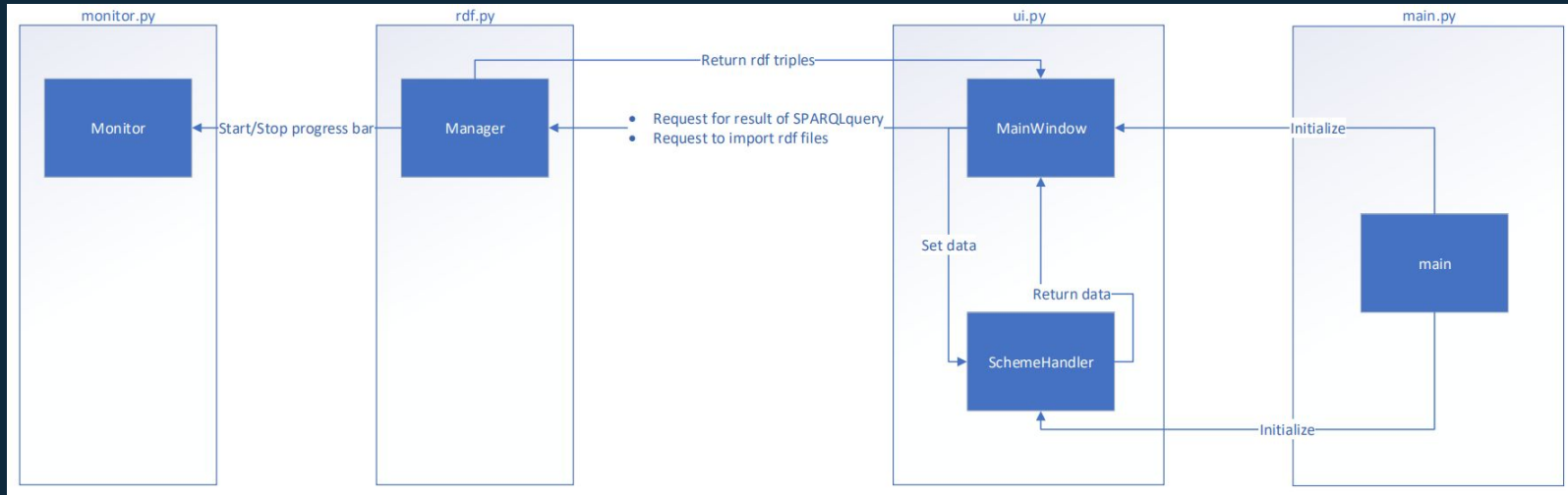
- ◇ Python
- ◇ Request
- ◇ RdfLib
- ◇ Pandas
- ◇ Auto-Py-to-Exe

## New Libraries:

- ◇ Tqdm - cli progress bar
- ◇ Dominate - generate dom using python syntax
- ◇ PyQt5 - gui library
- ◇ PyQtWebEngine - add web viewer to PyQt5



# Program Structure





# User Interfaces

SPARQL-with-LA-Public-Safety-Data

## LA Public Data Query Tool

RDF File Path:

Query:

41 results found

area	gender	count
77TH STREET	F	10
77TH STREET	M	4
CENTRAL	M	453
CENTRAL	F	281
DEVONSHIRE	M	5
DEVONSHIRE	F	2
FOOTHILL	M	3
FOOTHILL	F	1
HARBOR	F	6
HARBOR	M	2
HOLLENBECK	M	6
HOLLENBECK	F	3
HOLLYWOOD	M	4
HOLLYWOOD	F	1
MISSION	F	4
MISSION	M	1
N HOLLYWOOD	M	6
N HOLLYWOOD	F	4
NEWTON	M	2

Buttons: Import, Search, To Text, Chunk

Annotations:

- Path to RDF file
- Text Area to insert SPARQL Query
- HTML Viewer
- Button to import the RDF file
- Search button which enables the user to query information from data source
- Option to change the display of results into Text or HTML format
- Chunk to extract part of results



# Queries & Results



# Query 1:

Q: What is the time with the least amount of crime occurred?







# Query 1:

Query:

```
SELECT (COUNT(?report) AS ?number_of_crimes) ?hours WHERE {  
    {  
        SELECT (HOURS (?o) AS ?hours) ?report  
        WHERE {  
            ?report rdf:type ns1:CrimeReport .  
            ?report ns1:hasDateTime ?o  
        }  
    }  
}  
GROUP BY ?hours  
ORDER BY ?number_of_crimes  
LIMIT 5
```

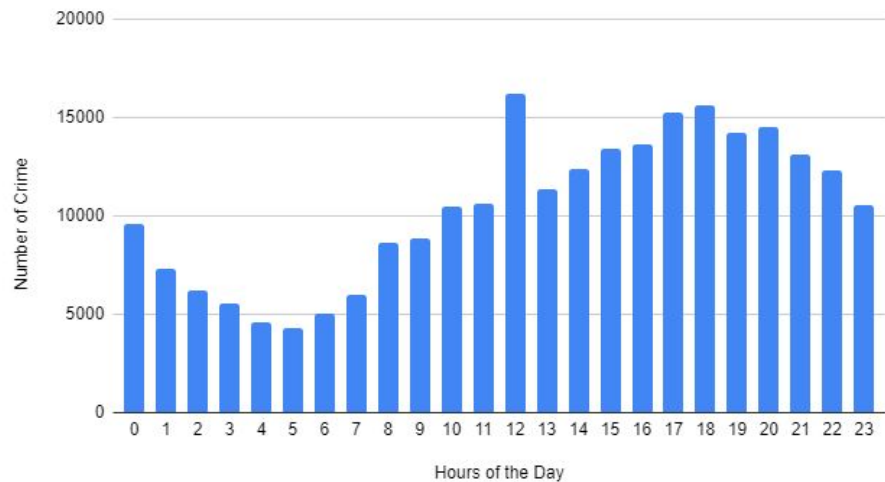


# Query 1:

Result:

number_of_crimes	hours
4296	5
4584	4
4976	6
5515	3
5968	7

Number of Crime vs. Hours of the Day





## Query 2:

Q: What is the ratio of the number of arrests of a person who committed a crime at a particular location to that of the total number of crimes located at that location?





## Query 2:

```
SELECT (?number_of_arrests/?number_of_crimes AS ?fraction)
WHERE {
  {
    SELECT (COUNT (?arrest_report) AS ?number_of_arrests)
    WHERE {
      ?arrest_report rdf:type ns1:ArrestReport .
      ?arrest_report ns1:hasLocation ?location .
      ?location ns1:hasAddress ?a_crime_address .
      FILTER (?a_crime_address = '6TH ST')
    }
  }
  .
  {
    SELECT (COUNT (?crime_report) AS ?number_of_crimes)
    WHERE {
      ?crime_report rdf:type ns1:CrimeReport .
      ?crime_report ns1:hasLocation ?location .
      ?location ns1:hasAddress ?c_crime_address .
      FILTER (?c_crime_address = '6TH ST')
    }
  }
}
```





## Query 2:

Result:

<b>fraction</b>
0.8684210526315789473684210526





## Query 3:

Q: For a given url reference, what are rdf triples that contain such a reference?

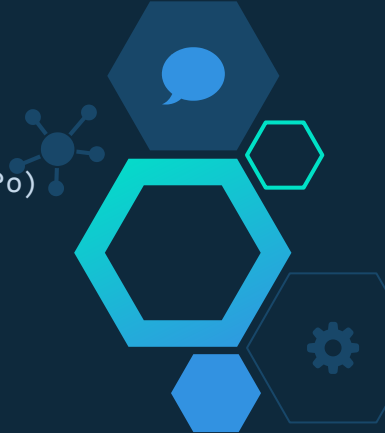




## Query 3:

Query: Let X be the url reference

```
SELECT ?s ?p ?o WHERE {  
  {  
    SELECT (COALESCE(X) as ?s) ?p ?o  
    WHERE {  
      X ?p ?o  
    }  
  } UNION {  
    SELECT ?s (COALESCE(X) as ?p) ?o  
    WHERE {  
      ?s X ?o  
    }  
  } UNION {  
    SELECT ?s ?p (COALESCE(X) as ?o)  
    WHERE {  
      ?s ?p X  
    }  
  }  
}
```





## Query 3:

Result:

s	p	o
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasPremise</a>	<a href="#">ns1:Premise-fe9e7928e7081b33ec2c15e0c0583de7</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasPart1-2</a>	2
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasDateReported</a>	2020-02-15T00:00:00
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">rdf:type</a>	<a href="#">ns1:CrimeReport</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasPerson</a>	<a href="#">ns1:Person-1fdb69221a3b65e3b067e0a764b281e3</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasStatus</a>	<a href="#">ns1:Status-1f50412150e198650794109875febf97</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasWeapon</a>	<a href="#">ns1:Weapon-7dc160b92a3f59f6682eea9f30c2c9dd</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasCrime</a>	<a href="#">ns1:Crime-1e3e249fb75c6a4b929212358ef71437</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasLocation</a>	<a href="#">ns1:Location-251c6008c85483eea74d1022617380bc</a>
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasMocodes</a>	0913 0421 0319
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasDateTime</a>	2020-02-15T21:00:00
<a href="#">ns1:Report-d370436c2f0b2b410581e7f32d4d6ed2</a>	<a href="#">ns1:hasID</a>	200306756







# Technical Challenges

Problem, problem, and more problems...



# Web Viewer

## Problems:

1. Since the web viewer is based on chromium, it has an inherent 2mb limit when you set content of the viewer directly.
2. The web viewer has a performance and an instability when displaying table with 10,000 rows or more.

## Solutions:

1. Implement “SchemeHandler” that is responsible for intercepting http request and send back data to the web viewer. It also allows controller to set data in which it supposes to reply with.
2. Split results into chunks of 1,000 data and only one chunk gets send to the web viewer at any given time.





# SPARQL Quirk

## Problems:

- ◆ Unlike SQL, SPARQL has limited set of flexible capabilities.
- ◆ There are special characters that mark the end of a term in a query such as “#”.
- ◆ Some SPARQL functions only work with `xsd:dateTime` rather than `xsd:date` or `xsd:time`.

## Solutions:

- ◆ Updated ‘#’ with ‘-’ RDF, as it has own importance in SPARQL.
- ◆ Merge `xsd:date` and `xsd:time` to `xsd:dateTime`.
- ◆ For `xsd:date` only, assume time is 00:00:00.





# Others

1. On some machine, there are python compatibility issues with current libraries and it requires a complete reinstall and downgrade python from 3.9 to 3.8.
2. We have to do a lot of learning since this is our first time working with these datasets or libraries/framework.





Thank you