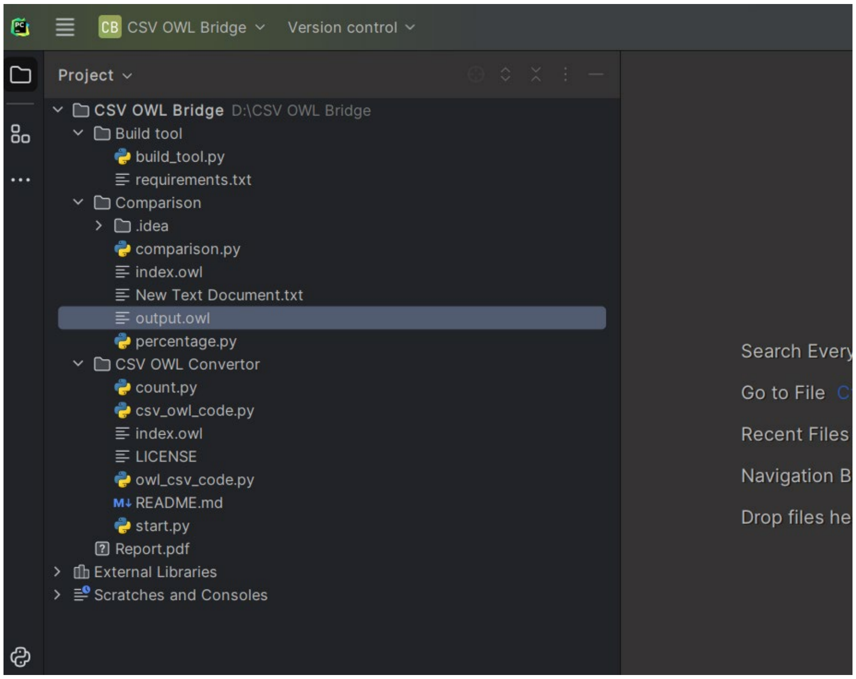
CSV – OWL Bridge

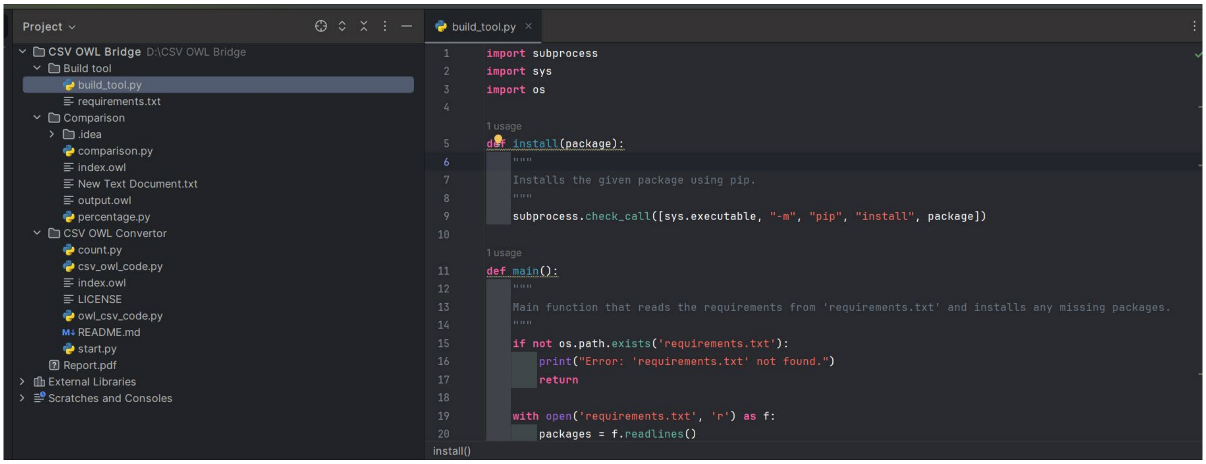
Project- IP
Topic- CSV – OWL Bridge
Professor- Raghava Mutharaju
Members- Anand (2020280), Rohit (2020538)

How to run:

Open the whole CSV OWL Bridge folder in as an project.

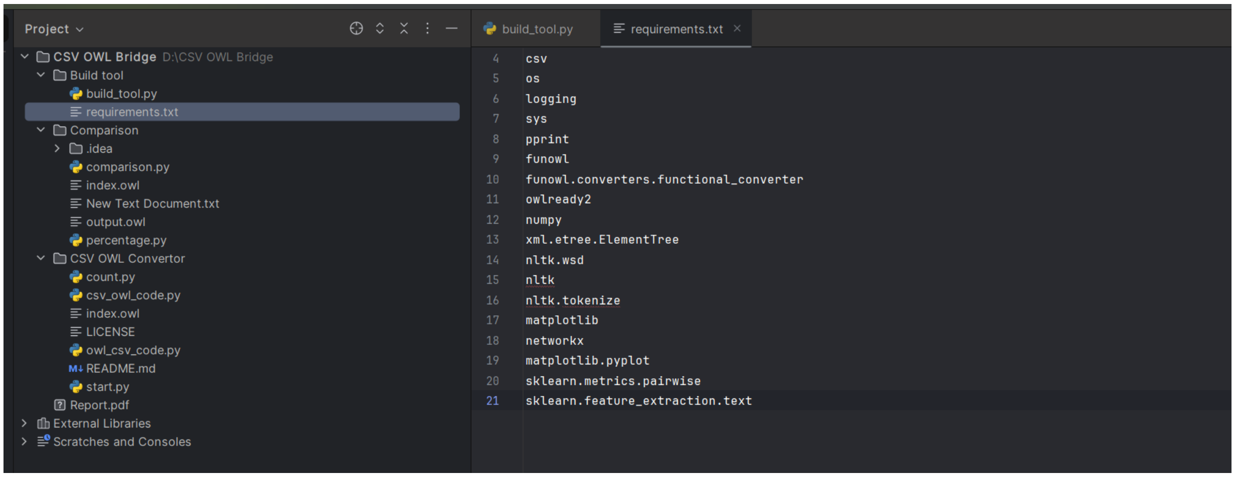


1. Firstly, we will Open the Build toll folder.
2. Then we will run the “build tool.py” python file to install all the necessary libraries that we will be needing for the project.



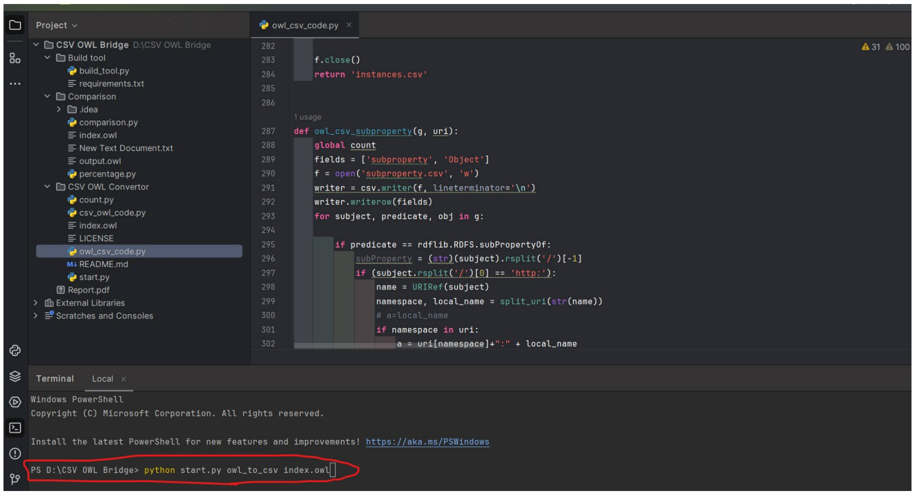
```python
import subprocess
import sys
import os

def install(package):
    """
    Installs the given package using pip.
    """
    subprocess.check_call([sys.executable, "-m", "pip", "install", package])

def main():
    """
    Main function that reads the requirements from 'requirements.txt' and installs any missing packages.
    """
    if not os.path.exists('requirements.txt'):
        print("Error: 'requirements.txt' not found.")
        return

    with open('requirements.txt', 'r') as f:
        packages = f.readlines()
```

3. To add more libraries just add the name of the libraries in the requirements.txt text file.



```
csv
os
logging
sys
pprint
funowl
funowl.converters.functional_converter
owlready2
numpy
xml.etree.ElementTree
nltk.wsd
nltk
nltk.tokenize
matplotlib
networkx
matplotlib.pyplot
sklearn.metrics.pairwise
sklearn.feature_extraction.text
```

4. Next, we will Open the CSV OWL Convertor folder.
5. Now, we will run “owl_csv_code.py” pythn file by typing the command “python start.py owl_to_csv index.owl” in the cmd. (index.owl file is already present)

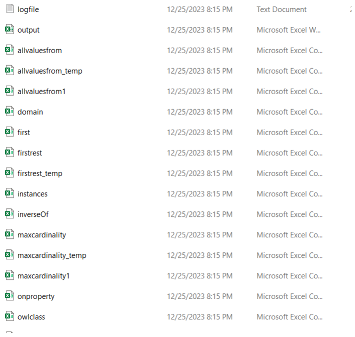


```
CSV OWL Bridge D:\CSV OWL Bridge
  Build tool
    build_tool.py
    requirements.txt
  Comparison
    .idea
    comparison.py
    index.owl
    New Text Document.txt
    output.owl
    percentage.py
  CSV OWL Convertor
    count.py
    csv_owl_code.py
    index.owl
    LICENSE
    owl_csv_code.py
    README.md
    start.py
  Report.pdf
External Libraries
Scratches and Consoles
```

```python
    f.close()
    return 'instances.csv'


1 usage
def owl_csv_subproperty(g, uri):
    global count
    fields = ['subproperty', 'Object']
    f = open('subproperty.csv', 'w')
    writer = csv.writer(f, lineterminator='\n')
    writer.writerow(fields)
    for subject, predicate, obj in g:

        if predicate == rdflib.RDFS.subPropertyOf:
            subProperty = (str)(subject).rsplit('/')[-1]
            if (subject.rsplit('/')[0] == 'http:'):
                name = URIRef(subject)
                namespace, local_name = split_uri(str(name))
                # a=local_name
                if namespace in uri:
                    a = uri[namespace]+":" + local_name
```

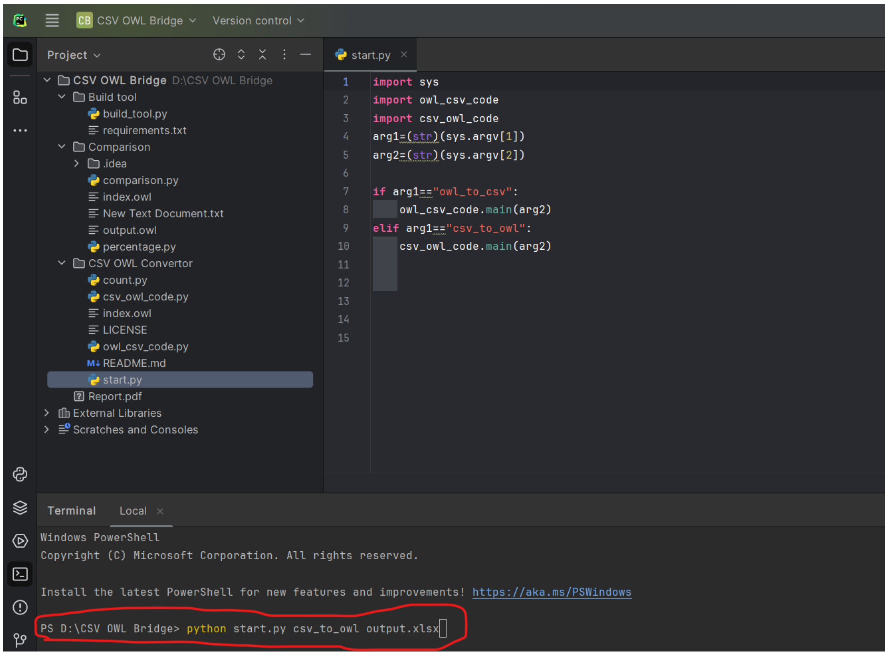
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\CSV OWL Bridge> python start.py owl_to_csv index.owl
```

6. Then after we get the output.csv and several other csv files as we required as the output and then we will now convert back to owl.



logfile	12/25/2023 8:15 PM	Text Document
output	12/25/2023 8:15 PM	Microsoft Excel W...
allvaluesfrom	12/25/2023 8:15 PM	Microsoft Excel Co...
allvaluesfrom_temp	12/25/2023 8:15 PM	Microsoft Excel Co...
allvaluesfrom1	12/25/2023 8:15 PM	Microsoft Excel Co...
domain	12/25/2023 8:15 PM	Microsoft Excel Co...
first	12/25/2023 8:15 PM	Microsoft Excel Co...
firstrest	12/25/2023 8:15 PM	Microsoft Excel Co...
firstrest_temp	12/25/2023 8:15 PM	Microsoft Excel Co...
instances	12/25/2023 8:15 PM	Microsoft Excel Co...
inverseOf	12/25/2023 8:15 PM	Microsoft Excel Co...
maxcardinality	12/25/2023 8:15 PM	Microsoft Excel Co...
maxcardinality_temp	12/25/2023 8:15 PM	Microsoft Excel Co...
maxcardinality1	12/25/2023 8:15 PM	Microsoft Excel Co...
onproperty	12/25/2023 8:15 PM	Microsoft Excel Co...
owlclass	12/25/2023 8:15 PM	Microsoft Excel Co...

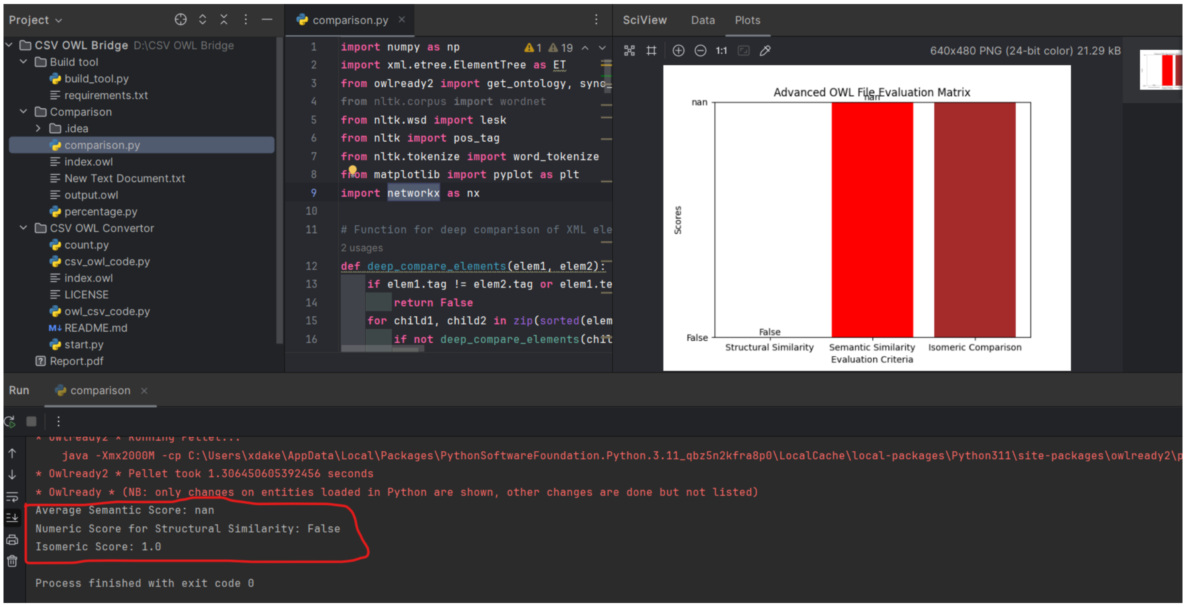
7. For csv to owl we will run “csv_owl_code.py” python file using the command “python start.py csv_to_owl output.xlsx” in cmd. This will give us a “output.owl” file.



```python
import sys
import owl_csv_code
import csv_owl_code
arg1=(str)(sys.argv[1])
arg2=(str)(sys.argv[2])

if arg1=="owl_to_csv":
    owl_csv_code.main(arg2)
elif arg1=="csv_to_owl":
    csv_owl_code.main(arg2)
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\CSV OWL Bridge> python start.py csv_to_owl output.xlsx
```

8. Now we will do the comparison. For this we will open the “Comparison folder”.
9. In this we will directly run the “comparison.py” for getting the structural, semantic and isomeric similarity score and graph. And the “percentage.py” will give the percentage of matching lines.



```python
import numpy as np
import xml.etree.ElementTree as ET
from owlready2 import get_ontology, sync_
from nltk.corpus import wordnet
from nltk.wsd import lesk
from nltk import pos_tag
from nltk.tokenize import word_tokenize
from matplotlib import pyplot as plt
import networkx as nx

# Function for deep comparison of XML ele
def deep_compare_elements(elem1, elem2):
    if elem1.tag != elem2.tag or elem1.te
        return False
    for child1, child2 in zip(sorted(elem
        if not deep_compare_elements(chil
```

```
* Owlready2 * Running Pellet...
    java -Xmx2000M -cp C:\Users\xdake\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\LocalCache\local-packages\Python311\site-packages\owlready2\p
* Owlready2 * Pellet took 1.306450605392456 seconds
* Owlready * (NB: only changes on entities loaded in Python are shown, other changes are done but not listed)
Average Semantic Score: nan
Numeric Score for Structural Similarity: False
Isomeric Score: 1.0

Process finished with exit code 0
```

Getting the output as:

1. average semantic score: NAN

Meaning: The average semantic similarity score between individual names in the two ontologies could not be computed in numerical terms. This is indicated by "nan," which stands for "Not a Number." It suggests that the semantic comparison algorithm may not have found suitable in numerical form comparison. But seeing the graph we can the score as 1. As they are almost identical.

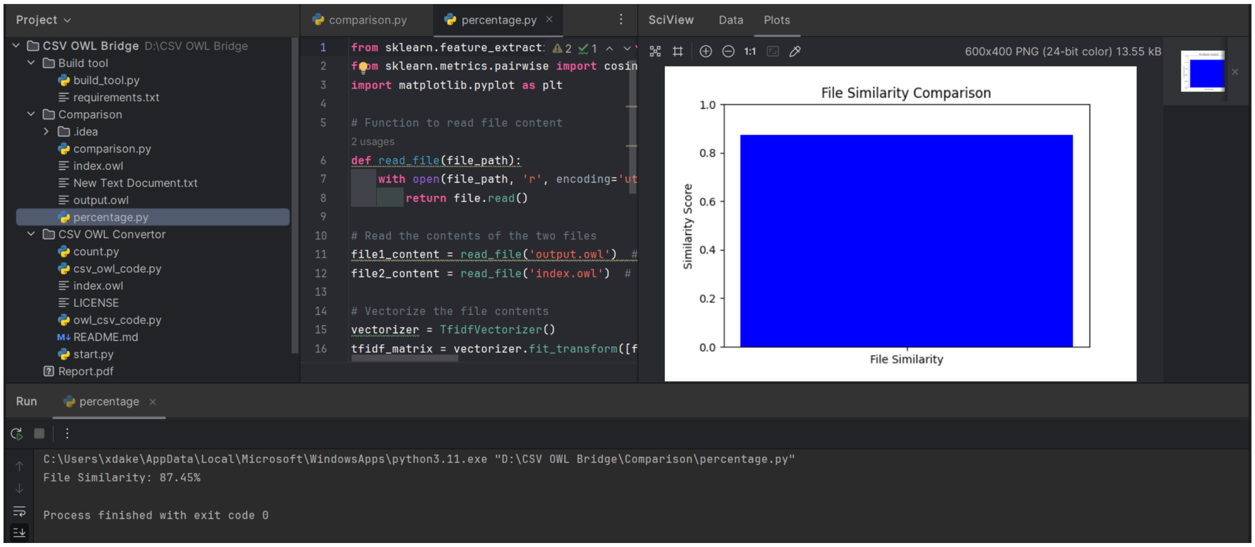
2. Numerical Score for Structural Similarity: False

Meaning: The structural similarity comparison between the XML structures of the two OWL files resulted in "False." This indicates that the XML structures are not identical. The deep comparison of XML elements returned a boolean result, and "False" suggests a lack of structural similarity and there are differences.

3. Isomeric Score: 1.0

Meaning: The isomeric comparison between the ontologies as graphs resulted in a score of 1.0. This implies that the ontologies are structurally isomorphic, meaning they have the same structure when represented as graphs. The value "1.0" indicates a perfect isomeric match.

Percentage Match:



```
from sklearn.feature_extract
from sklearn.metrics.pairwise import cosin
import matplotlib.pyplot as plt

# Function to read file content
def read_file(file_path):
    with open(file_path, 'r', encoding='ut
        return file.read()

# Read the contents of the two files
file1_content = read_file('output.owl')  #
file2_content = read_file('index.owl')  #

# Vectorize the file contents
vectorizer = TfidfVectorizer()
tfidf_matrix = vectorizer.fit_transform([f
```

```
C:\Users\xdake\AppData\Local\Microsoft\WindowsApps\python3.11.exe "D:\CSV OWL Bridge\Comparison\percentage.py"
File Similarity: 87.45%

Process finished with exit code 0
```

File Similarity: 87.45%

Meaning: The files have an 87.45% similarity based on their content. This similarity is computed using the TF-IDF representation of the files, and the cosine similarity metric.