Problem Description:

Implement an Information Extraction application using NLP features and techniques.

Input:

- Set of information templates
- Set of natural language statements

Output:

- Filled information templates

Proposed Solution:

1. We approached this problem using a combination of Wordnet based tools such as NLTK, SpaCy library, Stanford Dependency Parser for natural language processing.
2. We selected **Economic News Article Tone and Relevance** corpus downloaded directly from figure-eight website and extracted the articles that are relevant to our templates.
3. Our Templates:
   i. **Predict:** (Predictor, cause_subject, positive/negative impact, cause_object)
   ii. **Purchase**: (Price, Date, Purchaser, Items)
   iii. **Invest:** (Investor, capital, endeavor,Purpose,Date)
   iv. **Resign:** (Organization,Position,Entity,Location):
   v. **Launch**: (launcher_person/org,commodity/plan,effect,effected person/organization)
   vi. **Pass**: (Issuer, Bill/law, Date, Location)
   vii. **Elect:** (person, position, organization,elector, ordinal)
   viii. **Decline**: (commodity, from_point, to_point, location)
   ix. **Borrow**: (Amount, Lender, Borrower, Commodity)
   x. **Post**: (Amount, Type, Term/Date, Poster)
4. Tokenized the corpus into sentences and words.
5. We split corpus into sentences and used NLTK for extracting following features:

xi. POS Tags

xii. Lemma

xiii. Dependency parse trees

xiv. Hypernyms

xv. Hyponyms

xvi. Holonyms

xvii. Meronyms

6. For implementing part 4 we used a heuristic approach
   - Template Matching:
     - When the user inputs a sentence, we first tokenize the sentence into words, then find POS tags for all the words. We only took the words that has POS tags related to verb, singular and plural nouns (eliminated proper nouns). Stored the required words in check_tokens list.
     - We then took the stem of each word in check_tokens, found synonyms, hypernyms, hyponyms, meronyms, holonyms and stored in related_list. Now each token will have its related_list.
     - The code compares our templates with the related_list of each token and calls the matching template. If the inputted sentence contains words that match multiple templates, it calls all the matching templates and outputs properties for each template.
     - The function call for extraction code of particular template includes arguments as the input_sentence, the token present in the sentence which matched the sentence to that particular template. We call it true_token . For example, the user sentence is "**Berson also nailed his forecast for the change in the consumer-price index, which advanced 2.8% for the 12 months through May.**" The call for predict.py includes arguments (User_input_sentence, 'forecast'). Here true_token is 'forecast'.
   - Extraction of Subject:
     - Found the index of the true_token in the input_sentence and splitted the sentence into subject sentence and object sentence considering both active and passive voice.

    - Sentence in which true_token is in **Verb Form (Ex: Invested)**: From the subject_sentence, found the subject which is 'nsubj' of the true_token from dependency parse tree. Found noun_chunk form of the word extracted from dependency parse tree using spaCy library and some code which fetches head nouns forms by iterating through parse tree.
    - Sentence in which true_token is in **Noun Form (Ex: Investment):** Took noun_forms present in subject_sentence into a subject_list. If there exists a 'poss' or 'possesive' relation between a noun and the true_token, we consider that as the subject or else we output list of possible subjects.
- Extraction of Object:
    - Found the object similar to subject. Used object_sentence, 'dobj', 'pobj' and some other relations in the parse tree. Used noun_chunck and above specified methods to get full form of the object.
- Extraction specific to templates (Predict, Invest, Launch):
    - **Predict:** (Predictor, cause_subject, positive/negative impact, cause_object)**:** For example, in the sentence " Lufkin & Jenrette Inc. predicted the Fed will cut the discount rate to 5% by mid-February.", Predictor is " Lufkin & Jenrette Inc.", cause_subject is "the Fed", impact is "cut", cause_object is "the discount_rate". Predictor comes from subject_sentence. Remaining all come from object_sentence. We found subject and objects in the object_sentence using the same approach as specified above. Basically we applied the subject and object extraction code twice here.
        - **Approach 2**: For verb forms, used clausual complement, relative clause modifier, verbal modifier, clausual complement from the dependency tree to find out impact. After getting impact, found subject and object of it using general approach specified above. Approach 1 worked better.
    - **Invest:** (Investor, capital, endeavor,Purpose,Date): Extracted Endeavor using same technique as predict. Extracted capital and Date using NER in SpaCy library. Approach 2 worked better.
    - **Launch**: (launcher_person/org,commodity/plan,effect,effected person/organization)
- Extraction specific to templates (elect, resign):

  - **Elect:** (person, position, organization,elector, ordinal): Found the subject and object in general approach. For finding position, we used pre-defined list of positions that are applicable to economy and matched the sentence to the list. Found organization and ordinal using NER in SpaCy library.
  - **Resign:** (Organization,Position,Entity,Location): Extracted everything except location as specified above. Used NER in SpaCy to extract location.
- Extraction specific to template (decline):
  - **Decline**: (commodity, from_point, to_point, location): Extracted from_point and to_point using direct object and object of prepostion from dependency parse tree.
- Extraction specific to template (purchase):
  - **Purchase**: (Price, Date, Purchaser, Items): For example the sentence "*Amazon.com Inc.'s acquisition of Whole Foods Market Inc. for $13.7 billion was made in 2017*", Purchaser is Amazon.com Inc, Items is *Whole Foods Market Inc, Price is $13.7 billion and* Date is 2017.
- Extraction specific to template (post):
  - **Post**: (Amount, Type, Term/Date, Poster): For example the sentence "Ford Motor Co. posted a 2.6% decline in March sales on Monday", Amount was 2.6%, Type was decline, Date was Monday and Poster was Ford Motor Co.
- Extraction specific to template (pass):
  - **Pass**: (Issuer, Bill/law, Date, Location): For example the sentence "The President has threatened to veto the Employment bill passed by the House and the Senate on Monday, December 10th at Washington D.C.", Issuer was House and Senate, Bill/law was Employment Bill, Date was December 10th and Location was Washington.
- Extraction specific to template (Borrow):
  - **Borrow**: (Amount, Lender, Borrower, Commodity): For example the sentence "John borrowed two books from Jack last Monday.", Amount was 2, Lender was Jack, Borrower was John and Commodity was Books.

7. Finally, we evaluated the results for 10 examples for each template.

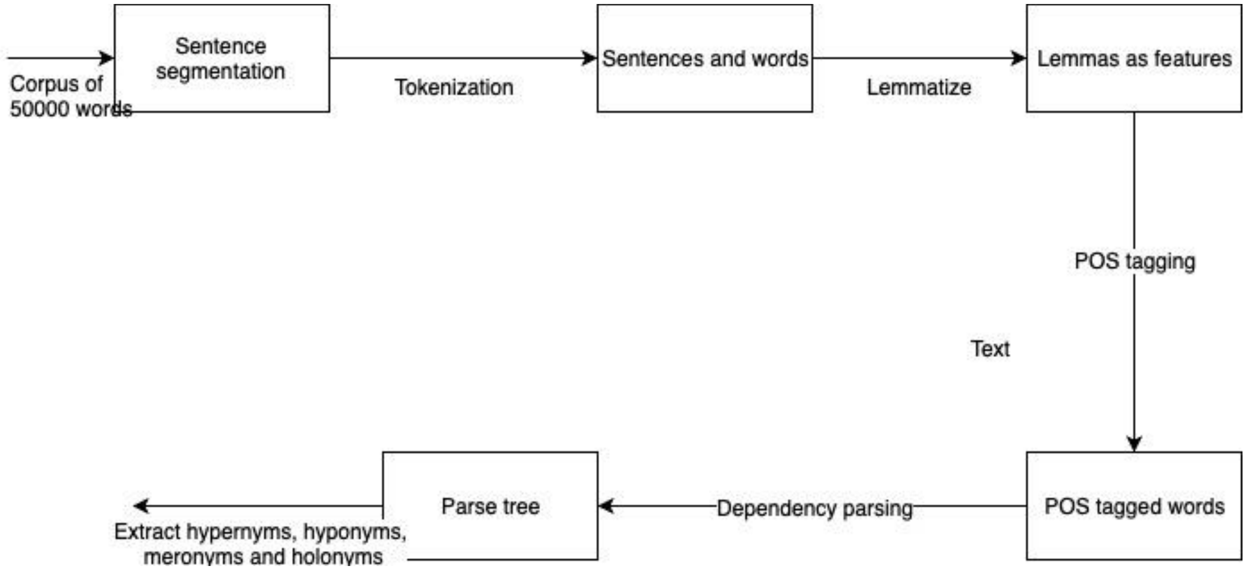
Implementation Details:

We used the following software tools for implementation:

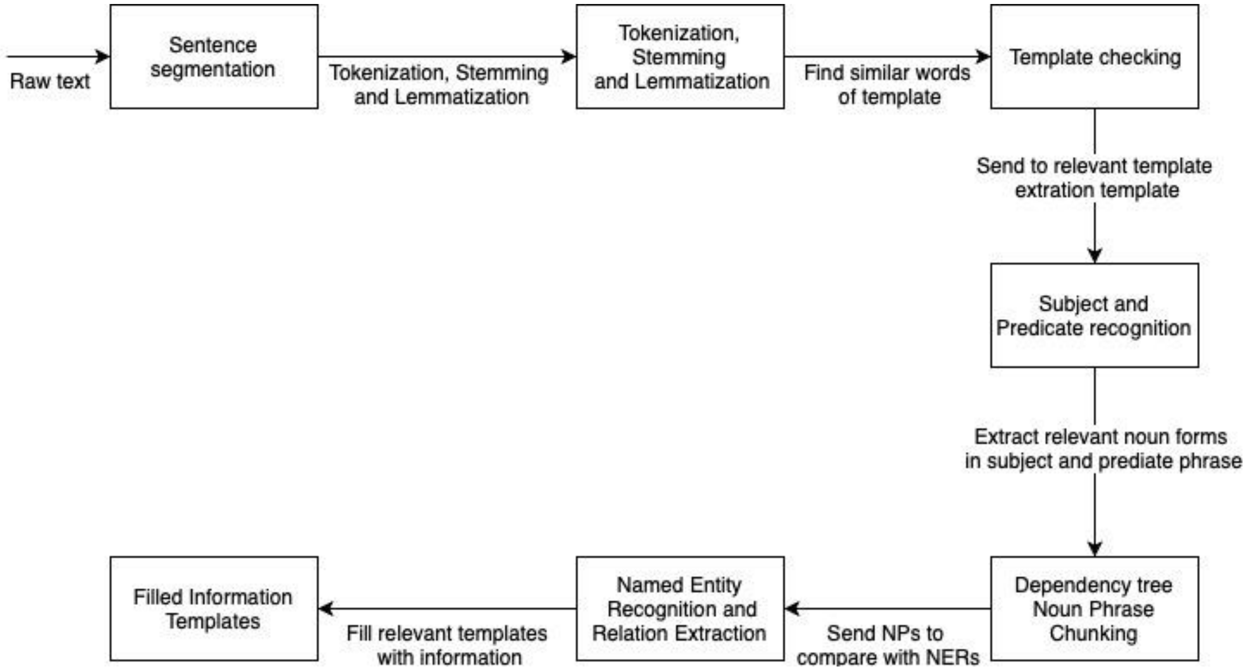
1. Python 2.7
2. NLTK
    a. WordNet
    b. WordNetLemmatizer
    c. PorterStemmer
    d. StanfordDependencyParser

Architectural Diagram:

Part 3:



Part 4:


Raw text
Sentence segmentation
Tokenization, Stemming and Lemmatization
Tokenization, Stemming and Lemmatization
Find similar words of template
Template checking
Send to relevant template extration template
Subject and Predicate recognition
Extract relevant noun forms in subject and prediate phrase
Dependency tree Noun Phrase Chunking
Send NPs to compare with NERs
Named Entity Recognition and Relation Extraction
Fill relevant templates with information
Filled Information Templates

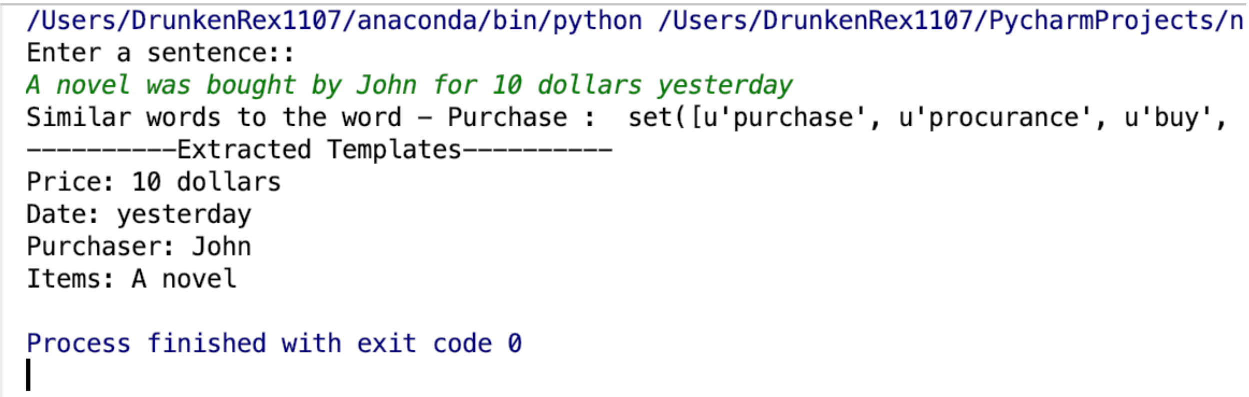
Project Files and their roles:

1. Feature_Extraction.py - Extracts the features for an input sentence or the whole corpus based on the option that user chooses.
2. Template_Matcher.py – This program matches the input sentence to templates and runs the respective template code.
3. 10 files named with their respective template (Ex: predict.py)- contains code for filling the respective template.

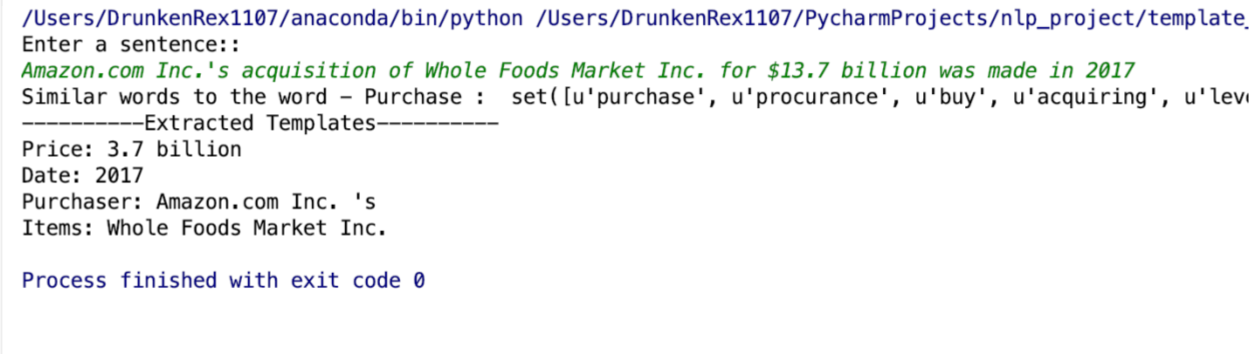
Experiments and Results:

We evaluated the results for each template with corresponding sentences.

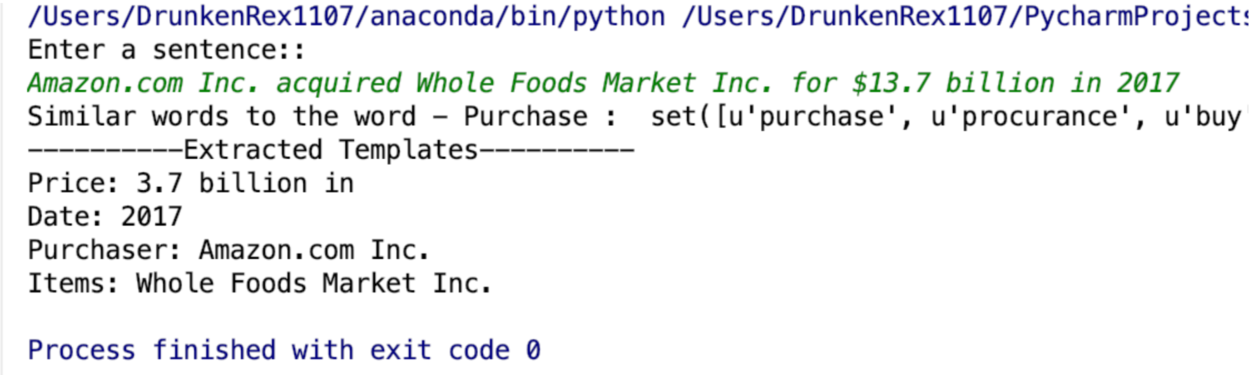
<u>**Purchase**: (Price, Date, Purchaser, Items):</u>



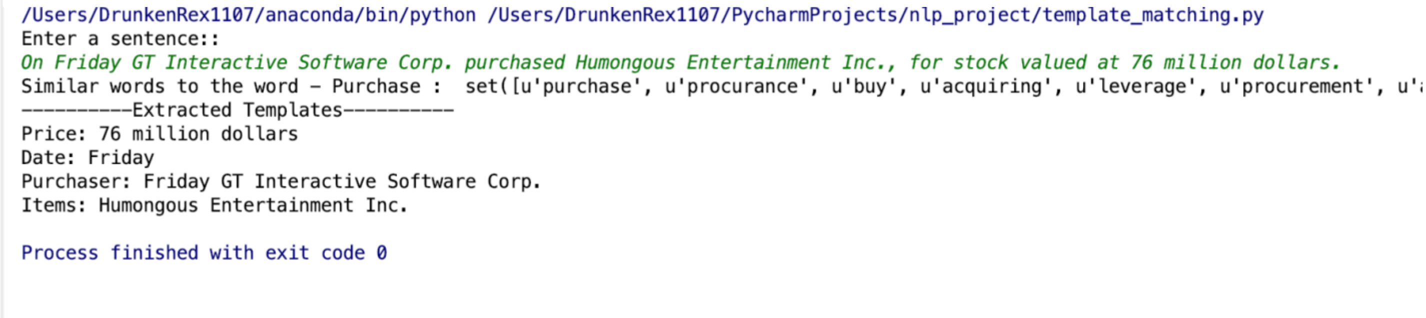
```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/n
Enter a sentence::
A novel was bought by John for 10 dollars yesterday
Similar words to the word - Purchase :  set([u'purchase', u'procurance', u'buy',
----------Extracted Templates----------
Price: 10 dollars
Date: yesterday
Purchaser: John
Items: A novel

Process finished with exit code 0
```



```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_
Enter a sentence::
Amazon.com Inc.'s acquisition of Whole Foods Market Inc. for $13.7 billion was made in 2017
Similar words to the word - Purchase :  set([u'purchase', u'procurance', u'buy', u'acquiring', u'leve
----------Extracted Templates----------
Price: 3.7 billion
Date: 2017
Purchaser: Amazon.com Inc. 's
Items: Whole Foods Market Inc.

Process finished with exit code 0
```

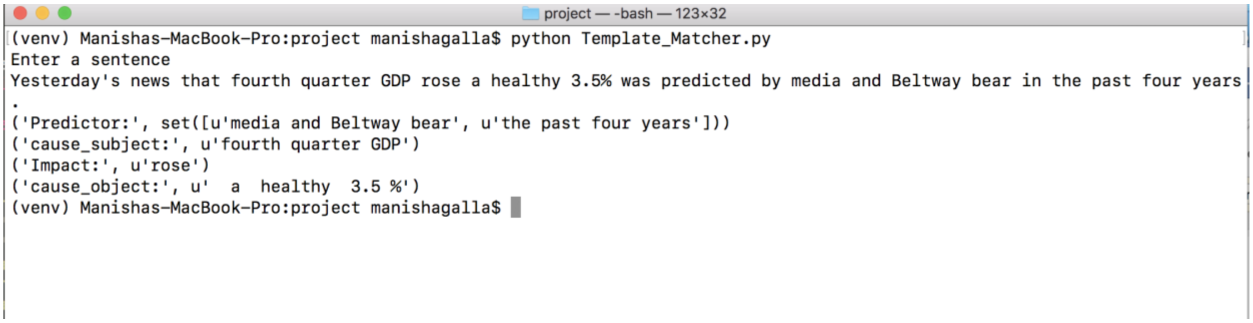


```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects
Enter a sentence::
Amazon.com Inc. acquired Whole Foods Market Inc. for $13.7 billion in 2017
Similar words to the word - Purchase :  set([u'purchase', u'procurance', u'buy
----------Extracted Templates----------
Price: 3.7 billion in
Date: 2017
Purchaser: Amazon.com Inc.
Items: Whole Foods Market Inc.

Process finished with exit code 0
```



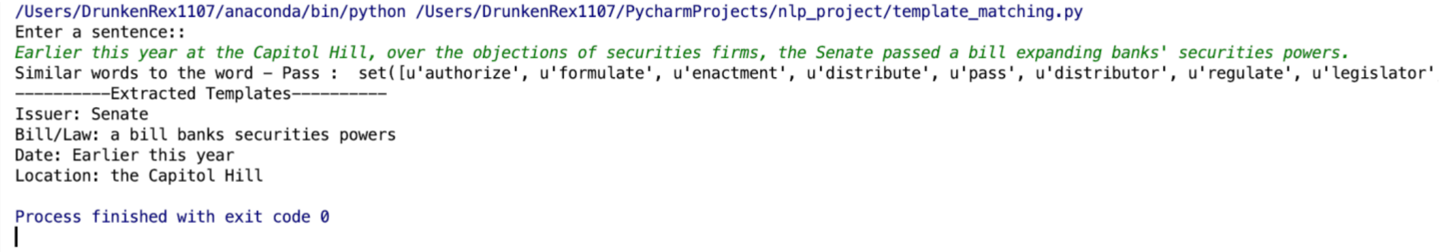
```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
On Friday GT Interactive Software Corp. purchased Humongous Entertainment Inc., for stock valued at 76 million dollars.
Similar words to the word - Purchase :  set([u'purchase', u'procurance', u'buy', u'acquiring', u'leverage', u'procurement', u'a
----------Extracted Templates----------
Price: 76 million dollars
Date: Friday
Purchaser: Friday GT Interactive Software Corp.
Items: Humongous Entertainment Inc.

Process finished with exit code 0
```

Predict: ((Predictor, cause_subject, positive/negative impact, cause_object,amount)

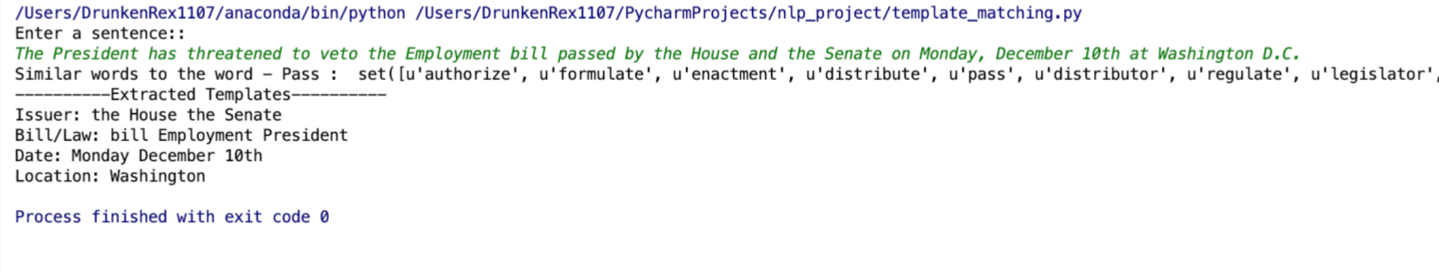


```
(venv) Manishas-MacBook-Pro:project manishagalla$ python Template_Matcher.py
Enter a sentence
Yesterday's news that fourth quarter GDP rose a healthy 3.5% was predicted by media and Beltway bear in the past four years
.
('Predictor:', set([u'media and Beltway bear', u'the past four years']))
('cause_subject:', u'fourth quarter GDP')
('Impact:', u'rose')
('cause_object:', u'  a  healthy  3.5 %')
(venv) Manishas-MacBook-Pro:project manishagalla$
```

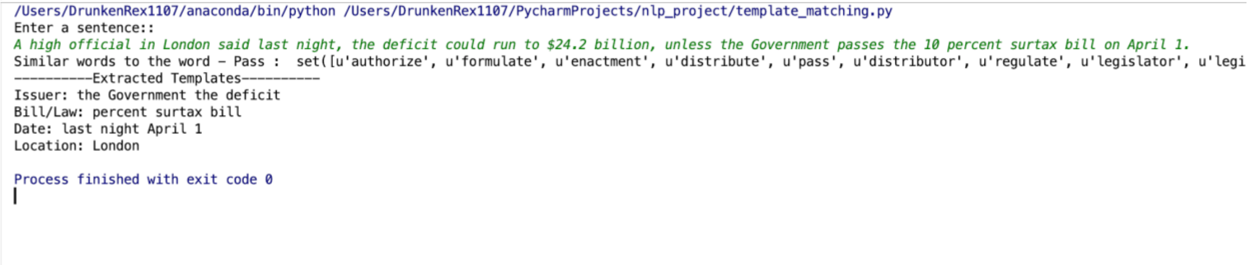
Pass: (Issuer, Bill/law, Date, Location)



```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
Earlier this year at the Capitol Hill, over the objections of securities firms, the Senate passed a bill expanding banks' securities powers.
Similar words to the word - Pass :  set([u'authorize', u'formulate', u'enactment', u'distribute', u'pass', u'distributor', u'regulate', u'legislator',
----------Extracted Templates----------
Issuer: Senate
Bill/Law: a bill banks securities powers
Date: Earlier this year
Location: the Capitol Hill

Process finished with exit code 0
```

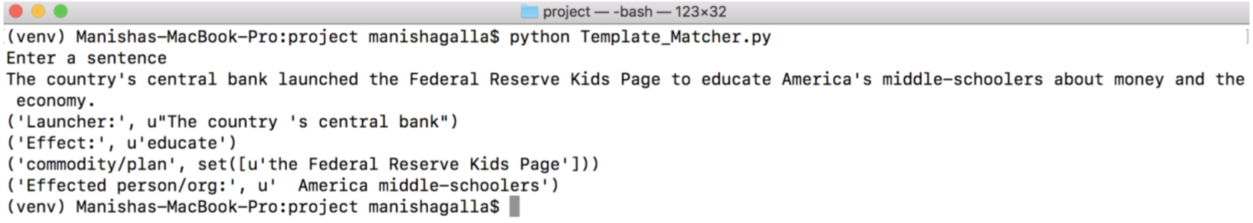


```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
The President has threatened to veto the Employment bill passed by the House and the Senate on Monday, December 10th at Washington D.C.
Similar words to the word - Pass :  set([u'authorize', u'formulate', u'enactment', u'distribute', u'pass', u'distributor', u'regulate', u'legislator',
----------Extracted Templates----------
Issuer: the House the Senate
Bill/Law: bill Employment President
Date: Monday December 10th
Location: Washington

Process finished with exit code 0
```



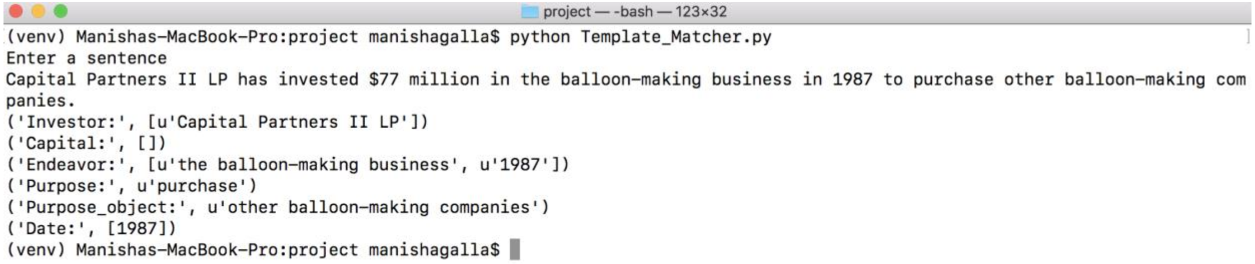
```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
A high official in London said last night, the deficit could run to $24.2 billion, unless the Government passes the 10 percent surtax bill on April 1.
Similar words to the word - Pass :  set([u'authorize', u'formulate', u'enactment', u'distribute', u'pass', u'distributor', u'regulate', u'legislator', u'legi
----------Extracted Templates----------
Issuer: the Government the deficit
Bill/Law: percent surtax bill
Date: last night April 1
Location: London

Process finished with exit code 0
```

Launch: (launcher_person/org,commodity/plan,effect,effected person/organization)



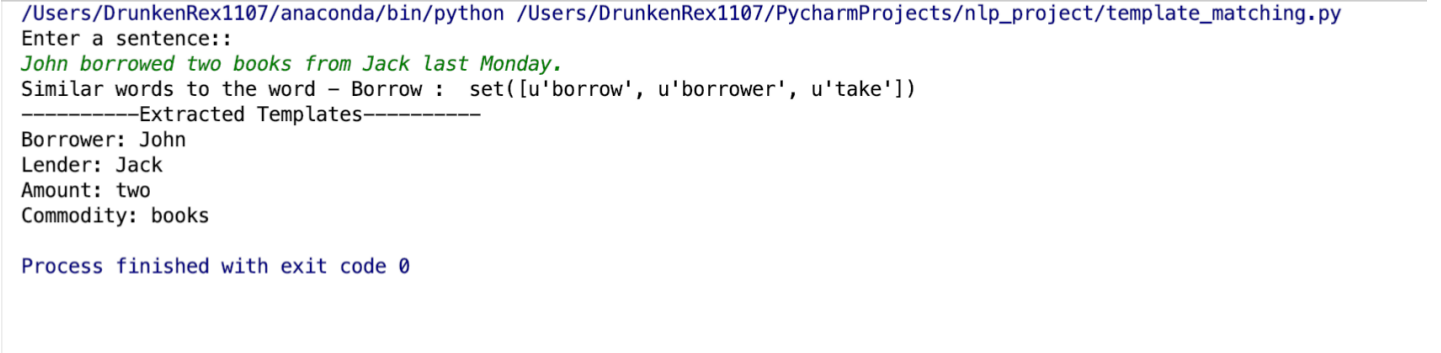
```
(venv) Manishas-MacBook-Pro:project manishagalla$ python Template_Matcher.py
Enter a sentence
The country's central bank launched the Federal Reserve Kids Page to educate America's middle-schoolers about money and the
 economy.
('Launcher:', u"The country 's central bank")
('Effect:', u'educate')
('commodity/plan', set([u'the Federal Reserve Kids Page']))
('Effected person/org:', u'  America middle-schoolers')
(venv) Manishas-MacBook-Pro:project manishagalla$
```

Invest: (Investor, capital, endeavor,Purpose,Date):

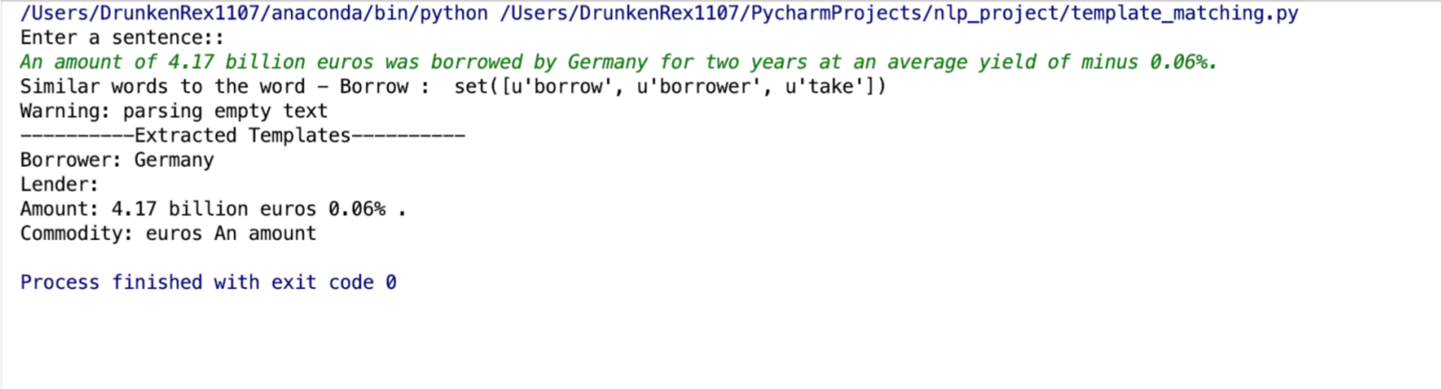


```
(venv) Manishas-MacBook-Pro:project manishagalla$ python Template_Matcher.py
Enter a sentence
Capital Partners II LP has invested $77 million in the balloon-making business in 1987 to purchase other balloon-making com
panies.
('Investor:', [u'Capital Partners II LP'])
('Capital:', [])
('Endeavor:', [u'the balloon-making business', u'1987'])
('Purpose:', u'purchase')
('Purpose_object:', u'other balloon-making companies')
('Date:', [1987])
(venv) Manishas-MacBook-Pro:project manishagalla$
```

Borrow: (Amount, Lender, Borrower, Commodity):

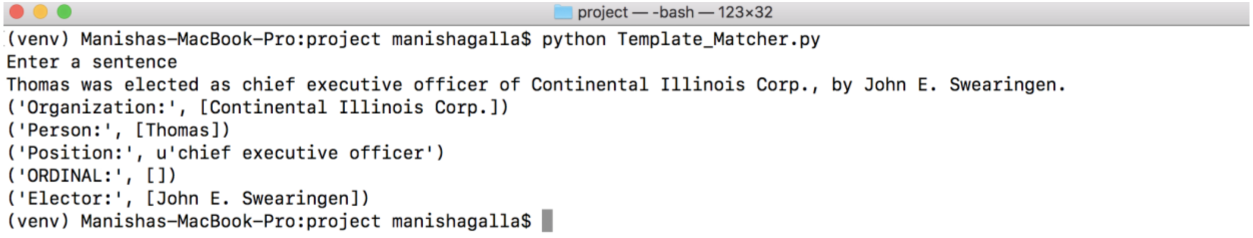


```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
John borrowed two books from Jack last Monday.
Similar words to the word - Borrow :  set([u'borrow', u'borrower', u'take'])
----------Extracted Templates----------
Borrower: John
Lender: Jack
Amount: two
Commodity: books

Process finished with exit code 0
```



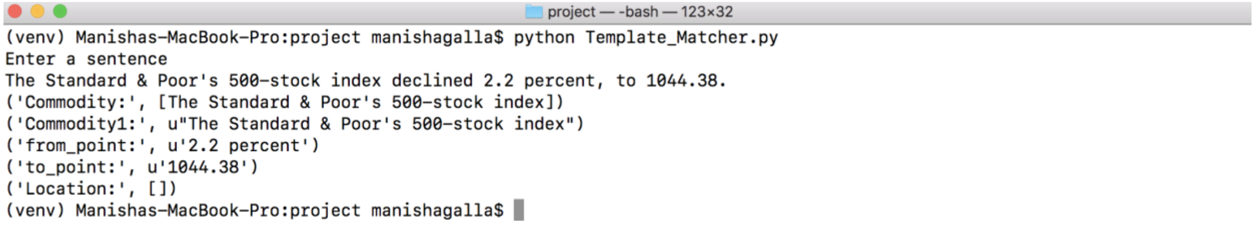
```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
An amount of 4.17 billion euros was borrowed by Germany for two years at an average yield of minus 0.06%.
Similar words to the word - Borrow :  set([u'borrow', u'borrower', u'take'])
Warning: parsing empty text
----------Extracted Templates----------
Borrower: Germany
Lender:
Amount: 4.17 billion euros 0.06% .
Commodity: euros An amount

Process finished with exit code 0
```

Elect: (person, position, organization,elector, ordinal):



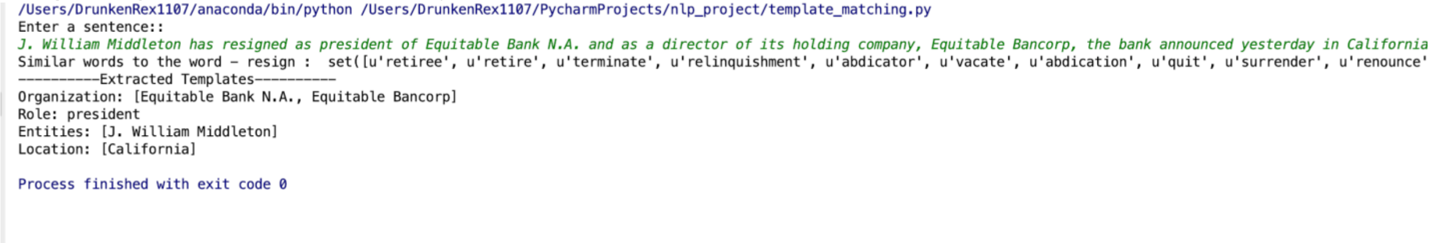
```
(venv) Manishas-MacBook-Pro:project manishagalla$ python Template_Matcher.py
Enter a sentence
Thomas was elected as chief executive officer of Continental Illinois Corp., by John E. Swearingen.
('Organization:', [Continental Illinois Corp.])
('Person:', [Thomas])
('Position:', u'chief executive officer')
('ORDINAL:', [])
('Elector:', [John E. Swearingen])
(venv) Manishas-MacBook-Pro:project manishagalla$
```

Decline: (commodity, from_point, to_point, location):

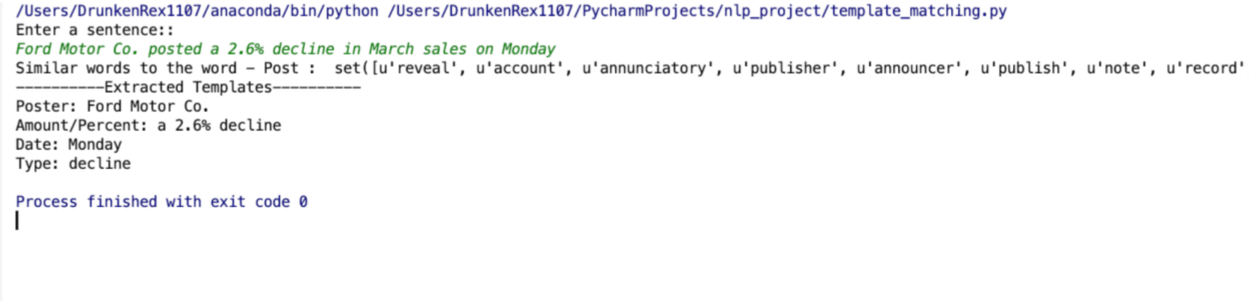


```
(venv) Manishas-MacBook-Pro:project manishagalla$ python Template_Matcher.py
Enter a sentence
The Standard & Poor's 500-stock index declined 2.2 percent, to 1044.38.
('Commodity:', [The Standard & Poor's 500-stock index])
('Commodity1:', u"The Standard & Poor's 500-stock index")
('from_point:', u'2.2 percent')
('to_point:', u'1044.38')
('Location:', [])
(venv) Manishas-MacBook-Pro:project manishagalla$
```

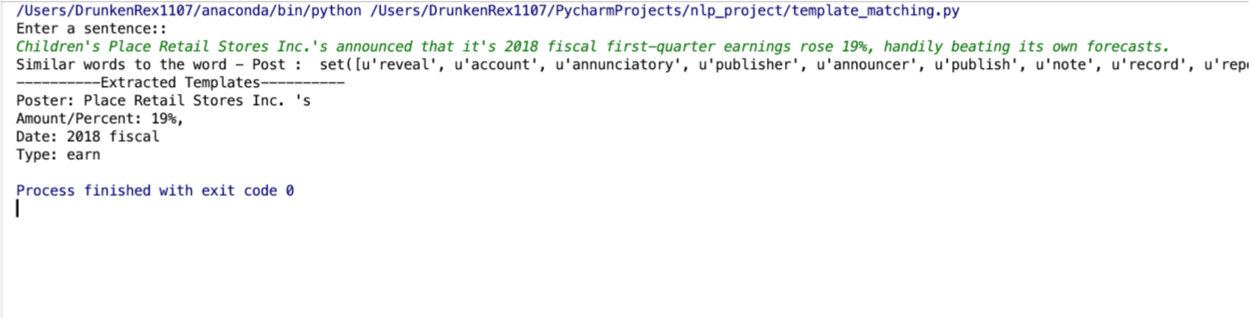
Resign: (Organization, Position, Entity, Location):



```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
J. William Middleton has resigned as president of Equitable Bank N.A. and as a director of its holding company, Equitable Bancorp, the bank announced yesterday in California
Similar words to the word - resign :  set([u'retiree', u'retire', u'terminate', u'relinquishment', u'abdicator', u'vacate', u'abdication', u'quit', u'surrender', u'renounce'
----------Extracted Templates----------
Organization: [Equitable Bank N.A., Equitable Bancorp]
Role: president
Entities: [J. William Middleton]
Location: [California]

Process finished with exit code 0
```



```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
Ford Motor Co. posted a 2.6% decline in March sales on Monday
Similar words to the word - Post :  set([u'reveal', u'account', u'annunciatory', u'publisher', u'announcer', u'publish', u'note', u'record'
----------Extracted Templates----------
Poster: Ford Motor Co.
Amount/Percent: a 2.6% decline
Date: Monday
Type: decline

Process finished with exit code 0
```



```
/Users/DrunkenRex1107/anaconda/bin/python /Users/DrunkenRex1107/PycharmProjects/nlp_project/template_matching.py
Enter a sentence::
Children's Place Retail Stores Inc.'s announced that it's 2018 fiscal first-quarter earnings rose 19%, handily beating its own forecasts.
Similar words to the word - Post :  set([u'reveal', u'account', u'annunciatory', u'publisher', u'announcer', u'publish', u'note', u'record', u'rep
----------Extracted Templates----------
Poster: Place Retail Stores Inc. 's
Amount/Percent: 19%,
Date: 2018 fiscal
Type: earn

Process finished with exit code 0
```

Post: (Amount, Type, Term/Date, Poster)

Problems encountered during Implementation:

- Identifying active and passive voice in sentences was difficult at first. We tried various ways to do it including extracting noun subjects, noun passive subjects, direct, indirect and passive objects. We also tried splitting the sentence before and after the corresponding template to identify the subject and predicate
- Chunking noun forms to get the head noun form from the Dependency tree
- Identifying Named entities in the document after/before chunking

Pending Issues:

- Complex multiple word entities extraction is difficult. Also currency symbol needs to be accompanied by the word after the value(For eg $10 dollars instead of just $10)
- Filled templates might also contain redundant information along with the correct extracted information.

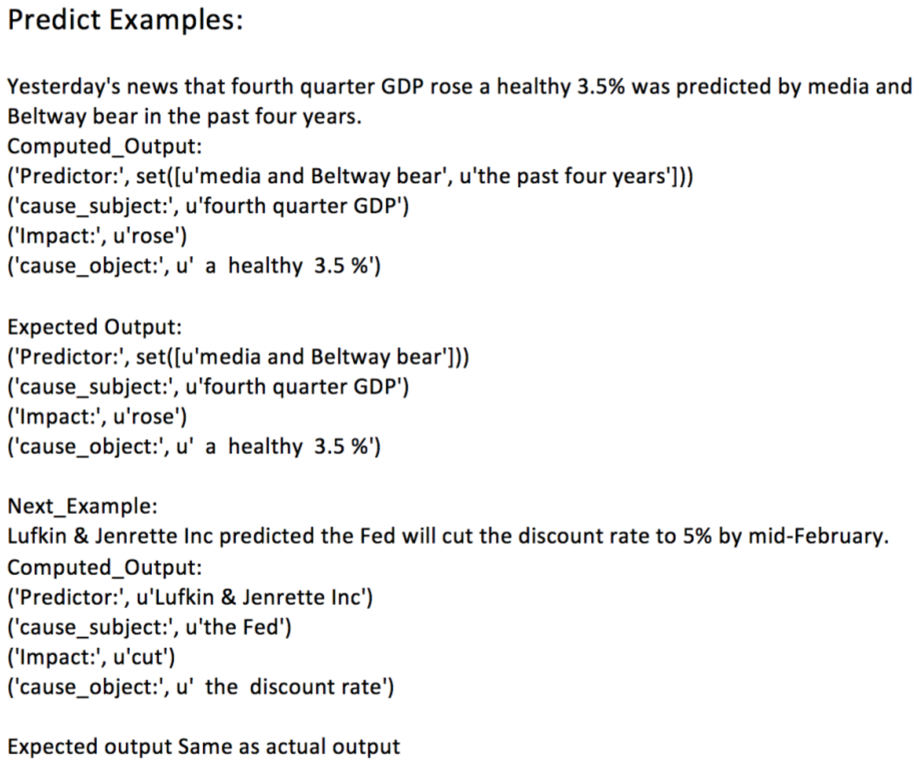
Potential Improvements:

- We can modify the Grammar used to parse and extract head noun chunks even better.
- Redundancies in extracted templates can be reduced by identifying and implementing better relation extraction algorithm.

Evaluation:

- The Evaluation is done for all the templates showing Computed and Expected Outputs. The Evaluation file is attached separately as it is huge.

- It looks as shown in the image:



Predict Examples:

Yesterday's news that fourth quarter GDP rose a healthy 3.5% was predicted by media and Beltway bear in the past four years.

Computed_Output:

```
('Predictor:', set([u'media and Beltway bear', u'the past four years']))
('cause_subject:', u'fourth quarter GDP')
('Impact:', u'rose')
('cause_object:', u'  a  healthy  3.5 %')
```

Expected Output:

```
('Predictor:', set([u'media and Beltway bear']))
('cause_subject:', u'fourth quarter GDP')
('Impact:', u'rose')
('cause_object:', u'  a  healthy  3.5 %')
```

Next_Example:

Lufkin & Jenrette Inc predicted the Fed will cut the discount rate to 5% by mid-February.

Computed_Output:

```
('Predictor:', u'Lufkin & Jenrette Inc')
('cause_subject:', u'the Fed')
('Impact:', u'cut')
('cause_object:', u'  the  discount rate')
```

Expected output Same as actual output