Approach and Methodologies:

1. **Text Preprocessing:**
    - Lowercasing: Uniformly convert all text to lowercase.
    - Punctuation Removal: Eliminate punctuation marks from the text.
    - Tokenization: Employ NLTK's word_tokenize to segment text into tokens.
    - Stopword Removal: Exclude common English stopwords.
2. **Building Inverted Index:**
    - For each preprocessed file, generate a list of unique terms.
    - Construct an inverted index associating each term with a list of documents where it appears.
    - Preserve the inverted index in a 'inverted_index.pkl' file using Pickle.
3. **Boolean Query Processing:**
    - Solicit user input for a boolean query and associated operations.
    - Preprocess the query akin to text preprocessing.
    - Execute boolean query processing utilizing the inverted index and designated operations (AND, OR, AND NOT, OR NOT).
    - Present the results, comprising the query string and retrieved documents.
4. **Building Positional Index:**
    - For each preprocessed file, craft a positional index mapping each term to its positions in the document.
    - Save the positional index in a 'positional_index.pkl' file using Pickle.
5. **Phrase Query Search:**
    - Collect user input for a phrase query.
    - Preprocess the query to acquire query tokens.
    - Conduct phrase query searches utilizing the positional index, verifying adjacent positions.
    - Exhibit the results, detailing the number of retrieved documents and their names.

Assumptions:

- NLTK is utilized for tokenization and stopwords removal.
- Stopwords are removed before index construction.

Results:

- **Inverted Index:**
  - The inverted index is successfully generated and stored in 'inverted_index.pkl'.
  - Boolean queries are processed, and outcomes are presented.
- **Positional Index:**
  - The positional index is effectively built and saved in 'positional_index.pkl'.
  - Phrase queries are processed leveraging the positional index, and outcomes are showcased.

Preprocessing:

```
File 1 :
Original string :
Loving these vintage springs on my vintage strat. They have a good tension and great stability. If you are floating your brid
ge and want the most out of your springs than these are the way to go.
Lower Case string :
loving these vintage springs on my vintage strat. they have a good tension and great stability. if you are floating your brid
ge and want the most out of your springs than these are the way to go.
String after removing the punctuations :
loving these vintage springs on my vintage strat they have a good tension and great stability if you are floating your bridge
and want the most out of your springs than these are the way to go
Tokens :
['loving', 'these', 'vintage', 'springs', 'on', 'my', 'vintage', 'strat', 'they', 'have', 'a', 'good', 'tension', 'and', 'gre
at', 'stability', 'if', 'you', 'are', 'floating', 'your', 'bridge', 'and', 'want', 'the', 'most', 'out', 'of', 'your', 'sprin
gs', 'than', 'these', 'are', 'the', 'way', 'to', 'go']
Tokens after removing the stopwords :
['loving', 'vintage', 'springs', 'vintage', 'strat', 'good', 'tension', 'great', 'stability', 'floating', 'bridge', 'want',
'springs', 'way', 'go']
Final tokens after removing the white space tokens :
['loving', 'vintage', 'springs', 'vintage', 'strat', 'good', 'tension', 'great', 'stability', 'floating', 'bridge', 'want',
```


Inverted Index:

```
In [36]: inverted_index
```

```
Out[36]: {'loving': ['file1.txt', 'file254.txt', 'file391.txt', 'file723.txt'],
          'vintage': ['file1.txt',
           'file150.txt',
           'file197.txt',
           'file278.txt',
           'file422.txt',
           'file439.txt',
           'file494.txt',
           'file51.txt',
           'file597.txt',
           'file638.txt',
           'file674.txt',
           'file725.txt',
           'file737.txt',
           'file827.txt',
           'file847.txt',
           'file895.txt',
           'file907.txt',
           'file936.txt'],
```

Boolean Query:

```
How many queries do you want to enter?2
Query: Way to Go?
Operations: And
Query: It's time for Great music.
Operations: AND NOT,OR
```

Result:

```
Query 1: way AND go
Number of documents retrieved for query 1: 10
Names of the documents retrieved for query 1: ['file139.txt', 'file144.txt', 'file890.txt', 'file563.txt', 'file79.txt', 'file8
35.txt', 'file413.txt', 'file764.txt', 'file623.txt', 'file1.txt']
Query 2: time AND NOT great OR music
Number of documents retrieved for query 2: 97
Names of the documents retrieved for query 2: ['file879.txt', 'file992.txt', 'file443.txt', 'file647.txt', 'file753.txt', 'file
525.txt', 'file927.txt', 'file245.txt', 'file342.txt', 'file886.txt', 'file959.txt', 'file154.txt', 'file926.txt', 'file974.tx
t', 'file347.txt', 'file638.txt', 'file163.txt', 'file835.txt', 'file143.txt', 'file5.txt', 'file22.txt', 'file129.txt', 'file7
84.txt', 'file896.txt', 'file308.txt', 'file290.txt', 'file47.txt', 'file935.txt', 'file332.txt', 'file373.txt', 'file847.txt',
'file502.txt', 'file910.txt', 'file735.txt', 'file844.txt', 'file462.txt', 'file830.txt', 'file45.txt', 'file25.txt', 'file677.
txt', 'file633.txt', 'file324.txt', 'file265.txt', 'file381.txt', 'file631.txt', 'file380.txt', 'file307.txt', 'file41.txt', 'f
ile880.txt', 'file74.txt', 'file864.txt', 'file145.txt', 'file937.txt', 'file706.txt', 'file684.txt', 'file468.txt', 'file691.t
xt', 'file365.txt', 'file406.txt', 'file818.txt', 'file160.txt', 'file203.txt', 'file666.txt', 'file778.txt', 'file859.txt', 'f
ile821.txt', 'file652.txt', 'file662.txt', 'file68.txt', 'file17.txt', 'file845.txt', 'file862.txt', 'file711.txt', 'file359.tx
t', 'file554.txt', 'file360.txt', 'file51.txt', 'file29.txt', 'file216.txt', 'file790.txt', 'file3.txt', 'file322.txt', 'file74
2.txt', 'file171.txt', 'file750.txt', 'file248.txt', 'file66.txt', 'file667.txt', 'file95.txt', 'file866.txt', 'file912.txt',
'file251.txt', 'file325.txt', 'file157.txt', 'file11.txt', 'file49.txt', 'file718.txt']
```

Positional Index:

```
In [45]: positional_index

Out[45]: {'loving': {'file1.txt': [1],
                     'file254.txt': [17],
                     'file391.txt': [3],
                     'file723.txt': [7]},
          'vintage': {'file1.txt': [2, 4],
                      'file150.txt': [11],
                      'file197.txt': [8, 43],
                      'file278.txt': [5],
                      'file422.txt': [9],
                      'file439.txt': [4, 33],
                      'file494.txt': [11],
                      'file51.txt': [28],
                      'file597.txt': [29],
                      'file638.txt': [63],
                      'file674.txt': [28],
                      'file725.txt': [18],
                      'file737.txt': [10],
                      'file827.txt': [34],
                      'file847.txt': [12, 53],
```

Phrase Query:

```
How many queries do you want to enter?2
Query: It's time for great music.
Query: Way to Go
```

Result:

```
Number of documents retrieved for query 1 using positional index: 0
Names of the documents retrieved for query 1 using positional index: []
Number of documents retrieved for query 2 using positional index: 4
Names of the documents retrieved for query 2 using positional index: ['file144.txt', 'file563.txt', 'file623.txt', 'file1.txt']
```