In [10]:
```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer, WordNetLemmatizer
import contractions
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
import re
```

In [4]:
```
# Download necessary NLTK data files
nltk.download('punkt')
nltk.download('stopwords')
nltk.download('wordnet')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\satch\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\satch\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\satch\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

Out[4]:
```
True
```

In [8]:
```
text = "Hey there! □ I can't believe it's already 2024. Did you see John's new blog post'
```

In [9]:
```
# Function to preprocess text
def preprocess_text(text):
    # 1. Expand Contractions
    text = contractions.fix(text)

    # 2. Remove URLs and Emails
    text = re.sub(r'http\S+|www\S+|https\S+|mailto:\S+', '', text, flags=re.MULTILINE)
    text = re.sub(r'\S+@\S+', '', text)

    # 3. Remove special characters and emojis
    text = re.sub(r'[^a-zA-Z\s]', '', text)

    # 4. Tokenization
    words = word_tokenize(text)

    # 5. Lowercasing
    words = [word.lower() for word in words]

    # 6. Removing Punctuation
    words = [word for word in words if word.isalnum()]

    # 7. Removing Stop Words
    stop_words = set(stopwords.words('english'))
    filtered_words = [word for word in words if word not in stop_words]

    # 8. Stemming
    stemmer = PorterStemmer()
    stemmed_words = [stemmer.stem(word) for word in filtered_words]

    # 9. Lemmatization
    lemmatizer = WordNetLemmatizer()
    lemmatized_words = [lemmatizer.lemmatize(word) for word in filtered_words]

    return ' '.join(lemmatized_words)

# Preprocess the text
```

```
preprocessed_text = preprocess_text(text)
print("Preprocessed Text:", preprocessed_text)
```

```
Preprocessed Text: hey believe already see john new blog postcheck also email mentioned
something stemming lemmatizationinteresting stuff wayi attending ai conference nyc next
month excited ai let u catch soon cheer john
```

In [11]:

```
# 10. Vectorization
# Using CountVectorizer
count_vectorizer = CountVectorizer()
count_vector = count_vectorizer.fit_transform([preprocessed_text])
print("Count Vectorizer - Feature Names:", count_vectorizer.get_feature_names_out())
print("Count Vectorizer - Vectorized Text:", count_vector.toarray())

# Using TfidfVectorizer
tfidf_vectorizer = TfidfVectorizer()
tfidf_vector = tfidf_vectorizer.fit_transform([preprocessed_text])
print("TF-IDF Vectorizer - Feature Names:", tfidf_vectorizer.get_feature_names_out())
print("TF-IDF Vectorizer - Vectorized Text:", tfidf_vector.toarray())
```

```
Count Vectorizer - Feature Names: ['ai' 'already' 'also' 'attending' 'believe' 'blog' 'c
atch' 'cheer'
 'conference' 'email' 'excited' 'hey' 'john' 'lemmatizationinteresting'
 'let' 'mentioned' 'month' 'new' 'next' 'nyc' 'postcheck' 'see'
 'something' 'soon' 'stemming' 'stuff' 'wayi']
Count Vectorizer - Vectorized Text: [[2 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1]]
TF-IDF Vectorizer - Feature Names: ['ai' 'already' 'also' 'attending' 'believe' 'blog'
'catch' 'cheer'
 'conference' 'email' 'excited' 'hey' 'john' 'lemmatizationinteresting'
 'let' 'mentioned' 'month' 'new' 'next' 'nyc' 'postcheck' 'see'
 'something' 'soon' 'stemming' 'stuff' 'wayi']
TF-IDF Vectorizer - Vectorized Text: [[0.34815531 0.17407766 0.17407766 0.17407766 0.174
07766 0.17407766
  0.17407766 0.17407766 0.17407766 0.17407766 0.17407766 0.17407766
  0.34815531 0.17407766 0.17407766 0.17407766 0.17407766 0.17407766
  0.17407766 0.17407766 0.17407766 0.17407766 0.17407766 0.17407766
  0.17407766 0.17407766 0.17407766]]
```