In [3]:
```
A = {1,2,3,4,6}
B = {1,2,5,8,9}
C = A.intersection(B)
D = A.union(B)
print('AnB = ', C)
print('AUB = ', D)
print('J(A,B) = ',float(len(C))/float(len(D)))
```

```
AnB =  {1, 2}
AUB =  {1, 2, 3, 4, 5, 6, 8, 9}
J(A,B) =  0.25
```

In [7]:
```
def jaccard_similarity(set1,set2):
    intersection = len(set1.intersection(set2))
    union = len(set1.union(set2))
    return intersection/union
set_a = {"Language","for","Computer","NLP","Science"}
set_b = {"NLP","for","Language","Data","ML","AI"}
similarity = jaccard_similarity(set_a,set_b)
print("Jaccard Similarity:",similarity)
```

```
Jaccard Similarity: 0.375
```

In [9]:
```
import numpy as np

from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine_similarity
```

In [37]:
```
# Step 1: Define predefined chatbot responses

responses = [

    "You can return an item within 7 days of purchase.",

    "Our return policy allows you to return items that are unopened and in their or

    "We offer free shipping on orders over $50.",

    "To track your order, you can visit the 'Order Tracking' page and enter your or

    "Our customer support team is available from 9 AM to 6 PM, Monday through Frida

]
```

In [39]:
```
# Step 2: Sample user input (this would come from a user in a real chatbot)

user_input = "How can I track my order?"
```

In [41]:
```
# Step 3: Preprocess the text using TF-IDF Vectorization

vectorizer = TfidfVectorizer(stop_words='english')  # Removing stopwords like 'the'

all_texts = responses + [user_input]  # Combine responses with the user input for v
```

In [43]:
```
# Step 4: Convert the text to TF-IDF vectors

tfidf_matrix = vectorizer.fit_transform(all_texts)
```

In [45]:
```
# Step 5: Compute cosine similarity between the user input and all responses

user_vector = tfidf_matrix[-1]  # The user input is the last text in the matrix

response_vectors = tfidf_matrix[:-1]  # All predefined responses



# Calculate cosine similarities

cosine_similarities = cosine_similarity(user_vector, response_vectors)
```

In [47]:
```
# Step 6: Find the most similar response

most_similar_idx = np.argmax(cosine_similarities)  # Find the index of the most sim
```

In [49]:
```
# Step 7: Display the most relevant response

print(f"User Query: {user_input}")

print(f"Most relevant response: {responses[most_similar_idx]}")
```

```
User Query: How can I track my order?
Most relevant response: To track your order, you can visit the 'Order Tracking' page
and enter your order number.
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: