In [1]:
```
#All necessary imports
import pandas as pd
import nltk
```

In [2]:
```
df = pd.read_excel('chatbot_ds(3)(2).xlsx', sheet_name='Sheet1')
```

In [3]:
```
df.head()
```

Out[3]:

	Intention	question	answer
0	Investment Solutions	I wish to invest on financial markets	You have the choice between three management o...
1	Investment Solutions	I want to maintain control over my decisions w...	You have the choice between three management o...
2	Investment Solutions	I want to access directly trading room experts	You have the choice between three management o...
3	Investment Solutions	I am a professional client, and I look for a d...	You have the choice between three management o...
4	Investment Solutions	I look for investment advices	You have the choice between three management o...

In [4]:
```
print(df)
```

```
73   Corporate solutions          What are your solutions for entrepr
eneurs
74   Corporate solutions  I am considering external growth opportunit
y f...
75   Corporate solutions  I am considering a change in my company cap
ita...
76   Corporate solutions                          I want to sale my c
ompany
77   Corporate solutions              I want to buy commercial real
estate
78   Corporate solutions             I want to sell commercial real
estate
79   Corporate solutions                     I want to take part in
an IPO
80   Corporate solutions                 I want to take part in bond
issue
81   Corporate solutions             I want to take part in an equity
issue

                                                  answer
```

In [5]:
```
df.describe()
```

Out[5]:

	Intention	question	answer
count	82	82	82
unique	10	82	10
top	How are you	What's up	I am fine thankyou . Is there something I can ...
freq	14	1	14

In [6]:
```
print(df.columns)
```

```
Index(['Intention', 'question', 'answer'], dtype='object')
```

Preprocessing of data set

In [7]:
```
questions_answers = ['question','answer'] #selecting only columns impor
```

In [8]:
```
qa = df[questions_answers]
```

In [9]:
```
qa.head()
```

Out[9]:

	question	answer
0	I wish to invest on financial markets	You have the choice between three management o...
1	I want to maintain control over my decisions w...	You have the choice between three management o...
2	I want to access directly trading room experts	You have the choice between three management o...
3	I am a professional client, and I look for a d...	You have the choice between three management o...
4	I look for investment advices	You have the choice between three management o...

In [10]:
```
pd.isnull(qa).describe() #no null data found
```

Out[10]:

	question	answer
count	82	82
unique	1	1
top	False	False
freq	82	82

In [11]:
```
qa.duplicated('question')
pd.options.display.max_colwidth = 100
```

In [12]:
```
qa.loc[qa['question']== "I wish to invest on financial markets" ]
```

Out[12]:

	question	answer
0	I wish to invest on financial markets	You have the choice between three management offer we an provide according to your investor's pr...

In [ ]:
```
#we have two options,either only take one answer and ignore remaining ar

#deleting all duplicate question and takind only first answer correspond

qa[qa.duplicated(['question'])]

qa.drop(qa[qa.duplicated(['question'])].index,inplace= True)
```

In [14]:
```
len(qa)
```

Out[14]:
```
82
```

In [15]:
```
qa.reset_index(inplace=True)
```

In [16]:
```
print(qa.columns)
```

```
Index(['index', 'question', 'answer'], dtype='object')
```

In [17]:
```
qa['index']= qa.index
```

In [18]:
```
df.head()
```

Out[18]:

	Intention	question	answer
0	Investment Solutions	I wish to invest on financial markets	You have the choice between three management offer we an provide according to your investor's pr...
1	Investment Solutions	I want to maintain control over my decisions while benefiting from personalized advices	You have the choice between three management offer we an provide according to your investor's pr...
2	Investment Solutions	I want to access directly trading room experts	You have the choice between three management offer we an provide according to your investor's pr...
3	Investment Solutions	I am a professional client, and I look for a direct access to a market professional for the nego...	You have the choice between three management offer we an provide according to your investor's pr...
4	Investment Solutions	I look for investment advices	You have the choice between three management offer we an provide according to your investor's pr...

In [19]:
```
qa.head()
```

Out[19]:

	index	question	answer
0	0	I wish to invest on financial markets	You have the choice between three management offer we an provide according to your investor's pr...
1	1	I want to maintain control over my decisions while benefiting from personalized advices	You have the choice between three management offer we an provide according to your investor's pr...
2	2	I want to access directly trading room experts	You have the choice between three management offer we an provide according to your investor's pr...
3	3	I am a professional client, and I look for a direct access to a market professional for the nego...	You have the choice between three management offer we an provide according to your investor's pr...
4	4	I look for investment advices	You have the choice between three management offer we an provide according to your investor's pr...

We have now done some preprocessing of dataset.

Movind on to processing input (question entered to out chatbot)

In [20]:
```
#Input the Question to chatbot

asked_question = input("Hi there! How can I help you ? ")
print(asked_question)
```

```
Hi there! How can I help you ? how can I contact you
how can I contact you
```

In [21]:
```
#for empty input return default
if(asked_question==''):
    print("Sorry I don't have a respone. I am constantly learning")
```

In [22]:
```
#for smileys - cause nltk removes all such special characters , we write

smileys = [':)','(:','^^','^_^',';)','(;','XD',':p',';p']

if asked_question.replace(" ","") in smileys:
    print(':)')
```

In [24]:
```
from nltk.tokenize import word_tokenize, sent_tokenize
```

In [ ]:
```
nltk.download('punkt')
sents = sent_tokenize(asked_question)
print(sents)
```

In [26]:
```
words =[word_tokenize(sent) for sent in sents]
print(words)
```

```
[['how', 'can', 'I', 'contact', 'you']]
```

In [27]:
```
from nltk.corpus import stopwords
```

In [28]:
```
from string import punctuation
```

In [ ]:
```
nltk.download('stopwords')
customStopWords=set(stopwords.words('english')+list(punctuation))# fails
```

In [30]:
```
words_not_stopWords = [word for word in word_tokenize(asked_question) if
print(words_not_stopWords)
```

```
['I', 'contact']
```

In [ ]:
```
nltk.download('averaged_perceptron_tagger')
nltk.pos_tag(words_not_stopWords)
```

In [32]:
```
tagged = nltk.pos_tag(words_not_stopWords)
filtered_words = list()
filtered_words = [item[0] for item in tagged if (item[1][0] == 'N')] # (
print(filtered_words)
```

```
[]
```

In [33]:
```
#since it failed to recognize contact, adding explicitly

if 'contact' in words_not_stopWords:
    filtered_words.append('contact')
```

In [34]:
```
for current_word in filtered_words:
    #check(current_word)
    print(current_word)


print(filtered_words)
```

```
contact
['contact']
```

In [35]:
```
#check separate for hi
```

In [36]:
```
def check(item, questions):
    count = 0
    ques_list = []
    for ques in questions:
        if item.lower() in str(ques).lower():
            count+=1
            ques_list.append(ques)

    print(count)
    return count, ques_list
```

In [37]:
```
def find_question_list(filtered_words, questions):
    max = 0
    selected_word = ''
    freq_word_questions = []
    for current_word in filtered_words:
        val,ques_list = check(current_word,questions)
        if (val > max):
            max = val
            selected_word=current_word
            freq_word_questions = ques_list
            print(selected_word)
    print(selected_word)
    print(freq_word_questions)
    return selected_word,freq_word_questions
```

In [38]:
```
selected_word,freq_word_questions = find_question_list(filtered_words, c
```

```
1
contact
contact
['how can I contact you']
```

In [39]:
```
if(selected_word.lower() == 'hi'):
    print('Hello, nice to meet you. How may I help you ?')
```

In [40]:
```
def find_occurrence_count(word,qa):
    for ques in qa.question:
        if word in str(ques):
            return 1
```

In [41]:
```
def word_in_question(filtered_words):
    count = 0
    for word in filtered_words:
        count+= find_occurrence_count(word,qa)
    return count
```

In [42]:
```
words_occurence_count = word_in_question(filtered_words)
print(words_occurence_count)
```

```
1
```

In [43]:
```
def find_next_iteration_questions(filtered_words,selected_word,freq_word
    if((len(freq_word_questions) <= 1) or (len(filtered_words) == 1)):
        return selected_word,freq_word_questions
    elif(len(freq_word_questions) > 1):
        filtered_words.remove(selected_word)
        next_selected_word,freq_word_questions_next_iter = find_question
        return find_next_iteration_questions(filtered_words,next_selecte
    else:
        return selected_word,freq_word_questions
```

In [44]:
```
def get_final_question_list(filtered_words,selected_word,freq_word_quest
    if(words_occurence_count == 1 or words_occurence_count == 0):
        return freq_word_questions
    else:
        final_selected_word,final_question = find_next_iteration_questio
        return final_question
```

In [45]:
```
chatbot_selected_question = get_final_question_list(filtered_words,selec
```

In [46]:
```
print(chatbot_selected_question)
```

```
['how can I contact you']
```

In [47]:
```
if(len(chatbot_selected_question) == 0):
    print("Sorry I don't have a respone. I am constantly learning")
```

In [48]:
```
qa.loc[qa['question']== chatbot_selected_question[0]].answer
```

Out[48]:
```
52    Here is how we can get in touch : by calling you back when you w
ill be available, meeting us at ...
Name: answer, dtype: object
```

In [49]:
```
#now we can also display a list of other suggestions, close to user's qu

print(freq_word_questions)
```

```
['how can I contact you']
```

In [50]:
```
freq_word_questions.remove(chatbot_selected_question[0]) #removing our s

suggestion_count = 1
if(len(freq_word_questions) > 0):
    for ques in freq_word_questions:
        print(str(suggestion_count) + " " + str(ques)+"\n")
        suggestion_count+=1
```

In [ ]: