In [1]:

```
import numpy as np
import pandas as pd
from wordcloud import WordCloud
import nltk
from nltk.corpus import stopwords
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer,TfidfVectorizer
from sklearn.metrics import accuracy_score
```

In [2]:

```
Data=pd.read_csv("C:/Users/nehal/Downloads/spam.csv")
```

In [3]:

```
Data.head()
```

Out[3]:

	v1	v2	Unnamed: 2	Unnamed: 3	Unnamed: 4
0	ham	Go until jurong point, crazy.. Available only ...	NaN	NaN	NaN
1	ham	Ok lar... Joking wif u oni...	NaN	NaN	NaN
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...	NaN	NaN	NaN
3	ham	U dun say so early hor... U c already then say...	NaN	NaN	NaN
4	ham	Nah I don't think he goes to usf, he lives aro...	NaN	NaN	NaN

In [4]:

```
Data=Data[['v1','v2']]
```

In [5]:

```
Data.head()
```

Out[5]:

	v1	v2
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...
3	ham	U dun say so early hor... U c already then say...
4	ham	Nah I don't think he goes to usf, he lives aro...

In [6]:

```
Data=Data.rename(columns={'v1' : 'Label',"v2" : 'Text'})
```

In [7]:

```
Data['Label'].value_counts(ascending=True)
```

Out[7]:

```
spam     747
ham     4825
Name: Label, dtype: int64
```

In [8]:

```
nltk.download("punkt")
import warnings
warnings.filterwarnings('ignore')
```

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

In [9]:

```
ham_words = ''
spam_words = ''
```

In [10]:

```
for val in Data[Data['Label'] == 'spam'].Text:
    Text = val.lower()
    tokens = nltk.word_tokenize(Text)
    for words in tokens:
        spam_words = spam_words + words + ' '

for val in Data[Data['Label'] == 'ham'].Text:
    Text = Text.lower()
    tokens = nltk.word_tokenize(Text)
    for words in tokens:
        ham_words = ham_words + words + ' '
```

In [11]:

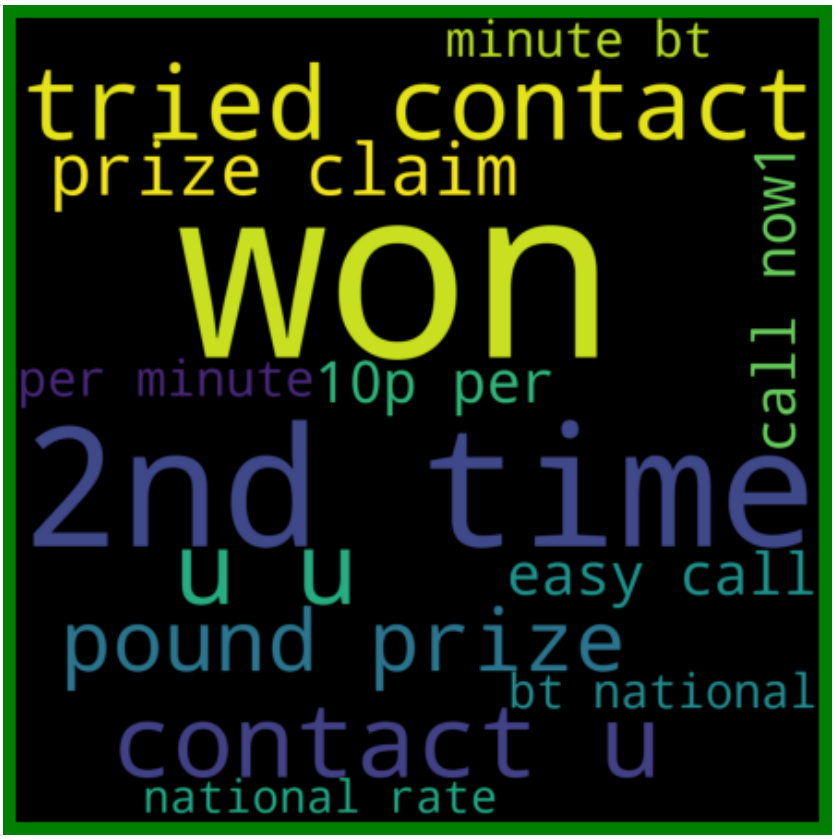
```
spam_wordcloud = WordCloud(width=1000, height=1000).generate(spam_words)
ham_wordcloud = WordCloud(width=1000, height=1000).generate(ham_words)
```

In [12]:

```
plt.figure( figsize=(10,8), facecolor='w')
plt.imshow(spam_wordcloud)
plt.axis("off")
plt.show()
```

In [13]:

```
plt.figure( figsize=(10,8), facecolor='g')
plt.imshow(ham_wordcloud)
plt.axis("off")
plt.show()
```



In [14]:

```
Data = Data.replace(['ham','spam'],[0, 1])
Data.head(10)
```

Out[14]:

	Label	Text
0	0	Go until jurong point, crazy.. Available only ...
1	0	Ok lar... Joking wif u oni...
2	1	Free entry in 2 a wkly comp to win FA Cup fina...
3	0	U dun say so early hor... U c already then say...
4	0	Nah I don't think he goes to usf, he lives aro...
5	1	FreeMsg Hey there darling it's been 3 week's n...
6	0	Even my brother is not like to speak with me. ...
7	0	As per your request 'Melle Melle (Oru Minnamin...
8	1	WINNER!! As a valued network customer you have...
9	1	Had your mobile 11 months or more? U R entitle...

In [15]:

```
nltk.download('stopwords')
import string
def text_process(Text):

    Text = Text.translate(str.maketrans('', '', string.punctuation))
    Text = [word for word in Text.split() if word.lower() not in stopwords.words('englis
h')]

    return " ".join(Text)
```

```
[nltk_data] Downloading package stopwords to
```

```
[nltk_data]     C:\Users\nehal\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

In [16]:

```
Data['Text'] = Data['Text'].apply(text_process)
Data.head()
```

Out[16]:

	Label	Text
0	0	Go jurong point crazy Available bugis n great ...
1	0	Ok lar Joking wif u oni
2	1	Free entry 2 wkly comp win FA Cup final tkts 2...
3	0	U dun say early hor U c already say
4	0	Nah dont think goes usf lives around though

In [17]:

```
Text = pd.DataFrame(Data['Text'])
Label = pd.DataFrame(Data['Label'])
```

In [18]:

```
vectorizer = TfidfVectorizer()
vectors = vectorizer.fit_transform(Data['Text'])
vectors.shape
```

Out[18]:

```
(5572, 9333)
```

In [19]:

```
features = vectors
```

In [20]:

```
X_train, X_test, y_train, y_test = train_test_split(features, Data['Label'], test_size=0
.2, random_state=111)
```

In [21]:

```
from sklearn.naive_bayes import MultinomialNB
from sklearn.tree import DecisionTreeClassifier
```

In [22]:

```
mnb = MultinomialNB(alpha=0.2)
dtc = DecisionTreeClassifier(min_samples_split=7, random_state=111)
```

In [23]:

```
clfs = { 'NB': mnb, 'DT': dtc}
```

In [24]:

```
def train(clf, features, targets):
    clf.fit(features, targets)

def predict(clf, features):
    return (clf.predict(features))
```

In [25]:

```
pred_scores_word_vectors = []
```

```
for k,v in clfs.items():
    train(v, X_train, y_train)
    pred = predict(v, X_test)
    pred_scores_word_vectors.append((k, [accuracy_score(y_test , pred)]))
```

In [26]:

```
pred_scores_word_vectors
```

Out[26]:

```
[('NB', [0.9856502242152466]), ('DT', [0.9650224215246637])]
```

In [27]:

```
def SpamOrNot(x):
    if x == 1:
        print ("Message is SPAM")
    elif x==0:
        print ("Message is NOT Spam")
```

In [28]:

```
newtext = ["prize"]
integers = vectorizer.transform(newtext)
```

In [29]:

```
x = mnb.predict(integers)
SpamOrNot(x)
```

```
Message is SPAM
```

In [ ]:

```

```