In [190]:

```
import urllib.request
import requests
from bs4 import BeautifulSoup
import re
import ast
url1 = "https://www.naukri.com/top-skill-jobs"
# We scrapped the data from naukri
```

In [3]:

```
import os
os.chdir("C:\\Users\\tamil\\OneDrive\\Documents\\Python Directory")
folder = "jp project/"
import seaborn as sns
import re
```

In [311]:

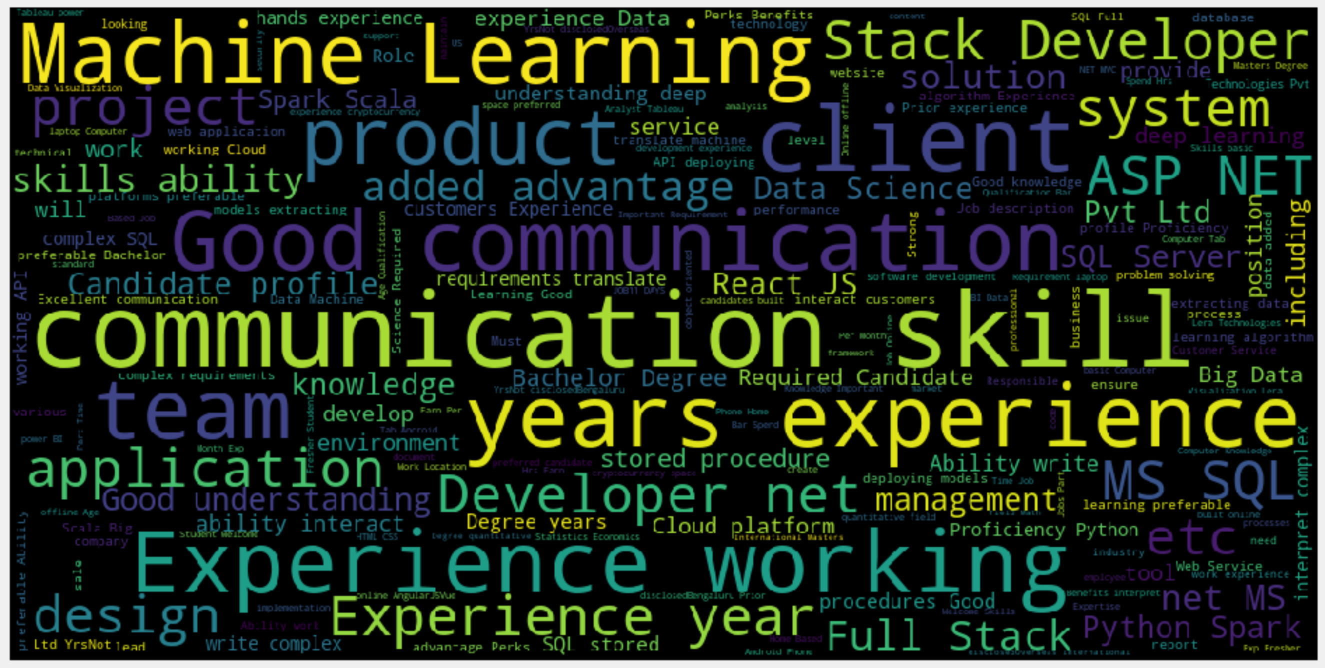
```
#Loading the scrapped data from the machine
f = open(folder+"naukri.txt")
f=f.read()

f_splitted=f.split("Job Description")
```

Demanding Job Cluster

In [313]:

```
import matplotlib.pyplot as plt
from wordcloud import WordCloud
wordcloud = WordCloud(width = 1000, height = 500).generate(" ".join(f_splitted))
wordcloud
plt.figure(figsize=(15,8))
plt.imshow(wordcloud)
plt.axis("off")
#plt.savefig("your_file_name"+".png", bbox_inches='tight')
plt.show()
plt.close()
```



It's not clear enough. Let's try to get more precise information from the data.

It's not clear enough. Let's try to get more precise information from the data.

Cleaning corpus

In [314]:

```
word_feature = []
for feature in f_splitted:
    feature = feature.replace("(\\\)"," ");
    feature = feature.replace("."," ");
    feature = feature.replace("\\\\"," ");
    feature = feature.replace("(\\\)"," ");
    feature = feature.replace("(\\\';\\\')"," ");
    feature = feature.replace("(\\n )"," ");
    feature = feature.replace("+","");
    feature = feature.replace("&","");
    feature = feature.replace("jobs","");
    feature = feature.replace("-","");
    feature = feature.replace("in","");
    feature = feature.replace("or","");
    feature = feature.replace("of","");
    feature = feature.replace("is","");
    feature = feature.replace("to","");
    feature = feature.replace("and","");
    feature = feature.replace("with","");
#      ["understg","Cidate","wkg","Ltd"]
    feature = feature.replace("understg","");
    feature = feature.replace("experience","");
    feature = feature.replace("Experience","");

    feature = feature.replace("Jobs","");
    feature = feature.replace("YrsNot","");
    feature = feature.replace("DAYS","");

    feature = feature.replace("Good","");
    feature = feature.replace("o","");


    feature = feature.replace("Cidate","");
    feature = feature.replace("wkg","");
    feature = feature.replace("Ltd","");
    feature = feature.replace('',"");

    feature = feature.replace('Job',"");
    feature = feature.replace('Description',"");
    feature = feature.replace('years',"");
    feature = feature.replace('knwledge',"");
    feature = feature.replace('Knwledge',"");
    feature = feature.replace('Shuld',"");
    feature = feature.replace('frm',"");
    feature = feature.replace('Skills',"");
    feature = feature.replace('added',"");
    feature = feature.replace('advantage',"");
    feature = feature.replace('client',"");
    feature = feature.replace('slutin',"");
    feature = feature.replace('Bachel',"");
    feature = feature.replace('Degree',"");
    feature = feature.replace('prject',"");
    feature = feature.replace('etc',"");

    feature = feature.replace('',"");

# ["knwledge","Knwledge","Shuld","frm","Skills"]
# ["added","advantage","client","slutin","Bachel","Degree","prject","etc"]

#      for i in listq:
#          feature = feature.replace(i,"")


    word_feature.append(feature)
# word_feature
```

In [315]:

```
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize

#Joining all words
all_words = ""
for words in word_feature:
    all_words +=words + " "

#Removing symbols inside corpus
all_words = re.sub('[^A-Za-z0-9]+', ' ', all_words)

#Removing numbers
all_words_2 = ""
for i in all_words:
    if i.isdigit() is False:
        all_words_2 += i

#Removing stopwords
stop_words = set(stopwords.words('english'))
word_tokens = word_tokenize(all_words_2)

filtered_sentence = [w for w in word_tokens if not w in stop_words]

filtered_sentence = []
for w in word_tokens:
    if w not in stop_words:
        filtered_sentence.append(w)

print(len(word_tokens))
print(len(filtered_sentence))
```

```
303272
274597
```

In [316]:

```
#Removing two letter words
filtered_sentence_2 = []

for word in filtered_sentence:
    if len(word) > 2:
        filtered_sentence_2.append(word)
# print(filtered_sentence_2)
```

In [317]:

```
#User-defined function to COUNT the words
def get_word_freq(list_data):
    freq = {}
    for word in list_data:
        if word not in freq:
            freq[word] = 1
        else:
            freq[word] +=1
    return freq

word_counts = get_word_freq(filtered_sentence_2)
```

In [318]:

```
#Sorting the dictionary
sorted_dict = {k: v for k, v in sorted(word_counts.items(), key=lambda item: item[1], reverse=True)
}
# print(sorted_dict)
```

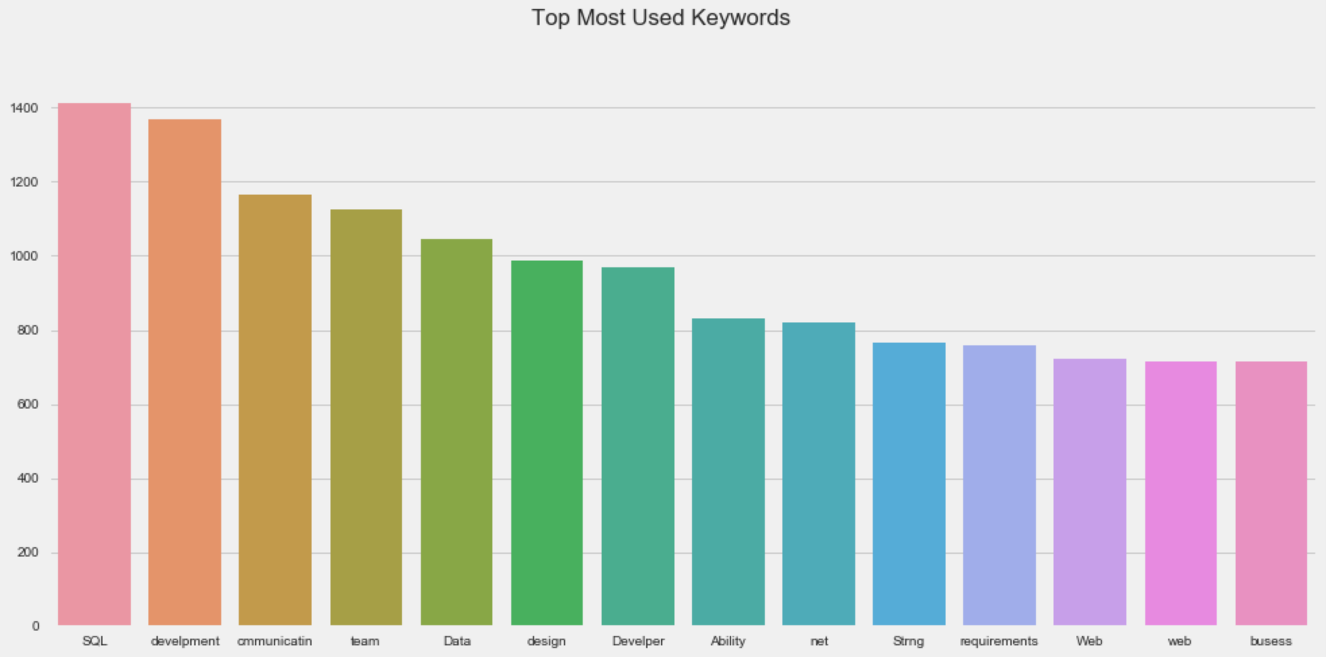
In [319]:

```
import itertools
#Slicing top most 15 keywords from the dictionary
keyword_dict = dict(itertools.islice(sorted_dict.items(), 15))
# print(keyword_dict)
```

Most Used Keywords

In [324]:

```
keys = list(keyword_dict.keys())
values = list(keyword_dict.values())
plt.style.use("fivethirtyeight")
plt.figure(figsize=(15,7))
sns.barplot(keys[1:],values[1:])
# plt.xticks(rotaion = 90);
plt.suptitle("Top Most Used Keywords")
# plt.ylabel("Frequency")
plt.savefig(folder+"keywords"+".png", bbox_inches='tight')
plt.show()
```



SQL is the top most keyword used.

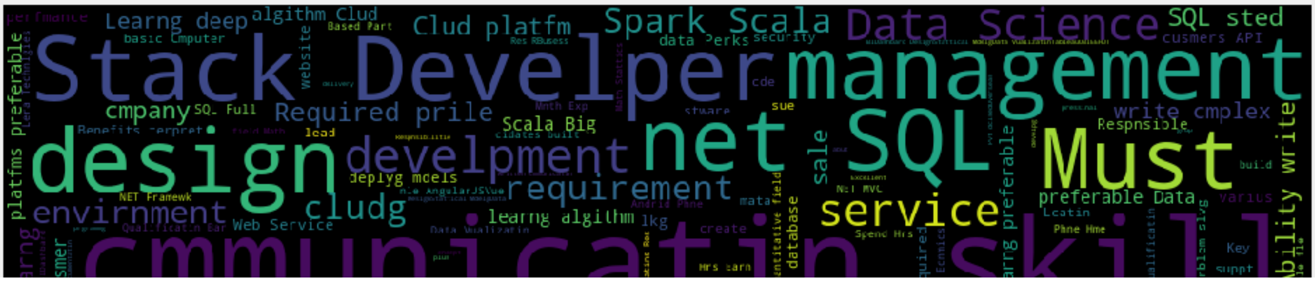
Communication Skills is hardly needed!

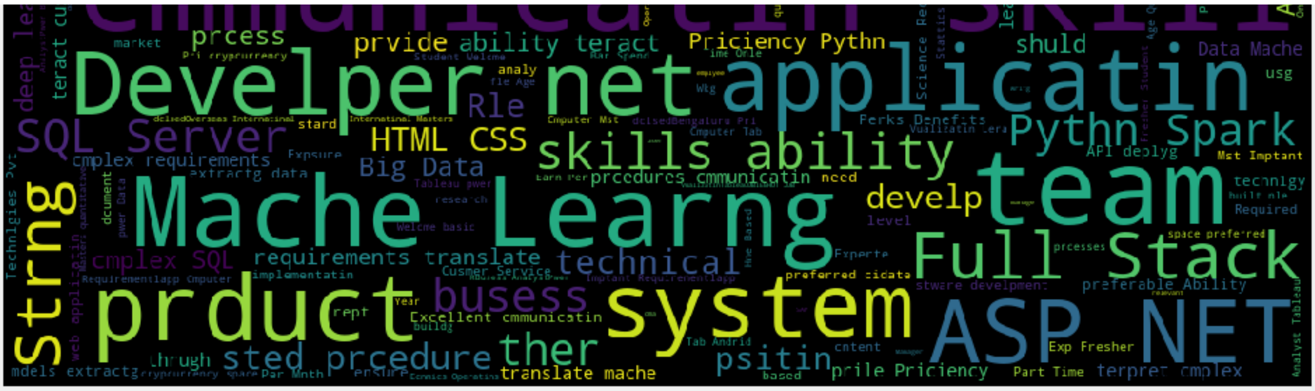
WEB DEVELOPMENT is another most used keyword in the portal.

Most Demanding Skills

In [323]:

```
wordcloud = WordCloud(width = 1000, height = 500).generate(" ".join(filtered_sentence_2))
wordcloud
plt.figure(figsize=(15,8))
plt.imshow(wordcloud)
plt.axis("off")
plt.savefig(folder+"demanding_skills"+".png", bbox_inches='tight')
plt.show()
plt.close()
```

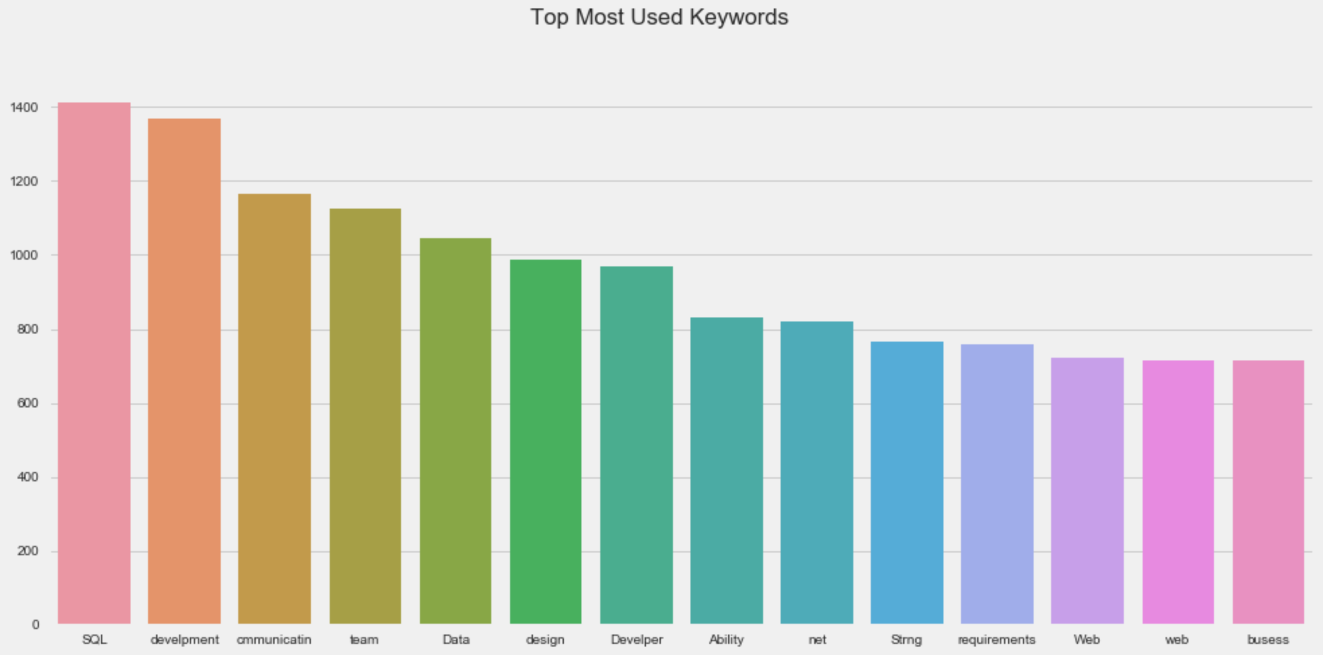


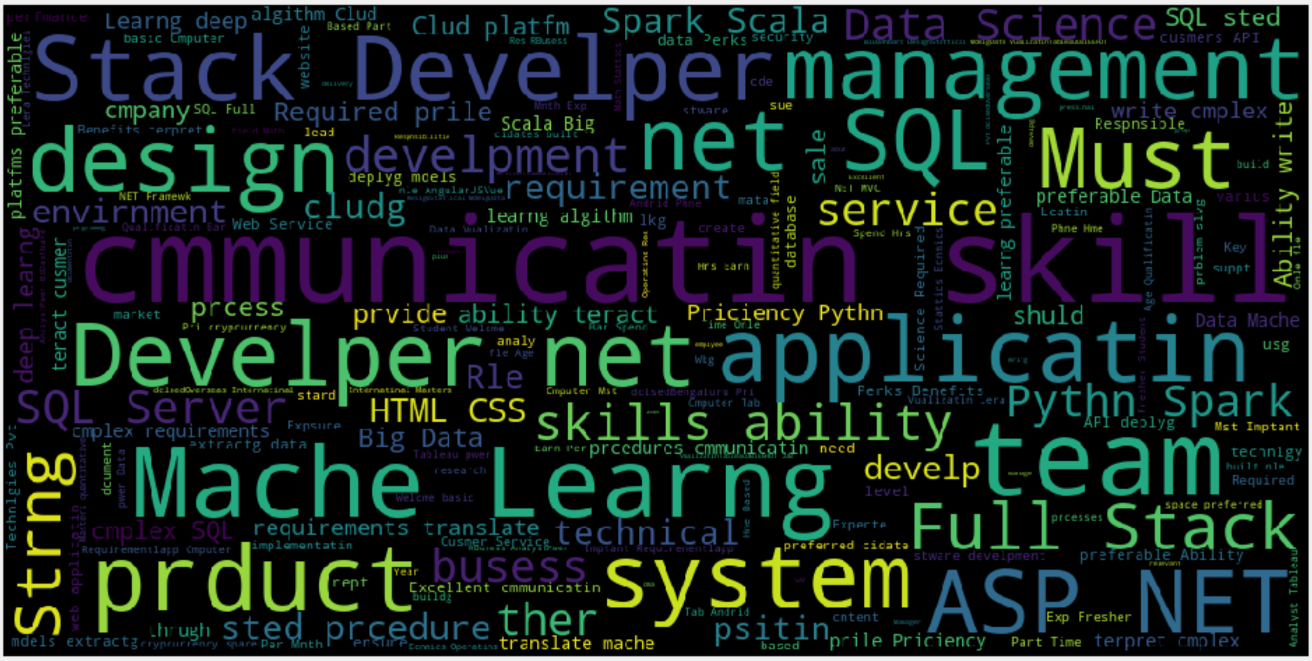


"o" missed in every words!! (Tokenization error)

:)

FINAL PLOTS





In [ ]: