In [50]:
```
import json
import os
import pandas as  pd
import spacy
import seaborn as sns
import string
from tqdm import tqdm
from textblob import TextBlob
from nltk.corpus import stopwords
import nltk
from nltk.stem import WordNetLemmatizer
from nltk import word_tokenize
import re
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import FunctionTransformer
from sklearn.base import BaseEstimator, TransformerMixin
from sklearn.pipeline import FeatureUnion
from sklearn.feature_extraction import DictVectorizer
import swifter
tqdm.pandas()
```

In [51]:
```
df = pd.read_excel("dataset.xlsx")
```

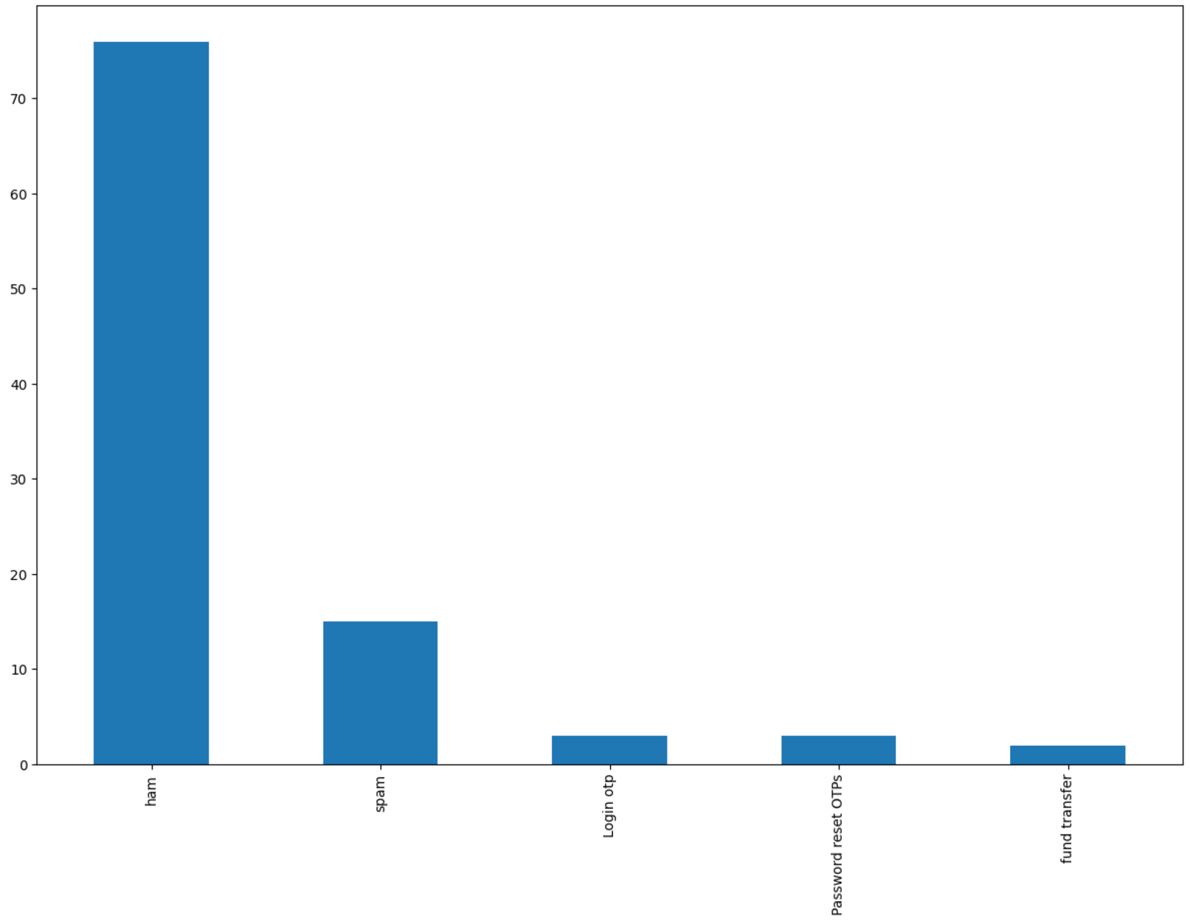
In [52]:
```
df.head()
```

Out[52]:

	categories	mssg
0	fund transfer	965439 is the OTP for transaction of INR 87650...
1	fund transfer	5465 is the OTP for transaction of INR 450 at ...
2	Login otp	Dear customer , use this One Time Password 854...
3	Login otp	Dear customer , use this One Time Password 458...
4	Login otp	Dear customer , use this One Time Password 453...

In [53]:
```
df['categories'].value_counts().plot( kind='bar', figsize=(15,10))
```

Out[53]:
```
<AxesSubplot:>
```



In [54]:
```
df.columns
```

Out[54]:
```
Index(['categories', 'mssg'], dtype='object')
```

In [55]:
```
df.describe()
```

Out[55]:

	categories	mssg
count	99	99
unique	5	99
top	ham	965439 is the OTP for transaction of INR 87650...
freq	76	1

In [56]:
```
df.isna().sum()
```

Out[56]:
```
categories    0
mssg          0
dtype: int64
```

In [57]:
```
df['categories'].unique()
```

Out[57]:
```
array(['fund transfer', 'Login otp', 'Password reset OTPs', 'spam', 'ham'],
      dtype=object)
```

In [58]:
```
stop_words_ = set(stopwords.words('english'))
wn = WordNetLemmatizer()
my_sw = ['make', 'amp',  'news','new' ,'time', 'u','s', 'photos',  'get', 'say']

def black_txt(token):
    return  token not in stop_words_ and token not in list(string.punctuation)  an
```

```
def clean_txt(text):
    clean_text = []
    clean_text2 = []
    text = re.sub("'", "",text)
    text=re.sub("(\\d|\\W)+"," ",text)
    clean_text = [ wn.lemmatize(word, pos="v") for word in word_tokenize(text.lowe
    clean_text2 = [word for word in clean_text if black_txt(word)]
    return " ".join(clean_text2)
```

In [59]:
```
def subj_txt(text):
    return  TextBlob(text).sentiment[1]

def polarity_txt(text):
    return TextBlob(text).sentiment[0]

def len_text(text):
    if len(text.split())>0:
         return len(set(clean_txt(text).split()))/ len(text.split())
    else:
         return 0
```

In [60]:
```
df['text'] = df['mssg']

df['text'] = df['text'].swifter.apply(clean_txt)
df['polarity'] = df['text'].swifter.apply(polarity_txt)
df['subjectivity'] = df['text'].swifter.apply(subj_txt)
df['len'] = df['text'].swifter.apply(lambda x: len(x))
```

```
Pandas Apply:   0%|          | 0/99 [00:00<?, ?it/s]
Pandas Apply:   0%|          | 0/99 [00:00<?, ?it/s]
Pandas Apply:   0%|          | 0/99 [00:00<?, ?it/s]
Pandas Apply:   0%|          | 0/99 [00:00<?, ?it/s]
```

In [61]:
```
X = df[['text', 'polarity', 'subjectivity','len']]
y =df['categories']
encoder = LabelEncoder()
y = encoder.fit_transform(y)

x_train, x_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=
v = dict(zip(list(y), df['categories'].to_list()))
```

In [62]:
```
text_clf = Pipeline([
...     ('vect', CountVectorizer(analyzer="word", stop_words="english")),
...     ('tfidf', TfidfTransformer(use_idf=True)),
...     ('clf', MultinomialNB(alpha=.01)),
... ])
```

In [63]:
```
text_clf.fit(x_train['text'].to_list(), list(y_train))
```

Out[63]:

```
▸ Pipeline
  ▸ CountVectorizer
  ▸ TfidfTransformer
  ▸ MultinomialNB
```

In [64]:
```
import numpy as np
```

```
In [65]: X_TEST = x_test['text'].to_list()
         Y_TEST = list(y_test)

In [66]: predicted = text_clf.predict(X_TEST)
         text_clf.score(X_TEST,Y_TEST)

Out[66]: 0.85

In [67]: np.mean(predicted == Y_TEST)

Out[67]: 0.85

In [71]: docs_new = ['789654 is otp for your transaction of rs 8980']
         predicted = text_clf.predict(docs_new)
         v[predicted[0]]

Out[71]: 'fund transfer'

In [72]: docs_new = ['hi there how are you!']
         predicted = text_clf.predict(docs_new)
         v[predicted[0]]

Out[72]: 'ham'

In [73]: docs_new = ['231456 is your otp to login your facebook account']
         predicted = text_clf.predict(docs_new)
         v[predicted[0]]

Out[73]: 'Login otp'

In [74]: docs_new = ['Ke bani crorepati ar jite lakho 500000 tk ke prize']
         predicted = text_clf.predict(docs_new)
         v[predicted[0]]

Out[74]: 'spam'

In [75]: docs_new = ['765439 is One time passaword to reset your linkedin account']
         predicted = text_clf.predict(docs_new)
         v[predicted[0]]

Out[75]: 'Password reset OTPs'

In [ ]:
```