In [1]:

```
import pandas as pd
import numpy as np
```

In [2]:

```
fsr = pd.read_csv('facial_similarity_reports.csv')
dr = pd.read_csv('doc_reports.csv')
```

In [3]:

```
fsr.head()
fsr.columns
```

Out[3]:

```
Index(['Unnamed: 0', 'user_id', 'result', 'face_comparison_result',
       'created_at', 'facial_image_integrity_result',
       'visual_authenticity_result', 'properties', 'attempt_id'],
      dtype='object')
```

In [4]:

```
dr.head()
```

Out[4]:

	Unnamed: 0	user_id	result	visual_authenticity_result	image_inte
0	0	ab23fae164e34af0a1ad1423ce9fd9f0	consider	consider	
1	1	15a84e8951254011b47412fa4e8f65b8	clear	clear	
2	2	ffb82fda52b041e4b9af9cb4ef298c85	clear	clear	
3	3	bd4a8b3e3601427e88aa1d9eab9f4290	clear	clear	
4	4	f52ad1c7e69543a9940c3e7f8ed28a39	clear	clear	

In [5]:

```
dr['created_at'] = pd.to_datetime(dr['created_at']).dt.date
```

In [6]:

```
#dr['created_at']
```

In [7]:

```
cnt_clear = 0
cnt_attempt = 0
clr_list = list()
atp_list = list()
date_group_att = dr.groupby(('created_at'),as_index=True).size()
```

In [8]:

```
date_group_clear = dr[dr['result']=='clear'].groupby(('created_at'),as_index=Tru
```

In [9]:

```
data = date_group_clear/date_group_att
```

In [10]:

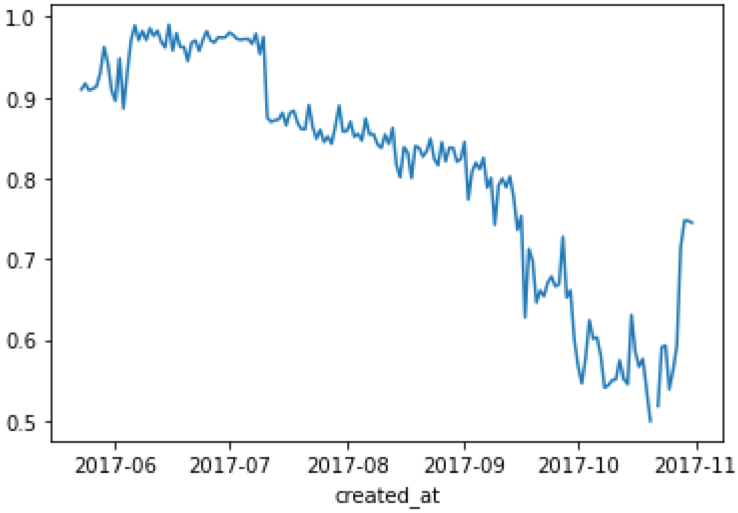
```
%matplotlib inline
```

In [11]:

```
dr_date=dr.created_at.unique()
```

In [12]:

```
import matplotlib.pyplot as plt
ax=data.plot()
#ax.set(xlim=(pd.Timestamp('2017-10-18'), pd.Timestamp('2017-10-23')))
#Data missing in above range
```



In [13]:

```
#date_group_varclear = dr[dr['visual_authenticity_result']=='clear'].groupby(('c
#data_var1 = date_group_varclear/date_group_att
#ax1 = data_var1.plot()
```

In [14]:

```
#date_group_sdclear = dr[dr['supported_document_result']=='clear'].groupby(('cre
#data_var2 = date_group_sdclear
#ax2 = data_var2.plot()
```

In [15]:

```
#date_group_fdclear = dr[dr['face_detection_result']!='clear'].groupby(('created
#data_var3 = date_group_fdclear/date_group_att
#ax3 = data_var3.plot()
```

In [16]:

```
#date_group_iqclear = dr[dr['image_quality_result']=='clear'].groupby(('created_
#data_var4 = date_group_iqclear/date_group_att
#ax4 = data_var4.plot()
```

In [17]:

```
#date_group_sbclear = dr[dr['sub_result']!='clear'].groupby(('created_at'),as_in
#data_var5 = date_group_sbclear
#ax5 = data_var5.plot()
```

In [18]:

```
#date_group_prclear = dr[dr['police_record_result']!='clear'].groupby(('created_
#data_var6 = date_group_sbclear-date_group_prclear
#ax6 = data_var6.plot()
```

In [19]:

```
#date_group_dcrclear = dr[dr['data_consistency_result']=='clear'].groupby(('crea
#data_var7 = date_group_dcrclear/date_group_att
#ax7 = data_var7.plot()
```

In [20]:

```
fsr['created_at'] = pd.to_datetime(fsr['created_at']).dt.date# Changing the date
```

In [21]:

```
#counting the number of attempts in a particular day fuction used .size()
date_gfacial_clear = fsr[fsr['result']=='clear'].groupby(('created_at'),as_index
```

In [22]:

```
date_gfacial_att = fsr.groupby(('created_at'),as_index=True).size()
```

In [23]:

```
data_fac = date_gfacial_clear/date_gfacial_att# Pass rate of the Visual assesmen
```

In [24]:

```
dd=pd.merge(dr,fsr,on='attempt_id',how='outer')#the merge was on attempt as the
# 2 attemps, merging on users will increase the count of the table hence uniques
```

In [25]:

```
#dd.count()
```

In [26]:

```
datal = dd.groupby(('created_at_x'),as_index=True).size()
data_all_clear = dd.loc[(dd.result_x=='clear') & (dd.result_y=='clear')]
```

In [27]:

```
dclear = data_all_clear.groupby('created_at_x',as_index=True).size()
```

In [28]:

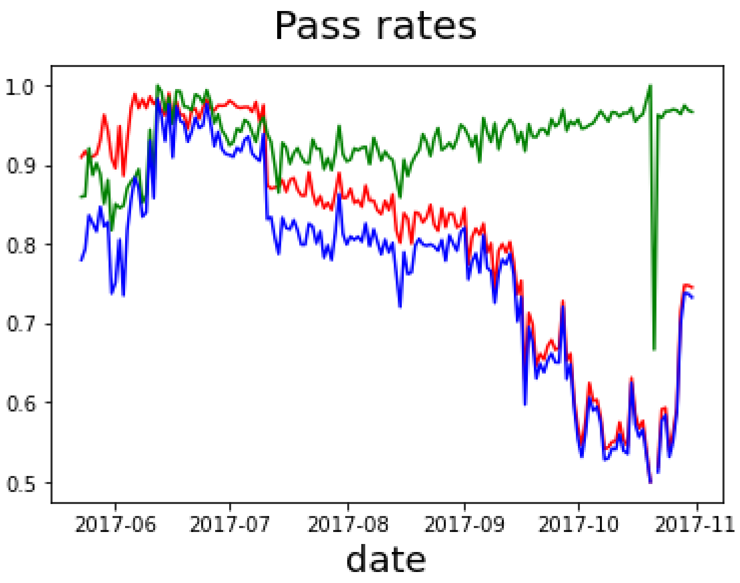
```
dataclear = dclear/datal

#dataclear.rename(columns={'created_date_x':'Date','':'Overall PR'},inplace=True
```

In [29]:

```
fig = plt.figure()
fig.suptitle('Pass rates', fontsize=20)


data.plot(color='r')#Document verification pass rate
data_fac.plot(color='g')#Visual verification pass rate
dataclear.plot(color='b')#Overall pass rate



plt.xlabel('date', fontsize=18)
```

Out[29]:

```
Text(0.5, 0, 'date')
```



In [30]:

```
conda install nbconvert
```

```
Collecting package metadata (current_repodata.json): done
Solving environment: done

# All requested packages already installed.


Note: you may need to restart the kernel to use updated packages.
```

In [ ]: