Import Libraries

In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Loading the Data Set

In [5]:
```
data_raw = pd.read_csv('../dataset/train_data.csv', sep='\t')
data = data_raw.copy()
pd.options.display.max_columns = 150

data.drop(data.columns[50:107], axis=1, inplace=True)
data.drop(data.columns[51:], axis=1, inplace=True)

print('Number of participants: ', len(data))
data.head()
```

```
Number of participants:  1015341
```

Out[5]:

	EXT1	EXT2	EXT3	EXT4	EXT5	EXT6	EXT7	EXT8	EXT9	EXT10	EST1	EST2	EST3	EST
0	4.0	1.0	5.0	2.0	5.0	1.0	5.0	2.0	4.0	1.0	1.0	4.0	4.0	2.
1	3.0	5.0	3.0	4.0	3.0	3.0	2.0	5.0	1.0	5.0	2.0	3.0	4.0	1.
2	2.0	3.0	4.0	4.0	3.0	2.0	1.0	3.0	2.0	5.0	4.0	4.0	4.0	2.
3	2.0	2.0	2.0	3.0	4.0	2.0	2.0	4.0	1.0	4.0	3.0	3.0	3.0	2.
4	3.0	3.0	3.0	3.0	5.0	3.0	3.0	5.0	3.0	4.0	1.0	5.0	5.0	3.

Cleaning the Data Set

In [6]:
```
print('Is there any missing value? \n ', data.isnull().values.any())
print('How many missing values? \n ', data.isnull().values.sum())
data.dropna(inplace=True)
print('Number of participants after eliminating missing values? \n ', len(data
```

```
Is there any missing value?
  True
How many missing values?
  89227
Number of participants after eliminating missing values?
  1013481
```

Groups and Questions

In [7]:

```
ext_questions = {'EXT1' : 'I am the life of the party',
                 'EXT2' : 'I dont talk a lot',
                 'EXT3' : 'I feel comfortable around people',
                 'EXT4' : 'I keep in the background',
                 'EXT5' : 'I start conversations',
                 'EXT6' : 'I have little to say',
                 'EXT7' : 'I talk to a lot of different people at parties',
                 'EXT8' : 'I dont like to draw attention to myself',
                 'EXT9' : 'I dont mind being the center of attention',
                 'EXT10': 'I am quiet around strangers'}

est_questions = {'EST1' : 'I get stressed out easily',
                 'EST2' : 'I am relaxed most of the time',
                 'EST3' : 'I worry about things',
                 'EST4' : 'I seldom feel blue',
                 'EST5' : 'I am easily disturbed',
                 'EST6' : 'I get upset easily',
                 'EST7' : 'I change my mood a lot',
                 'EST8' : 'I have frequent mood swings',
                 'EST9' : 'I get irritated easily',
                 'EST10': 'I often feel blue'}

agr_questions = {'AGR1' : 'I feel little concern for others',
                 'AGR2' : 'I am interested in people',
                 'AGR3' : 'I insult people',
                 'AGR4' : 'I sympathize with others feelings',
                 'AGR5' : 'I am not interested in other peoples problems',
                 'AGR6' : 'I have a soft heart',
                 'AGR7' : 'I am not really interested in others',
                 'AGR8' : 'I take time out for others',
                 'AGR9' : 'I feel others emotions',
                 'AGR10': 'I make people feel at ease'}

csn_questions = {'CSN1' : 'I am always prepared',
                 'CSN2' : 'I leave my belongings around',
                 'CSN3' : 'I pay attention to details',
                 'CSN4' : 'I make a mess of things',
                 'CSN5' : 'I get chores done right away',
                 'CSN6' : 'I often forget to put things back in their proper p
                 'CSN7' : 'I like order',
                 'CSN8' : 'I shirk my duties',
                 'CSN9' : 'I follow a schedule',
                 'CSN10' : 'I am exacting in my work'}

opn_questions = {'OPN1' : 'I have a rich vocabulary',
                 'OPN2' : 'I have difficulty understanding abstract ideas',
                 'OPN3' : 'I have a vivid imagination',
                 'OPN4' : 'I am not interested in abstract ideas',
                 'OPN5' : 'I have excellent ideas',
                 'OPN6' : 'I do not have a good imagination',
                 'OPN7' : 'I am quick to understand things',
                 'OPN8' : 'I use difficult words',
                 'OPN9' : 'I spend time reflecting on things',
```

```
'OPN10': 'I am full of ideas'}
```

Group Names and Columns

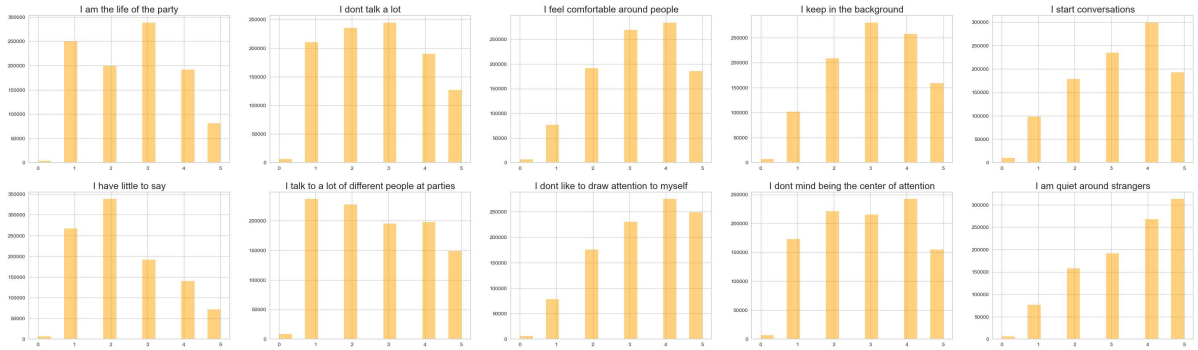
In [8]:
```
EXT = [column for column in data if column.startswith('EXT')]
EST = [column for column in data if column.startswith('EST')]
AGR = [column for column in data if column.startswith('AGR')]
CSN = [column for column in data if column.startswith('CSN')]
OPN = [column for column in data if column.startswith('OPN')]
```

Visualize the questions and answers distribution data

In [9]:
```
def vis_questions(groupname, questions, color):
    plt.figure(figsize=(40,60))
    for i in range(1, 11):
        plt.subplot(10,5,i)
        plt.hist(data[groupname[i-1]], bins=14, color= color, alpha=.5)
        plt.title(questions[groupname[i-1]], fontsize=18)
```

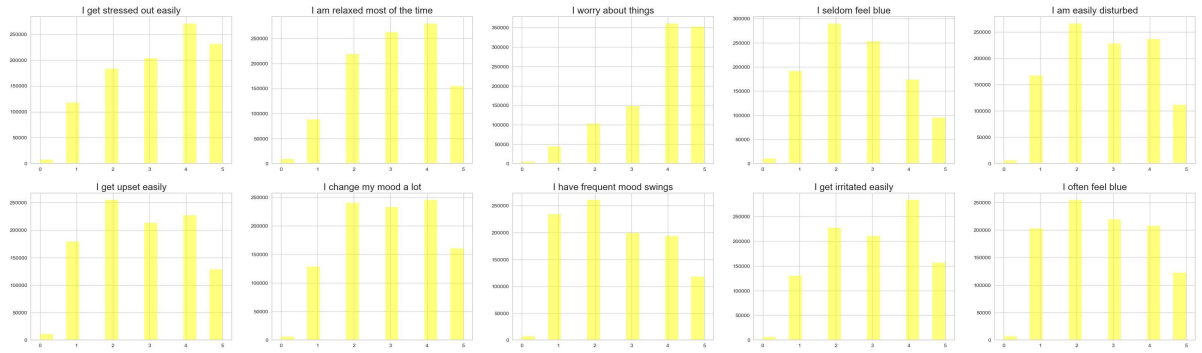
Q&A Distribution related to Extroversion Personality

In [10]:
```
vis_questions(EXT, ext_questions, 'orange')
```

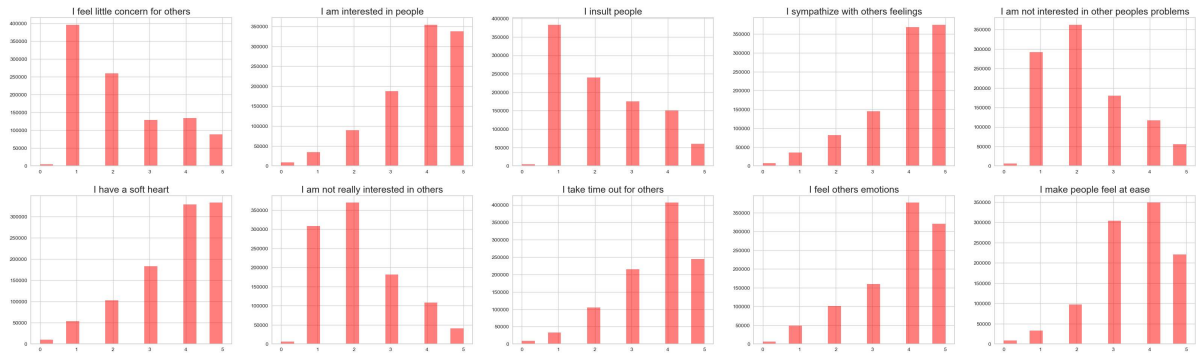


Q&A Distribution related to Neuroticism Personality

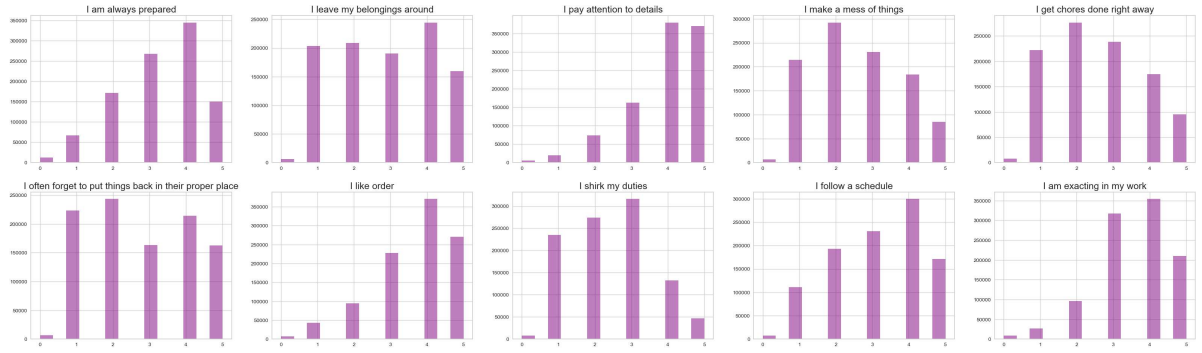
In [11]:
```
vis_questions(EST, est_questions, 'yellow')
```



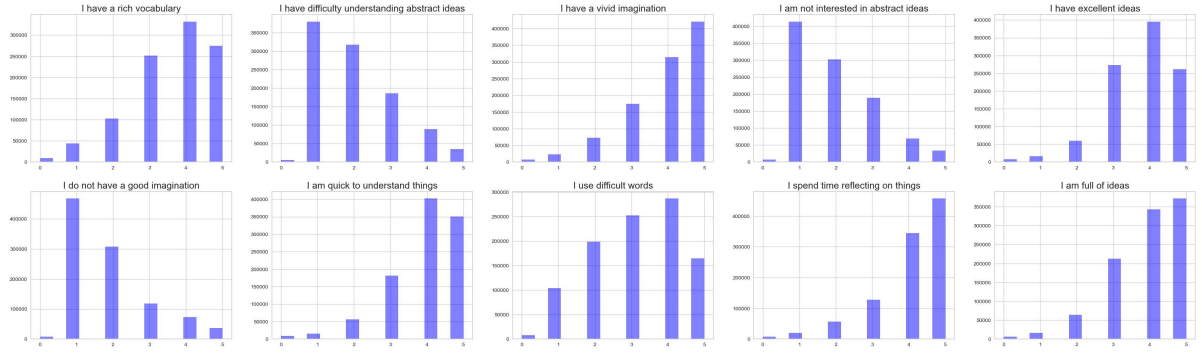
In [12]:
```
vis_questions(AGR, agr_questions, 'red')
```



In [13]:
```
vis_questions(CSN, csn_questions, 'purple')
```



In [14]:
```
vis_questions(OPN, opn_questions, 'blue')
```

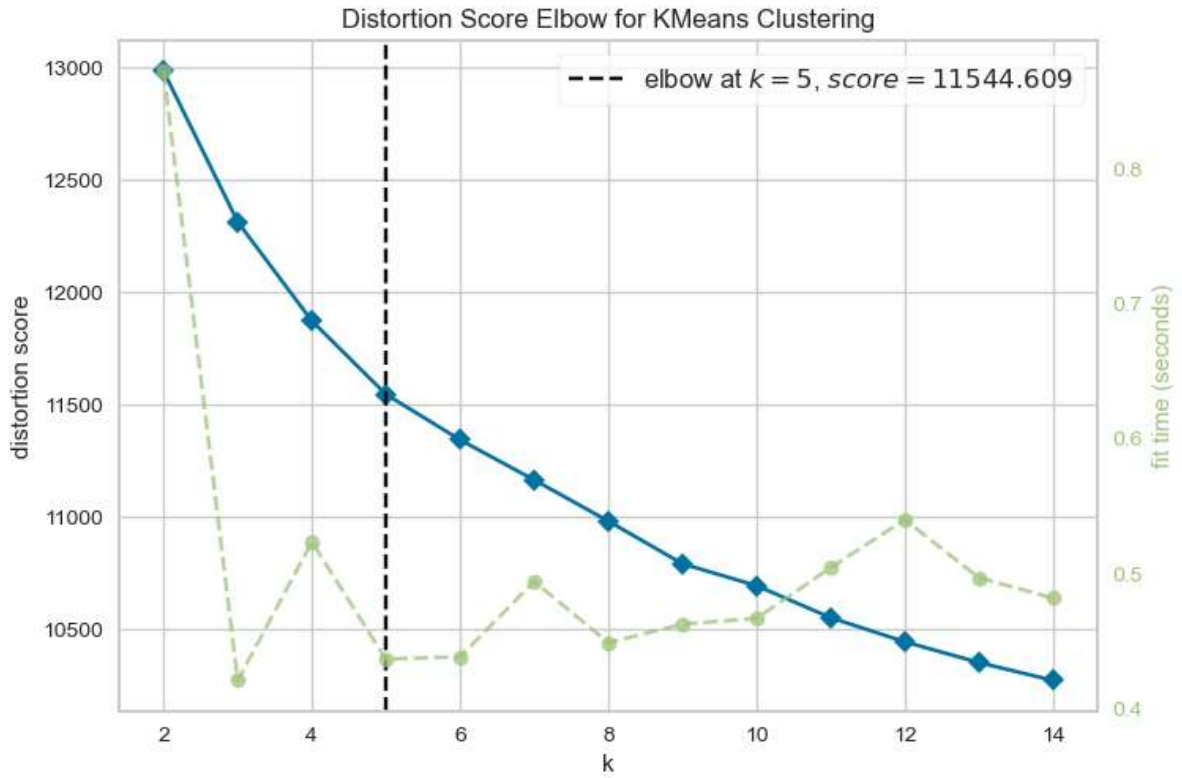


Defining Clustering

In [15]:
```
from sklearn.preprocessing import MinMaxScaler

df = data.drop('country', axis=1)
columns = list(df.columns)

scaler = MinMaxScaler(feature_range=(0,1))
df = scaler.fit_transform(df)
df = pd.DataFrame(df, columns=columns)
df_sample = df[:5000]
```

In [16]:
```
from sklearn.cluster import KMeans
from yellowbrick.cluster import KElbowVisualizer

kmeans = KMeans()
visualizer = KElbowVisualizer(kmeans, k=(2,15))
visualizer.fit(df_sample)
visualizer.poof()
```

```
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: F
utureWarning: The default value of `n_init` will change from 10 to 'auto' in
1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```



Out[16]: <Axes: title={'center': 'Distortion Score Elbow for KMeans Clustering'}, xlabel='k', ylabel='distortion score'>

Results are showing that 5 clusters looks optimum for the data set and we already working on OCEAN model of 5 different personalities traits.

Train the Model

In [18]:
```
from sklearn.cluster import KMeans

df_model = data.drop('country', axis=1)

kmeans = KMeans(n_clusters=5)
k_fit = kmeans.fit(df_model)
```

```
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\cluster\_kmeans.py:1412: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  super()._check_params_vs_input(X, default_n_init=10)
```

In [19]:
```
pd.options.display.max_columns = 10
predictions = k_fit.labels_
df_model['Clusters'] = predictions
df_model.head()
```

Out[19]:

	EXT1	EXT2	EXT3	EXT4	EXT5	...	OPN7	OPN8	OPN9	OPN10	Clusters
0	4.0	1.0	5.0	2.0	5.0	...	5.0	3.0	4.0	5.0	2
1	3.0	5.0	3.0	4.0	3.0	...	4.0	2.0	5.0	3.0	3
2	2.0	3.0	4.0	4.0	3.0	...	5.0	3.0	4.0	4.0	3
3	2.0	2.0	2.0	3.0	4.0	...	4.0	4.0	3.0	3.0	0
4	3.0	3.0	3.0	3.0	5.0	...	5.0	3.0	5.0	5.0	2

5 rows × 51 columns

Results of the Model and Predictions

How many individual do we have for each cluster?

In [20]:
```
df_model.Clusters.value_counts()
```

Out[20]:
```
4    227181
3    212816
2    209702
1    200635
0    163147
Name: Clusters, dtype: int64
```

Grouping results into clusters

In [21]:
```
pd.options.display.max_columns = 150
df_model.groupby('Clusters').mean()
```

Out[21]:

Clusters	EXT1	EXT2	EXT3	EXT4	EXT5	EXT6	EXT7	EXT8	EXT
0	2.194665	3.279815	2.793401	3.397464	2.589064	2.758169	2.124569	3.517956	2.6546(
1	1.810223	3.625409	2.233409	4.053226	2.285733	3.192035	1.744586	4.067471	2.1776(
2	3.474163	1.940883	4.339515	2.216231	4.265753	1.664867	3.888613	2.806702	3.7454
3	2.118252	3.314281	3.091083	3.667821	2.951160	2.712428	2.198782	4.005841	2.3221(
4	3.447529	1.917713	3.790092	2.509356	4.038925	1.834027	3.649482	2.771931	3.7596

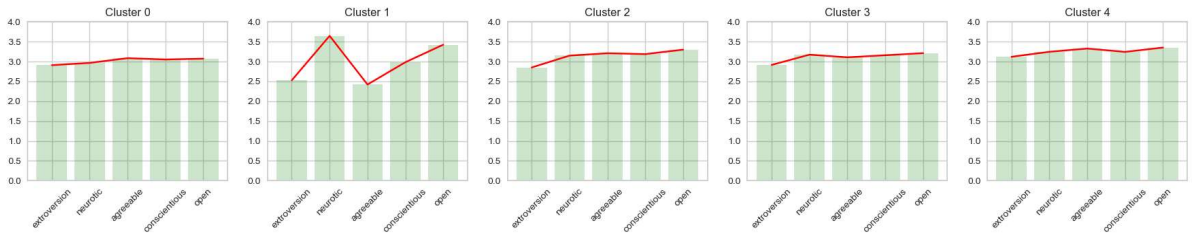
Sum up the each question groups

In [22]:
```
col_list = list(df_model)
ext = col_list[0:10]
est = col_list[10:20]
agr = col_list[20:30]
csn = col_list[30:40]
opn = col_list[40:50]

data_sums = pd.DataFrame()
data_sums['extroversion'] = df_model[ext].sum(axis=1)/10
data_sums['neurotic'] = df_model[est].sum(axis=1)/10
data_sums['agreeable'] = df_model[agr].sum(axis=1)/10
data_sums['conscientious'] = df_model[csn].sum(axis=1)/10
data_sums['open'] = df_model[opn].sum(axis=1)/10
data_sums['clusters'] = predictions
data_sums.groupby('clusters').mean()
```

Out[22]:

clusters	extroversion	neurotic	agreeable	conscientious	open
0	2.909009	2.527920	2.851194	2.914710	3.120438
1	2.966319	3.647053	3.149676	3.173664	3.246015
2	3.085414	2.421786	3.208768	3.106559	3.327056
3	3.051976	2.983073	3.187312	3.158763	3.243259
4	3.072375	3.424805	3.299968	3.211236	3.352354

In [23]:
```
dataclusters = data_sums.groupby('clusters').mean()
plt.figure(figsize=(22,3))
for i in range(0, 5):
    plt.subplot(1,5,i+1)
    plt.bar(dataclusters.columns, dataclusters.iloc[:, i], color='green', alph
    plt.plot(dataclusters.columns, dataclusters.iloc[:, i], color='red')
    plt.title('Cluster ' + str(i))
    plt.xticks(rotation=45)
    plt.ylim(0,4);
```



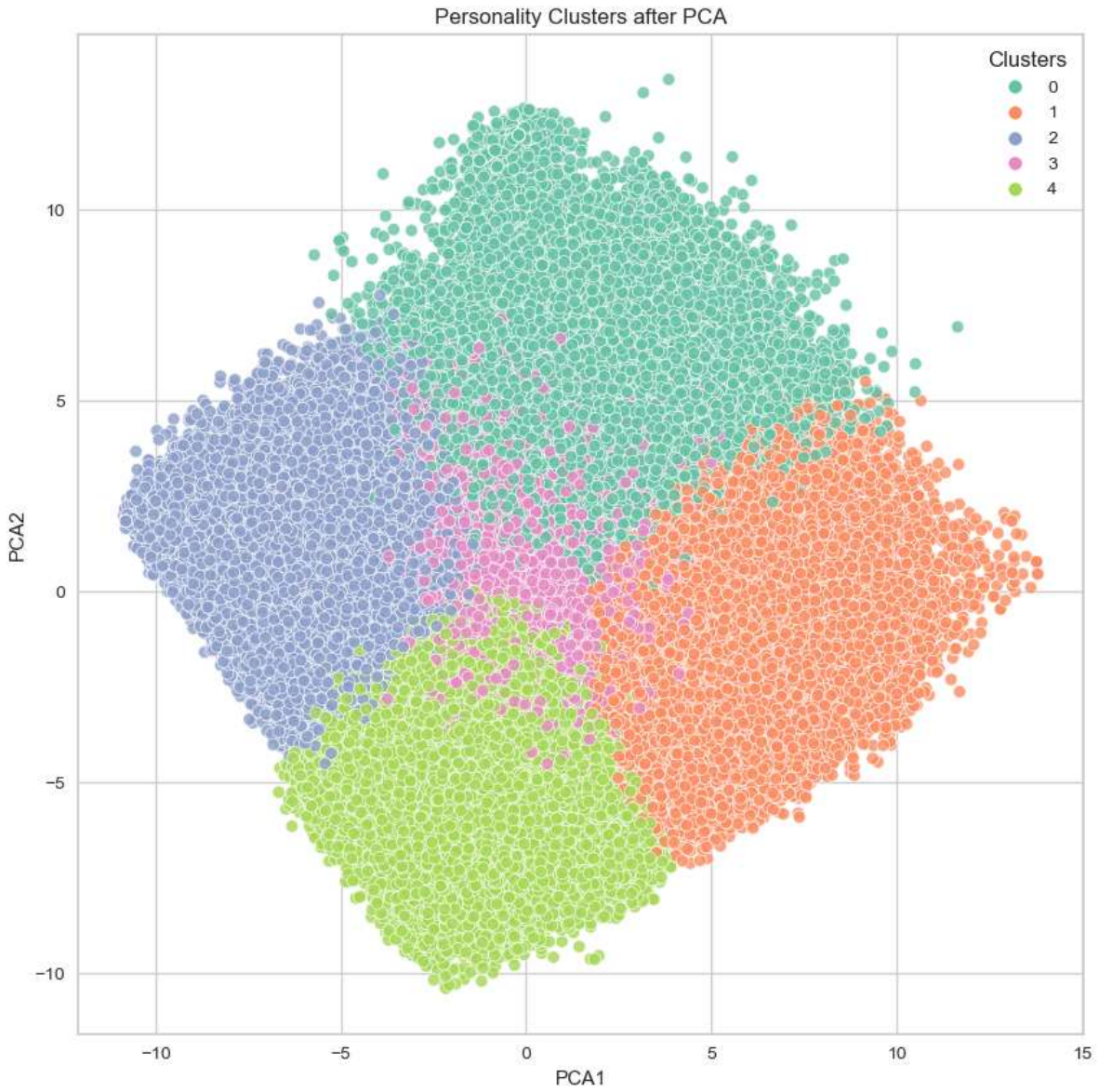
Visualizing the Cluster Predictions

In [24]:
```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
pca_fit = pca.fit_transform(df_model)

df_pca = pd.DataFrame(data=pca_fit, columns=['PCA1', 'PCA2'])
df_pca['Clusters'] = predictions
df_pca.head()
```

Out[24]:

	PCA1	PCA2	Clusters
0	-5.415861	-0.443574	2
1	0.316616	2.309447	3
2	-0.586280	1.733510	3
3	1.495434	0.866191	0
4	-4.461952	2.665236	2

In [25]:
```
plt.figure(figsize=(10,10))
sns.scatterplot(data=df_pca, x='PCA1', y='PCA2', hue='Clusters', palette='Set2
plt.title('Personality Clusters after PCA');
```



Save the Model

In [26]:
```
import pickle

Pkl_Filename = "Pickle_RL_Model.pkl"

with open(Pkl_Filename, 'wb') as file:
    pickle.dump(k_fit, file)
```

Testing the Model

In [27]:
```
my_data = pd.read_excel('../dataset/test_data.xlsx')
my_data-
```

Out[27]:

	EXT1	EXT2	EXT3	EXT4	EXT5	EXT6	EXT7	EXT8	EXT9	EXT10	EST1	EST2	EST3	EST
0	2	3	3	2	2	5	2	4	5	4	2	3	4	

In [28]:
```
my_personality = k_fit.predict(my_data)
print('My Personality Cluster: ', my_personality)
```

```
My Personality Cluster:  [3]
```

Sum of question groups

In [29]:
```
# Summing up the my question groups
col_list = list(my_data)
ext = col_list[0:10]
est = col_list[10:20]
agr = col_list[20:30]
csn = col_list[30:40]
opn = col_list[40:50]

my_sums = pd.DataFrame()
my_sums['extroversion'] = my_data[ext].sum(axis=1)/10
my_sums['neurotic'] = my_data[est].sum(axis=1)/10
my_sums['agreeable'] = my_data[agr].sum(axis=1)/10
my_sums['conscientious'] = my_data[csn].sum(axis=1)/10
my_sums['open'] = my_data[opn].sum(axis=1)/10
my_sums['cluster'] = my_personality

my_sums
```

Out[29]:

	extroversion	neurotic	agreeable	conscientious	open	cluster
0	3.2	2.4	3.4	3.3	3.4	3

Visual the cluster

In [30]:
```
my_sum = my_sums.drop('cluster', axis=1)
plt.bar(my_sum.columns, my_sum.iloc[0,:], color='green', alpha=0.2)
plt.plot(my_sum.columns, my_sum.iloc[0,:], color='red')
plt.title('Cluster 4')
plt.xticks(rotation=45)
plt.ylim(0,4);
```