Project Name: TURKEY STUDENT EVALUATION

Project Done By:

ANKIT KUMAR SINHA

SONAL

DHEERAJ

JYOTI ¶

Problem Statement:

- We have to build a Unsupervised Model which should mimimum number of clusters among the students data

Dataset description:

- There are total 5820 instances in the dataset
- Thrity three baseline variables instr(instructor), class, nb,repeat, attendace,difficulty and 28 Questions have been answered by each of the 5820 students(instances).

One can find more about the dataset here (http://archive.ics.uci.edu/ml/datasets/Turkiye+Student+Evaluation).

Steps of our project:

- Importing the dataset and understanding the dataset
- Exploratory Data Analysis
- Modelling and Evaluation

IMPORTING THE DATASETS AND UNDERSTANDING THE DATASETS

In [2]:

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

Loading the data set into data frame

In [5]:

```
Turkye_Data=pd.read_csv("turkiye-student-evaluation_generic.csv")
```

In [7]:

```
Turkye_Data.head(10)
```

Out[7]:

	instr	class	nb.repeat	attendance	difficulty	Q1	Q2	Q3	Q4	Q5	...	Q19	Q20	Q21	Q22
0	1	2	1	0	4	3	3	3	3	3	...	3	3	3	3
1	1	2	1	1	3	3	3	3	3	3	...	3	3	3	3
2	1	2	1	2	4	5	5	5	5	5	...	5	5	5	5
3	1	2	1	1	3	3	3	3	3	3	...	3	3	3	3
4	1	2	1	0	1	1	1	1	1	1	...	1	1	1	1
5	1	2	1	3	3	4	4	4	4	4	...	4	4	4	4
6	1	2	1	1	3	4	4	4	4	4	...	4	4	4	4
7	1	2	1	1	3	5	5	5	5	5	...	5	5	5	5
8	1	2	1	1	3	4	4	4	4	4	...	4	4	4	4
9	1	2	1	4	4	4	4	4	4	4	...	4	4	4	4

10 rows × 33 columns

Data Qunatitavie Analysis

In [8]:

```
Turkye_Data.describe()
```

Out[8]:

	instr	class	nb.repeat	attendance	difficulty	Q1	Q2
count	5820.000000	5820.000000	5820.000000	5820.000000	5820.000000	5820.000000	5820.000000
mean	2.485567	7.276289	1.214089	1.675601	2.783505	2.929897	3.073883
std	0.718473	3.688175	0.532376	1.474975	1.348987	1.341077	1.285251
min	1.000000	1.000000	1.000000	0.000000	1.000000	1.000000	1.000000
25%	2.000000	4.000000	1.000000	0.000000	1.000000	2.000000	2.000000
50%	3.000000	7.000000	1.000000	1.000000	3.000000	3.000000	3.000000
75%	3.000000	10.000000	1.000000	3.000000	4.000000	4.000000	4.000000
max	3.000000	13.000000	3.000000	4.000000	5.000000	5.000000	5.000000

8 rows × 33 columns

In [66]:

```
Turkye_Data.shape
```

Out[66]:

```
(5820, 33)
```

This cell gives the information about the empty attributes in data

In [12]:

```
Turkye_Data.isnull().sum()
```

Out[12]:

```
instr         0
class         0
nb.repeat     0
attendance    0
difficulty    0
Q1            0
Q2            0
Q3            0
Q4            0
Q5            0
Q6            0
Q7            0
Q8            0
Q9            0
Q10           0
Q11           0
Q12           0
Q13           0
Q14           0
Q15           0
Q16           0
Q17           0
Q18           0
Q19           0
Q20           0
Q21           0
Q22           0
Q23           0
Q24           0
Q25           0
Q26           0
Q27           0
Q28           0
dtype: int64
```

No empty attribute is present in the data as each fiels as 0 result

EXPLORATORY DATA ANALYSIS

In [13]:

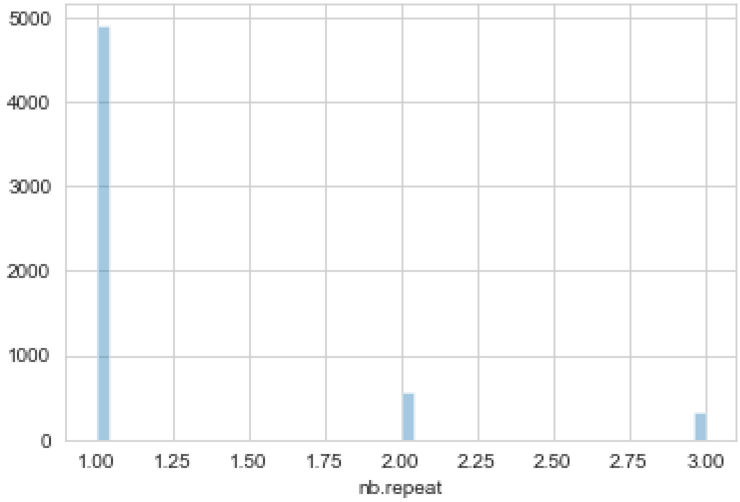
```
import seaborn as sns
sns.set_style('whitegrid')
```

In [69]:

```
sns.distplot(a=Turkye_Data['nb.repeat'],kde=False)
```

Out[69]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2665bacb0f0>
```



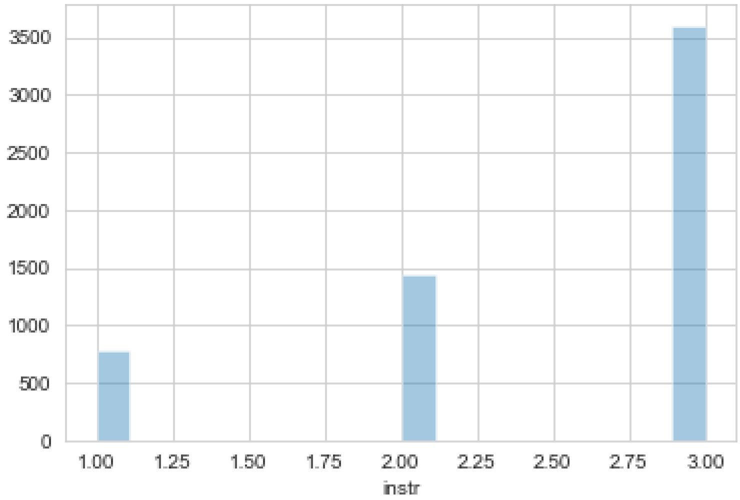
Number of times student taking different courses with the help of histogram

In [18]:

```
sns.distplot(a=Turkye_Data['instr'],kde=False)
```

Out[18]:

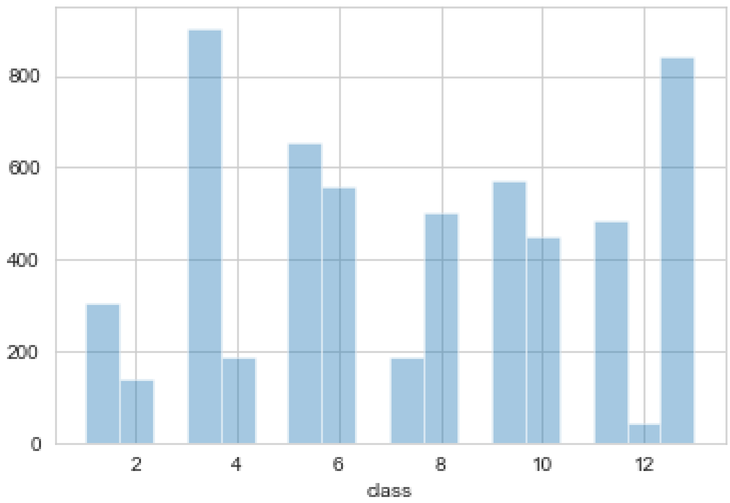
```
<matplotlib.axes._subplots.AxesSubplot at 0x26655f055f8>
```



Number of students rated different instuctors with the help of histogram

In [19]:
```
sns.distplot(a=Turkye_Data['class'],kde=False)
```

Out[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2665584e860>
```



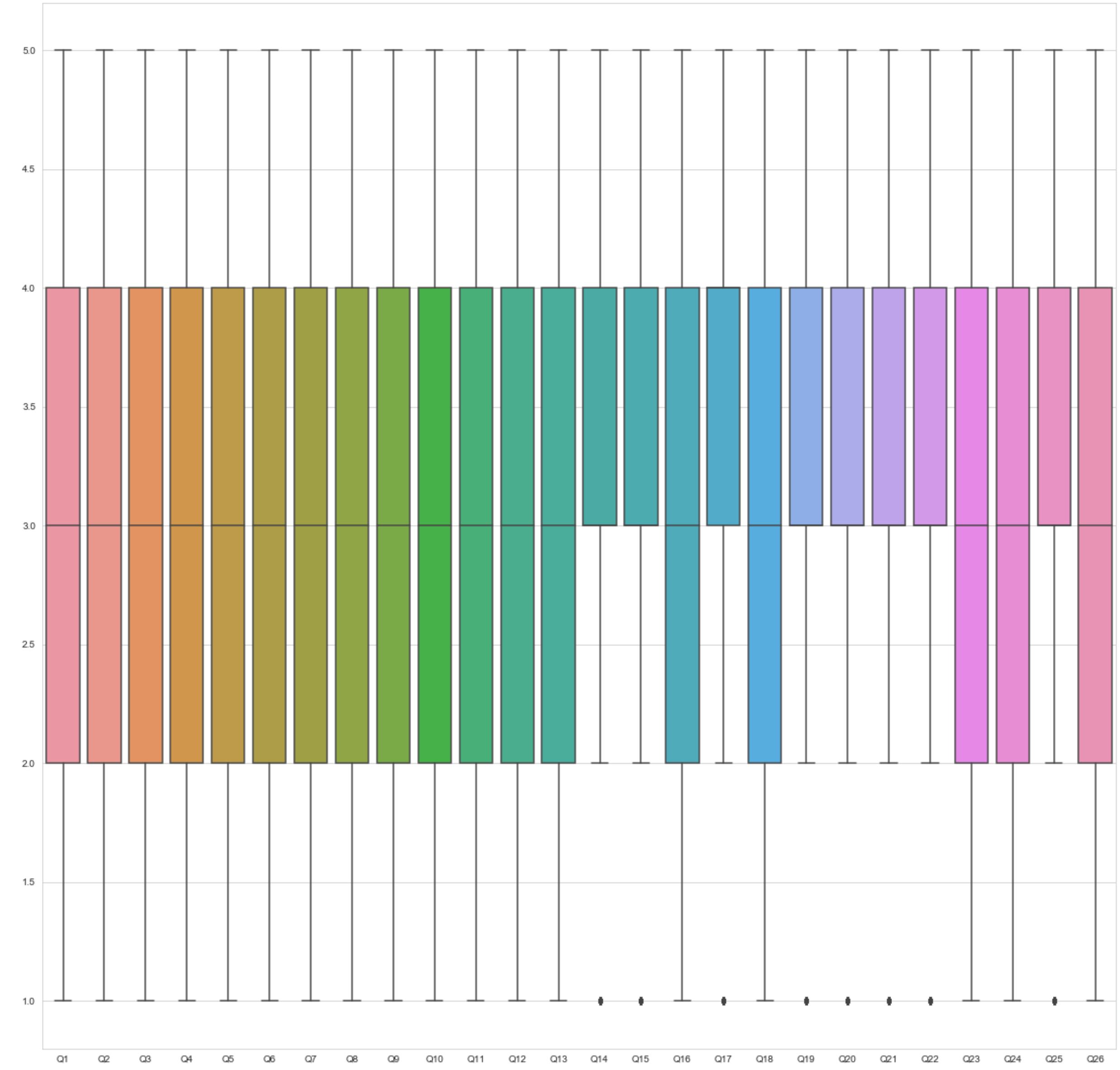
Number of students in each class with the help of histogram

In [65]:

```
plt.figure(figsize=(20, 20))
sns.boxplot(data=Turkye_Data.iloc[:,5:31 ])
```

Out[65]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2665a759978>
```



Graph to see how the rating has been given by student for each questions

Lets understand how the students have reponded for the questions against classes

In [38]:

```
questionmeans = []
classlist = []
questions = []
totalplotdata = pd.DataFrame(list(zip(classlist,questions,questionmeans))
                      ,columns=['class','questions', 'mean'])
for class_num in range(1,13):
    class_data = Turkye_Data[(Turkye_Data["class"]==class_num)]

    questionmeans = []
    classlist = []
    questions = []

    for num in range(1,28):
        questions.append(num)
    for col in range(5,32):
        questionmeans.append(class_data.iloc[:,col].mean())
    classlist += 32 * [class_num]
    print(classlist)
    plotdata = pd.DataFrame(list(zip(classlist,questions,questionmeans))
                      ,columns=['class','questions', 'mean'])
    totalplotdata = totalplotdata.append(plotdata, ignore_index=True)
```

```
[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1]
[2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2]
[3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3,
3, 3, 3, 3, 3, 3]
[4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 4,
4, 4, 4, 4, 4, 4]
[5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5, 5,
5, 5, 5, 5, 5, 5]
[6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6, 6,
6, 6, 6, 6, 6, 6]
[7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7,
7, 7, 7, 7, 7, 7]
[8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8,
8, 8, 8, 8, 8, 8]
[9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9, 9,
9, 9, 9, 9, 9, 9]
[10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 1
0, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10]
[11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 1
1, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11, 11]
[12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 1
2, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12, 12]
```

In [39]:

```
totalplotdata
```

Out[39]:

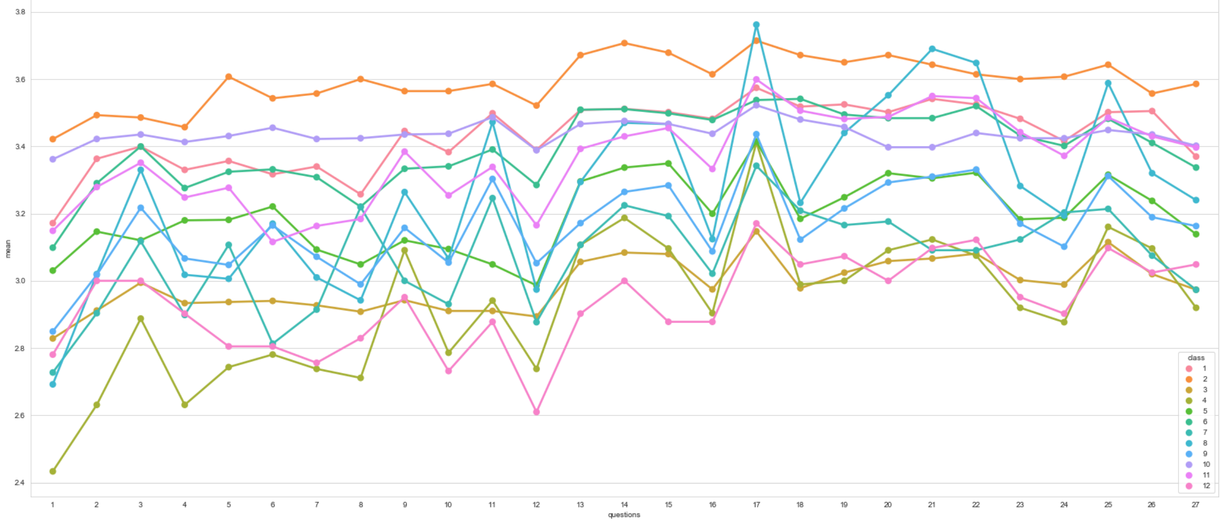
	class	questions	mean
0	1	1	3.171617
1	1	2	3.363036
2	1	3	3.399340
3	1	4	3.330033
4	1	5	3.356436
5	1	6	3.316832
6	1	7	3.339934
7	1	8	3.257426
8	1	9	3.445545
9	1	10	3.382838
10	1	11	3.498350
11	1	12	3.389439
12	1	13	3.508251
13	1	14	3.511551
14	1	15	3.501650
15	1	16	3.481848
16	1	17	3.574257
17	1	18	3.518152
18	1	19	3.524752
19	1	20	3.501650
20	1	21	3.541254
21	1	22	3.524752
22	1	23	3.481848
23	1	24	3.415842
24	1	25	3.501650
25	1	26	3.504950
26	1	27	3.369637
27	2	1	3.421429
28	2	2	3.492857
29	2	3	3.485714
...	...	...	...
294	11	25	3.485537
295	11	26	3.429752
296	11	27	3.396694

	class	questions	mean
297	12	1	2.780488
298	12	2	3.000000
299	12	3	3.000000
300	12	4	2.902439
301	12	5	2.804878
302	12	6	2.804878
303	12	7	2.756098
304	12	8	2.829268
305	12	9	2.951220
306	12	10	2.731707
307	12	11	2.878049
308	12	12	2.609756
309	12	13	2.902439
310	12	14	3.000000
311	12	15	2.878049
312	12	16	2.878049
313	12	17	3.170732
314	12	18	3.048780
315	12	19	3.073171
316	12	20	3.000000
317	12	21	3.097561
318	12	22	3.121951
319	12	23	2.951220
320	12	24	2.902439
321	12	25	3.097561
322	12	26	3.024390
323	12	27	3.048780

324 rows × 3 columns

In [40]:
```
plt.figure(figsize=(28,12))
sns.pointplot(x="questions",y="mean",data=totalplotdata,hue="class")
```

Out[40]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x26656925780>
```



This graph shows that students of class 2 has rated each questions quite well while students of class 4 has rated each questions very poor.Such more pattern can be find within the data

Modelling and Model Evaluation

In [45]:
```
Question=Turkye_Data.iloc[:,5:33]
```

In [46]:
```
Question.head()
```

Out[46]:

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	...	Q19	Q20	Q21	Q22	Q23	Q24	Q25	Q26
0	3	3	3	3	3	3	3	3	3	3	...	3	3	3	3	3	3	3	3
1	3	3	3	3	3	3	3	3	3	3	...	3	3	3	3	3	3	3	3
2	5	5	5	5	5	5	5	5	5	5	...	5	5	5	5	5	5	5	5
3	3	3	3	3	3	3	3	3	3	3	...	3	3	3	3	3	3	3	3
4	1	1	1	1	1	1	1	1	1	1	...	1	1	1	1	1	1	1	1

5 rows × 28 columns

Reducind DImenstion of Data using PCA

In [47]:
```
from sklearn.decomposition import PCA
pca=PCA(n_components=2)
```

In [48]:

```
Question_modified=pca.fit_transform(Question)
```

In [50]:

```
Question_modified.shape
```

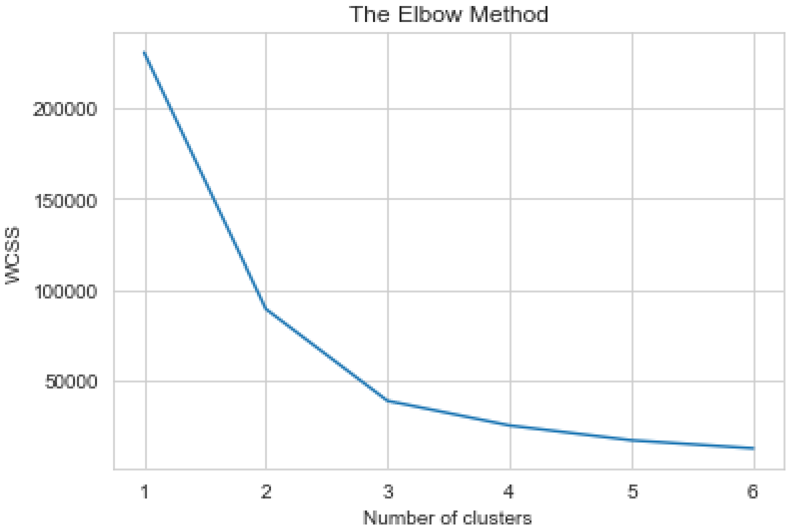
Out[50]:

```
(5820, 2)
```

Claculating Least Number of Cluster fit Bset for Data Using WCSS and Elbow Method

In [53]:

```
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 7):
    kmeans = KMeans(n_clusters = i, init = 'k-means++', random_state = 42)
    kmeans.fit(Question_modified)
    wcss.append(kmeans.inertia_)
plt.plot(range(1, 7), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
```



This graph shows that at clusters=3 the elbow method suggest a good accuracy for the data

Fitting the Data in Kmeans Algorithm

In [55]:

```
kmeans=KMeans(n_clusters=3)
y_kmeans = kmeans.fit_predict(Question_modified)
```

In [57]:

```
y_kmeans.shape
```

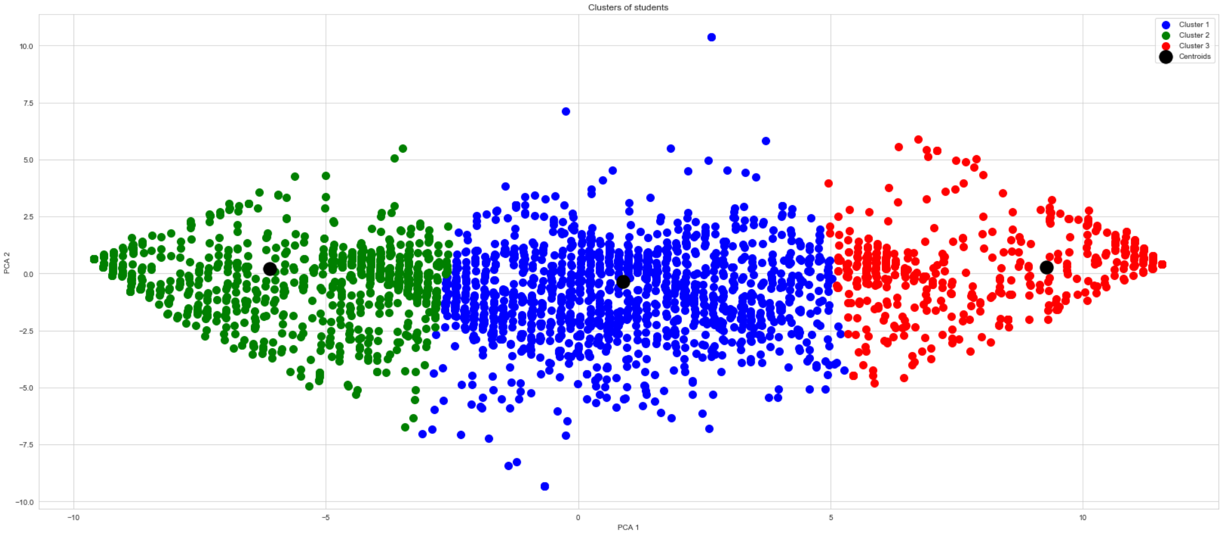
Out[57]:

```
(5820,)
```

FINAL PLOT OF THE DATASETS WITH THEIR RESPECTIVE CLUSTERS

In [62]:

```
plt.figure(figsize=(28,12))
plt.scatter(Question_modified[y_kmeans == 0, 0],Question_modified[y_kmeans =
plt.scatter(Question_modified[y_kmeans == 1, 0],Question_modified[y_kmeans =
plt.scatter(Question_modified[y_kmeans == 2, 0],Question_modified[y_kmeans =
plt.scatter(kmeans.cluster_centers_[:, 0], kmeans.cluster_centers_[:, 1], s
plt.title('Clusters of students')
plt.xlabel('PCA 1')
plt.ylabel('PCA 2')
plt.legend()
plt.show()
```



In [64]:

```
Question_modified.shape
```

Out[64]:

```
(5820, 2)
```

In [ ]:

```

```