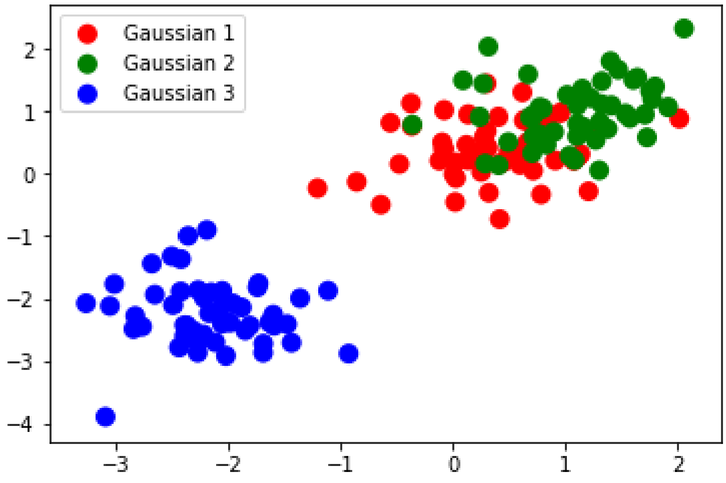
Sample 3 Gaussians

In [46]:

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.mixture import GaussianMixture
from scipy.stats import norm

x_1=np.random.normal(0.3, 0.5, size=50)
y_1=np.random.normal(0.3, 0.5, size=50)

x_2=np.random.normal(1, 0.5, size=50)
y_2=np.random.normal(1, 0.5, size=50)

x_3=np.random.normal(-2.2, 0.5, size=50)
y_3=np.random.normal(-2.2, 0.5, size=50)

fig, ax=plt.subplots()
ax.scatter(x_1, y_1, s=100, edgecolor='', label='Gaussian 1', c='r')
ax.scatter(x_2, y_2, s=100, edgecolor='', label='Gaussian 2', c='g')
ax.scatter(x_3, y_3, s=100, edgecolor='', label='Gaussian 3', c='b')
plt.legend(loc='best')
plt.show()

X=[]
Y=[]

for i in range(len(x_1)):
    X.append([x_1[i], y_1[i]])
    Y.append(0)

for i in range(len(x_2)):
    X.append([x_2[i], y_2[i]])
    Y.append(1)

for i in range(len(x_3)):
    X.append([x_3[i], y_3[i]])
    Y.append(2)
```



Silhouette method for selecting k

In [52]:

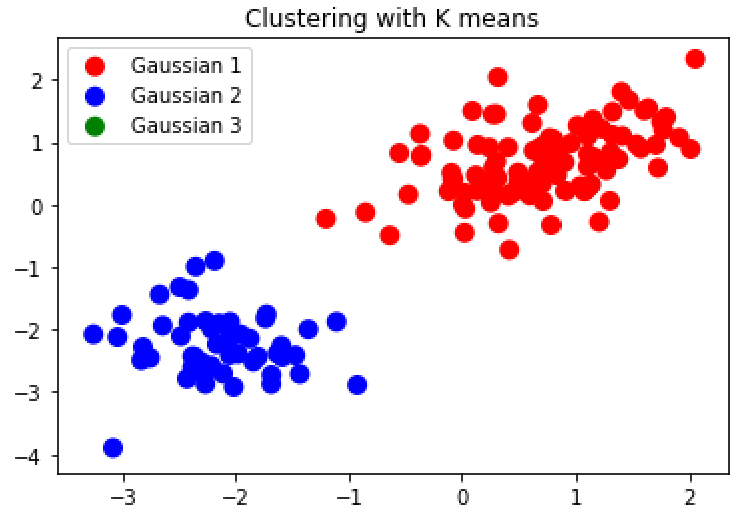
```
from sklearn.metrics import silhouette_samples, silhouette_score
from sklearn.cluster import KMeans
from sklearn.mixture import GaussianMixture

n_clusters_range=[2,3,4,5]

for n_clusters in n_clusters_range:
    clusterer=KMeans(n_clusters=n_clusters, random_state=0)
    cluster_labels=clusterer.fit_predict(X)

    silhouette_avg = silhouette_score(X, cluster_labels)
    print("For n_clusters =", n_clusters,
          "The average silhouette_score is :", silhouette_avg)


print('---------------------------------------------------------------------')

for n_clusters in n_clusters_range:
    clusterer=GaussianMixture(n_components=n_clusters, random_state=0)
    cluster_labels=clusterer.fit_predict(X)

    silhouette_avg = silhouette_score(X, cluster_labels)
    print("For n_clusters =", n_clusters,
          "The average silhouette_score is :", silhouette_avg)
```

```
For n_clusters = 2 The average silhouette_score is : 0.743598175540867
4
For n_clusters = 3 The average silhouette_score is : 0.509566437322498
2
For n_clusters = 4 The average silhouette_score is : 0.452578571944613
1
For n_clusters = 5 The average silhouette_score is : 0.325387041863751
45
---------------------------------------------------------------------
For n_clusters = 2 The average silhouette_score is : 0.743598175540867
4
For n_clusters = 3 The average silhouette_score is : 0.508451979039761
4
For n_clusters = 4 The average silhouette_score is : 0.368567739978862
9
For n_clusters = 5 The average silhouette_score is : 0.315456461454415
36
```

# K-Means clustering with k=2 clusters

In [58]:

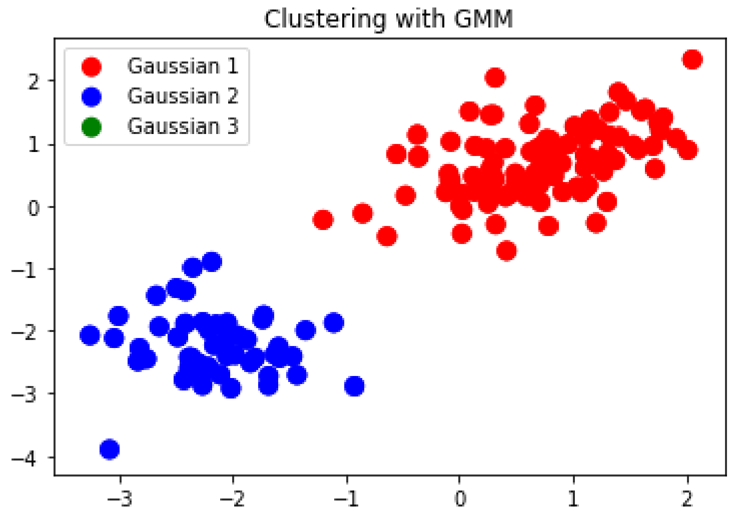
```
from sklearn.cluster import KMeans

kmeans=KMeans(n_clusters=2, random_state=0).fit(X)

cx1=[]
cx2=[]
cx3=[]
cy1=[]
cy2=[]
cy3=[]

for i in range(len(kmeans.labels_)):
    try:
        if(kmeans.labels_[i]==0):
            cx1.append(X[i][0])
            cy1.append(X[i][1])
        elif(kmeans.labels_[i]==1):
            cx2.append(X[i][0])
            cy2.append(X[i][1])
        else:
            cx3.append(X[i][0])
            cy3.append(X[i][1])
    except Exception as e:
        print(e)
        continue
fig, ax=plt.subplots()
ax.scatter(cx1, cy1, s=100, edgecolor='', label='Gaussian 1', c='r')
ax.scatter(cx2, cy2, s=100, edgecolor='', label='Gaussian 2', c='b')
ax.scatter(cx3, cy3, s=100, edgecolor='', label='Gaussian 3', c='g')
plt.legend(loc='best')
plt.title('Clustering with K means')
plt.show()
```



GMM soft clustering with k=2 clusters

In [59]:

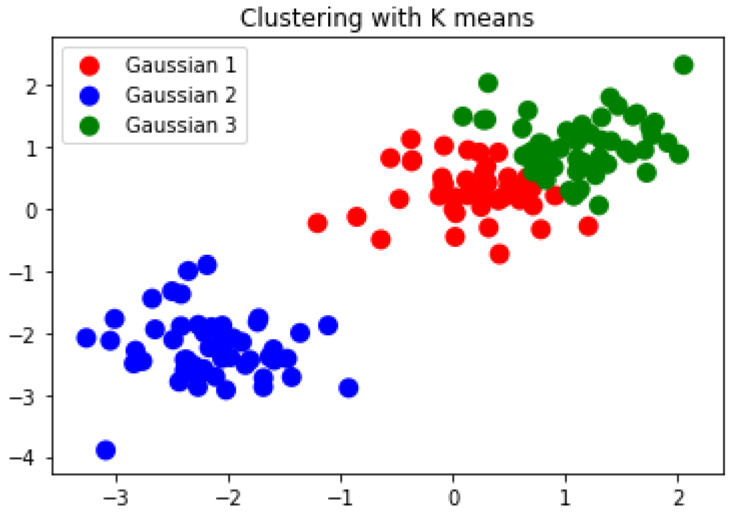
```
from sklearn.mixture import GaussianMixture

gmm_labels=GaussianMixture(n_components=2, random_state=0).fit_predict(X)

for i in range(len(gmm_labels)):
    try:
        if(gmm_labels[i]==0):
            cx1.append(X[i][0])
            cy1.append(X[i][1])
        elif(gmm_labels[i]==1):
            cx2.append(X[i][0])
            cy2.append(X[i][1])
        else:
            cx3.append(X[i][0])
            cy3.append(X[i][1])
    except Exception as e:
        print(e)
        continue

fig, ax=plt.subplots()
ax.scatter(cx1, cy1, s=100, edgecolor='', label='Gaussian 1', c='r')
ax.scatter(cx2, cy2, s=100, edgecolor='', label='Gaussian 2', c='b')
ax.scatter(cx3, cy3, s=100, edgecolor='', label='Gaussian 3', c='g')
plt.legend(loc='best')
plt.title('Clustering with GMM')
plt.show()
```



K-Means clustering with k=3 clusters

In [57]:

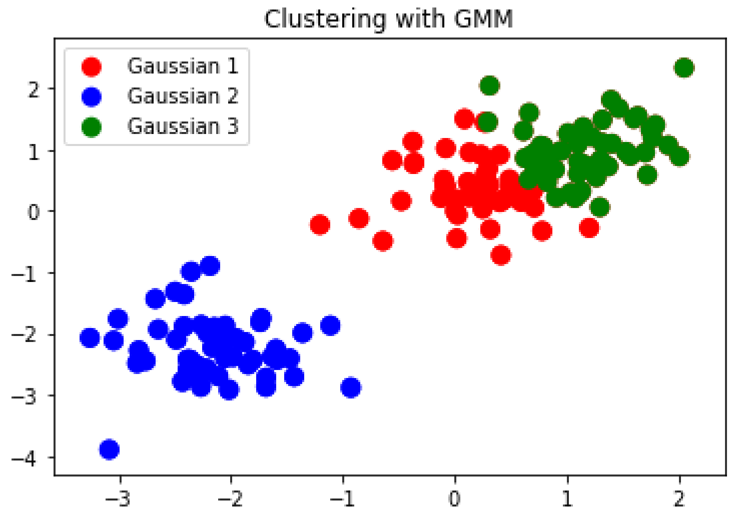
```
from sklearn.cluster import KMeans

kmeans=KMeans(n_clusters=3, random_state=0).fit(X)

cx1=[]
cx2=[]
cx3=[]
cy1=[]
cy2=[]
cy3=[]

for i in range(len(kmeans.labels_)):
    try:
        if(kmeans.labels_[i]==0):
            cx1.append(X[i][0])
            cy1.append(X[i][1])
        elif(kmeans.labels_[i]==1):
            cx2.append(X[i][0])
            cy2.append(X[i][1])
        else:
            cx3.append(X[i][0])
            cy3.append(X[i][1])
    except Exception as e:
        print(e)
        continue
fig, ax=plt.subplots()
ax.scatter(cx1, cy1, s=100, edgecolor='', label='Gaussian 1', c='r')
ax.scatter(cx2, cy2, s=100, edgecolor='', label='Gaussian 2', c='b')
ax.scatter(cx3, cy3, s=100, edgecolor='', label='Gaussian 3', c='g')
plt.legend(loc='best')
plt.title('Clustering with K means')
plt.show()
```



GMM soft clustering with k=3 clusters

In [56]:

```
from sklearn.mixture import GaussianMixture

gmm_labels=GaussianMixture(n_components=3, random_state=0).fit_predict(X)

for i in range(len(gmm_labels)):
    try:
        if(gmm_labels[i]==0):
            cx1.append(X[i][0])
            cy1.append(X[i][1])
        elif(gmm_labels[i]==1):
            cx2.append(X[i][0])
            cy2.append(X[i][1])
        else:
            cx3.append(X[i][0])
            cy3.append(X[i][1])
    except Exception as e:
        print(e)
        continue

fig, ax=plt.subplots()
ax.scatter(cx1, cy1, s=100, edgecolor='', label='Gaussian 1', c='r')
ax.scatter(cx2, cy2, s=100, edgecolor='', label='Gaussian 2', c='b')
ax.scatter(cx3, cy3, s=100, edgecolor='', label='Gaussian 3', c='g')
plt.legend(loc='best')
plt.title('Clustering with GMM')
plt.show()
```



Submission:

The required materials are provided as:

a) The parameters for the three Gaussians are: i) Gaussian 1: mu=[0.3, 0.3], sigma=[0.5, 0.5] ii) Gaussian 2: mu=[1, 1], sigma=[0.5, 0.5] iii) Gaussian 3: mu=[-2.2, -2.2], sigma=[0.5, 0.5]

b) Low dimensional visualization of sampled gaussians is displayed in cell 1 above.

c) Kmeans and GMM clusterings are shown in the above cells(cells 3,4,5 and 6) [k was selected according to top two silhouette scores i.e. k=2 and k=3 are shown]

d) We can clearly see from the above plots that for k=2, the performance of both the clustering methods is the same. However, for k=3, GMM slightly outperforms K-means since K-means performs very hard clustering on the datapoints and hence fails to correctly identify the overlapping points from Gaussian 1 and 2 correctly. This is because K-means performs deterministic hard clustering whereas GMM performs probabilistic soft clustering. Hence, for this particular instance GMM performs slightly better than K-means.