In [ ]:
```
'''
NAME: MANE SHIVRAJ PANDURANG
ROLL NO.37
COURSE: AI&DS, SUB:ML(Machine Learning)
CLASS: BE
'''
```

In [ ]:
```
'''
PRACTICAL NO:04
A.Implement K-Means clustering on Iris.csv dataset. Determine the number of clusters
using the elbow method.

'''
```

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
```

In [12]:
```
iris_data = pd.read_csv("Iris.csv")
X = iris_data.iloc[:, :-1]  # Features
y = iris_data.iloc[:, -1]
```

In [13]:
```
X.head()
```

Out[13]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
0	1	5.1	3.5	1.4	0.2
1	2	4.9	3.0	1.4	0.2
2	3	4.7	3.2	1.3	0.2
3	4	4.6	3.1	1.5	0.2
4	5	5.0	3.6	1.4	0.2

In [14]:
```
y.head()
```

Out[14]:
```
0    Iris-setosa
1    Iris-setosa
2    Iris-setosa
3    Iris-setosa
4    Iris-setosa
Name: Species, dtype: object
```

In [15]:
```
X.describe()
```

Out[15]:

	Id	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm
count	150.000000	150.000000	150.000000	150.000000	150.000000
mean	75.500000	5.843333	3.054000	3.758667	1.198667
std	43.445368	0.828066	0.433594	1.764420	0.763161
min	1.000000	4.300000	2.000000	1.000000	0.100000
25%	38.250000	5.100000	2.800000	1.600000	0.300000
50%	75.500000	5.800000	3.000000	4.350000	1.300000
75%	112.750000	6.400000	3.300000	5.100000	1.800000
max	150.000000	7.900000	4.400000	6.900000	2.500000

In [16]:
```
y.describe()
```

Out[16]:
```
count             150
unique              3
top       Iris-setosa
freq               50
Name: Species, dtype: object
```

In [17]:
```
X.info()
```

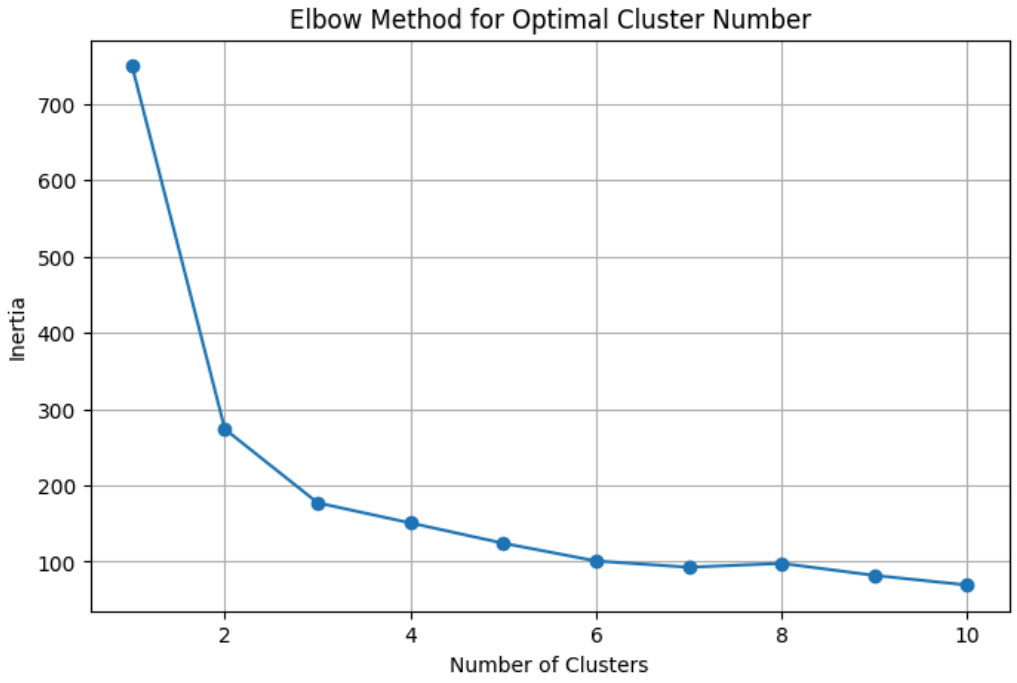
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   Id             150 non-null    int64  
 1   SepalLengthCm  150 non-null    float64
 2   SepalWidthCm   150 non-null    float64
 3   PetalLengthCm  150 non-null    float64
 4   PetalWidthCm   150 non-null    float64
dtypes: float64(4), int64(1)
memory usage: 6.0 KB
```

no null values

In [18]:
```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

In [19]:
```
inertia = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(X_scaled)
    inertia.append(kmeans.inertia_)
```

In [20]:
```
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), inertia, marker='o')
plt.xlabel('Number of Clusters')
plt.ylabel('Inertia')
plt.title('Elbow Method for Optimal Cluster Number')
plt.grid(True)
plt.show()
```



optimal number of clusters are 3 according to the elbow plot

In [21]:
```
optimal_k = 3
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
kmeans.fit(X_scaled)
```

Out[21]:
```
KMeans
KMeans(n_clusters=3, random_state=42)
```

In [22]:
```
iris_data['Cluster'] = kmeans.labels_
centroids = kmeans.cluster_centers_
```

In [23]:
```
plt.figure(figsize=(8, 5))
for i in range(optimal_k):
    plt.scatter(X_scaled[iris_data['Cluster'] == i, 0], X_scaled[iris_data['Cluster'] == i, 1], label=f'Cluster {i}')
plt.scatter(centroids[:, 0], centroids[:, 1], s=200, c='red', label='Centroids')
plt.xlabel('Feature 1')
plt.ylabel('Feature 2')
plt.title('K-Means Clustering')
plt.legend()
plt.grid(True)
plt.show()
```