Implement K-Means clustering on customer dataset. Determine the number of clusters using the elbow method. Dataset link : https://www.kaggle.com/code/heeraldedhia/kmeans-clustering-for-customer-data/input

In [5]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.cluster import KMeans
```

In [6]:
```
data = pd.read_csv("C:/Users/Atharva/OneDrive/Desktop/LP3 code/Mall_Customers.csv")
print(data.head())
```

```
   CustomerID  Gender  Age  Annual Income (k$)  Spending Score (1-100)
0           1    Male   19                  15                      39
1           2    Male   21                  15                      81
2           3  Female   20                  16                       6
3           4  Female   23                  16                      77
4           5  Female   31                  17                      40
```

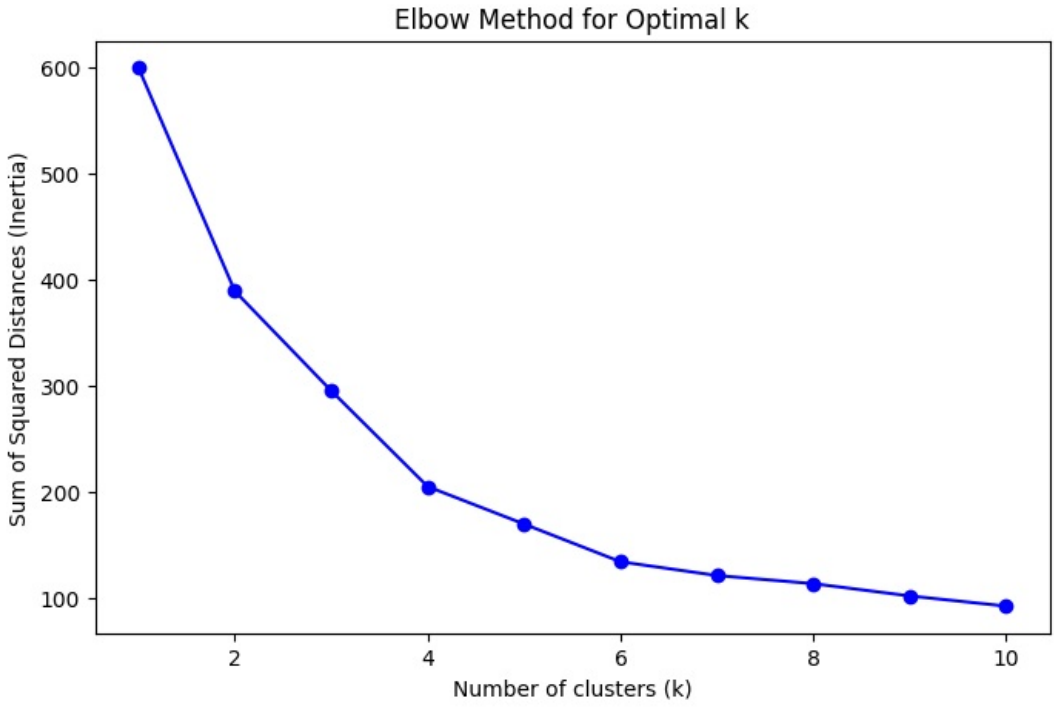
In [7]:
```
# Convert the 'Gender' column to numerical values (e.g., Male = 0, Female = 1)
le = LabelEncoder()
data['Gender'] = le.fit_transform(data['Gender'])
```

In [8]:
```
# Select the relevant features for clustering
features = data[['Age', 'Annual Income (k$)', 'Spending Score (1-100)']]
```

In [9]:
```
# Standardize the features for better clustering performance
scaler = StandardScaler()
scaled_features = scaler.fit_transform(features)
```

In [10]:
```
# Determine the optimal number of clusters using the elbow method
sse = []
for k in range(1, 11):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(scaled_features)
    sse.append(kmeans.inertia_)
```

In [11]:
```
# Plot the elbow curve
plt.figure(figsize=(8, 5))
plt.plot(range(1, 11), sse, marker='o', color='b')
plt.xlabel("Number of clusters (k)")
plt.ylabel("Sum of Squared Distances (Inertia)")
plt.title("Elbow Method for Optimal k")
plt.show()
```

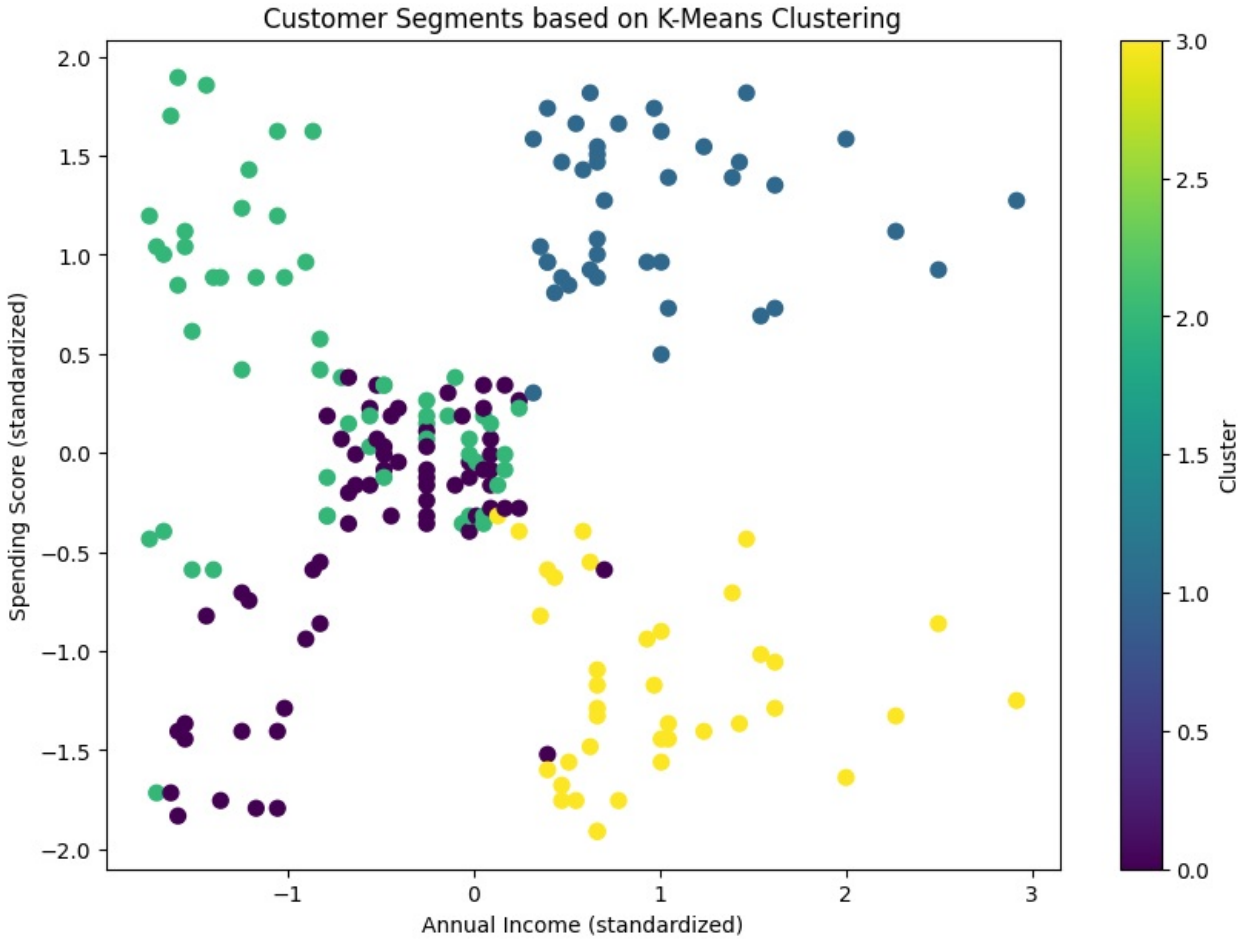


In [12]:
```
# From the elbow plot, choose the optimal number of clusters (e.g., k=4)
optimal_k = 4  # Adjust this based on the elbow plot
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
data['Cluster'] = kmeans.fit_predict(scaled_features)
```

In [13]:
```
# Show the first few rows with cluster labels
print(data[['CustomerID', 'Age', 'Annual Income (k$)', 'Spending Score (1-100)', 'Cluster']].head())
```

	CustomerID	Age	Annual Income (k$)	Spending Score (1-100)	Cluster
0	1	19	15	39	2
1	2	21	15	81	2
2	3	20	16	6	2
3	4	23	16	77	2
4	5	31	17	40	2

In [14]:
```
# Optional: Visualize the clusters (for 2D)
plt.figure(figsize=(10, 7))
plt.scatter(scaled_features[:, 1], scaled_features[:, 2], c=data['Cluster'], cmap='viridis', s=50)
plt.xlabel("Annual Income (standardized)")
plt.ylabel("Spending Score (standardized)")
plt.title("Customer Segments based on K-Means Clustering")
plt.colorbar(label="Cluster")
plt.show()
```



In [ ]: