```python
db_scores = []
clusters_range = range(2, 11)
for k in clusters_range:
    kmeans = KMeans(n_clusters=k, random_state=42)
    clusters = kmeans.fit_predict(feature_matrix)
    db_index = davies_bouldin_score(feature_matrix, clusters)
    db_scores.append(db_index)
```

```python
optimal_k = clusters_range[np.argmin(db_scores)]
kmeans = KMeans(n_clusters=optimal_k, random_state=42)
clusters = kmeans.fit_predict(feature_matrix)
customer_features["Cluster"] = clusters
```

```python
final_db_index = davies_bouldin_score(feature_matrix, clusters)
```

```python
pca = PCA(n_components=2)
reduced_features = pca.fit_transform(feature_matrix)
customer_features["PCA1"] = reduced_features[:, 0]
customer_features["PCA2"] = reduced_features[:, 1]
```

```python
plt.figure(figsize=(10, 6))
sns.scatterplot(data=customer_features, x="PCA1", y="PCA2", hue="Cluster", palette="viridi
plt.title(f"Customer Clusters (K={optimal_k})")
plt.xlabel("PCA Component 1")
plt.ylabel("PCA Component 2")
plt.legend(title="Cluster")
plt.show()
```

```python
print(f"Optimal Number of Clusters: {optimal_k}")
print(f"Davies-Bouldin Index: {final_db_index}")
```

```python
customer_features[["CustomerID", "Cluster"]].to_csv("Customer_Clusters.csv", index=False)
```