In [1]:
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
import plotly.graph_objects as go
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import warnings
warnings.filterwarnings('ignore')
```

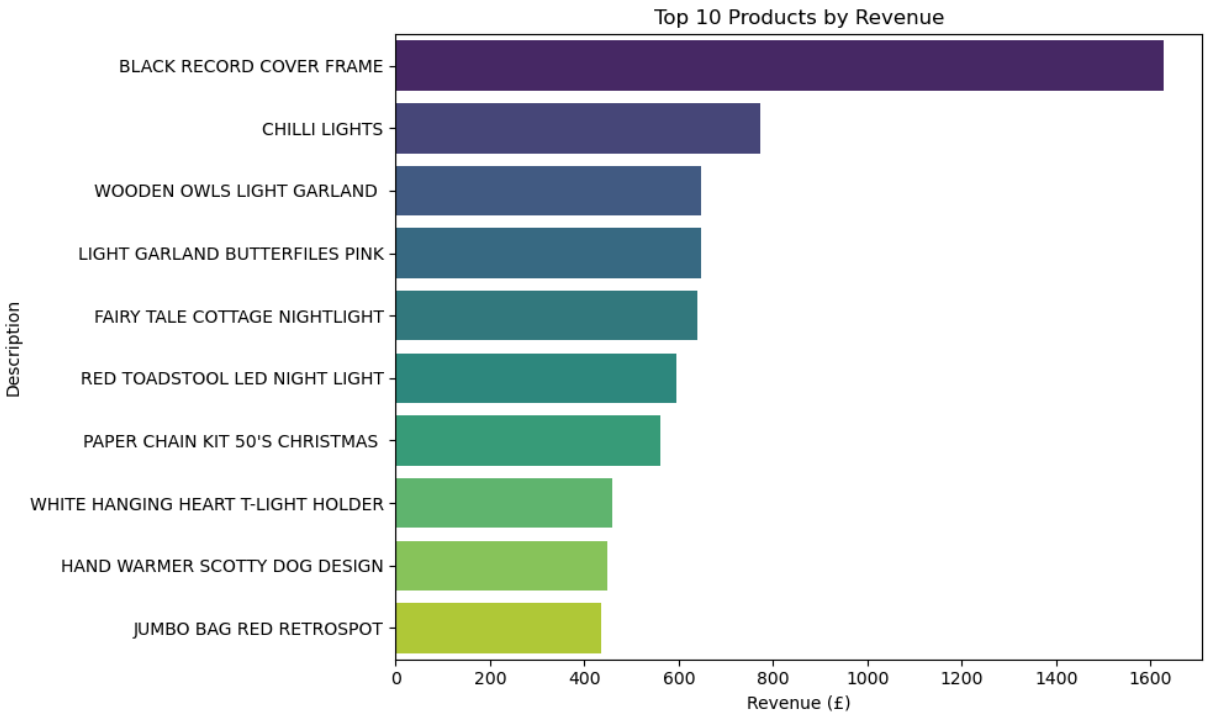
In [3]:
```
# Load Data
df = pd.read_excel(r"C:\Users\mukki\OneDrive\Desktop\Online Retail.xlsx", sheet_nam
df.dropna(subset=['CustomerID'], inplace=True)
df = df[(df['Quantity'] > 0) & (df['UnitPrice'] > 0)]
df['TotalSales'] = df['Quantity'] * df['UnitPrice']
df['InvoiceDate'] = pd.to_datetime(df['InvoiceDate'])
```

In [5]:
```
# Basic KPIs
total_sales = df['TotalSales'].sum()
unique_customers = df['CustomerID'].nunique()
top_country = df['Country'].value_counts().idxmax()
print(f"Total Sales: £{total_sales:,.2f}")
print(f"Unique Customers: {unique_customers}")
print(f"Top Country: {top_country}")
```

```
Total Sales: £32,618.26
Unique Customers: 68
Top Country: United Kingdom
```

In [7]:
```
daily_sales = df.resample('D', on='InvoiceDate')['TotalSales'].sum().reset_index()
fig = px.line(daily_sales, x='InvoiceDate', y='TotalSales', title='Daily Sales Tren
fig.show()
```

In [9]:
```
top_products = df.groupby('Description')['TotalSales'].sum().sort_values(ascending=
plt.figure(figsize=(10, 6))
sns.barplot(x=top_products.values, y=top_products.index, palette='viridis')
plt.title('Top 10 Products by Revenue')
plt.xlabel("Revenue (£)")
plt.tight_layout()
plt.show()
```



In [11]:

```
rfm = df.groupby('CustomerID').agg({
    'InvoiceDate': lambda x: (df['InvoiceDate'].max() - x.max()).days,
    'InvoiceNo': 'nunique',
    'TotalSales': 'sum'
})
rfm.columns = ['Recency', 'Frequency', 'Monetary']
scaler = StandardScaler()
rfm_scaled = scaler.fit_transform(rfm)

# KMeans clustering
kmeans = KMeans(n_clusters=4, random_state=42)
rfm['Cluster'] = kmeans.fit_predict(rfm_scaled)

# PCA for 2D plot
pca = PCA(n_components=2)
rfm['PCA1'], rfm['PCA2'] = zip(*pca.fit_transform(rfm_scaled))

px.scatter(rfm, x='PCA1', y='PCA2', color=rfm['Cluster'].astype(str),
           title="Customer Segments (PCA View)", hover_data=rfm.columns).show()
```

In [ ]: