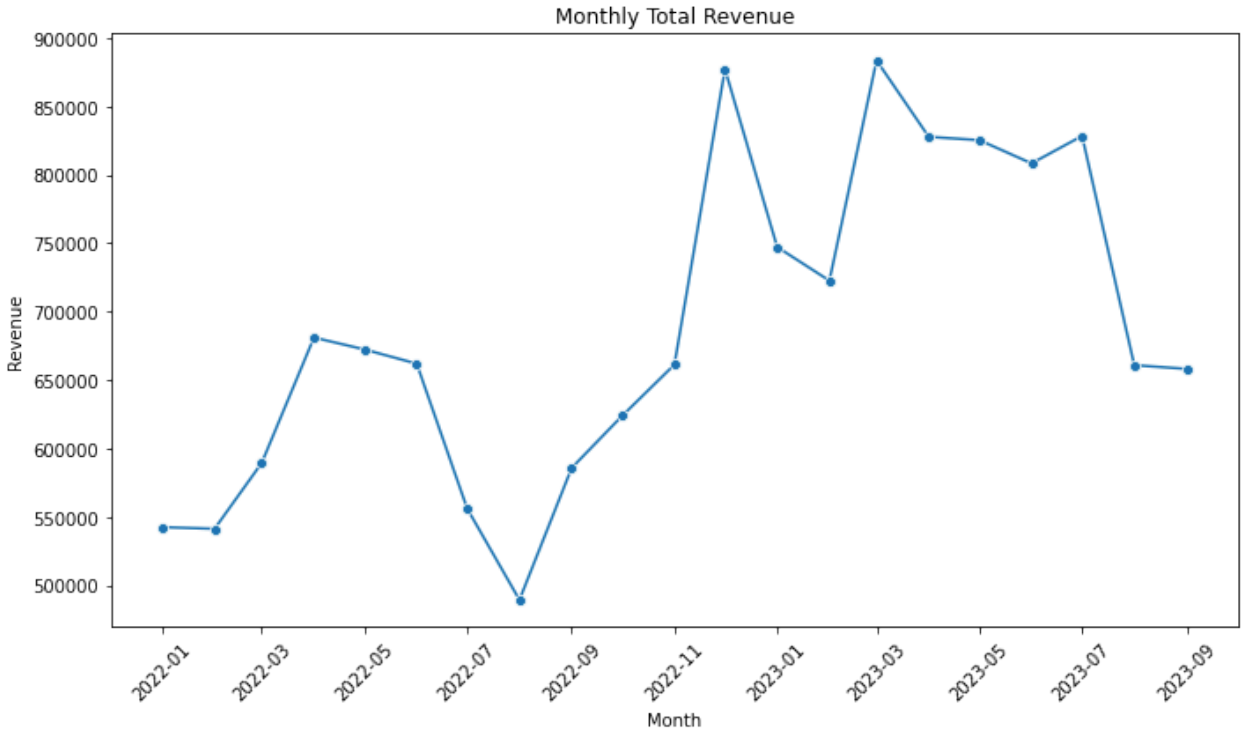
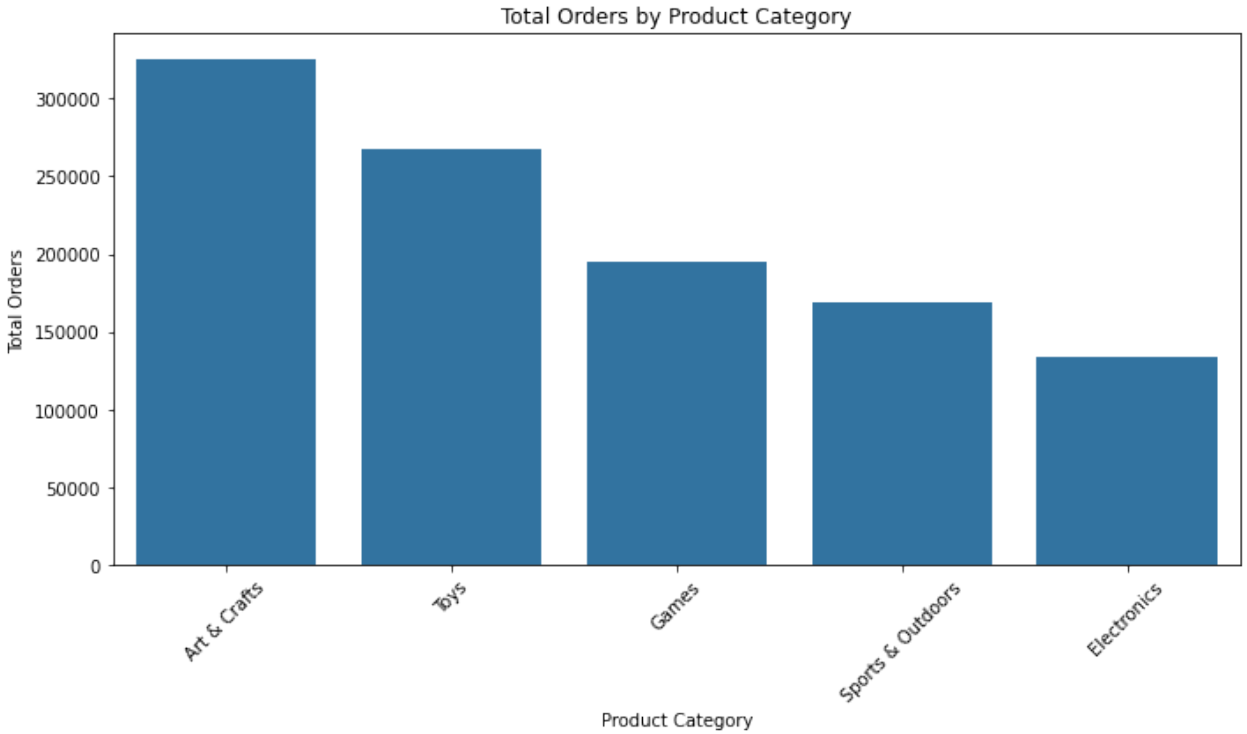
In [1]:

```
# Toy Store KPI Report Script
# Objective: Build an interactive KPI report for Maven Toys to track sales, revenue, and profit

import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load datasets
calendar_df = pd.read_csv('calendar.csv')
sales_df = pd.read_csv('sales.csv')
inventory_df = pd.read_csv('inventory.csv')
stores_df = pd.read_csv('stores.csv')
products_df = pd.read_csv('products.csv')

# Data Cleaning
# Convert 'Product_Cost' and 'Product_Price' to numeric
products_df['Product_Cost'] = products_df['Product_Cost'].replace('[\$,]', '', regex=True).astype(float)
products_df['Product_Price'] = products_df['Product_Price'].replace('[\$,]', '', regex=True).astype(float)

# Convert 'Date' in sales_df and calendar_df to datetime
sales_df['Date'] = pd.to_datetime(sales_df['Date'])
calendar_df['Date'] = pd.to_datetime(calendar_df['Date'])

# Add 'Start of Month' and 'Start of Week' to calendar_df
calendar_df['Start_of_Month'] = calendar_df['Date'].values.astype('datetime64[M]')
calendar_df['Start_of_Week'] = calendar_df['Date'] - pd.to_timedelta(calendar_df['Date'].dt.weekday, unit='D')

# Merge sales with products to calculate Revenue and Profit
sales_df = sales_df.merge(products_df, on='Product_ID', how='left')
sales_df['Revenue'] = sales_df['Units'] * sales_df['Product_Price']
sales_df['Profit'] = sales_df['Revenue'] - (sales_df['Units'] * sales_df['Product_Cost'])

# KPI Metrics
total_orders = sales_df['Sale_ID'].nunique()
total_revenue = sales_df['Revenue'].sum()
total_profit = sales_df['Profit'].sum()

# Monthly Aggregation
monthly_metrics = sales_df.groupby(sales_df['Date'].dt.to_period('M')).agg({
    'Sale_ID': 'nunique',
    'Revenue': 'sum',
    'Profit': 'sum'
}).reset_index().rename(columns={'Sale_ID': 'Total_Orders'})

# Convert Period to datetime for plotting
monthly_metrics['Date'] = monthly_metrics['Date'].dt.to_timestamp()

# Visualization
```

```python
plt.figure(figsize=(10, 6))
sns.lineplot(data=monthly_metrics, x='Date', y='Revenue', marker='o')
plt.title('Monthly Total Revenue')
plt.xlabel('Month')
plt.ylabel('Revenue')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Orders by Product Category
category_orders = sales_df.groupby('Product_Category')['Units'].sum().sort_values(ascending=False)
plt.figure(figsize=(10, 6))
sns.barplot(x=category_orders.index, y=category_orders.values)
plt.title('Total Orders by Product Category')
plt.xlabel('Product Category')
plt.ylabel('Total Orders')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Calculate Revenue for Arts & Crafts in February 2023
arts_crafts_feb_revenue = sales_df[(sales_df['Product_Category'] == 'Art & Crafts') &
                                   (sales_df['Date'].dt.month == 2) &
                                   (sales_df['Date'].dt.year == 2023)]['Revenue'].sum()

print(f"Total Revenue for Arts & Crafts in February 2023: {arts_crafts_feb_revenue}")
```


Monthly Total Revenue
Revenue
900000
850000
800000
750000
700000
650000
600000
550000
500000
2022-01
2022-03
2022-05
2022-07
2022-09
2022-11
2023-01
2023-03
2023-05
2023-07
2023-09
Month



Total Revenue for Arts & Crafts in February 2023: 187713.09999999998