SuperFoodsMax

December 14, 2025

In [29]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
```

In [30]:
```
# File paths
data_path = "dataset_2019_2022.csv"
output_dir = "outputs"
```

In [31]:
```
# Create output folder if it does not exist
import os
os.makedirs(output_dir, exist_ok=True)
```

In [32]:
```
print("Setup complete")
```

```
Setup complete
```

In [33]:
```
# Load the CSV file
df = pd.read_csv(data_path)

#Quick check of the data
df.head()
```

Out[33]:

	customer_id	product_id	basket_id	loyalty	household_type	age_band	department	brand	commodity	store	price
0	15803	1131974	57266	Loyalist	1 adult with kids	19-24	Grocery	private	Baked bread/buns/rolls	374	0.99
1	15803	1051516	57266	Loyalist	1 adult with kids	19-24	Produce	national	Vegetables - all others	374	0.70
2	15803	967254	57266	Loyalist	1 adult with kids	19-24	Pharmaceutical	national	Cold and flu	374	1.68
3	15803	1134222	57266	Loyalist	1 adult with kids	19-24	Grocery	private	Paper housewares	374	2.59
4	15803	1003421	57266	Loyalist	1 adult with kids	19-24	Grocery	national	Soup	374	0.60

```
  transaction_date
0        5/10/2020
1       24/10/2020
2       18/10/2020
3       23/10/2020
4       27/10/2020
```

In [34]:
```
# Convert transaction date to datetime format
df["transaction_date"] = pd.to_datetime(
    df["transaction_date"],
    dayfirst=True,
    errors="coerce"
)
```

In [35]:
```
# Convert transaction date to datetime format
df["transaction_date"] = pd.to_datetime(
    df["transaction_date"],
    dayfirst=True,
    errors="coerce"
)

# Make sure price is numeric
df["price"] = pd.to_numeric(df["price"], errors="coerce")

# Remove rows with missing key values
df = df.dropna(
    subset=["customer_id", "basket_id", "transaction_date", "price", "loyalty"]
)

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 77750 entries, 0 to 77749
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   customer_id       77750 non-null  int64
 1   product_id        77750 non-null  int64
 2   basket_id         77750 non-null  int64
 3   loyalty           77750 non-null  object
 4   household_type    77750 non-null  object
 5   age_band          77750 non-null  object
 6   department        77750 non-null  object
 7   brand             77750 non-null  object
 8   commodity         77750 non-null  object
 9   store             77750 non-null  int64
 10  price             77750 non-null  float64
 11  transaction_date  77750 non-null  datetime64[ns]
```

```
dtypes: datetime64[ns](1), float64(1), int64(4), object(6)
memory usage: 7.1+ MB
```

```
In [36]: # Combine line items into one order per basket
         orders = (
             df.groupby(
                 ["customer_id", "basket_id", "transaction_date"],
                 as_index=False
             )
             .agg(
                 order_value=("price", "sum"),
                 items=("product_id", "count"),
                 loyalty=("loyalty", lambda x: x.mode().iloc[0])
             )
         )

         # Create a month column for trend analysis
         orders["year_month"] = orders["transaction_date"].dt.to_period("M").dt.to_timestamp()

         orders.head()
```

Out[36]:

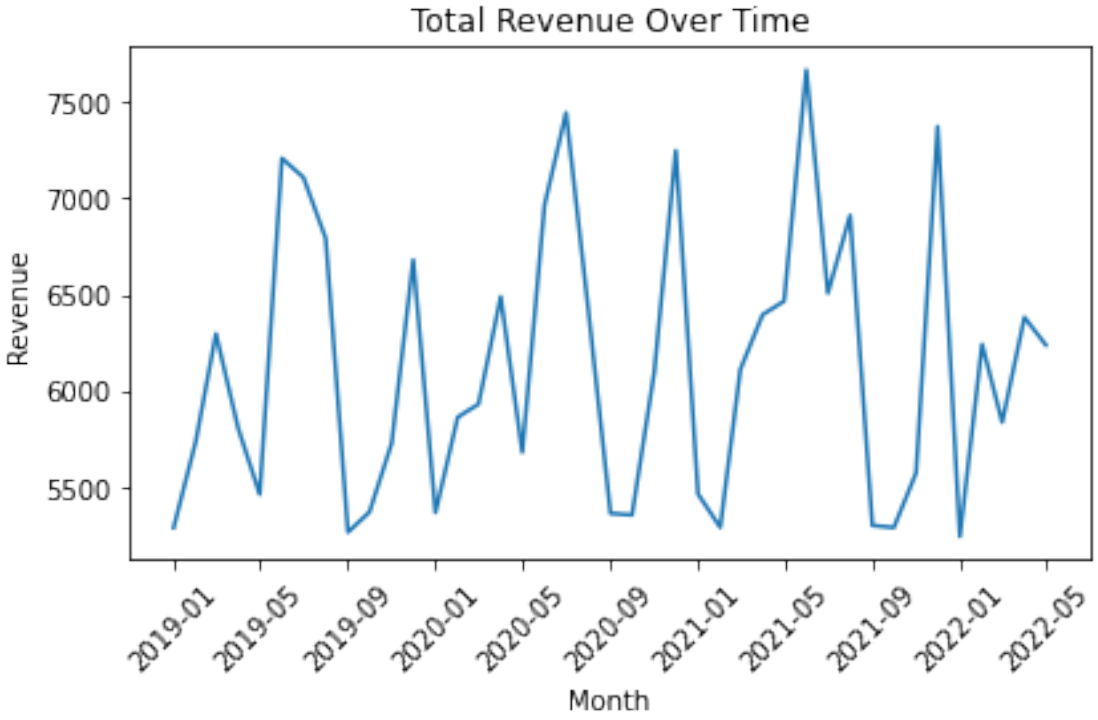
	customer_id	basket_id	transaction_date	order_value	items	loyalty	year_month
0	15803	57266	2020-10-01	1.99	1	Loyalist	2020-10-01
1	15803	57266	2020-10-02	5.00	1	Loyalist	2020-10-01
2	15803	57266	2020-10-04	3.98	1	Loyalist	2020-10-01
3	15803	57266	2020-10-05	0.99	1	Loyalist	2020-10-01
4	15803	57266	2020-10-07	3.99	1	Loyalist	2020-10-01

```
In [37]: print("Number of line items:", df.shape)
         print("Number of orders:", orders.shape)

         orders["loyalty"].value_counts()
```

```
Number of line items: (77750, 12)
Number of orders: (55058, 7)
```

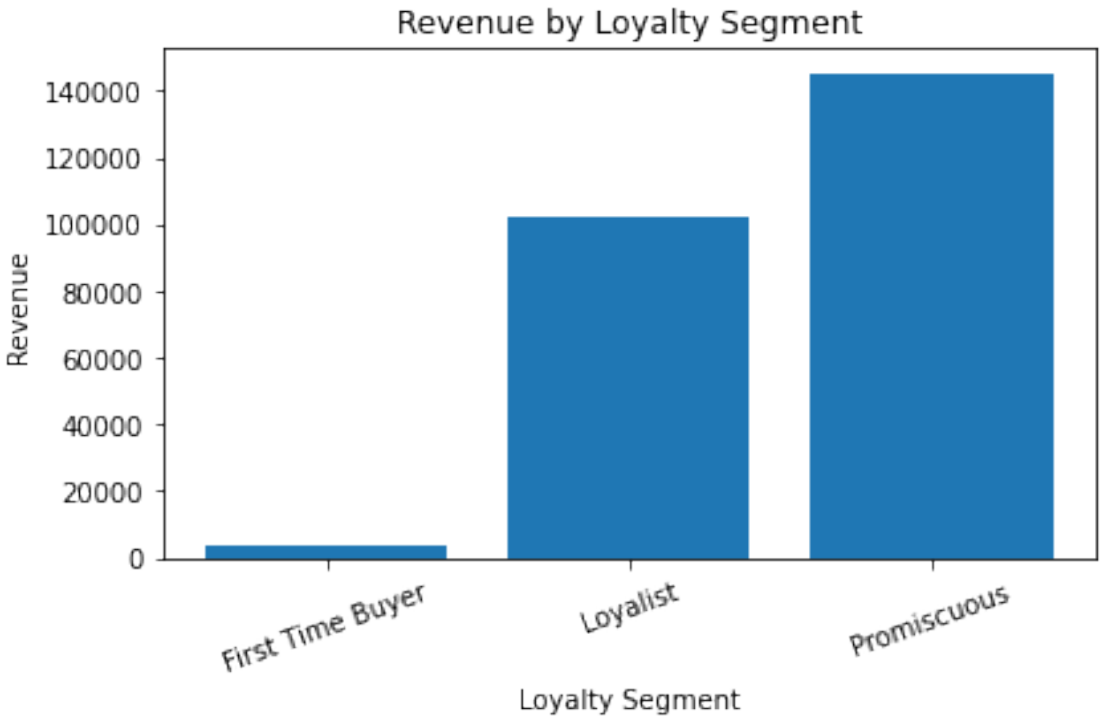
```
Out[37]: Promiscuous         31859
         Loyalist            22390
         First Time Buyer      809
         Name: loyalty, dtype: int64
```

In [38]:
```
monthly_revenue = (
    orders.groupby("year_month", as_index=False)
    .agg(revenue=("order_value", "sum"))
)

plt.figure()
plt.plot(monthly_revenue["year_month"], monthly_revenue["revenue"])
plt.title("Total Revenue Over Time")
plt.xlabel("Month")
plt.ylabel("Revenue")
plt.xticks(rotation=45)
plt.tight_layout()

plt.savefig(os.path.join(output_dir, "chart_1_revenue_trend.png"))
plt.show()
```



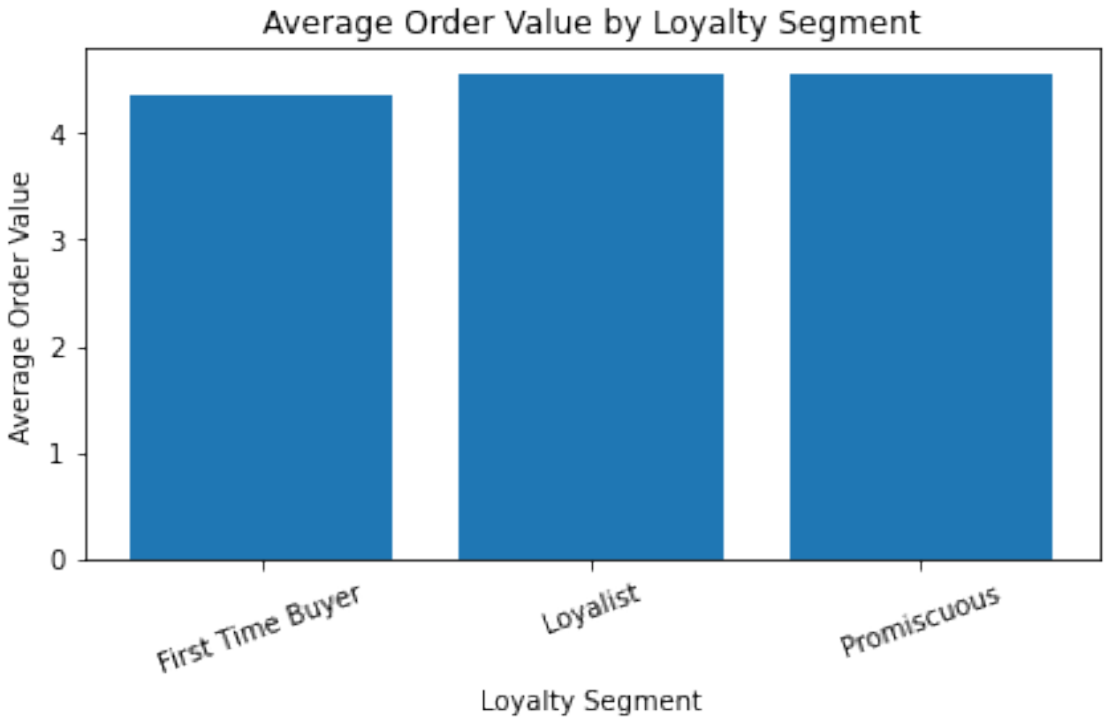
In [39]:
```
revenue_by_loyalty = (
    orders.groupby("loyalty", as_index=False)
    .agg(revenue=("order_value", "sum"))
)

plt.figure()
plt.bar(revenue_by_loyalty["loyalty"], revenue_by_loyalty["revenue"])
plt.title("Revenue by Loyalty Segment")
```

```
plt.xlabel("Loyalty Segment")
plt.ylabel("Revenue")
plt.xticks(rotation=20)
plt.tight_layout()

plt.savefig(os.path.join(output_dir, "chart_2_revenue_by_loyalty.png"))
plt.show()
```



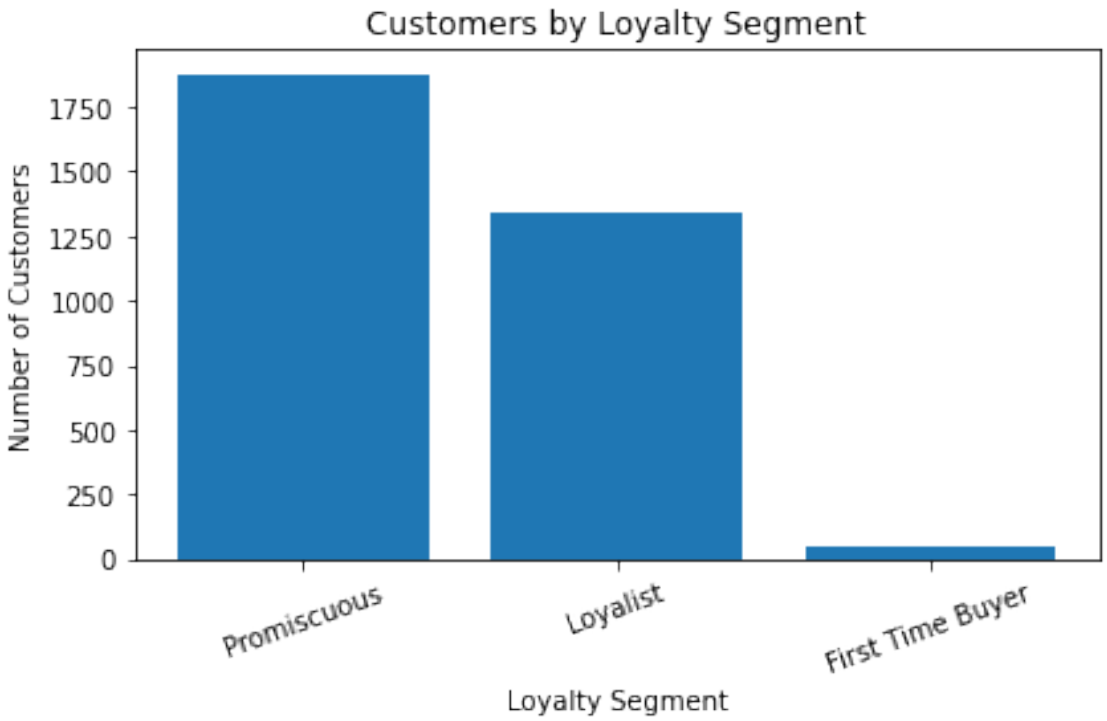
In [40]:
```
average_order_value = (
    orders.groupby("loyalty", as_index=False)
    .agg(avg_order_value=("order_value", "mean"))
)

plt.figure()
plt.bar(
    average_order_value["loyalty"],
    average_order_value["avg_order_value"]
)
plt.title("Average Order Value by Loyalty Segment")
plt.xlabel("Loyalty Segment")
plt.ylabel("Average Order Value")
plt.xticks(rotation=20)
plt.tight_layout()
```

```
plt.savefig(os.path.join(output_dir, "chart_3_aov_by_loyalty.png"))
plt.show()
```



In [41]:
```
# Assign each customer a loyalty segment
customer_segments = (
    orders.groupby("customer_id")["loyalty"]
    .agg(lambda x: x.mode().iloc[0])
    .reset_index(name="loyalty_segment")
)

customer_counts = (
    customer_segments["loyalty_segment"]
    .value_counts()
    .reset_index()
)

customer_counts.columns = ["loyalty_segment", "customers"]

plt.figure()
plt.bar(
    customer_counts["loyalty_segment"],
    customer_counts["customers"]
)
plt.title("Customers by Loyalty Segment")
```

```
plt.xlabel("Loyalty Segment")
plt.ylabel("Number of Customers")
plt.xticks(rotation=20)
plt.tight_layout()

plt.savefig(os.path.join(output_dir, "chart_4_customers_by_loyalty.png"))
plt.show()
```



In [42]:
```
print("Charts saved in folder:", output_dir)
os.listdir(output_dir)
```

```
Charts saved in folder: outputs
```

Out[42]:
```
['chart_1_revenue_trend.png',
 'chart_2_revenue_by_loyalty.png',
 'chart_4_customers_by_loyalty.png',
 'chart_3_aov_by_loyalty.png']
```

In [ ]: