In [124…

```
# importing libraries
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import pandas as pd
import numpy as np
import sqlite3
```

In [126…

```
# Define the file path for the excel
file_path = r"D:\toki\testings\python\new\fashion(p).xlsx"
df = pd.read_excel(file_path)
```

In [127…

```
# Checking if the data frame works
print(df.head())
print(df.info())
```

```
     Brand_ID Brand_Name    Country  Year Sustainability_Rating  \
0  BRAND-0001    Brand_1  Australia  2018                     D
1  BRAND-0001    Brand_1  Australia  2018                     D
2  BRAND-0002    Brand_2      Japan  2015                     D
3  BRAND-0003    Brand_3        USA  2024                     A
4  BRAND-0004    Brand_4      Italy  2023                     D

   Material_Type Eco_Friendly_Manufacturing  Carbon_Footprint_MT  \
0         Tencel                        No                 1.75
1         Tencel                        No                 1.75
2  Vegan Leather                       Yes               124.39
3  Vegan Leather                        No               336.66
4  Bamboo Fabric                        No               152.04

   Water_Usage_Liters  Waste_Production_KG Recycling_Programs  Product_Lines  \
0          4511152.79             97844.11                 No              2
1          4511152.79             97844.11                 No              2
2          1951566.31             37267.75                 No             15
3           467454.52                  NaN                 No              2
4           899576.90             32665.45                 No             13

   Average_Price_USD Market_Trend Certifications
0              38.33      Growing           GOTS
1              38.33      Growing           GOTS
2             250.07      Growing           GOTS
3             146.16      Growing         B Corp
4             165.52       Stable       OEKO-TEX
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5001 entries, 0 to 5000
Data columns (total 15 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Brand_ID                    5000 non-null   object
 1   Brand_Name                  5000 non-null   object
 2   Country                     5001 non-null   object
 3   Year                        5001 non-null   int64
 4   Sustainability_Rating       5001 non-null   object
 5   Material_Type               5001 non-null   object
 6   Eco_Friendly_Manufacturing  5001 non-null   object
 7   Carbon_Footprint_MT         4998 non-null   float64
 8   Water_Usage_Liters          4999 non-null   float64
 9   Waste_Production_KG         5000 non-null   float64
 10  Recycling_Programs          5001 non-null   object
 11  Product_Lines               5001 non-null   int64
 12  Average_Price_USD           4998 non-null   float64
 13  Market_Trend                5001 non-null   object
 14  Certifications              3997 non-null   object
dtypes: float64(4), int64(2), object(9)
memory usage: 586.2+ KB
None
```

In [128…

```
# Checking for missing values
print("Missing values in each column:\n", df.isnull().sum())
```

```
Missing values in each column:
 Brand_ID                        1
Brand_Name                       1
Country                          0
Year                             0
Sustainability_Rating            0
Material_Type                    0
Eco_Friendly_Manufacturing       0
Carbon_Footprint_MT              3
Water_Usage_Liters               2
Waste_Production_KG              1
Recycling_Programs               0
Product_Lines                    0
Average_Price_USD                3
Market_Trend                     0
Certifications                1004
dtype: int64
```

In [129…
```python
# Removing duplicates
df.drop_duplicates(inplace=True)
```

In [130…
```python
# Removing extra column
df.drop(columns=['Brand_Name'], inplace=True)
```

In [131…
```python
# Filling in missing values with error handling
columns_to_fill = {
    'Carbon_Footprint_MT': df['Carbon_Footprint_MT'].mean(),
    'Waste_Production_KG': df['Waste_Production_KG'].median(),
    'Water_Usage_Liters': df['Water_Usage_Liters'].median(),
    'Average_Price_USD': df['Average_Price_USD'].mean(),
    'Certifications': "Not-Certified"
}

for column, value in columns_to_fill.items():
    if column in df.columns:
        df[column].fillna(value, inplace=True)
```

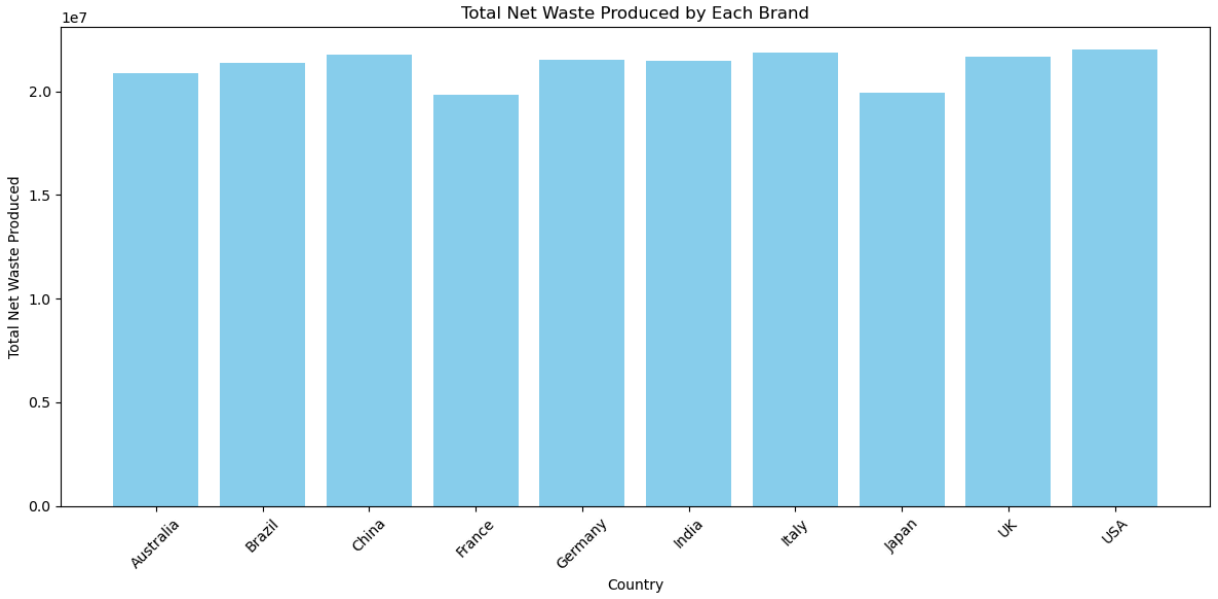
In [132…
```python
# Generating missing Brand_ID's
df['Brand_ID'] = df['Brand_ID'].fillna(df.apply(lambda row: f"BRAND-{str(row.name).
```

In [133…
```python
# Feature engineering #1: Net waste produced
recycle_est = 0.3
df['RecPrgBIN'] = df['Recycling_Programs'].map({'Yes': 1, 'No': 0})
df['Net_WastePD'] = df['Waste_Production_KG'] * (1 - df['RecPrgBIN'] * recycle_est)
```

In [134…
```python
# Grouping Country with the Net Waste in a new variable
waste_summary = df.groupby('Country')['Net_WastePD'].sum().reset_index()
```

In [135…
```python
# Bar Chart
plt.figure(figsize=(12, 6))
plt.bar(waste_summary['Country'], waste_summary['Net_WastePD'], color='skyblue')
plt.xlabel('Country')
plt.ylabel('Total Net Waste Produced')
plt.title('Total Net Waste Produced by Each Brand')
```

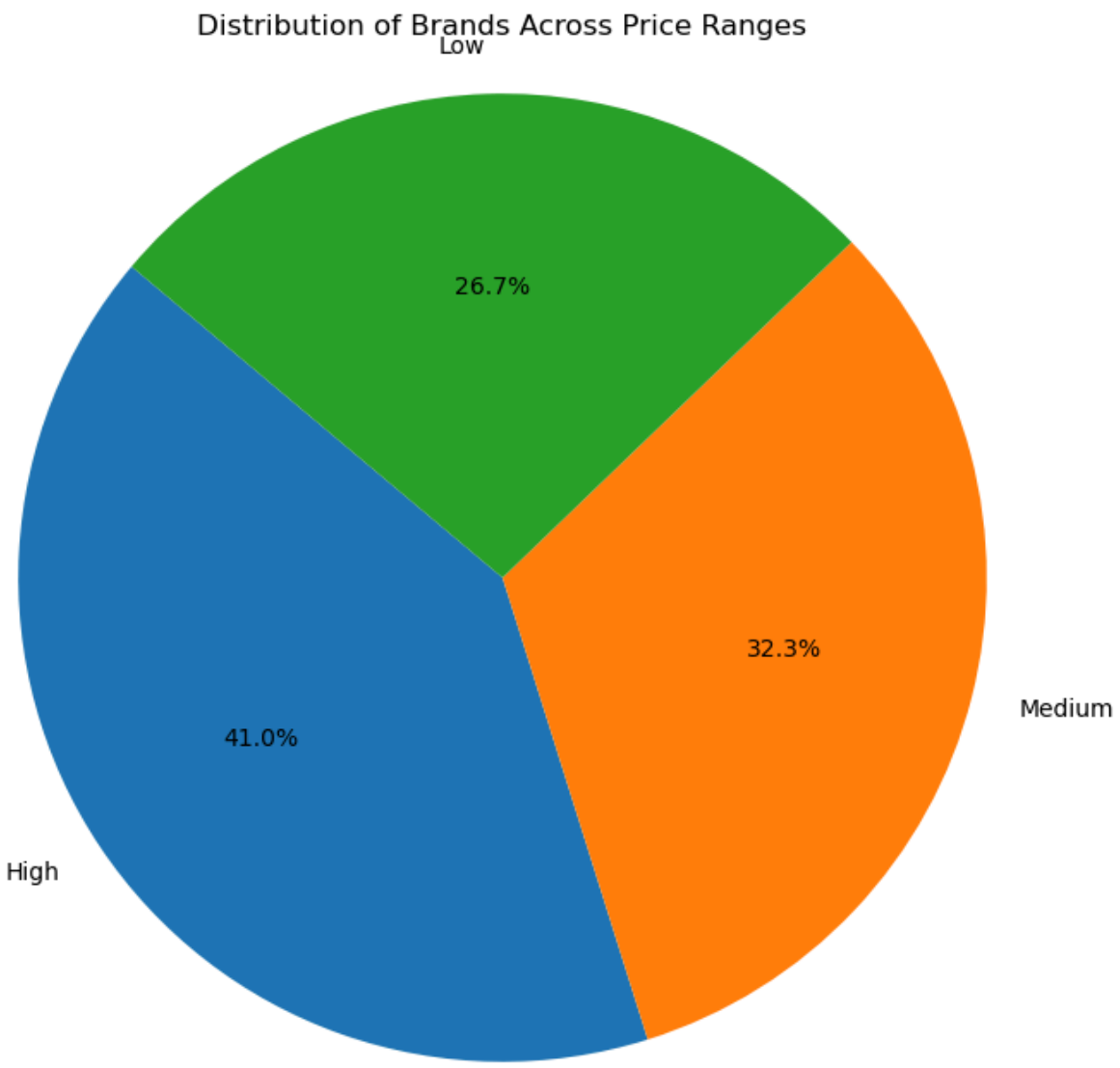
```
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [137…
```
# Feature engineering #2: Price ranges
price_bins = [20, 150, 300, 500]
price_labels = ['Low', 'Medium', 'High']
df['Price_Range'] = pd.cut(df['Average_Price_USD'], bins=price_bins, labels=price_l
```

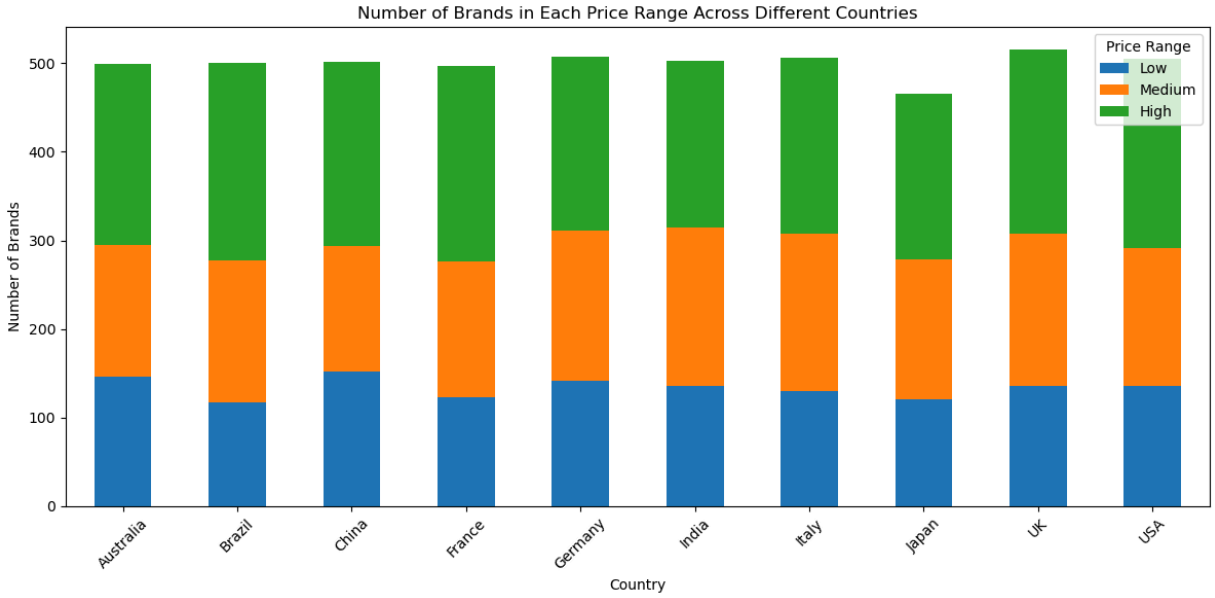
In [145…
```
# Counting NO. of Brands in each Price Range
price_range_counts = df['Price_Range'].value_counts()
```

In [147…
```
# Pie Chart

plt.figure(figsize=(8, 8))
plt.pie(price_range_counts, labels=price_range_counts.index, autopct='%1.1f%%', sta
plt.title('Distribution of Brands Across Price Ranges')
plt.axis('equal')
plt.show()
```



In [148…
```python
# Grouping by Country and Price Range
country_price_range_counts = df.groupby(['Country', 'Price_Range'], observed=False)
```

In [151…
```python
# Stacked Bar Chart

country_price_range_counts.plot(kind='bar', stacked=True, figsize=(12, 6))
plt.title('Number of Brands in Each Price Range Across Different Countries')
plt.xlabel('Country')
plt.ylabel('Number of Brands')
plt.xticks(rotation=45)
plt.legend(title='Price Range')
plt.tight_layout()
plt.show()
```

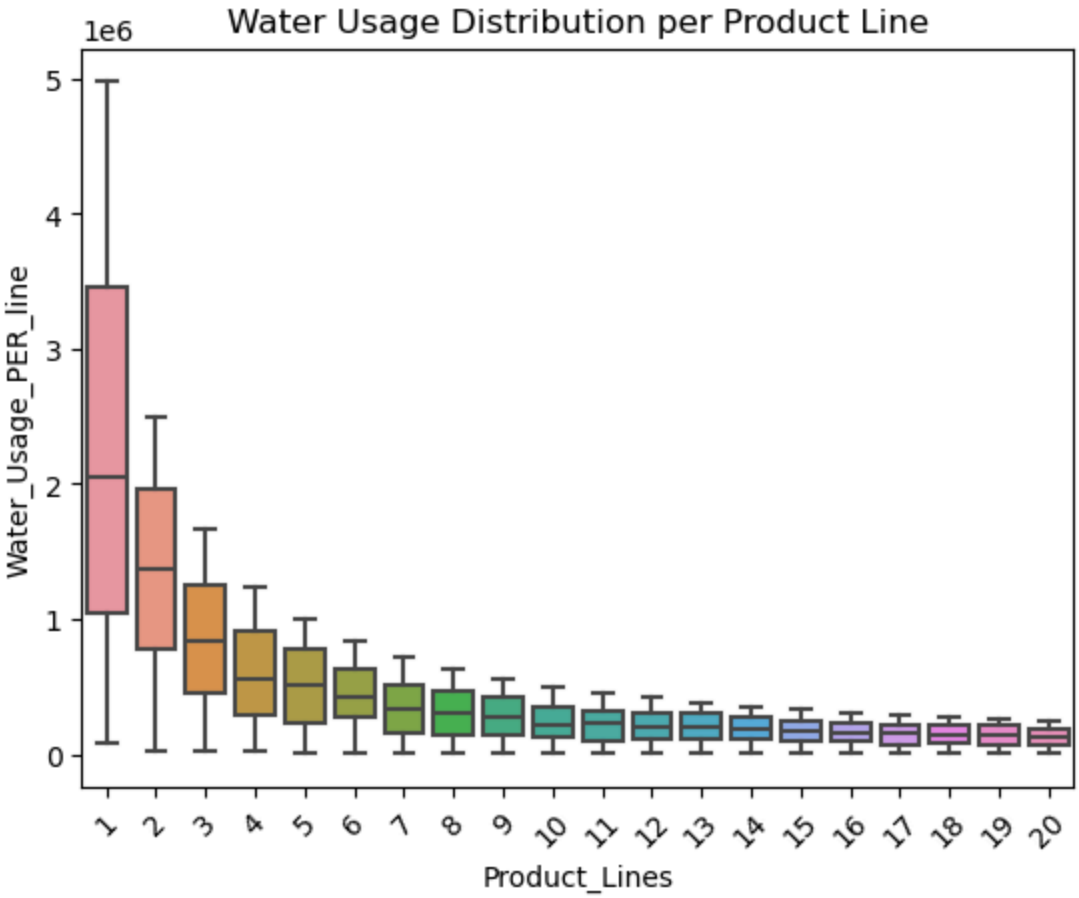


In [152…

```
# Feature engineering #3: Water usage per product line & water usage per dollar
df['Water_Usage_PER_line'] = df['Water_Usage_Liters'] / df['Product_Lines']
df['Water_Usage_PER_dollar'] = df['Water_Usage_Liters'] / df['Average_Price_USD']
```

In [154…

```
# Box Plot

sns.boxplot(x='Product_Lines', y='Water_Usage_PER_line', data=df)
plt.title('Water Usage Distribution per Product Line')
plt.xticks(rotation=45)
plt.show()
```



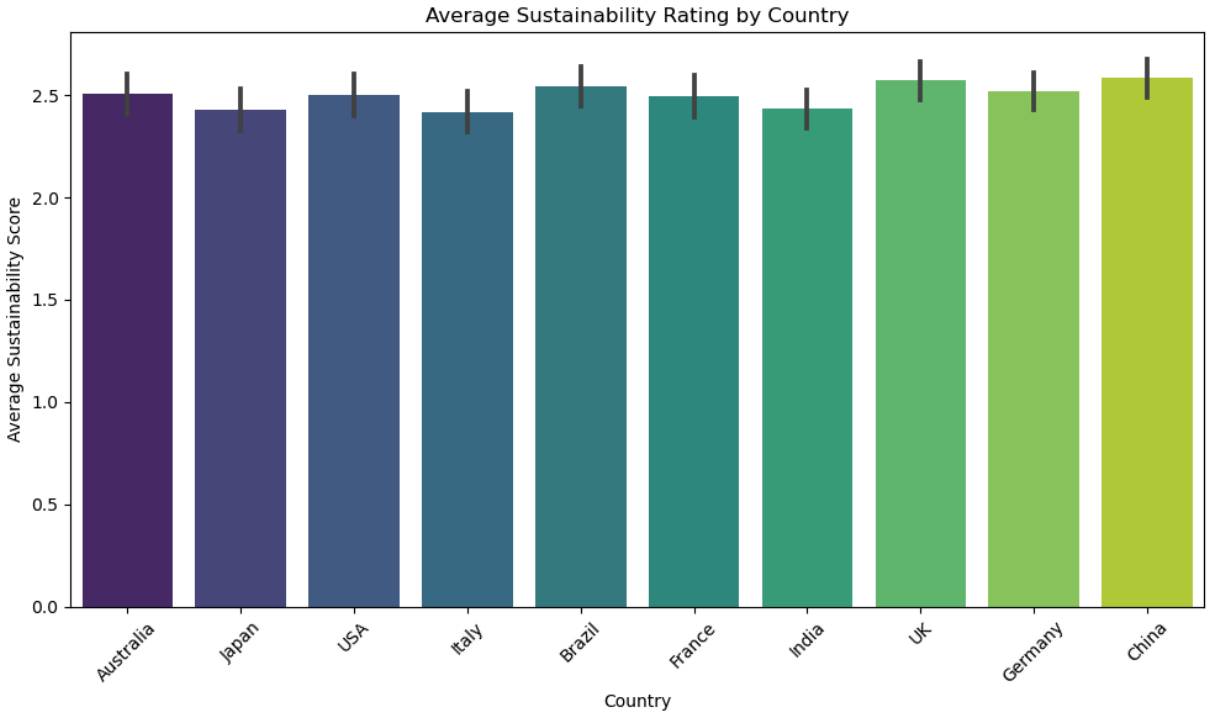
In [178…
```
# Round all relevant numeric columns to 2 decimal places
numeric_cols = ['Carbon_Footprint_MT', 'Waste_Production_KG', 'Water_Usage_Liters',
                'Average_Price_USD', 'Net_WastePD', 'Water_Usage_PER_line', 'Water_
df[numeric_cols] = df[numeric_cols].round(2)
```

In [160…
```
# Exploratory analysis #1: Average sustainability ratings by country
sus_map = {'A': 1, 'B': 2, 'C': 3, 'D': 4}
df['Sustain_Score'] = df['Sustainability_Rating'].map(sus_map)
average_ratings = df.groupby('Country')['Sustain_Score'].mean().reset_index()
sorted_ratings = average_ratings.sort_values(by='Sustain_Score', ascending=False)
print("Average Sustainability Ratings by Country:\n", sorted_ratings)
```

```
Average Sustainability Ratings by Country:
      Country  Sustain_Score
2       China       2.583665
8          UK       2.574757
1      Brazil       2.544000
4     Germany       2.520710
0   Australia       2.507014
9         USA       2.502970
3      France       2.492958
5       India       2.435388
7       Japan       2.431330
6       Italy       2.418972
```

In [161…
```
# Bar Chart
plt.figure(figsize=(10, 6))
sns.barplot(x='Country', y='Sustain_Score', data=df, palette='viridis')
```

```
plt.title('Average Sustainability Rating by Country')
plt.ylabel('Average Sustainability Score')
plt.xlabel('Country')
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



In [163…
```
# Exploratory analysis #2: Common materials used by brands with high sustainability
df_pro = df[df['Sustain_Score'].isin([1, 2])]
material_groups = df_pro.groupby('Material_Type')['Brand_ID'].count().reset_index(n
sorted_materials = material_groups.sort_values(by='Brand Count', ascending=False)
print("Common Materials Used by High Sustainability Brands:\n", sorted_materials)
```

```
Common Materials Used by High Sustainability Brands:
         Material_Type  Brand Count
3  Recycled Polyester          438
1                Hemp          431
0       Bamboo Fabric          419
2      Organic Cotton          417
5       Vegan Leather          410
4              Tencel          394
```

In [166…
```
# Grouping high-end Brands by Material Type to count there use
material_count = df_pro.groupby('Material_Type')['Brand_ID'].count().reset_index(na
```

In [167…
```
# Tree Map
fig = px.treemap(material_count,
                 path=['Material_Type'],
                 values='Brand Count',
                 title="Common Materials Used by High-Rated Brands",
                 color='Brand Count',
                 color_continuous_scale='Blues')

fig.show()
```

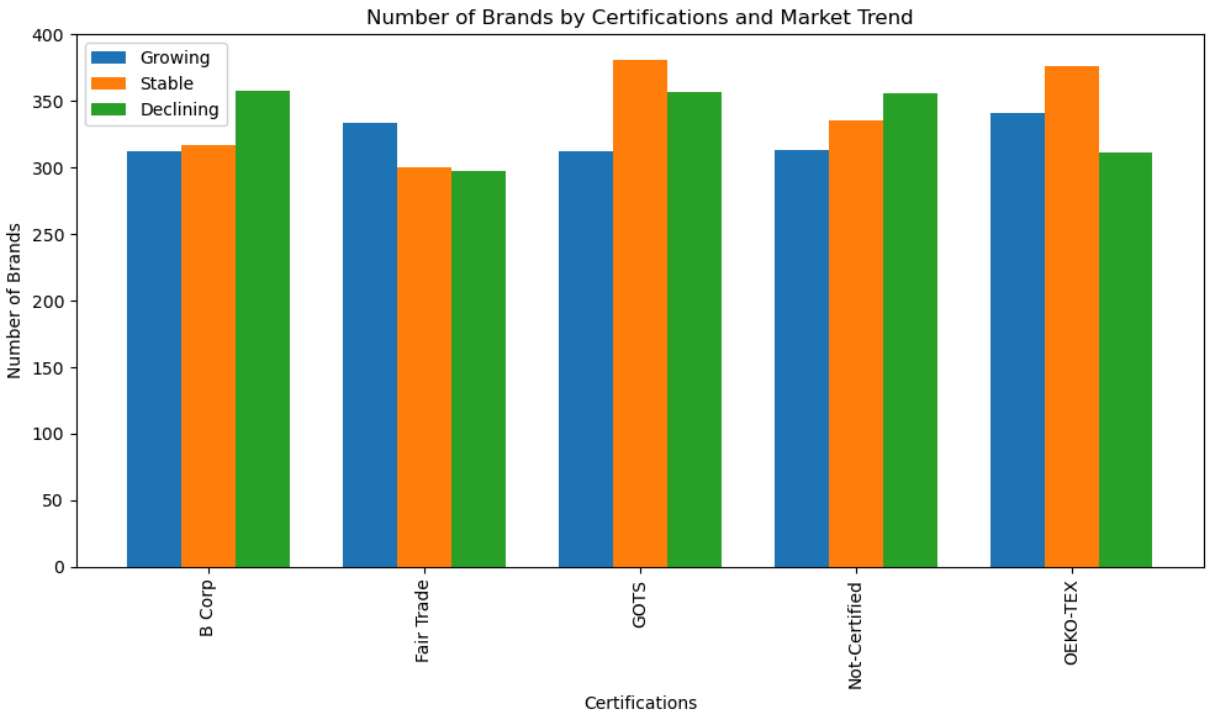
In [168…

```
# Exploratory analysis #3: Which certifications are most common among brands with g
trend_map = {'Growing': 1.5, 'Stable': 1, 'Declining': -0.5}
df['Trend_Map'] = df['Market_Trend'].map(trend_map)
df_grow = df[df['Trend_Map'] == 1.5]
df_stable = df[df['Trend_Map'] == 1]
df_decline = df[df['Trend_Map'] == -0.5]
certified_stableBrands = df_stable.groupby('Certifications')['Brand_ID'].count()
certified_growingBrands = df_grow.groupby('Certifications')['Brand_ID'].count()
certified_declineBrands = df_decline.groupby('Certifications')['Brand_ID'].count()
print('Growing // Certificate Count')
print(certified_growingBrands)
print('Stable // Certificate Count')
print(certified_stableBrands)
print('Decline // Certificate Count')
print(certified_declineBrands)
```

```
Growing // Certificate Count
Certifications
B Corp           312
Fair Trade       334
GOTS             312
Not-Certified    313
OEKO-TEX         341
Name: Brand_ID, dtype: int64
Stable // Certificate Count
Certifications
B Corp           317
Fair Trade       300
GOTS             381
Not-Certified    335
OEKO-TEX         376
Name: Brand_ID, dtype: int64
Decline // Certificate Count
Certifications
B Corp           358
Fair Trade       297
GOTS             357
Not-Certified    356
OEKO-TEX         311
Name: Brand_ID, dtype: int64
```

In [169…

```python
# Merging the data into a single DataFrame for easy plotting
certified_df = pd.DataFrame({
    'Growing': certified_growingBrands,
    'Stable': certified_stableBrands,
    'Declining': certified_declineBrands
}).fillna(0)

# Grouped Bar Chart
certifications = certified_df.index
x = np.arange(len(certifications))  # label locations
width = 0.25  # width of the bars
fig, ax = plt.subplots(figsize=(10, 6))
rects1 = ax.bar(x - width, certified_df['Growing'], width, label='Growing')
rects2 = ax.bar(x, certified_df['Stable'], width, label='Stable')
rects3 = ax.bar(x + width, certified_df['Declining'], width, label='Declining')
ax.set_xlabel('Certifications')
ax.set_ylabel('Number of Brands')
ax.set_title('Number of Brands by Certifications and Market Trend')
ax.set_xticks(x)
ax.set_xticklabels(certifications, rotation=90)
ax.legend()
plt.tight_layout()
plt.show()
```



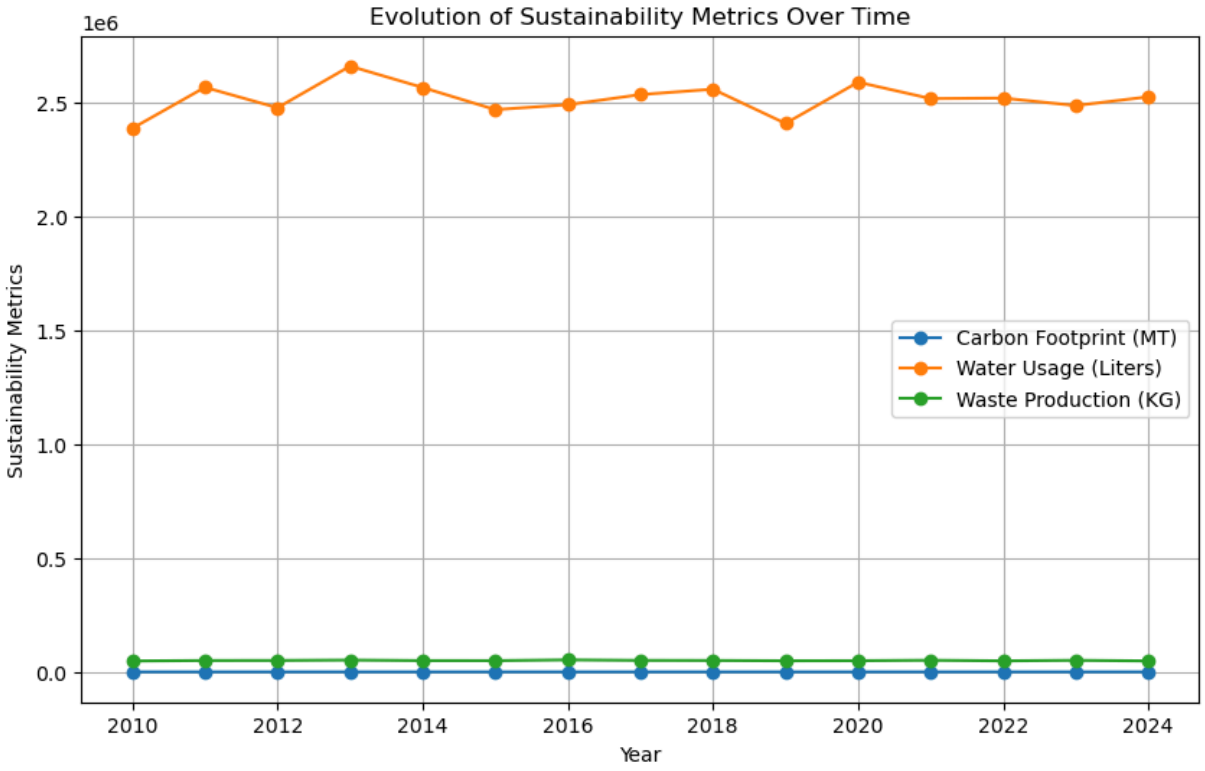
In [170…

```
# Exploratory analysis #4: How have sustainability metrics evolved over time across
sustainability_metrics = ['Year', 'Carbon_Footprint_MT', 'Water_Usage_Liters', 'Was
sustainability_trends = df[sustainability_metrics].groupby('Year').mean().reset_ind
print(sustainability_trends)
```

	Year	Carbon_Footprint_MT	Water_Usage_Liters	Waste_Production_KG
0	2010	252.946400	2.387696e+06	48101.137657
1	2011	239.340173	2.567852e+06	50077.442968
2	2012	258.879286	2.478142e+06	50388.991964
3	2013	242.231550	2.660909e+06	52072.664195
4	2014	255.412387	2.567160e+06	49759.162810
5	2015	247.967710	2.469890e+06	49564.657839
6	2016	263.101724	2.491450e+06	52905.869885
7	2017	246.065702	2.535976e+06	50709.001287
8	2018	253.031183	2.559679e+06	50267.270000
9	2019	245.528079	2.409346e+06	49037.260199
10	2020	251.134524	2.590538e+06	49314.014082
11	2021	250.436534	2.518427e+06	51247.054688
12	2022	246.274955	2.520546e+06	48641.170593
13	2023	244.950781	2.487783e+06	50955.585435
14	2024	255.842066	2.526200e+06	48302.327305

In [184…

```
# Line Plot
plt.figure(figsize=(10, 6))
plt.plot(sustainability_trends['Year'], sustainability_trends['Carbon_Footprint_MT'
plt.plot(sustainability_trends['Year'], sustainability_trends['Water_Usage_Liters']
plt.plot(sustainability_trends['Year'], sustainability_trends['Waste_Production_KG'

plt.xlabel('Year')
plt.ylabel('Sustainability Metrics')
plt.title('Evolution of Sustainability Metrics Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



In [186…

```
# Establishing connection with sqlite3
conn = sqlite3.connect(r"D:\toki\testings\python\new\fashion(p).db")
```

In [188…

```
# Move pandas DataFrame into SQLite
df.to_sql('brands', conn, if_exists='replace', index=False)
```

Out[188…

```
5000
```

In [190…

```
# Query the database to double check
cursor = conn.cursor()
cursor.execute("SELECT * FROM brands LIMIT 5;")
rows = cursor.fetchall()
for row in rows:
    print(row)
```

```
('BRAND-0001', 'Australia', 'D', 'Tencel', 'No', 1.75, 4511152.79, 97844.11, 'No',
2, 38.33, 'Growing', 'GOTS', 0, 97844.11, 'Low', 2255576.4, 117692.48, 4, 1.5)
('BRAND-0002', 'Japan', 'D', 'Vegan Leather', 'Yes', 124.39, 1951566.31, 37267.75,
'No', 15, 250.07, 'Growing', 'GOTS', 0, 37267.75, 'Medium', 130104.42, 7804.08, 4,
1.5)
('BRAND-0003', 'USA', 'A', 'Vegan Leather', 'No', 336.66, 467454.52, 50470.95, 'No',
2, 146.16, 'Growing', 'B Corp', 0, 50470.95, 'Low', 233727.26, 3198.24, 1, 1.5)
('BRAND-0004', 'Italy', 'D', 'Bamboo Fabric', 'No', 152.04, 899576.9, 32665.45, 'N
o', 13, 165.52, 'Stable', 'OEKO-TEX', 0, 32665.45, 'Medium', 69198.22, 5434.85, 4,
1.0)
('BRAND-0005', 'USA', 'D', 'Bamboo Fabric', 'Yes', 415.63, 1809219.9, 37295.47, 'Ye
s', 19, 259.35, 'Stable', 'Fair Trade', 1, 26106.83, 'Medium', 95222.1, 6975.87, 4,
1.0)
```

In [192…

```python
# Save cleaned DataFrame to Excel
df.to_excel(r"D:\toki\testings\python\new\fashion(q).xlsx", index=False)
```

In [198…

```python
# Close the connection
conn.close()
```

In [ ]:

```python
# the end :3
```