# Importing Data

In [1]:

```
import pandas as pd
import matplotlib.pyplot as plt
```

In [ ]:

```
df = pd.read_csv(r'../Orders.csv')
pd.options.display.max_columns = None
df
```

Out[ ]:

|  | Date: | 28/9/2023 | Unnamed: 2 | Unnamed: 3 | Unnamed: 4 | Unnamed: 5 | Unnamed: 6 | Unnai |
|---|---|---|---|---|---|---|---|---|
| **0** | Time | 5:22 PM | NaN | NaN | NaN | NaN | NaN |  |
| **1** | NaN | NaN | NaN | NaN | NaN | NaN | NaN |  |
| **2** | NaN | NaN | NaN | NaN | NaN | NaN | NaN |  |
| **3** | Order ID | Order Date | Country | City | Branch | Lat | Lng | Custo First N |
| **4** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 |  |
| **...** | ... | ... | ... | ... | ... | ... | ... |  |
| **20003** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | ah |
| **20004** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 |  |
| **20005** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | hu |
| **20006** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 |  |
| **20007** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | ah |

20008 rows × 19 columns

---

# Fix Data Columns and Index

In [3]:

```
# you can use skip rows from the import func
df.drop(range(3),inplace=True)
df.reset_index(inplace=True,drop=True)
df.columns = df.iloc[0]
df.reset_index(inplace=True,drop=True)
df.drop(0,inplace=True)
df.reset_index(inplace=True,drop=True)
df.columns = df.columns.str.strip()
df
```

Out[3]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Customer First Name | Cust N |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | lina | alrr |
| **1** | 2 | 1/1/2023 | Saudi Arabia | riyadh | rd01 | 24.7136 | 46.6753 | omar | |
| **2** | 3 | 1/1/2023 | Saudi Arabia | riyadh | rd03 | 24.7743 | 46.7386 | iman | iisı |
| **3** | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | |
| **4** | 4 | 1/1/2023 | United Arab Emirates | abu dhabi | ad01 | 24.4539 | 54.3773 | ahmad | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | ahmad | |
| **20000** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | ali | |
| **20001** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | husayn | s |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 | fatin | ba |
| **20003** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | ahmad | sh |

20004 rows × 19 columns

---

# Explore Data and Handling missing values

In [4]:

```
print("-"*50)
print(df.shape)
print("-"*50)
print(df.columns)
print("-"*50)
df.isna().sum()
```

```
--------------------------------------------------
(20004, 19)
--------------------------------------------------
Index(['Order ID', 'Order Date', 'Country', 'City', 'Branch', 'Lat', 'Lng',
       'Customer First Name', 'Customer Last Name', 'Email', 'Phone Number',
       'Category', 'Sub Category', 'Item', 'SalesPerson ID', 'Quantity',
       'Unit Price', 'Discount', 'Status'],
      dtype='object', name=0)
--------------------------------------------------
```

Out[4]:

```
0
Order ID                  4
Order Date                4
Country                   4
City                      4
Branch                    4
Lat                       4
Lng                       4
Customer First Name       4
Customer Last Name        4
Email                     4
Phone Number              4
Category                  4
Sub Category              4
Item                      4
SalesPerson ID            4
Quantity                  4
Unit Price                4
Discount               1115
Status                    4
dtype: int64
```

In [5]:

```
df.dropna(how="all",inplace=True)
df.isna().sum()
```

Out[5]:

```
0
Order ID                  0
Order Date                0
Country                   0
City                      0
Branch                    0
Lat                       0
Lng                       0
Customer First Name       0
Customer Last Name        0
Email                     0
Phone Number              0
Category                  0
Sub Category              0
Item                      0
SalesPerson ID            0
Quantity                  0
Unit Price                0
Discount               1111
Status                    0
dtype: int64
```

In [6]:

```
df.fillna(0,inplace=True)
df.isna().sum()
```

Out[6]:

```
0
Order ID               0
Order Date             0
Country                0
City                   0
Branch                 0
Lat                    0
Lng                    0
Customer First Name    0
Customer Last Name     0
Email                  0
Phone Number           0
Category               0
Sub Category           0
Item                   0
SalesPerson ID         0
Quantity               0
Unit Price             0
Discount               0
Status                 0
dtype: int64
```

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20000 entries, 0 to 20003
Data columns (total 19 columns):
 #   Column               Non-Null Count  Dtype 
---  ------               --------------  ----- 
 0   Order ID             20000 non-null  object
 1   Order Date           20000 non-null  object
 2   Country              20000 non-null  object
 3   City                 20000 non-null  object
 4   Branch               20000 non-null  object
 5   Lat                  20000 non-null  object
 6   Lng                  20000 non-null  object
 7   Customer First Name  20000 non-null  object
 8   Customer Last Name   20000 non-null  object
 9   Email                20000 non-null  object
 10  Phone Number         20000 non-null  object
 11  Category             20000 non-null  object
 12  Sub Category         20000 non-null  object
 13  Item                 20000 non-null  object
 14  SalesPerson ID       20000 non-null  object
 15  Quantity             20000 non-null  object
 16  Unit Price           20000 non-null  object
 17  Discount             20000 non-null  object
 18  Status               20000 non-null  object
dtypes: object(19)
memory usage: 3.1+ MB
```

In [8]:

```
df['Quantity'] = df['Quantity'].astype(int)
df['Unit Price'] = df['Unit Price'].astype(float)
```

```
df['Discount'] = df['Discount'].astype(float)
```

In [9]:

```
print(df['Status'].unique())
print("-"*50)
print(df['Category'].unique())
print("-"*50)
print(df['Sub Category'].unique())
print("-"*50)
print(df['Item'].unique())
print("-"*50)
```

```
['FALSE' 'TRUE']
--------------------------------------------------
['Tablet' 'Smartphone' 'Digital Camera' 'Headphones' 'Smartwatch' 'Laptop'
 'Smart Speaker' 'VR Headset' 'Fitness Tracker' 'Gaming Console']
--------------------------------------------------
['Apple iPad' 'Samsung Galaxy' 'Panasonic Lumix' 'Samsung Galaxy Tab'
 'Sennheiser HD' 'Garmin Fenix' 'Anker Soundcore' 'HP Envy'
 'Apple HomePod' 'Lenovo Legion' 'Fossil Gen' 'Amazon Fire' 'Lenovo Tab'
 'Oculus Rift' 'Sony Cyber-shot' 'JBL Live' 'Sony WH' 'Apple MacBook'
 'OnePlus' 'Fitbit Versa' 'Fujifilm X Series' 'Samsung Galaxy Tab S'
 'Huawei Watch' 'Bose QuietComfort' 'Xiaomi Mi' 'Pimax Vision' 'ASUS ROG'
 'Sony Alpha' 'Xiaomi Redmi' 'Garmin Vivosmart' 'HP Reverb G2 Omnicept'
 'JBL Link' 'Withings Move' 'Garmin Venu' 'Sony PlayStation' 'Google Nest'
 'Amazon Fire HD' 'Valve Index' 'HTC Vive' 'HP Pavilion' 'Sonos One'
 'Microsoft Surface' 'Nikon Coolpix' 'Amazon Echo' 'Lenovo ThinkPad'
 'Garmin Approach' 'Amazfit GTR' 'Harman Kardon' 'Garmin Forerunner'
 'Apple iPhone' 'Oculus Rift S' 'Microsoft Xbox' 'Google Pixel'
 'Oculus Quest' 'Huawei Band' 'HTC Vive Pro' 'Nintendo Switch' 'Dell XPS'
 'Bose SoundSport' 'Nikon Z Series' 'Fitbit Charge' 'Olympus OM-D'
 'Acer Swift' 'Jabra Elite' 'Apple Watch' 'Retro Console' 'Apple iPad Pro'
 'Canon EOS' 'Samsung Gear' 'Google Nest Hub' 'HP Reverb'
 'Canon EOS R Series' 'Amazon Echo Show' 'Oppo Find' 'Bose Home Speaker'
 'Xiaomi Mi Band' 'Samsung Odyssey' 'Amazfit Bip']
--------------------------------------------------
['  iPad   Pro   12.9" ' 'Galaxy  S21  Ultra  ' 'Panasonic Lumix GH5 '
 ' Galaxy Tab A8 ' ' Sennheiser   HD   450BT  ' ' Garmin   Fenix   6S   '
 'Panasonic Lumix S1H' 'Anker   Soundcore   Liberty   Air   2   Pro   '
 '  Envy   x360  ' 'Apple  HomePod  mini ' ' Lenovo   Legion   5  '
 '   Galaxy  Tab  S7' '  Fossil Gen 5E  ' ' Garmin   Fenix   7'
 '   Amazon Fire 7 ' '  Galaxy  Tab  A7' ' Sennheiser HD 660S '
 'Lenovo Tab M10 FHD Plus   ' 'Oculus  Rift  S   '
 ' Anker Soundcore Life Q35  ' 'Sony  Cyber-shot  RX100  VII   '
 '   JBL Live 500BT ' ' Sony   WH-CH710N   '
 'Anker Soundcore Liberty Air Pro' 'Lenovo   Legion   5 '
 '   MacBook   Air   13"   ' '  Amazon Fire HD 10   '
 ' OnePlus  10  Pro  ' 'Fitbit Versa 2  ' '  Fitbit  Versa  3 '
 'Galaxy S21 Ultra' 'Sennheiser HD 599 SE' 'Fujifilm X-T4'
 'Galaxy Tab S7+' 'Huawei Watch Fit 2 Pro' 'QC35 II' 'Xiaomi Mi 12'
 'Pimax Vision 5K Super Plus' 'ASUS ROG Strix Scar 17' 'Sony Alpha A6400'
 'MacBook Pro 16"' 'Huawei Watch GT 3 Pro' 'Xiaomi Redmi 10'
 'Garmin Vivosmart 4' 'ASUS ROG Zephyrus G14' 'HP Reverb G2 Omnicept'
 'JBL Link 20' 'Galaxy Watch Active 2' 'Withings Move' 'Garmin Venu 2S'
 'PlayStation 4 Pro' 'Galaxy Tab S7' 'OnePlus 10 Pro' 'Nest Audio'
 'Amazon Fire HD 8 Plus' 'Valve Index VR Kit 2' 'HTC Vive Cosmos Elite'
 'Xiaomi Redmi Note 11 Pro' 'HP Pavilion x360' 'Sonos Move'
 'Amazon Fire 7' 'Surface Laptop 4' 'Nikon Coolpix P1000' 'OnePlus Nord 2'
 'Echo Dot (4th Gen)' 'Sennheiser HD 800 S' 'ThinkPad X13'
 'Garmin Approach S12' 'Galaxy Tab A7' 'HTC Vive Cosmos' 'Oculus Rift S'
 'Amazfit GTR 3' 'Harman Kardon Citation' 'Garmin Forerunner 945'
 'Apple HomePod' 'iPhone SE' 'PlayStation 5' 'iPhone 12' 'Oculus Rift 2'
 'Xbox Series S' 'ThinkPad X1 Carbon' 'iPad Air' 'Google Pixel 5a'
 'Apple HomePod mini' 'PlayStation 3' 'Fitbit Versa 3'
 'Anker Soundcore Liberty Air 2 Pro' 'Oculus Quest 2 (256GB)'
 'Huawei Band 4 Pro' 'Harman Kardon Allure' 'Fossil Gen 5E'
 'Google Pixel 6 Pro' 'HTC Vive Pro 2' 'Nintendo Switch OLED Pro'
 'PlayStation 3 Slim' 'Xbox One X' 'Galaxy M32' 'XPS 15' 'Surface Go 2'
 'MacBook Air 13"' 'Bose SoundSport Free' 'HP Pavilion 15' 'Sonos One SL'
 'Garmin Approach S62' 'Nikon Z50' 'Fitbit Charge 4'
 'Olympus OM-D E-M1 Mark III' 'Fossil Gen 6' 'Fitbit Charge 3'
 'Sony Cyber-shot RX100 VII' 'Acer Swift 3' 'Nintendo Switch Pro'
 'Garmin Fenix 7' 'Garmin Forerunner 55' 'Xbox 360' 'OnePlus 9 Pro'
 'Echo Show 8' 'Galaxy Tab A8' 'Jabra Elite 85t' 'Apple Watch SE'
 'Lenovo Legion 5' 'Lenovo Legion 7i' 'Jabra Elite 75t' 'Oculus Quest 3'
 'Nikon Z7 II' 'iPad Pro 12.9"' 'Super NES Classic' 'Fujifilm X100V'
 'PlayStation VR' 'Apple Watch Series 7' 'Oculus Quest 2 (64GB)' 'XPS 13'
 'iPad Pro 11" (5th Gen)' 'Fitbit Versa 2' 'iPhone 13 Pro Max'
 'Canon EOS M50 Mark II' 'Nikon Coolpix B500' 'Lenovo Tab P11 Plus'
 'Samsung Gear S4' 'Anker Soundcore Life Q35' 'Garmin Vivosmart HR' 'QC45'
 'Google Nest Hub Max' 'JBL Live 650BTNC' 'Nintendo Switch OLED'
 'Samsung Gear Fit 4' 'HTC Vive Focus 3' 'Acer Swift 5' 'HP Reverb G2'
 'JBL Link 500' 'Canon EOS R5' 'Amazon Echo Show 8 (2nd Gen)'
 'Galaxy Z Fold 3' 'Nintendo Switch Lite' 'Nest Hub (2nd Gen)'
 'Xbox Original' 'Anker Soundcore Life Q30' 'PlayStation 2'
 'Galaxy Watch 4' 'Pimax Vision 5K Super' 'Valve Index VR Kit'
 'Huawei Watch GT 2e' 'Sega Genesis Mini' 'Huawei Band 6' 'Sony WH-CH710N'
 'Oppo Find X4 Neo' 'ASUS ROG Strix G15' 'Xbox Series X' 'Amazfit GTR 2e'
 'PlayStation 4 Slim' 'Sennheiser HD 660S' 'Lenovo Tab M10 FHD Plus'
 'Galaxy Tab S8+' 'Bose Home Speaker 300' 'Lenovo Tab M7'
 'Bose SoundSport Wireless' 'Garmin Fenix 6S' 'Xiaomi Mi Band 5'
 'Envy x360' 'Oppo Find X4 Pro' 'Lenovo Tab P11 Pro' 'JBL Live 500BT'
 'Withings Move ECG' 'Sony Cyber-shot HX99' 'Galaxy A52'
 'Xiaomi Mi Band 6' 'Amazon Fire HD 10' 'Panasonic Lumix GH5'
 'HTC Vive Pro' 'Sennheiser HD 450BT' 'Samsung Odyssey G9' 'Canon EOS R6'
 'OnePlus Nord CE 5G' 'Surface Pro 7' 'Huawei Watch GT 3' 'Galaxy A72'
 'Xiaomi Mi 11 Lite' 'Sony Alpha A7 III' 'ASUS ROG Flow Z13'
 'PlayStation 4' 'Google Nest Hub (2nd Gen)' 'iPad Pro 12.9" (5th Gen)'
 'Surface Book 3' 'Bose Home Speaker 500' 'Amazon Fire HD 8'
 'Amazfit Bip U Pro' 'Pimax Vision 8KX' 'Galaxy Tab S8' 'Envy 15'
 'Sony WH-1000XM4' 'Amazon Echo Show 10 (3rd Gen)' 'Amazfit Bip S'
 'Garmin Venu 2' 'Panasonic Lumix ZS200' 'Canon EOS R']
--------------------------------------------------
```

---

## Clean the Data

In [10]:

```
df
```

Out[10]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Customer First Name | Cust N |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | lina | alrr |
| **1** | 2 | 1/1/2023 | Saudi Arabia | riyadh | rd01 | 24.7136 | 46.6753 | omar | |
| **2** | 3 | 1/1/2023 | Saudi Arabia | riyadh | rd03 | 24.7743 | 46.7386 | iman | iis |
| **4** | 4 | 1/1/2023 | United Arab Emirates | abu dhabi | ad01 | 24.4539 | 54.3773 | ahmad | |
| **5** | 5 | 1/1/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | sami | a |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | ahmad | |
| **20000** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | ali | |
| **20001** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | husayn | s |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 | fatin | ba |
| **20003** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | ahmad | sh |

20000 rows × 19 columns

In [11]:

```
df['Customer First Name'] = df['Customer First Name'].str.capitalize()
df['Customer Last Name'] = df['Customer Last Name'].str.capitalize()
df['Customer First Name'] = df["Customer First Name"] + ' ' + df['Customer Last Nam
df.drop(columns="Customer Last Name",inplace=True)
df = df.rename(columns={"Customer First Name":"Full Name"})
df
```

Out[11]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | LINA.A |
| **1** | 2 | 1/1/2023 | Saudi Arabia | riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | OM |
| **2** | 3 | 1/1/2023 | Saudi Arabia | riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | IMAI |
| **4** | 4 | 1/1/2023 | United Arab Emirates | abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | AHM |
| **5** | 5 | 1/1/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | A |
| **20000** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | h |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | |
| **20003** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | Al |

20000 rows × 18 columns

In [12]:

```
df['Unit Price'] = df['Unit Price'].abs()
df['Discount'] = ((df['Discount'] / df['Unit Price']) * 100).round(2)
df['Total Price'] = (df["Quantity"] * (1 - df['Discount'] / 100) * df["Unit Price"]
df.insert(17, 'Total Price', df.pop('Total Price'))
df['Total Price'] = df['Total Price'].round()

df
```

Out[12]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | LINA.A |
| **1** | 2 | 1/1/2023 | Saudi Arabia | riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | OM |
| **2** | 3 | 1/1/2023 | Saudi Arabia | riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | IMAI |
| **4** | 4 | 1/1/2023 | United Arab Emirates | abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | AHM |
| **5** | 5 | 1/1/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | A |
| **20000** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | h |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | |
| **20003** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | Al |

20000 rows × 19 columns

In [13]:

```
df['Email'] = df['Email'].str.replace(r'@', '@g', regex=True)
df['Email'] = df['Email'].str.replace(r'\.', '', regex=True)
df['Email'] = df['Email'].str.lower()
df
```

Out[13]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | linaal |
| **1** | 2 | 1/1/2023 | Saudi Arabia | riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | om |
| **2** | 3 | 1/1/2023 | Saudi Arabia | riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | imani |
| **4** | 4 | 1/1/2023 | United Arab Emirates | abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | ahma |
| **5** | 5 | 1/1/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | sam |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | ahr |
| **20000** | 19997 | 1/9/2023 | Syria | homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | husayı |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | fatin |
| **20003** | 20000 | 1/9/2023 | USA | washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | ahmac |

20000 rows × 19 columns

In [14]:

```
df['City'] = df['City'].str.capitalize()
df
```

Out[14]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | linaa |
| **1** | 2 | 1/1/2023 | Saudi Arabia | Riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | om |
| **2** | 3 | 1/1/2023 | Saudi Arabia | Riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | iman |
| **4** | 4 | 1/1/2023 | United Arab Emirates | Abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | ahm |
| **5** | 5 | 1/1/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | sam |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | Casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | ah |
| **20000** | 19997 | 1/9/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | Las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | husay |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | Jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | fatir |
| **20003** | 20000 | 1/9/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | ahmad |

20000 rows × 19 columns

In [15]:

```
df['Status'] = df['Status'].str.capitalize()
df
```

Out[15]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | linaa |
| **1** | 2 | 1/1/2023 | Saudi Arabia | Riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | om |
| **2** | 3 | 1/1/2023 | Saudi Arabia | Riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | iman |
| **4** | 4 | 1/1/2023 | United Arab Emirates | Abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | ahm |
| **5** | 5 | 1/1/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | sam |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | Casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | ah |
| **20000** | 19997 | 1/9/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | Las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | husay |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | Jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | fatir |
| **20003** | 20000 | 1/9/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | ahmac |

20000 rows × 19 columns

In [16]:

```
df['Phone Number'] = df['Phone Number'].str.replace("Tel:","")
df['Phone Number'] = df['Phone Number'].apply(lambda x: x[0:4]+"-"+x[4:8]+"-"+x[8:]
df
```

Out[16]:

| | Order ID | Order Date | Country | City | Branch | Lat | Lng | Full Name | |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | 1/1/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Lina Alrrashid | linaa |
| **1** | 2 | 1/1/2023 | Saudi Arabia | Riyadh | rd01 | 24.7136 | 46.6753 | Omar Eurul | om |
| **2** | 3 | 1/1/2023 | Saudi Arabia | Riyadh | rd03 | 24.7743 | 46.7386 | Iman Iismaeil | iman |
| **4** | 4 | 1/1/2023 | United Arab Emirates | Abu dhabi | ad01 | 24.4539 | 54.3773 | Ahmad Rihan | ahma |
| **5** | 5 | 1/1/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Sami Altawil | sam |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | |
| **19999** | 19996 | 1/9/2023 | Morocco | Casablanca | cs01 | 33.5731 | 7.5898 | Ahmad Iad | ah |
| **20000** | 19997 | 1/9/2023 | Syria | Homs | hs01 | 34.7326 | 36.7136 | Ali Kiali | |
| **20001** | 19998 | 1/9/2023 | USA | Las vegas | lv01 | 36.1699 | -115.1398 | Husayn Salayk | husay |
| **20002** | 19999 | 1/9/2023 | Saudi Arabia | Jeddah | jd03 | 21.4858 | 39.1925 | Fatin Bahriin | fatir |
| **20003** | 20000 | 1/9/2023 | USA | Washington | wh01 | 38.9072 | -77.0369 | Ahmad Shakur | ahmac |

20000 rows × 19 columns

---

# Aggregation and Grouping

In [17]:

```
agg_df = df[df['Status'] == "True"][["Country", "Category", "Sub Category", "Item",
agg_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 14013 entries, 1 to 20003
Data columns (total 7 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Country       14013 non-null  object 
 1   Category      14013 non-null  object 
 2   Sub Category  14013 non-null  object 
 3   Item          14013 non-null  object 
 4   Quantity      14013 non-null  int64  
 5   Unit Price    14013 non-null  float64
 6   Total Price   14013 non-null  float64
dtypes: float64(2), int64(1), object(4)
memory usage: 875.8+ KB
```

In [18]:

```
x = agg_df.groupby(['Country']).agg({"Quantity":["sum"]})
x.sort_values(by=("Quantity", "sum"),ascending=False)
```

Out[18]:

| | Quantity |
|---|---|
| | sum |
| Country | |
| Saudi Arabia | 8298 |
| USA | 5850 |
| United Arab Emirates | 3869 |
| Syria | 3502 |
| Egypt | 3278 |
| France | 1526 |
| Morocco | 877 |

In [19]:

```
x = agg_df.groupby(['Country']).agg({"Total Price":["sum"]})
x.sort_values(by=("Total Price", "sum"),ascending=False)
```

Out[19]:

|  | Total Price |
|---|---|
|  | sum |
| **Country** |  |
| **Saudi Arabia** | 3821753.0 |
| **USA** | 2764045.0 |
| **United Arab Emirates** | 1701035.0 |
| **Syria** | 1569260.0 |
| **Egypt** | 1526289.0 |
| **France** | 730478.0 |
| **Morocco** | 404978.0 |

In [20]:

```
agg_df.groupby("Category").sum(numeric_only=True)
```

Out[20]:

|  | Quantity | Unit Price | Total Price |
|---|---|---|---|
| **Category** |  |  |  |
| **Digital Camera** | 2508 | 2073808.0 | 2911441.0 |
| **Fitness Tracker** | 2620 | 172242.0 | 252816.0 |
| **Gaming Console** | 2751 | 376134.0 | 552555.0 |
| **Headphones** | 2697 | 362531.0 | 528883.0 |
| **Laptop** | 3222 | 2025155.0 | 3091559.0 |
| **Smart Speaker** | 2531 | 252291.0 | 356713.0 |
| **Smartphone** | 2718 | 999473.0 | 1460660.0 |
| **Smartwatch** | 2634 | 438410.0 | 645424.0 |
| **Tablet** | 2810 | 728992.0 | 1042454.0 |
| **VR Headset** | 2709 | 1122681.0 | 1675333.0 |

In [21]:

```
z = agg_df.groupby(["Country","Category"]).sum(numeric_only=True)
z.sort_values(by="Total Price",ascending=False)
```

Out[21]:

|  |  | Quantity | Unit Price | Total Price |
|---|---|---|---|---|
| **Country** | **Category** |  |  |  |
| **Saudi Arabia** | **Laptop** | 983 | 624685.0 | 955775.0 |
|  | **Digital Camera** | 842 | 684111.0 | 953027.0 |
| **USA** | **Laptop** | 658 | 439364.0 | 678399.0 |
|  | **Digital Camera** | 503 | 428124.0 | 603703.0 |
| **Saudi Arabia** | **VR Headset** | 752 | 353659.0 | 482421.0 |
| **...** | **...** | ... | ... | ... |
| **Morocco** | **Headphones** | 70 | 10697.0 | 13149.0 |
|  | **Smart Speaker** | 104 | 8014.0 | 12195.0 |
| **France** | **Fitness Tracker** | 144 | 9240.0 | 11916.0 |
| **Morocco** | **Gaming Console** | 60 | 10393.0 | 11641.0 |
|  | **Fitness Tracker** | 95 | 5045.0 | 8176.0 |

70 rows × 3 columns

In [22]:

```
y = agg_df.groupby(["Category"]).sum(numeric_only=True)
y.sort_values(by="Total Price",ascending=False)
```

Out[22]:

|  | Quantity | Unit Price | Total Price |
|---|---|---|---|
| **Category** |  |  |  |
| **Laptop** | 3222 | 2025155.0 | 3091559.0 |
| **Digital Camera** | 2508 | 2073808.0 | 2911441.0 |
| **VR Headset** | 2709 | 1122681.0 | 1675333.0 |
| **Smartphone** | 2718 | 999473.0 | 1460660.0 |
| **Tablet** | 2810 | 728992.0 | 1042454.0 |
| **Smartwatch** | 2634 | 438410.0 | 645424.0 |
| **Gaming Console** | 2751 | 376134.0 | 552555.0 |
| **Headphones** | 2697 | 362531.0 | 528883.0 |
| **Smart Speaker** | 2531 | 252291.0 | 356713.0 |
| **Fitness Tracker** | 2620 | 172242.0 | 252816.0 |

In [23]:

```
x = agg_df.groupby(["Item"]).sum(numeric_only=True)
x.sort_values(by="Total Price",ascending=False)
```

Out[23]:

| | Quantity | Unit Price | Total Price |
|---|---|---|---|
| **Item** | | | |
| **Canon EOS R5** | 133 | 295926.0 | 428088.0 |
| **Panasonic Lumix S1H** | 112 | 223936.0 | 306926.0 |
| **ASUS ROG Flow Z13** | 163 | 182427.0 | 283155.0 |
| **MacBook Pro 16"** | 135 | 155935.0 | 277033.0 |
| **XPS 15** | 191 | 137514.0 | 229532.0 |
| **...** | ... | ... | ... |
| **Galaxy Tab A7** | 1 | 179.0 | 165.0 |
| **Amazon Fire HD 10** | 1 | 149.0 | 139.0 |
| **Sony WH-CH710N** | 1 | 129.0 | 129.0 |
| **Sennheiser HD 450BT** | 1 | 129.0 | 125.0 |
| **Amazon Fire 7** | 1 | 49.0 | 46.0 |

206 rows × 3 columns

In [24]:

```
i = agg_df.groupby(["Country"]).agg({"Total Price":["sum"]})
print(i.sort_values(by=("Total Price","sum"),ascending=False))
```

```
0                     Total Price
                              sum
Country
Saudi Arabia            3821753.0
USA                     2764045.0
United Arab Emirates    1701035.0
Syria                   1569260.0
Egypt                   1526289.0
France                   730478.0
Morocco                  404978.0
```

In [25]:

```
# df.to_csv('Cleaned Data.csv',index=False)
```

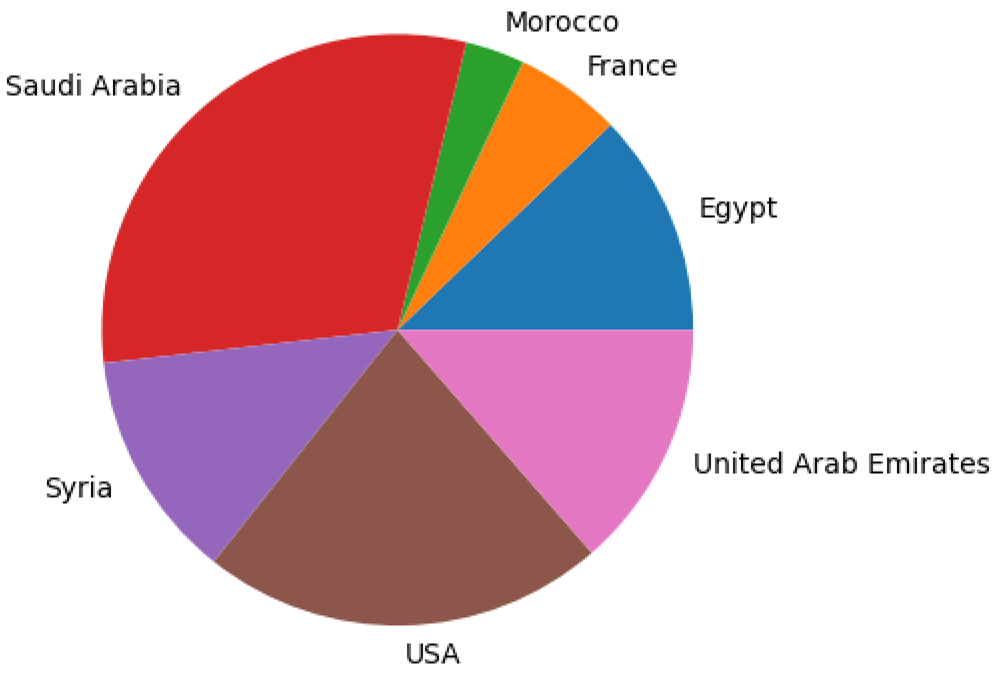
# Data Visualization

In [26]:

```
i.plot.pie(y="Total Price",legend=False, ylabel='')
```

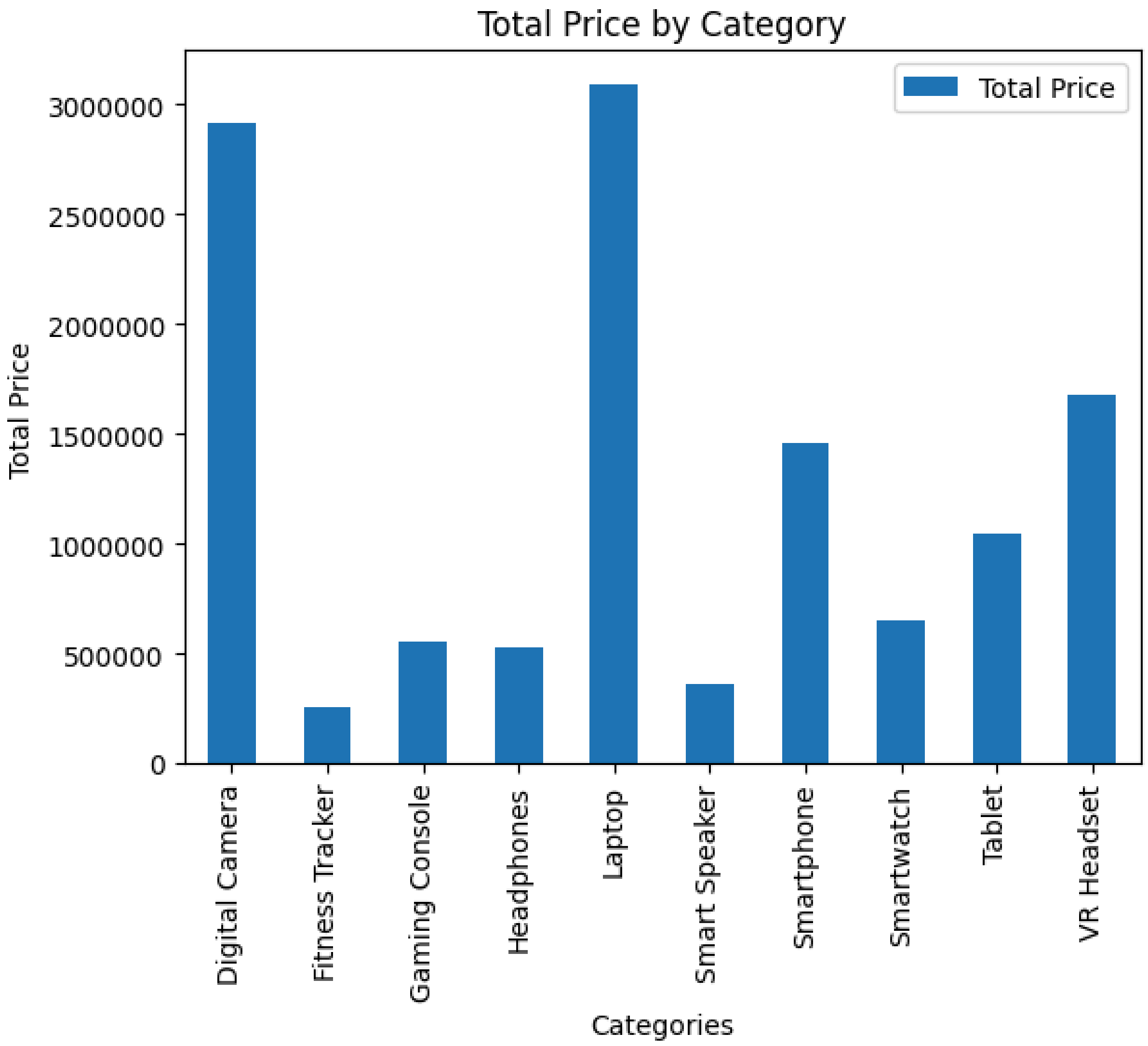
Out[26]:

```
<Axes: >
```



In [27]:

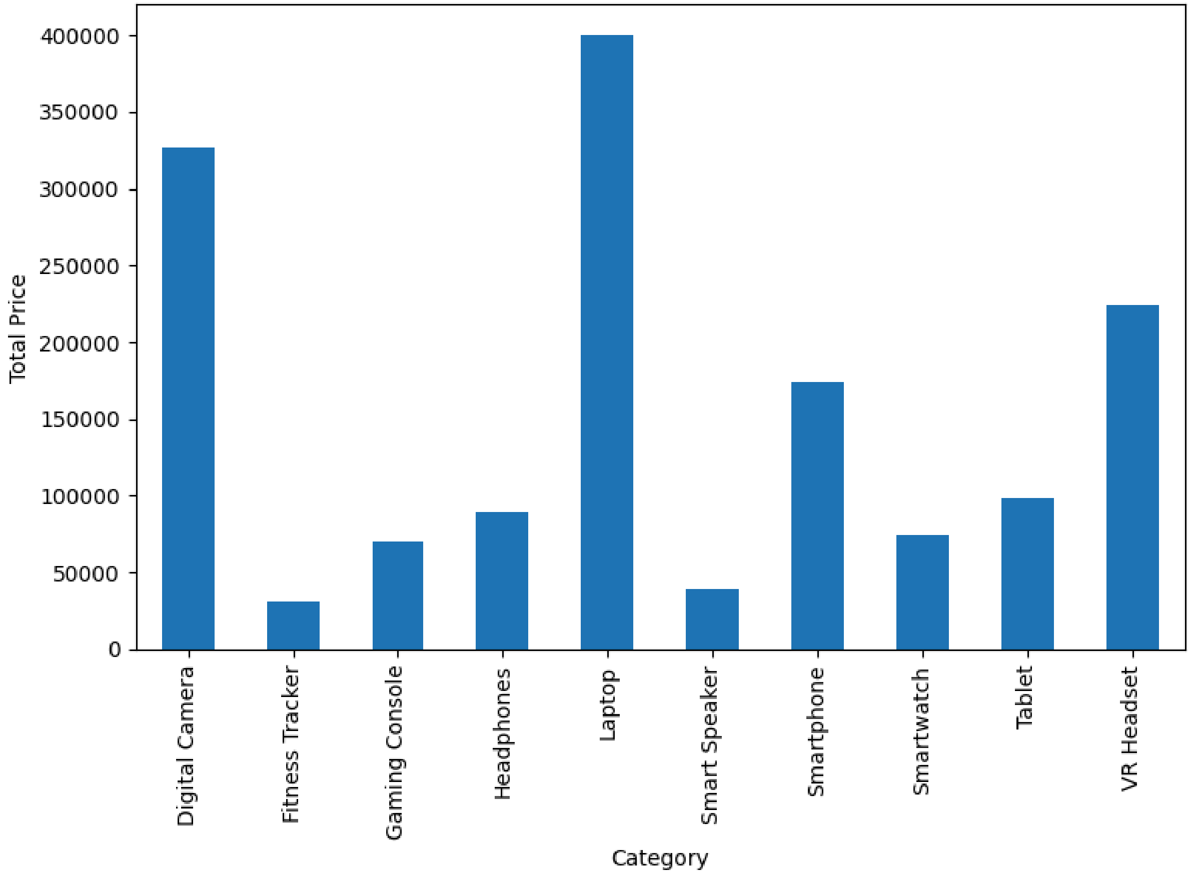
```
y.plot(kind="bar",y='Total Price')

plt.ticklabel_format(style='plain', axis='y')

plt.xlabel("Categories")
plt.ylabel("Total Price")

plt.title("Total Price by Category")
plt.show()
```



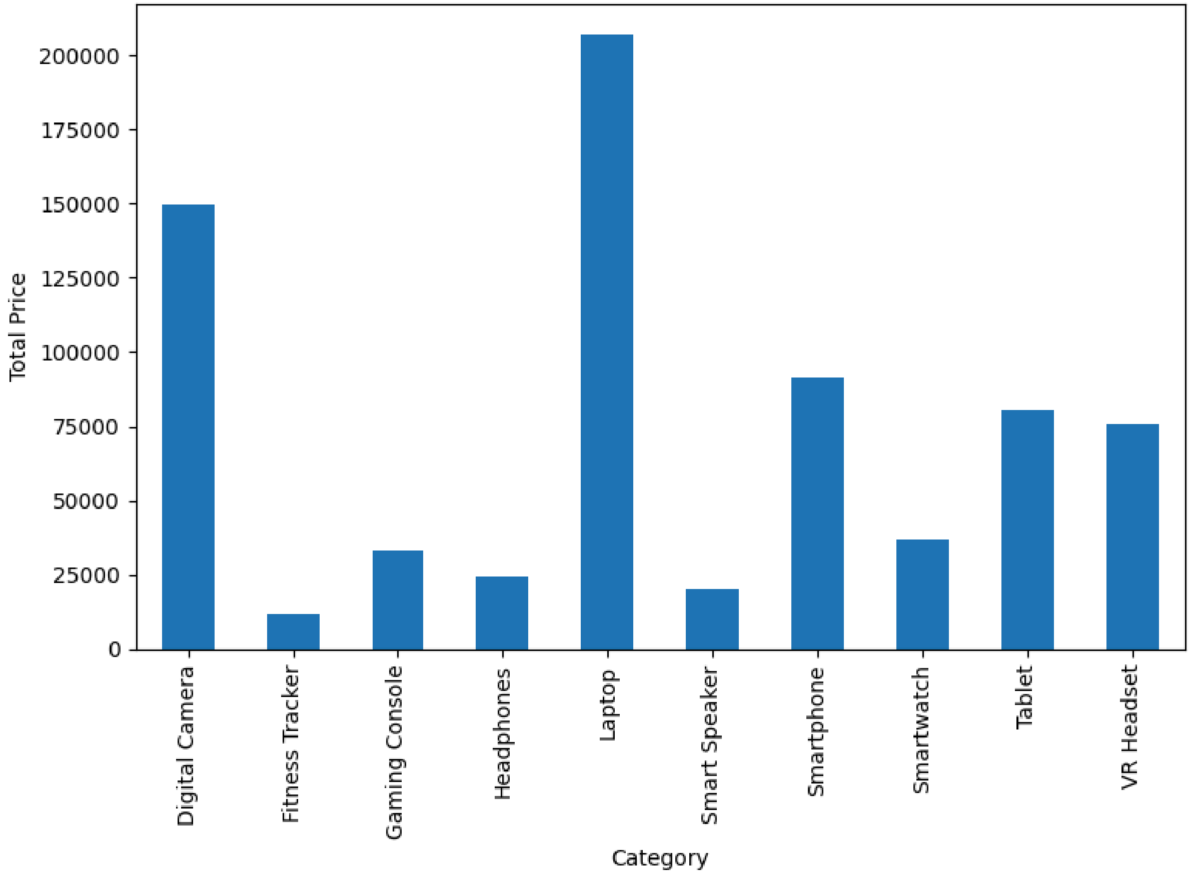
In [ ]:

```
# Group by 'Country' and 'Category', then sum 'Total Price'
z = agg_df.groupby(["Country", "Category"]).sum(numeric_only=True).reset_index()

# Get the list of unique countries
countries = z['Country'].unique()

# Create a subplot for each country
fig, axes = plt.subplots(nrows=len(countries), ncols=1, figsize=(8, 6 * len(countri

# If there's only one country, axes will not be an array, so we wrap it in a list
if len(countries) == 1:
    axes = [axes]

# Plot each country's data in its own subplot
for i, country in enumerate(countries):
    # Filter data for the current country
    country_data = z[z['Country'] == country]

    # Plot the bar chart for the current country
    ax = axes[i]
    country_data.plot(kind="bar", x='Category', y='Total Price', legend=False, ax=a
    # Disable scientific notation on the y-axis
    ax.ticklabel_format(style='plain', axis='y')

    # Add labels and title
    ax.set_xlabel("Category")
    ax.set_ylabel("Total Price")
    ax.set_title(f"Total Price by Category in {country}")

# Adjust layout to prevent overlap
plt.tight_layout()
plt.show()
```

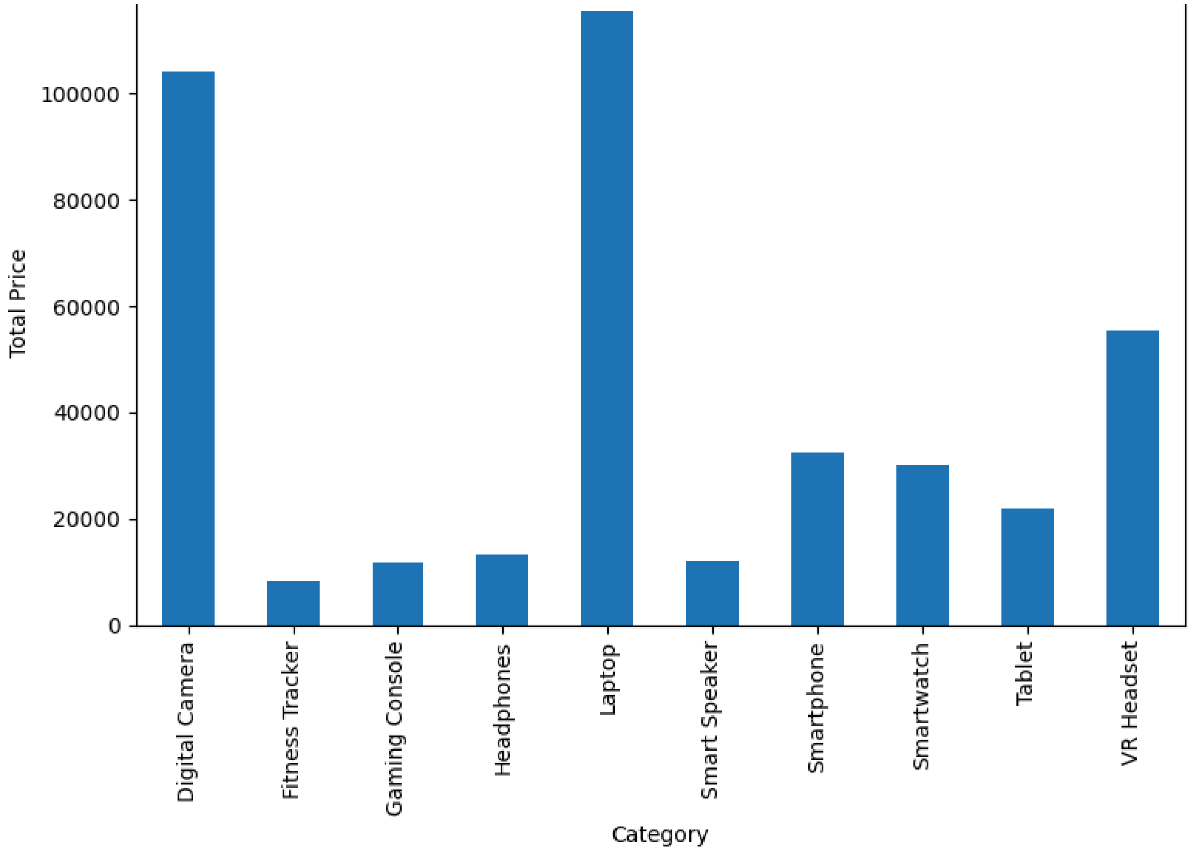
Total Price by Category in Egypt

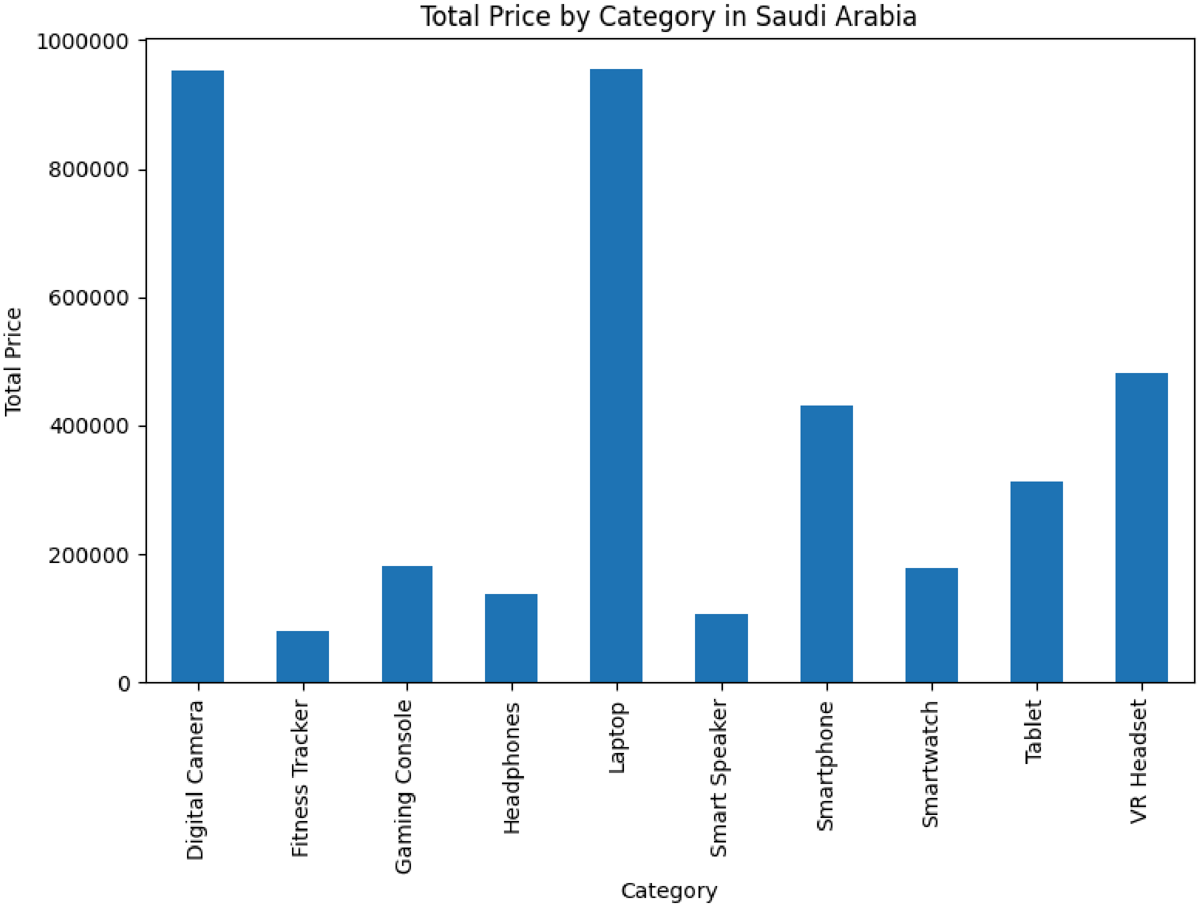


Total Price by Category in France

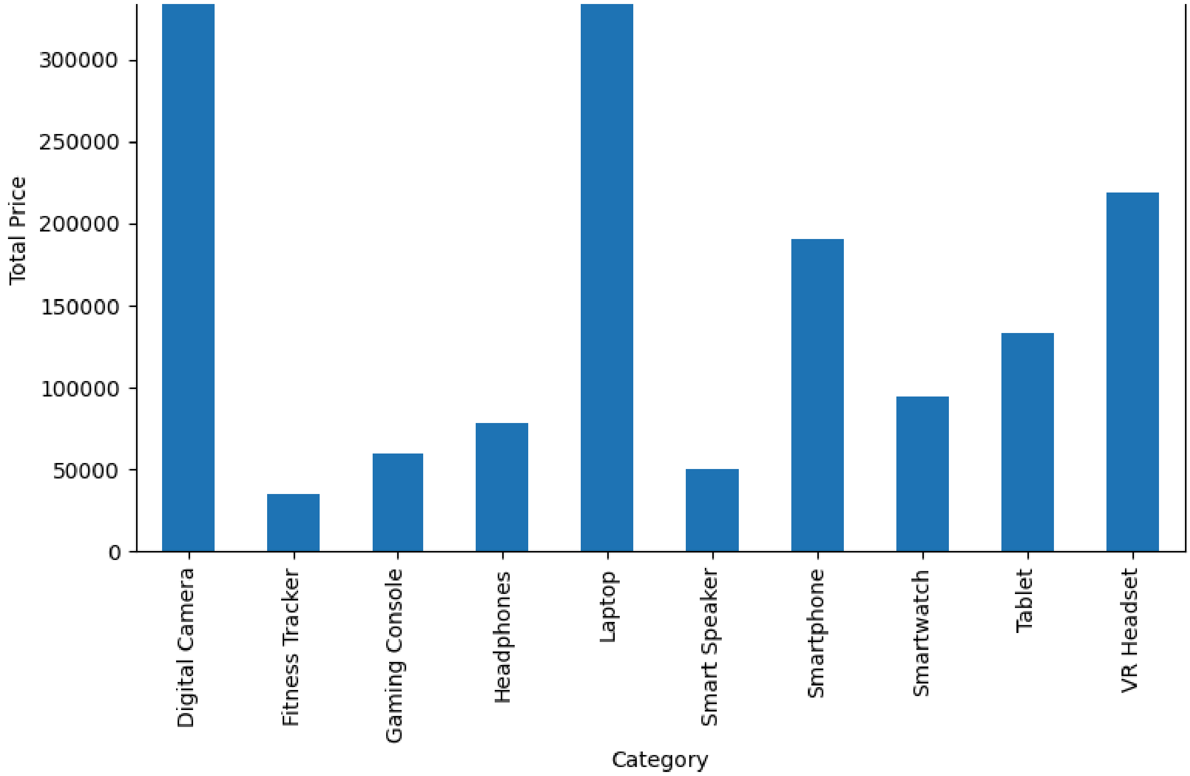


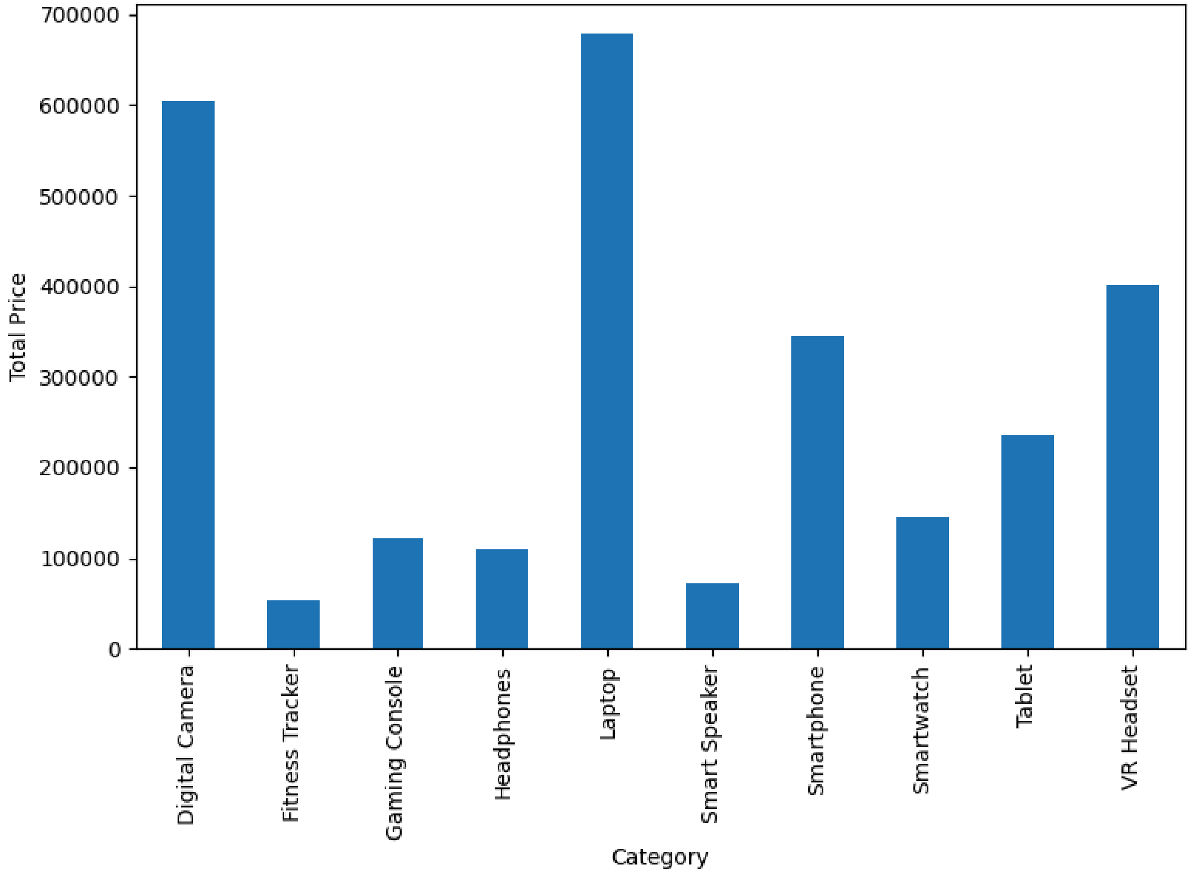
Total Price by Category in Morocco

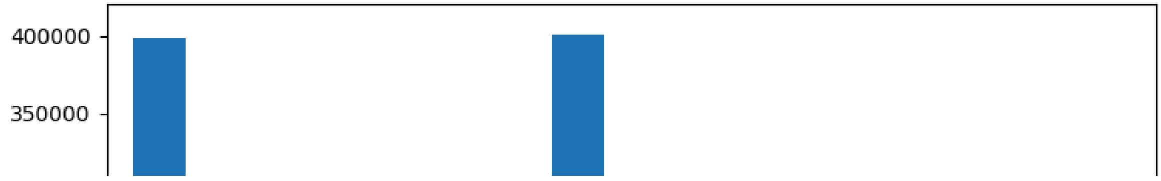


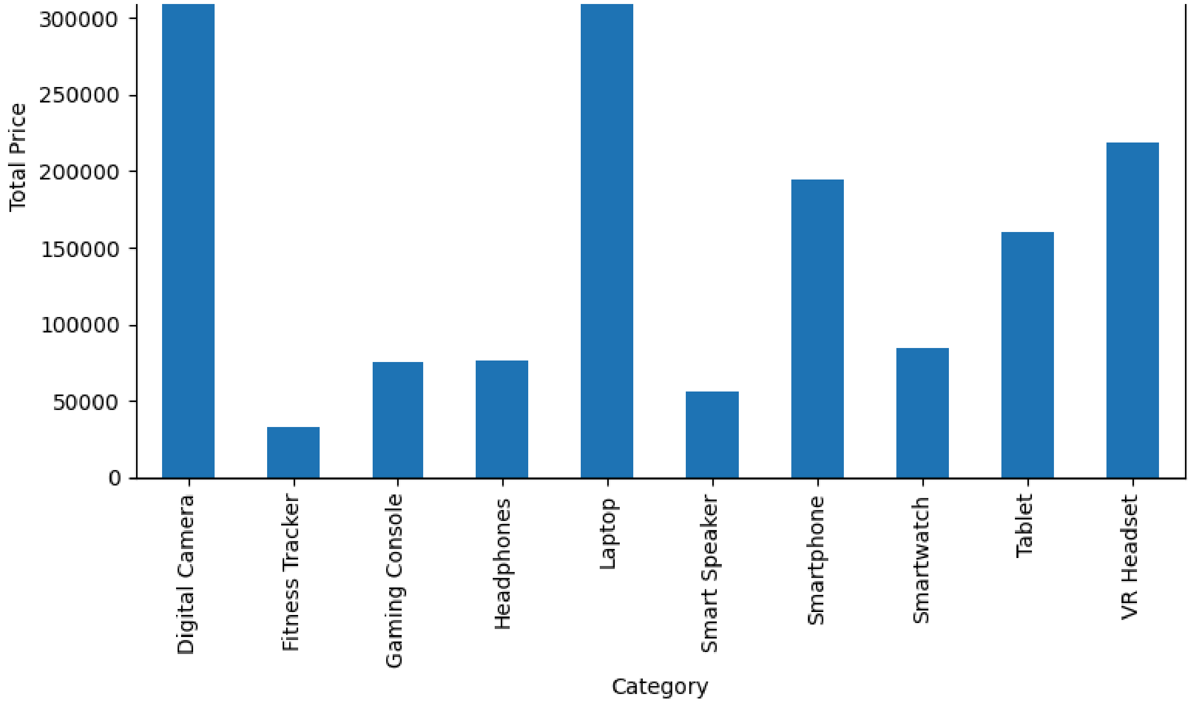

Total Price by Category in Saudi Arabia

Total Price by Category in Syria




Total Price by Category in USA


Total Price by Category in United Arab Emirates


Total Price
300000
250000
200000
150000
100000
50000
0
Digital Camera
Fitness Tracker
Gaming Console
Headphones
Laptop
Smart Speaker
Smartphone
Smartwatch
Tablet
VR Headset
Category