In [1]:
```
                    E-Commerce Supply Chain Analysis
                         By : Mohammad Areeb
            Linkedin: www.linkedin.com/in/mohammadareeb2544
                  Github: https://github.com/areeb399
```

In [ ]:

In [ ]:
```
Objective:
    1. To find the revenue generated by different product type.
    2. To analyze the sales by product type.
    3. To find out the total revenue generated from shipping carriers.
    4. To analyze revenue generated by each SKU (Stock Keeping Unit).
    5. To analyze the shupping cost of carriers.
    6. To find out the cost distribution by transportation modes.
    8. To analyze the defect rate of the product during shipping.
```

In [ ]:

In [2]:
```
#Importing Libraries

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [ ]:

In [3]:
```
#Importing Data
df = pd.read_csv("supply_chain_data.csv")
df
```

Out[3]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3
...	...	...	...	...	...	...	...	...	...
95	haircare	SKU95	77.903927	65	672	7386.363944	Unknown	15	14
96	cosmetics	SKU96	24.423131	29	324	7698.424766	Non-binary	67	2
97	haircare	SKU97	3.526111	56	62	4370.916580	Male	46	19
98	skincare	SKU98	19.754605	43	913	8525.952560	Female	53	1
99	haircare	SKU99	68.517833	17	627	9185.185829	Unknown	55	8

100 rows × 24 columns

In [ ]:

In [4]:

```
#Top 10 Data

df.head(10)
```

Out[4]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	q
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10	
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13	
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3	
5	haircare	SKU5	1.699976	87	147	2828.348746	Non-binary	90	27	
6	skincare	SKU6	4.078333	48	65	7823.476560	Male	11	15	
7	cosmetics	SKU7	42.958384	59	426	8496.103813	Female	93	17	
8	cosmetics	SKU8	68.717597	78	150	7517.363211	Female	5	10	
9	skincare	SKU9	64.015733	35	980	4971.145988	Unknown	14	27	

10 rows × 24 columns

In [ ]:

In [5]:

```
#Bottom 10 Data
df.tail(10)
```

Out[5]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times
90	skincare	SKU90	13.881914	56	320	9592.633570	Non-binary	66	18
91	cosmetics	SKU91	62.111965	90	916	1935.206794	Male	98	22
92	cosmetics	SKU92	47.714233	44	276	2100.129755	Male	90	25
93	haircare	SKU93	69.290831	88	114	4531.402134	Unknown	63	17
94	cosmetics	SKU94	3.037689	97	987	7888.356547	Unknown	77	26
95	haircare	SKU95	77.903927	65	672	7386.363944	Unknown	15	14
96	cosmetics	SKU96	24.423131	29	324	7698.424766	Non-binary	67	2
97	haircare	SKU97	3.526111	56	62	4370.916580	Male	46	19
98	skincare	SKU98	19.754605	43	913	8525.952560	Female	53	1
99	haircare	SKU99	68.517833	17	627	9185.185829	Unknown	55	8

10 rows × 24 columns

In [ ]:

In [6]:

```
#Statistical Summary of the DataFrame
df.describe()
```

Out[6]:

	Price	Availability	Number of products sold	Revenue generated	Stock levels	Lead times	Order quantities	Shi
count	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.000000	100.0
mean	49.462461	48.400000	460.990000	5776.048187	47.770000	15.960000	49.220000	5.7
std	31.168193	30.743317	303.780074	2732.841744	31.369372	8.785801	26.784429	2.7
min	1.699976	1.000000	8.000000	1061.618523	0.000000	1.000000	1.000000	1.0
25%	19.597823	22.750000	184.250000	2812.847151	16.750000	8.000000	26.000000	3.7
50%	51.239831	43.500000	392.500000	6006.352023	47.500000	17.000000	52.000000	6.0
75%	77.198228	75.000000	704.250000	8253.976921	73.000000	24.000000	71.250000	8.0
max	99.171329	100.000000	996.000000	9866.465458	100.000000	30.000000	96.000000	10.0

In [ ]:

In [7]:

```
#Checking the dimensions or shape of a DataFrame

df.shape
```

Out[7]: (100, 24)

In [8]:

```
#DataFrame has 100 rows and 3 columns
```

In [9]:

```
#Removing rows containing any missing values (NaN values)

print(df.dropna(inplace = True))
```

None

In [10]:

```
#Checking Missing Values

df.isna()
```

Out[10]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	Ord quantitie
0	False	False	False	False	False	False	False	False	False	Fals
1	False	False	False	False	False	False	False	False	False	Fals
2	False	False	False	False	False	False	False	False	False	Fals
3	False	False	False	False	False	False	False	False	False	Fals
4	False	False	False	False	False	False	False	False	False	Fals
...	...	...	...	...	...	...	...	...	...	
95	False	False	False	False	False	False	False	False	False	Fals
96	False	False	False	False	False	False	False	False	False	Fals
97	False	False	False	False	False	False	False	False	False	Fals
98	False	False	False	False	False	False	False	False	False	Fals
99	False	False	False	False	False	False	False	False	False	Fals

100 rows × 24 columns

In [11]:

```
#True : If cell has a missing value (NaN), and False : If it's not a missing value.
```

In [ ]:

In [12]:

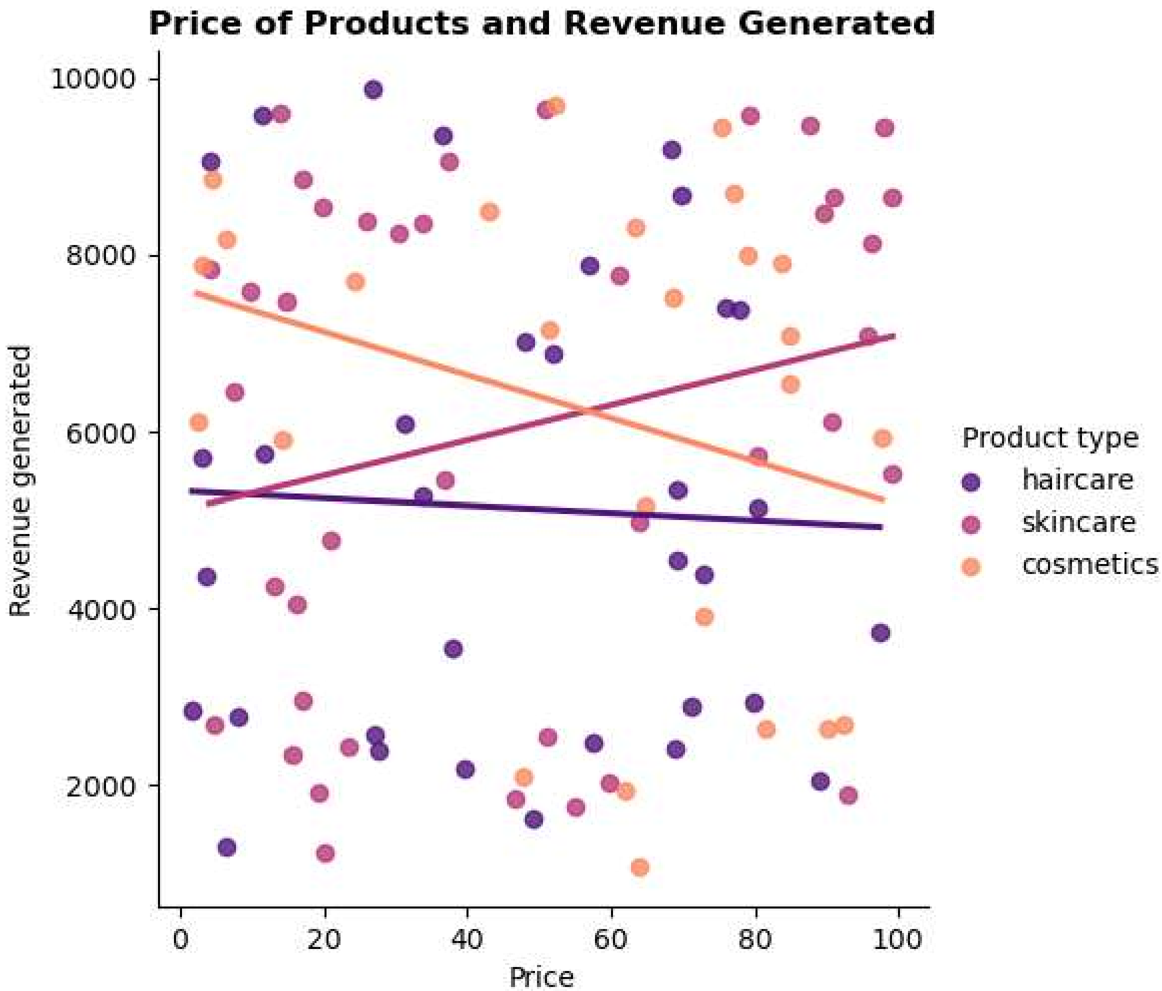
```
#Count of number of missing(Nan) values in each column

df.isna().sum()
```

Out[12]:
```
Product type               0
SKU                        0
Price                      0
Availability               0
Number of products sold    0
Revenue generated          0
Customer demographics      0
Stock levels               0
Lead times                 0
Order quantities           0
Shipping times             0
Shipping carriers          0
Shipping costs             0
Supplier name              0
Location                   0
Lead time                  0
Production volumes         0
Manufacturing lead time    0
Manufacturing costs        0
Inspection results         0
Defect rates               0
Transportation modes       0
Routes                     0
Costs                      0
dtype: int64
```

In [ ]:

In [13]:
```
#Creating a Visualization on Price of Products and Revenue Generated by them:

plt.figure(figsize = (10,10))
sns.lmplot(x='Price', y='Revenue generated', data=df, hue='Product type', ci=None,
plt.title("Price of Products and Revenue Generated", fontweight = 'bold')
plt.show()
```

```
<Figure size 1000x1000 with 0 Axes>
```



In [ ]:
```
Conclusion:
    Skincare products contribute significantly to the company's revenue, being the
    while cosmetics and haircare products generate comparatively lower revenue.
```

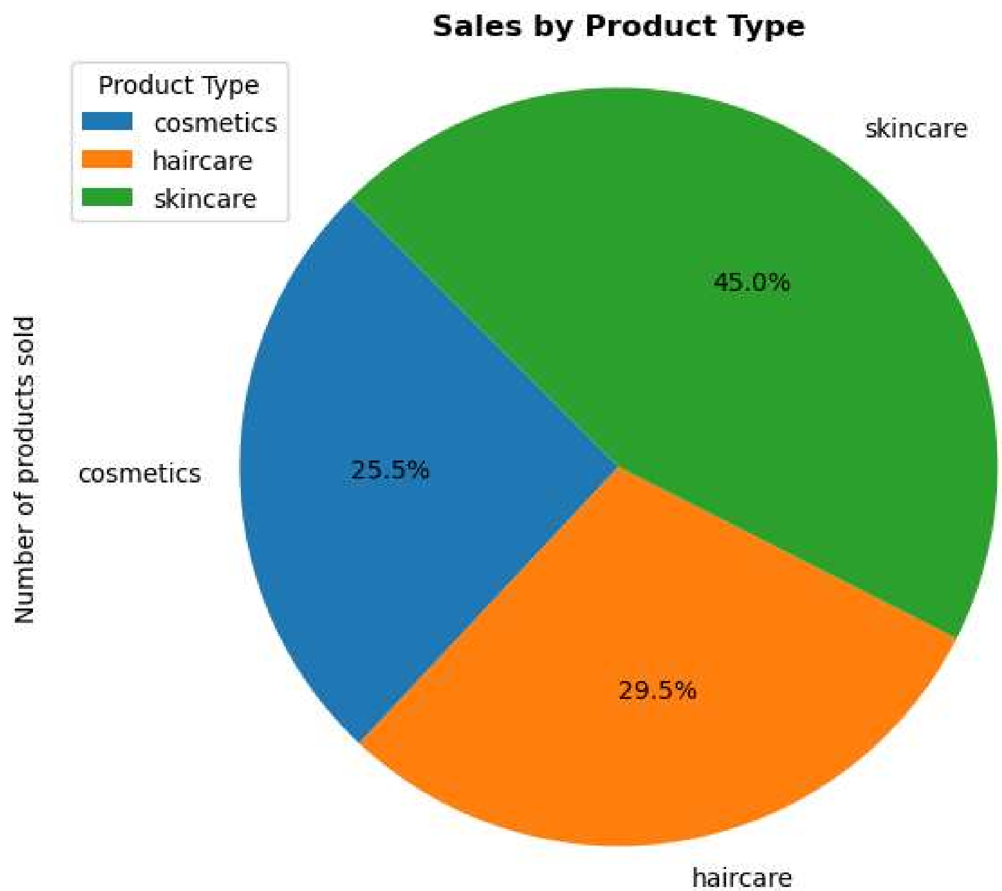
In [ ]:
```

```

In [15]:
```
# Grouping the DataFrame by 'Product type' and calculate the sum of 'Number of prod

sales_data = df.groupby('Product type')['Number of products sold'].sum()

print(sales_data)
```

```
Product type
cosmetics    11757
haircare     13611
skincare     20731
Name: Number of products sold, dtype: int64
```

In [16]:
```
#Creating Visualization Based on Sales Data

plt.figure(figsize = (8,6))

sales_data.plot(kind='pie', autopct='%0.1f%%', startangle = 135)

plt.axis('equal')
```

```
plt.legend(sales_data.index , title='Product Type', loc='upper left')

plt.title("Sales by Product Type", fontweight = 'bold')

plt.show()
```



In [ ]:
```
Conclusion :
    Skincare leads the sales charts,with haircare following closely in second place
    and cosmetics claiming the last spot in terms of revenue generation.
```

In [ ]:

In [17]:
```
df.head()
```

Out[17]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	qu
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10	
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13	
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3	

5 rows × 24 columns

In [ ]:

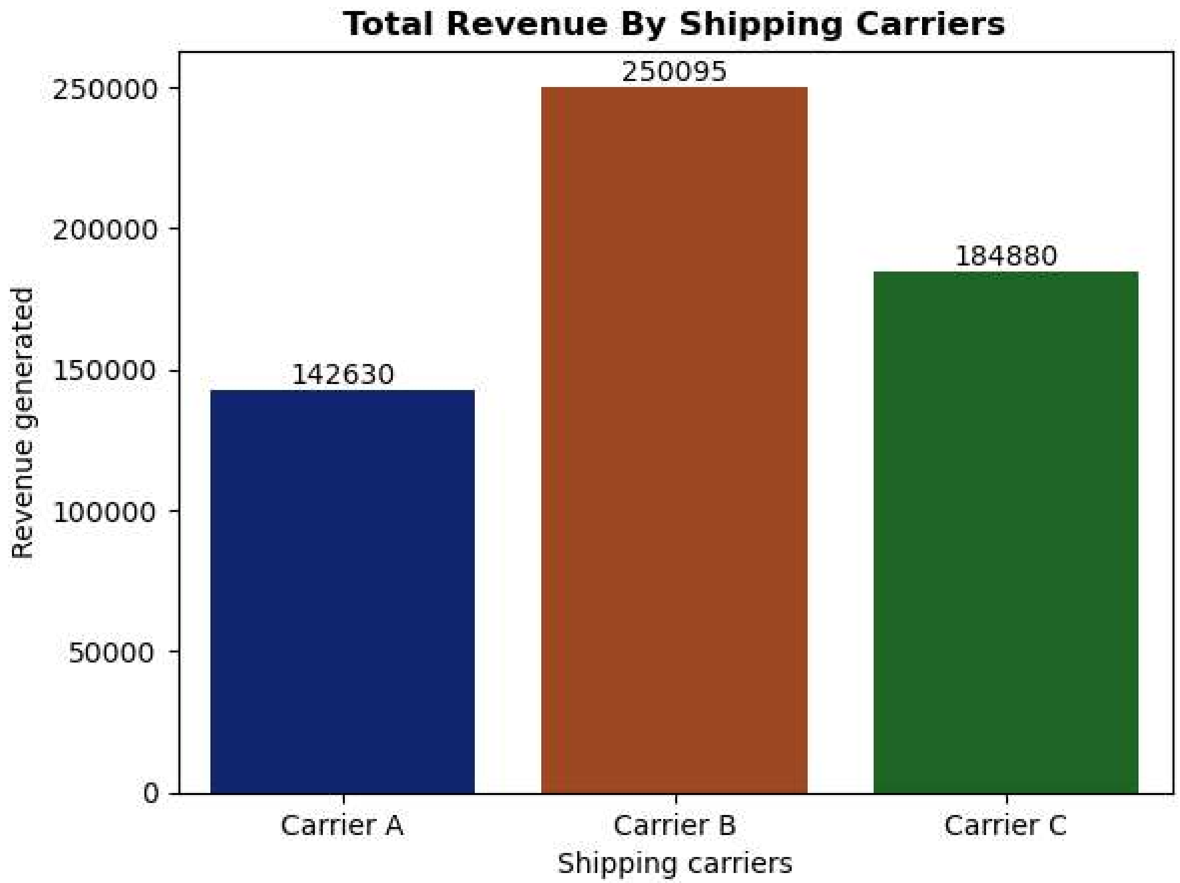
In [18]:

```
#Groupby operation for calculating the sum of the "Revenue generated" for each "Shi

total_revenue = df.groupby('Shipping carriers')['Revenue generated'].sum().reset_in

print(total_revenue)
```

```
  Shipping carriers  Revenue generated
0         Carrier A      142629.994607
1         Carrier B      250094.646988
2         Carrier C      184880.177143
```

In [19]:

```
#Creating Visualization on Revenue Generated through Shipping Carriers

fig = sns.barplot(x = total_revenue['Shipping carriers'] , y = total_revenue['Reven

# Add count labels on top of each bar
for bars in fig.containers:
    fig.bar_label(bars)

plt.title("Total Revenue By Shipping Carriers", fontweight = 'bold')
plt.show()
```



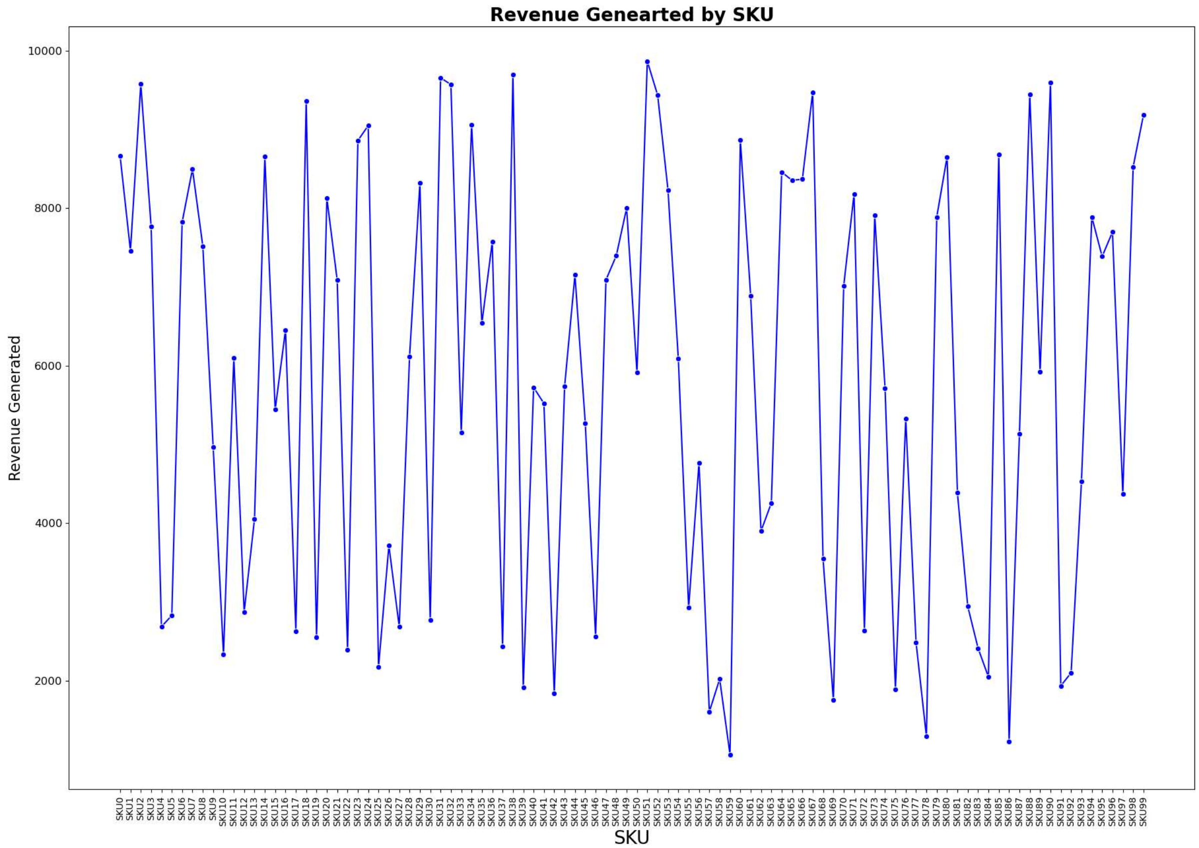
In [ ]:

```
Conclusion:
    Based on the above visualization, we observe that Carrier B generated the highe
    Following that, Carrier C is ranked second in revenue generation, while Carrier
```

In [ ]:

In [20]:

```python
#Creating Visualization on Revenue Generated through Shipping Carriers

plt.figure(figsize=(22, 15))
sns.lineplot(x = 'SKU', y = 'Revenue generated', data = df , marker='o', color='b')

plt.xticks(fontsize = 10, rotation = 90)
plt.yticks(fontsize = 12)

plt.xlabel('SKU', fontsize = 20)
plt.ylabel('Revenue Generated', fontsize = 16)
plt.title('Revenue Genearted by SKU',fontsize = 20, fontweight = 'bold')



plt.show()
```

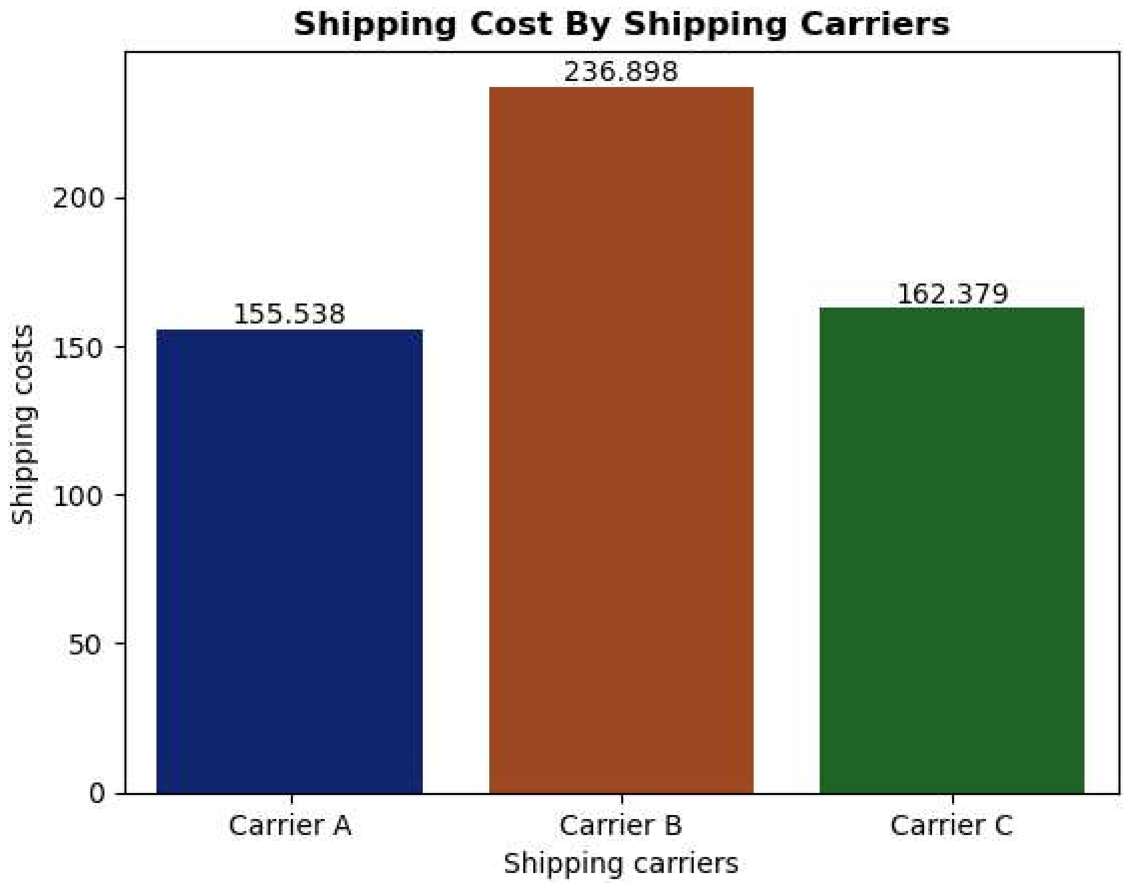


In [ ]:

In [21]:
```
#Groupby operation for calculating the sum of the "Shipping costs" for each "Shippi

shipping_cost = df.groupby('Shipping carriers')['Shipping costs'].sum().reset_index

print(shipping_cost)
```

```
  Shipping carriers  Shipping costs
0         Carrier A      155.537831
1         Carrier B      236.897620
2         Carrier C      162.379457
```

In [22]:
```
#Creating Visualization on Shipping Cost through Shipping Carriers

fig = sns.barplot(x = shipping_cost['Shipping carriers'] , y = shipping_cost['Shipp

# Add count labels on top of each bar
for bars in fig.containers:
    fig.bar_label(bars)

plt.title("Shipping Cost By Shipping Carriers", fontweight = 'bold')
plt.show()
```



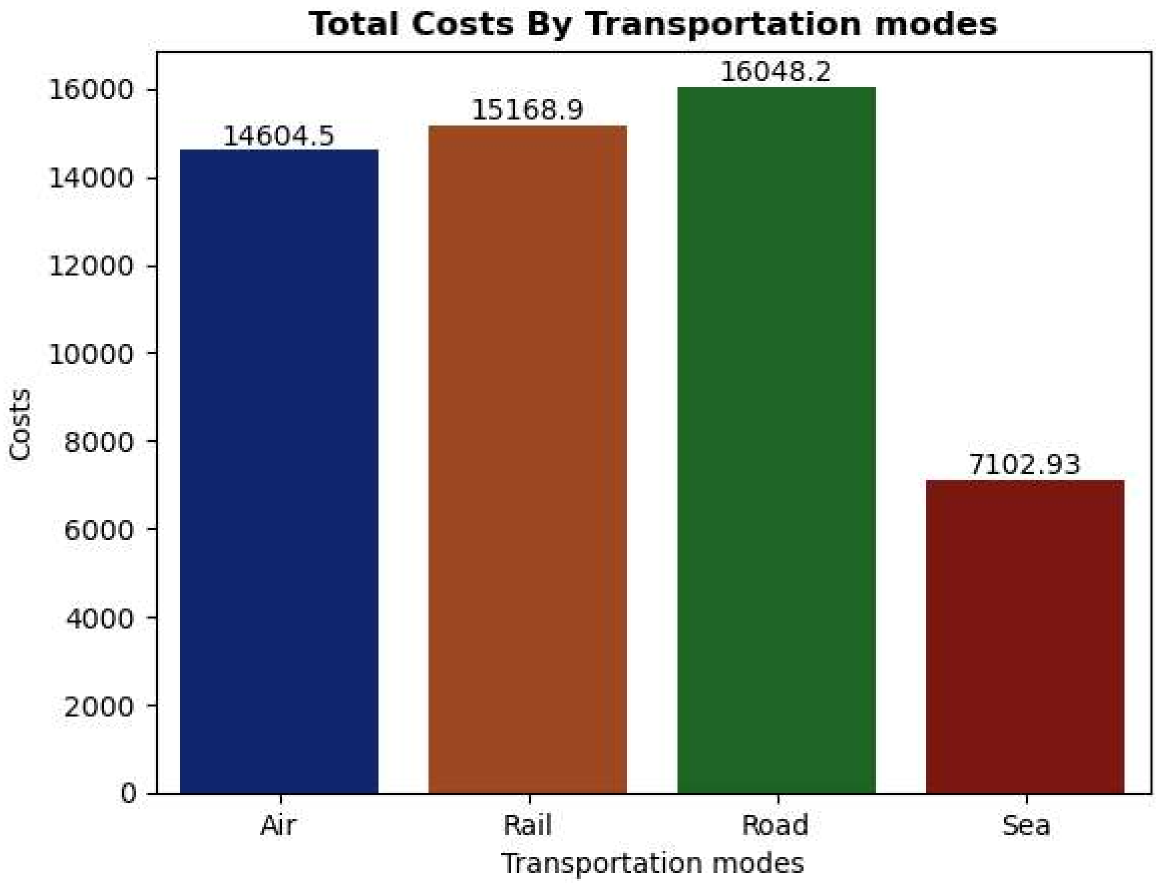
In [ ]:
```
Conclusion :
    The shipping cost of Carrier B is the highest at 236.898, followed by Carrier C
```

In [ ]:
```
```

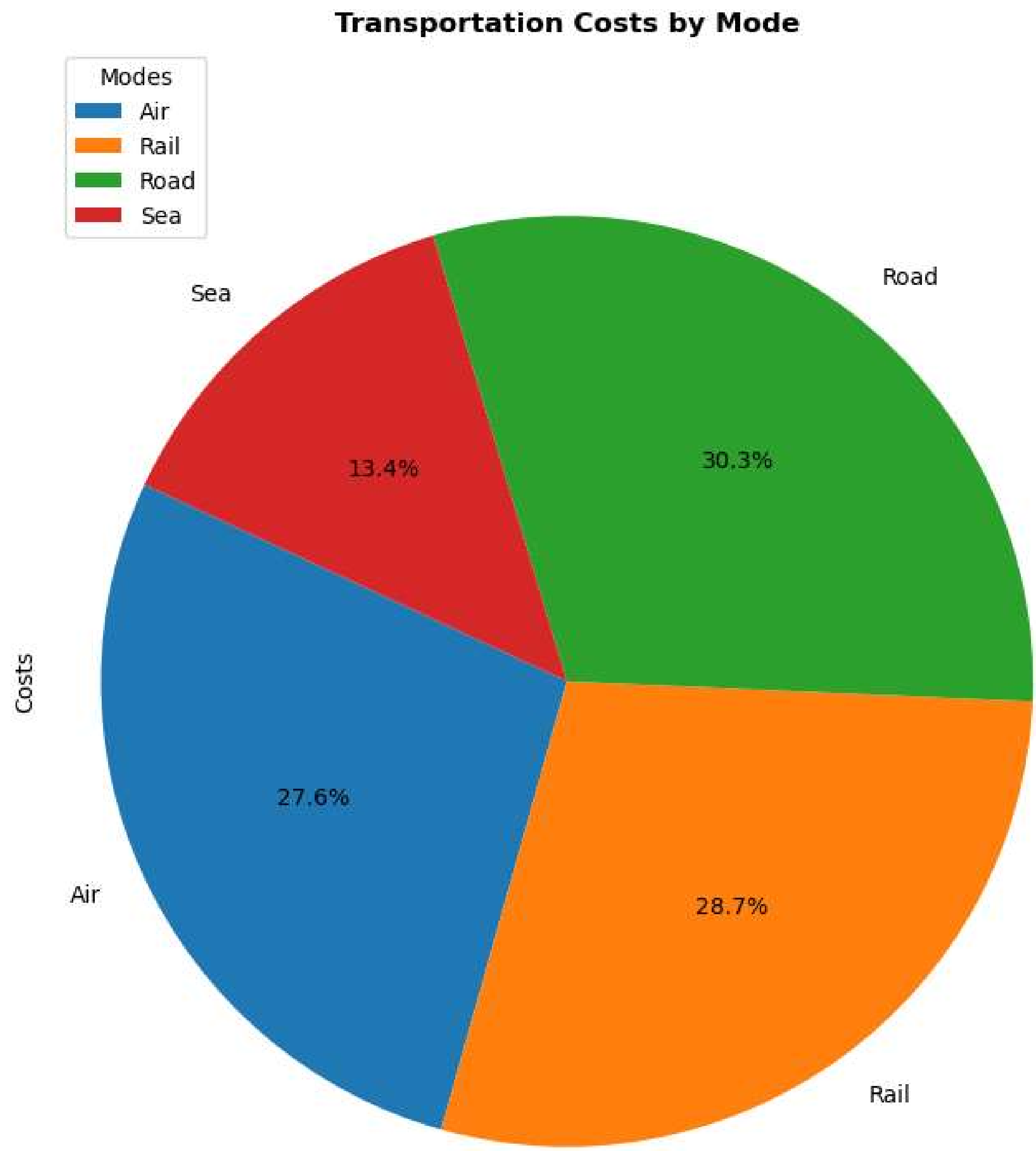
In [30]:
```
#Grouping the "Transportation modes" column and then calculating the sum of the "Co

transportation_cost = df.groupby('Transportation modes')['Costs'].sum()

print(transportation_cost)
```

```
Transportation modes
Air     14604.527498
Rail    15168.931559
Road    16048.193639
Sea      7102.925520
Name: Costs, dtype: float64
```

In [24]:
```
#Creating Visualization on Transportation Costs by differnt Transportation Modes

fig = sns.barplot(x = transportation_cost['Transportation modes'] , y = transportat

# Add count labels on top of each bar
for bars in fig.containers:
    fig.bar_label(bars)

plt.title("Total Costs By Transportation modes", fontweight = 'bold')
plt.show()
```



In [34]:
```
# Plotting the pie chart for transportation_cost

plt.figure(figsize=(8, 10))


transportation_cost.plot(kind='pie', autopct='%0.1f%%', startangle=155)

plt.axis('equal')


plt.legend(transportation_cost.index, title='Modes', loc='upper left')

plt.title("Transportation Costs by Mode", fontweight='bold')

plt.show()
```



In [ ]:
```
Conclusion :
    Based on the visualization above, it is evident that transportation by road inc
    followed by rail, air, and finally, sea transportation.
```

In [ ]:
```
```

In [35]:
```
df.head()
```

Out[35]:

	Product type	SKU	Price	Availability	Number of products sold	Revenue generated	Customer demographics	Stock levels	Lead times	qu
0	haircare	SKU0	69.808006	55	802	8661.996792	Non-binary	58	7	
1	skincare	SKU1	14.843523	95	736	7460.900065	Female	53	30	
2	haircare	SKU2	11.319683	34	8	9577.749626	Unknown	1	10	
3	skincare	SKU3	61.163343	68	83	7766.836426	Non-binary	23	13	
4	skincare	SKU4	4.805496	26	871	2686.505152	Non-binary	5	3	

5 rows × 24 columns

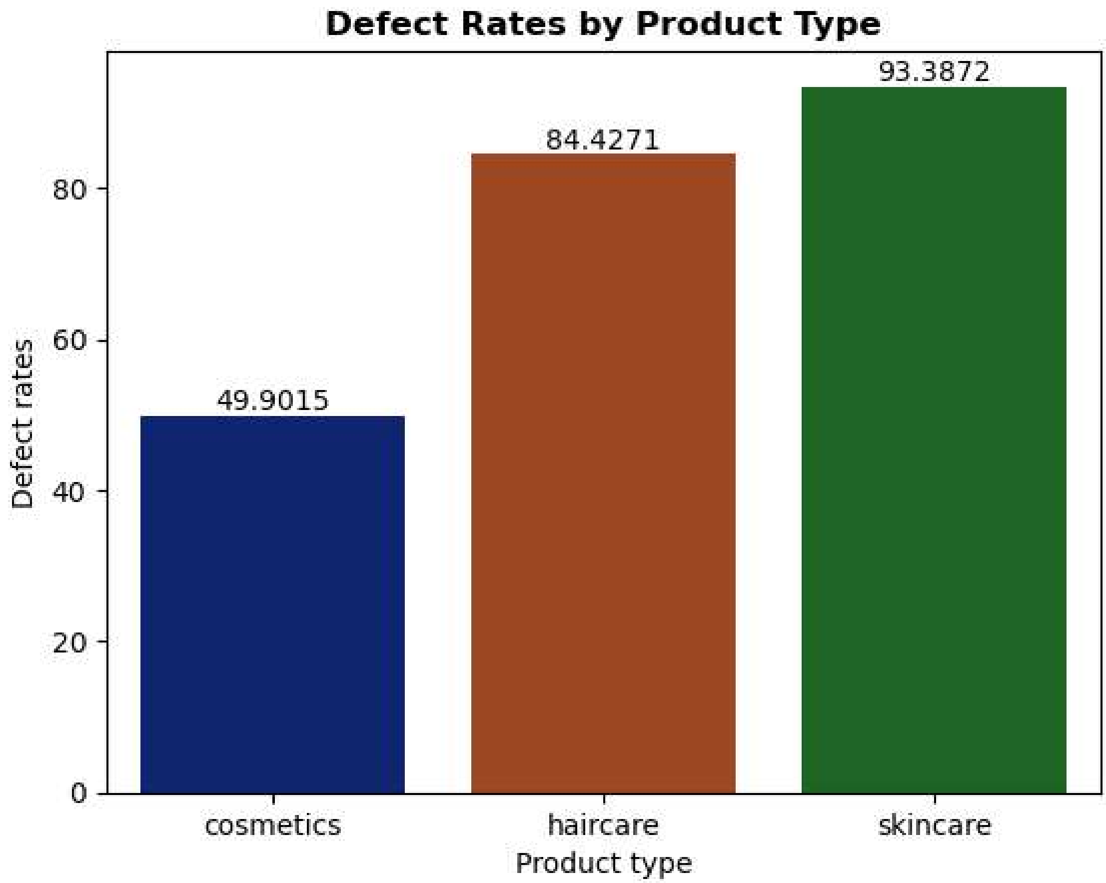
In [37]:

```
#Grouping the "Product type" column and then calculating the sum of "Defect rates"

Total_Defect_Rates = df.groupby('Product type')['Defect rates'].sum().reset_index()

print(Total_Defect_Rates)
```

```
  Product type  Defect rates
0    cosmetics     49.901461
1     haircare     84.427107
2     skincare     93.387231
```

In [39]:

```
#Creating Visualization on Defect Rates by Product Type

fig = sns.barplot(x = Total_Defect_Rates['Product type'] , y = Total_Defect_Rates['

# Add count labels on top of each bar
for bars in fig.containers:
    fig.bar_label(bars)

plt.title("Defect Rates by Product Type", fontweight = 'bold')
plt.show()
```



In [ ]:
```
Conclusion :
    Based on the above visualization, we observe that the defect rate is highest in
    followed by haircare products in the second position, and cosmetics with the lo
```