Project Title

Amazon Sales Analysis

Required Library

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

Load Data

In [2]:
```
df = pd.read_csv(r"C:\Users\dell\Downloads\amazon_sales_2025_INR.csv")
```

In [3]:
```
#read first 5 rows
df.head()
```

Out[3]:

	Order_ID	Date	Customer_ID	Product_Category	Product_Name	Quan
0	ORD100000	2025-01-25	CUST2796	Home & Kitchen	Cookware Set	
1	ORD100001	2025-08-28	CUST9669	Beauty	Hair Dryer	
2	ORD100002	2025-02-27	CUST5808	Electronics	Tablet	
3	ORD100003	2025-02-24	CUST5889	Electronics	Headphones	
4	ORD100004	2025-06-15	CUST9005	Clothing	Saree	

In [4]:
```
#read last 5 rows
df.tail()
```

Out[4]:

	Order_ID	Date	Customer_ID	Product_Category	Product_Name
14995	ORD114995	2025-04-12	CUST2822	Beauty	Lipstick
14996	ORD114996	2025-08-29	CUST6143	Beauty	Shampoo
14997	ORD114997	2025-01-27	CUST6747	Books	Science Textbook
14998	ORD114998	2025-06-21	CUST2748	Beauty	Hair Dryer
14999	ORD114999	2025-08-07	CUST9174	Home & Kitchen	Mixer Grinder

Data Preparation & Cleaning Process

In [5]:
```
#shape of Data set
df.shape
```

Out[5]:
```
(15000, 14)
```

In [6]:
```
print(f"The Datasets have \nTotal Number of Rows: {df.shape[0]}  \nTotal Numbe
```

```
The Datasets have
Total Number of Rows: 15000
Total Numbers of columns 14
```

In [7]:
```
#Name of Columns
df.columns.tolist()
```

Out[7]:
```
['Order_ID',
 'Date',
 'Customer_ID',
 'Product_Category',
 'Product_Name',
 'Quantity',
 'Unit_Price_INR',
 'Total_Sales_INR',
 'Payment_Method',
 'Delivery_Status',
 'Review_Rating',
 'Review_Text',
 'State',
 'Country']
```

In [8]:
```
#check info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   Order_ID          15000 non-null  object 
 1   Date              15000 non-null  object 
 2   Customer_ID       15000 non-null  object 
 3   Product_Category  15000 non-null  object 
 4   Product_Name      15000 non-null  object 
 5   Quantity          15000 non-null  int64  
 6   Unit_Price_INR    15000 non-null  float64
 7   Total_Sales_INR   15000 non-null  float64
 8   Payment_Method    15000 non-null  object 
 9   Delivery_Status   15000 non-null  object 
 10  Review_Rating     15000 non-null  int64  
 11  Review_Text       15000 non-null  object 
 12  State             15000 non-null  object 
 13  Country           15000 non-null  object 
dtypes: float64(2), int64(2), object(10)
memory usage: 1.6+ MB
```

In [9]:
```
#check data type
df.dtypes
```

Out[9]:
```
Order_ID            object
Date                object
Customer_ID         object
Product_Category    object
Product_Name        object
Quantity             int64
Unit_Price_INR     float64
Total_Sales_INR    float64
Payment_Method      object
Delivery_Status     object
Review_Rating        int64
Review_Text         object
State               object
Country             object
dtype: object
```

In [10]:
```
#convert date Data Type in Datetime
df["Date"]= pd.to_datetime(df["Date"])
```

In [11]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 15000 entries, 0 to 14999
Data columns (total 14 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Order_ID          15000 non-null  object
 1   Date              15000 non-null  datetime64[ns]
 2   Customer_ID       15000 non-null  object
 3   Product_Category  15000 non-null  object
 4   Product_Name      15000 non-null  object
 5   Quantity          15000 non-null  int64
 6   Unit_Price_INR    15000 non-null  float64
 7   Total_Sales_INR   15000 non-null  float64
 8   Payment_Method    15000 non-null  object
 9   Delivery_Status   15000 non-null  object
 10  Review_Rating     15000 non-null  int64
 11  Review_Text       15000 non-null  object
 12  State             15000 non-null  object
 13  Country           15000 non-null  object
dtypes: datetime64[ns](1), float64(2), int64(2), object(9)
memory usage: 1.6+ MB
```

In [12]:
```
#Check missing values
df.isnull().sum()
```

Out[12]:
```
Order_ID            0
Date                0
Customer_ID         0
Product_Category    0
Product_Name        0
Quantity            0
Unit_Price_INR      0
Total_Sales_INR     0
Payment_Method      0
Delivery_Status     0
Review_Rating       0
Review_Text         0
State               0
Country             0
dtype: int64
```

In [13]:
```
#check duplicate rows
total_duplicate=df.duplicated().sum()
print(f"Number of Total Duplicate rows {total_duplicate}")
```

```
Number of Total Duplicate rows 0
```

In [14]:
```
#Add Feature columns
df["Month"] = df["Date"].dt.month_name().str[:3]
df["Week"] = df["Date"].dt.day_name().str[:3]
df["Days"] = df["Date"].dt.day
```

Exploratory Data Analysis & Visualization

In [15]:
```
df.describe().T
```

Out[15]:

	count	mean	min	25%	50%
Date	15000	2025-07-02 04:46:56.639999744	2025-01-01 00:00:00	2025-04-02 18:00:00	2025-07-03 00:00:00
Quantity	15000.0	2.984667	1.0	2.0	3.0
Unit_Price_INR	15000.0	24955.313715	202.57	12512.9375	24878.755
Total_Sales_INR	15000.0	74544.120233	204.05	27087.8525	57293.57
Review_Rating	15000.0	3.040133	1.0	2.0	3.0
Days	15000.0	15.602	1.0	8.0	16.0

In [16]:
```
df.describe(include="object").T
```

Out[16]:

	count	unique	top	freq
Order_ID	15000	15000	ORD100000	1
Customer_ID	15000	7259	CUST4795	8
Product_Category	15000	5	Electronics	3036
Product_Name	15000	25	Children's Book	636
Payment_Method	15000	4	Cash on Delivery	3827
Delivery_Status	15000	3	Delivered	5075
Review_Text	15000	25	Satisfied with the product	658
State	15000	28	Sikkim	596
Country	15000	1	India	15000
Month	15000	12	Aug	1312
Week	15000	7	Sun	2194

In [17]:
```
#set theme for charts
sns.set_theme(style="darkgrid")
```

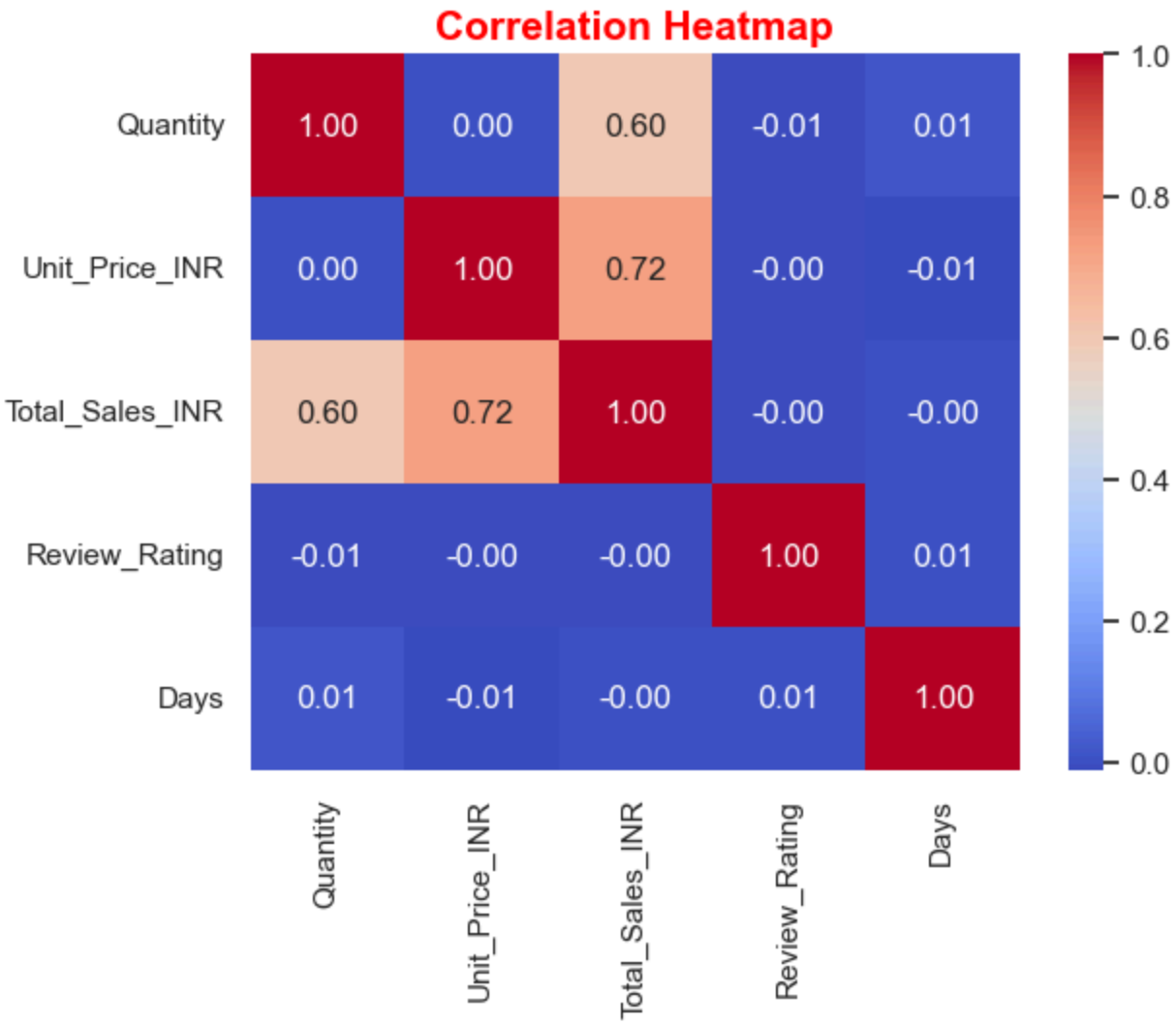
Distribution of Numerical Columns

In [18]:
```
plt.figure(figsize=(8,6))
sns.histplot(x = "Total_Sales_INR",data = df)
plt.title("Distribution of Price",color = "Red", fontweight = "bold")
plt.xlabel("Price")
plt.ylabel("Frequency")
plt
```

Out[18]: <module 'matplotlib.pyplot' from 'c:\\Users\\dell\\AppData\\Local\\Programs\\Python\\Python314\\Lib\\site-packages\\matplotlib\\pyplot.py'>

correlation Heatmap

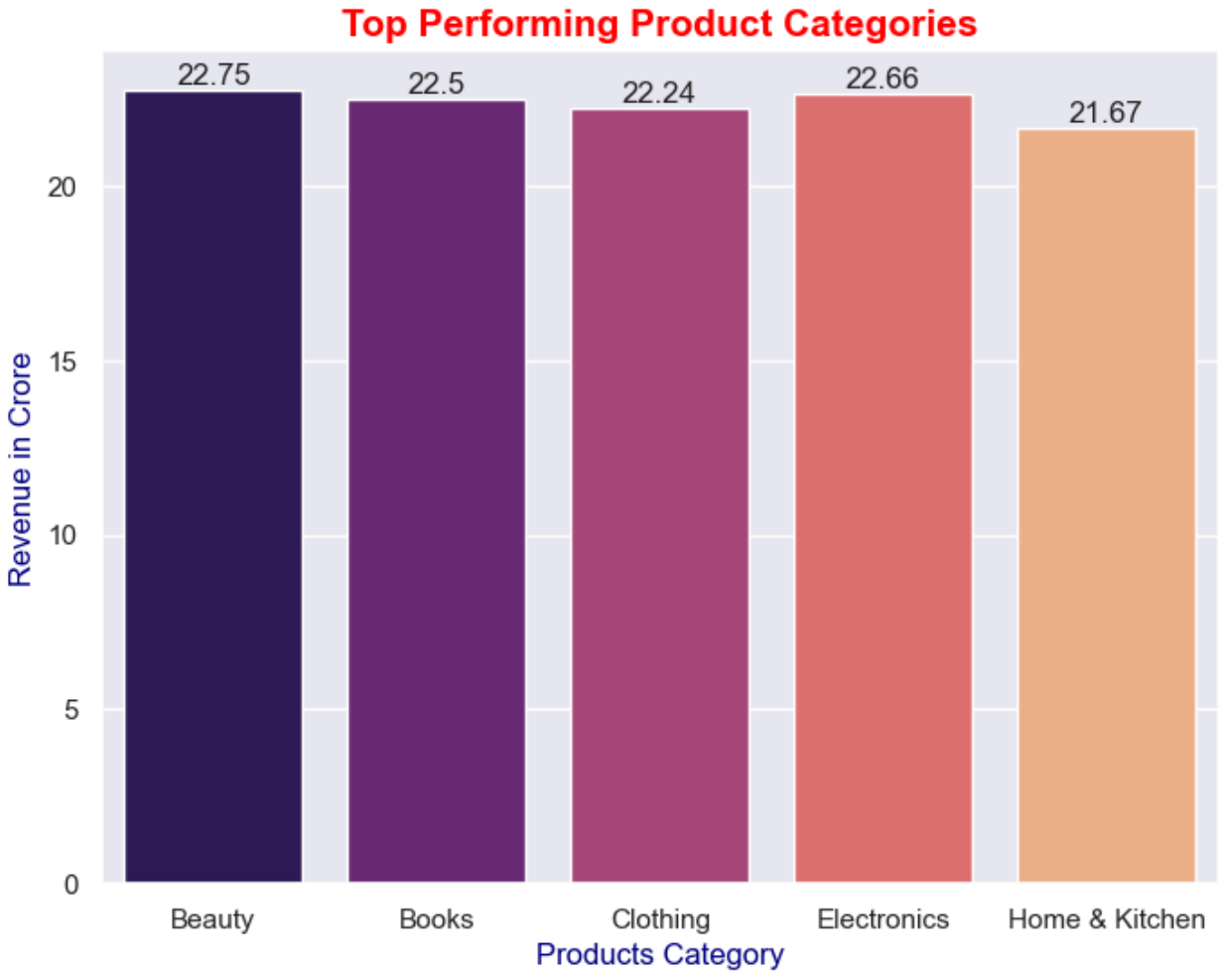
In [20]:
```
num_col= df.select_dtypes(["int", "float"])
correlation_matrix =  num_col.corr()
sns.heatmap(correlation_matrix,annot=True,
fmt=".2f",cmap="coolwarm")
plt.title("Correlation Heatmap", color="red",
size=15, fontweight="bold")
plt.show()
```

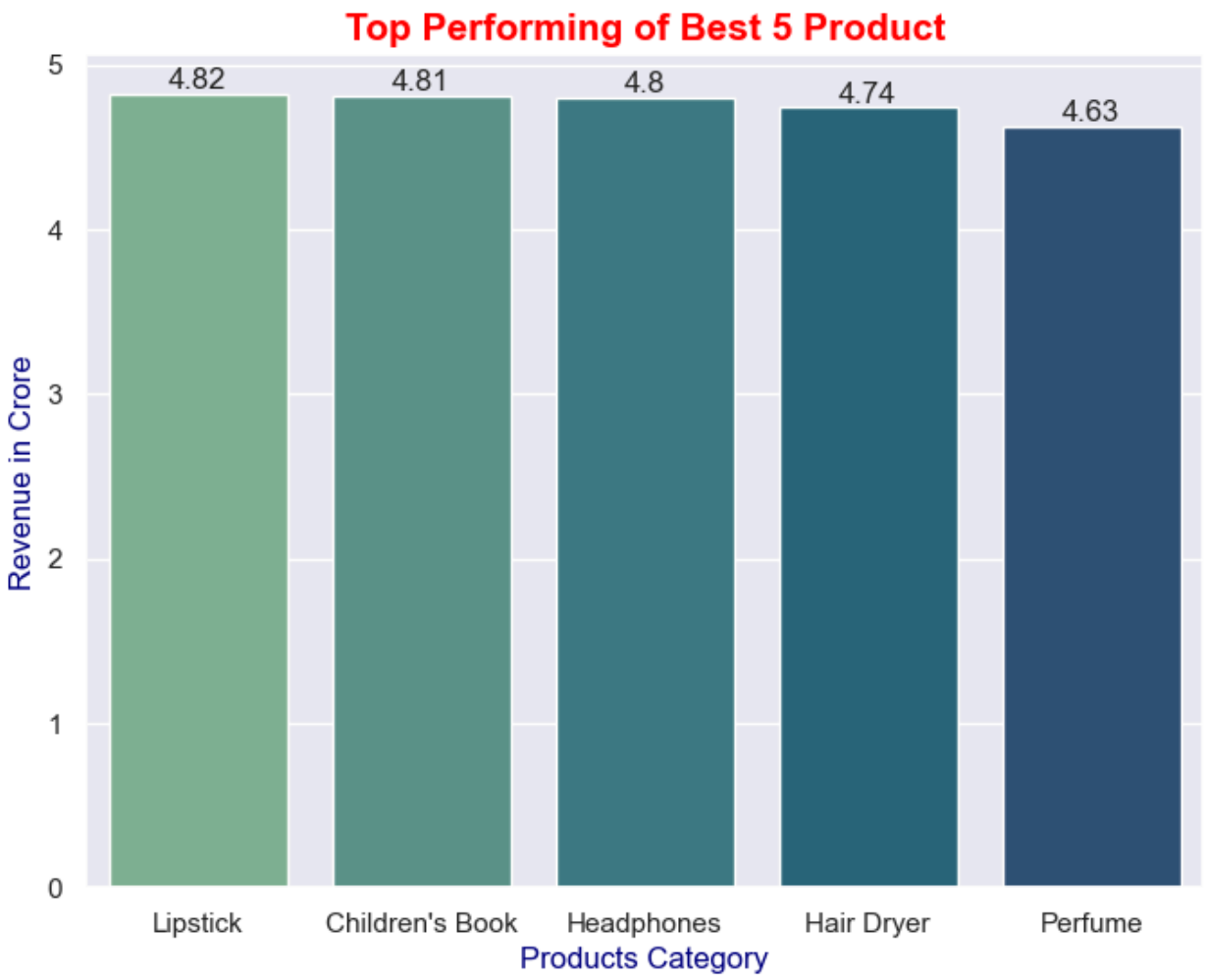


Top Performing Categories & Products

In [21]:
```
#Top Performance Products Category
Top_Cat_Product= df.groupby("Product_Category")["Total_Sales_INR"].sum().reset
#Convert in Crore
Top_Cat_Product["Total_Sales_INR"] = (Top_Cat_Product["Total_Sales_INR"]/1e7).
#Show in Charts
plt.figure(figsize=(8,6))
ax =sns.barplot(x = "Product_Category", y = "Total_Sales_INR",
data = Top_Cat_Product, palette="magma")
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Top Performing Product Categories", size=15,
color="red", fontweight="bold")
plt.xlabel("Products Category", size =12,
color="navy")
plt.ylabel("Revenue in Crore", size =12,
color="navy")
plt.show()
```

```
#Top 5 best performing Products
Top_Product= df.groupby("Product_Name")["Total_Sales_INR"].sum().reset_index()
Top_Product = Top_Product.sort_values(by= "Total_Sales_INR", ascending=False)[
#Convert in Crore
Top_Product["Total_Sales_INR"] = (Top_Product["Total_Sales_INR"]/1e7).round(2)
#Show in Charts
plt.figure(figsize=(8,6))
ax =sns.barplot(x = "Product_Name", y = "Total_Sales_INR",
data = Top_Product, palette="crest")
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Top Performing of Best 5 Product", size=15,
color="red", fontweight="bold")
plt.xlabel("Products Category", size =12,
color="navy")
plt.ylabel("Revenue in Crore", size =12,
color="navy")
plt.show()
```



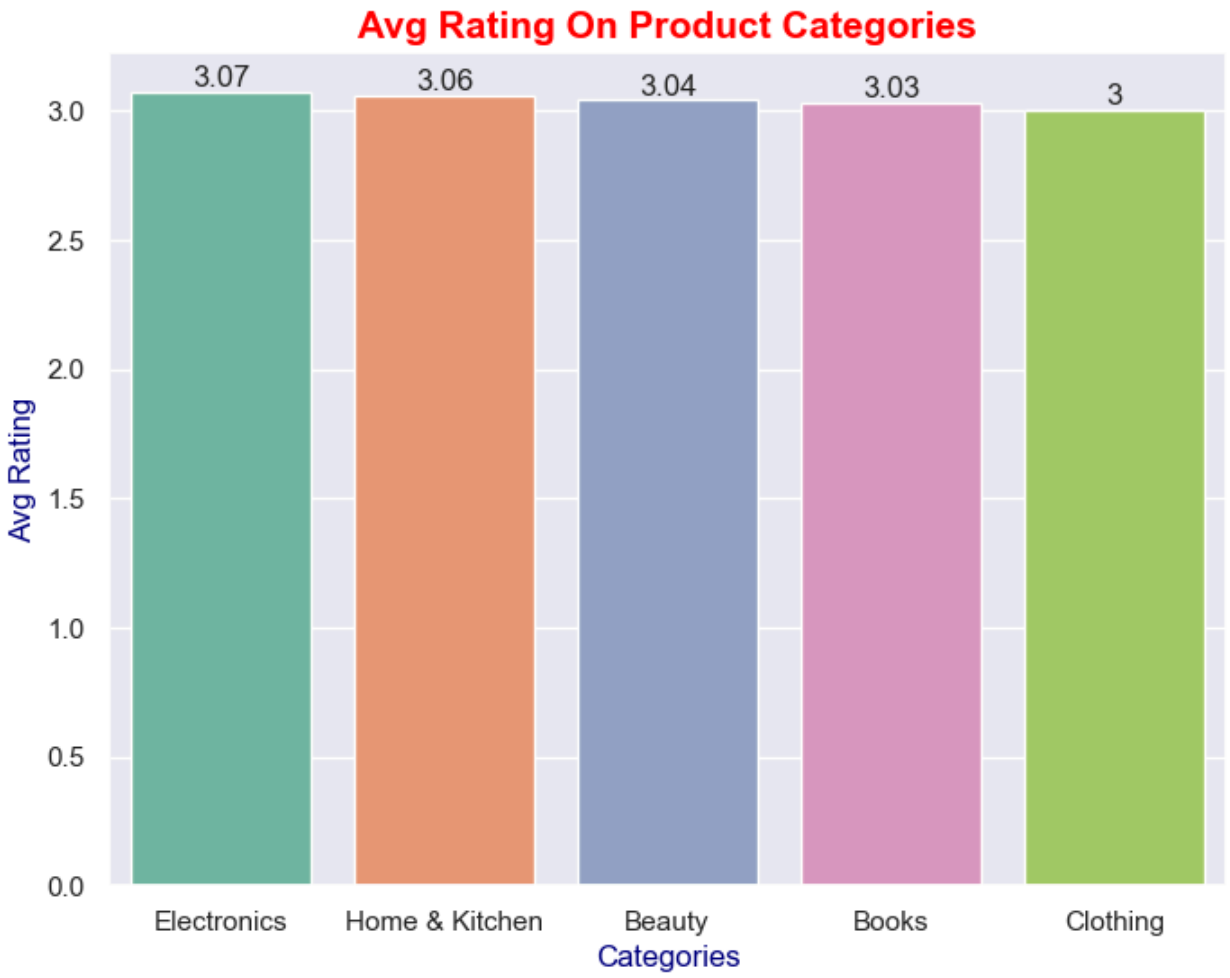


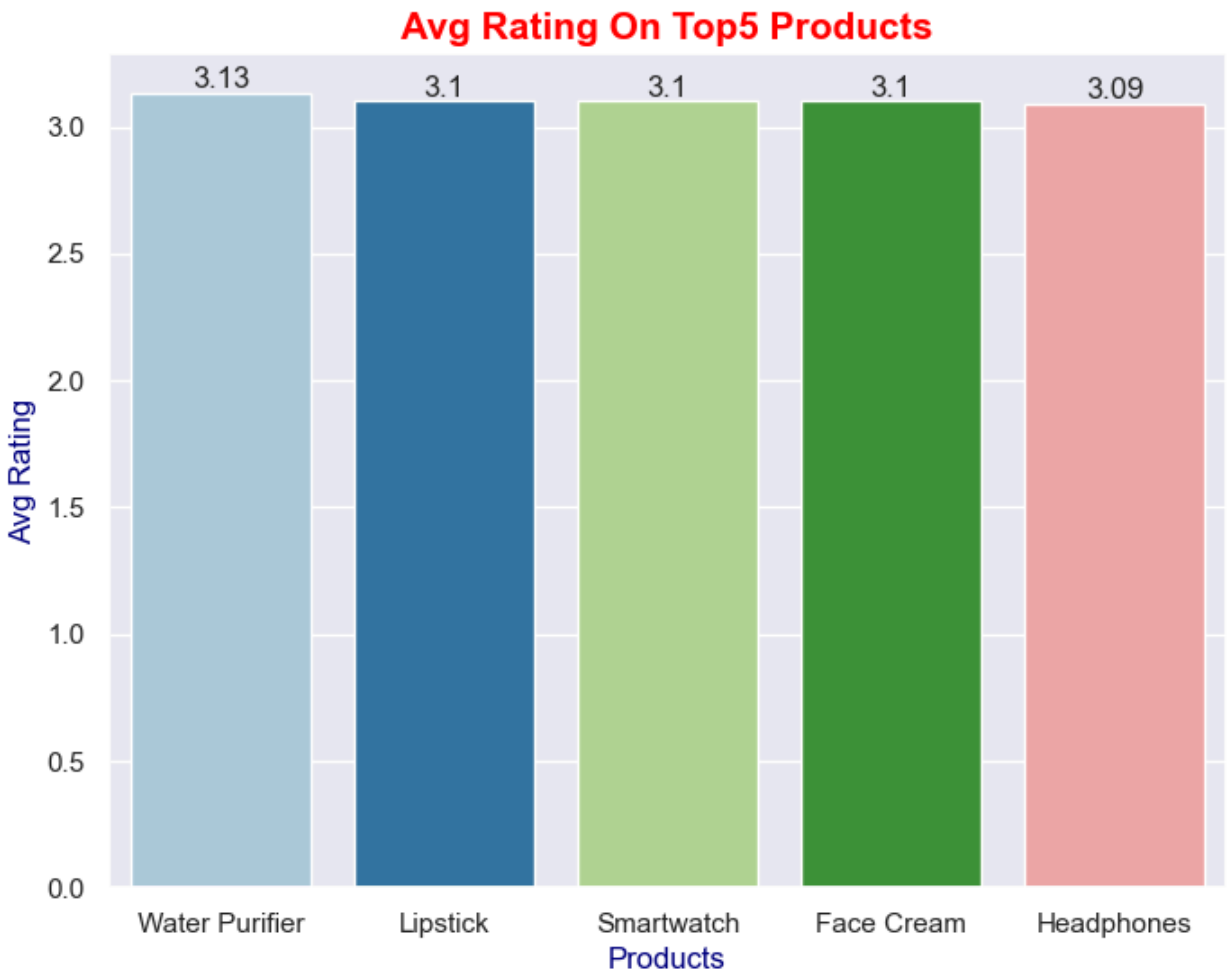
Customer Engagement and Feedback

In [23]:
```
#Avg Rating on Category
Avg_Category_Rating = df.groupby("Product_Category")["Review_Rating"].mean().r
Avg_Category_Rating = Avg_Category_Rating.sort_values(by="Review_Rating", asce
plt.figure(figsize=(8,6))
ax=sns.barplot(x = "Product_Category",y="Review_Rating",
data = Avg_Category_Rating, palette="Set2")
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Avg Rating On Product Categories",size = 15,
color="red", fontweight="bold")
plt.xlabel("Categories", size =12, color="navy")
plt.ylabel("Avg Rating", size =12, color="navy")
plt.show()

#Avg Rating on Products
Avg_Product_Rating = df.groupby("Product_Name")["Review_Rating"].mean().reset_
Avg_Product_Rating = Avg_Product_Rating.sort_values(by="Review_Rating", ascend
plt.figure(figsize=(8,6))
ax=sns.barplot(x = "Product_Name",y="Review_Rating",
data = Avg_Product_Rating, palette="Paired")
```

```
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Avg Rating On Top5 Products",size = 15,
color="red", fontweight="bold")
plt.xlabel("Products", size =12, color="navy")
plt.ylabel("Avg Rating", size =12, color="navy")
plt.show()
```





In [31]:
```
Review_Dist = df.groupby("Review_Text")["Review_Rating"].sum().reset_index()
Review_Dist = Review_Dist.sort_values(by="Review_Rating",ascending=False)[0:10
Review_Dist
```

Out[31]:

	Review_Text	Review_Rating
4	Excellent product!	3265
6	Fantastic quality!	3205
8	Highly recommend!	3185
23	Worth every rupee!	3075
9	Loved it!	2910
18	Satisfied with the product	2632
7	Good quality	2452
15	Pretty decent	2436
24	Would buy again	2376
20	Value for money	2192

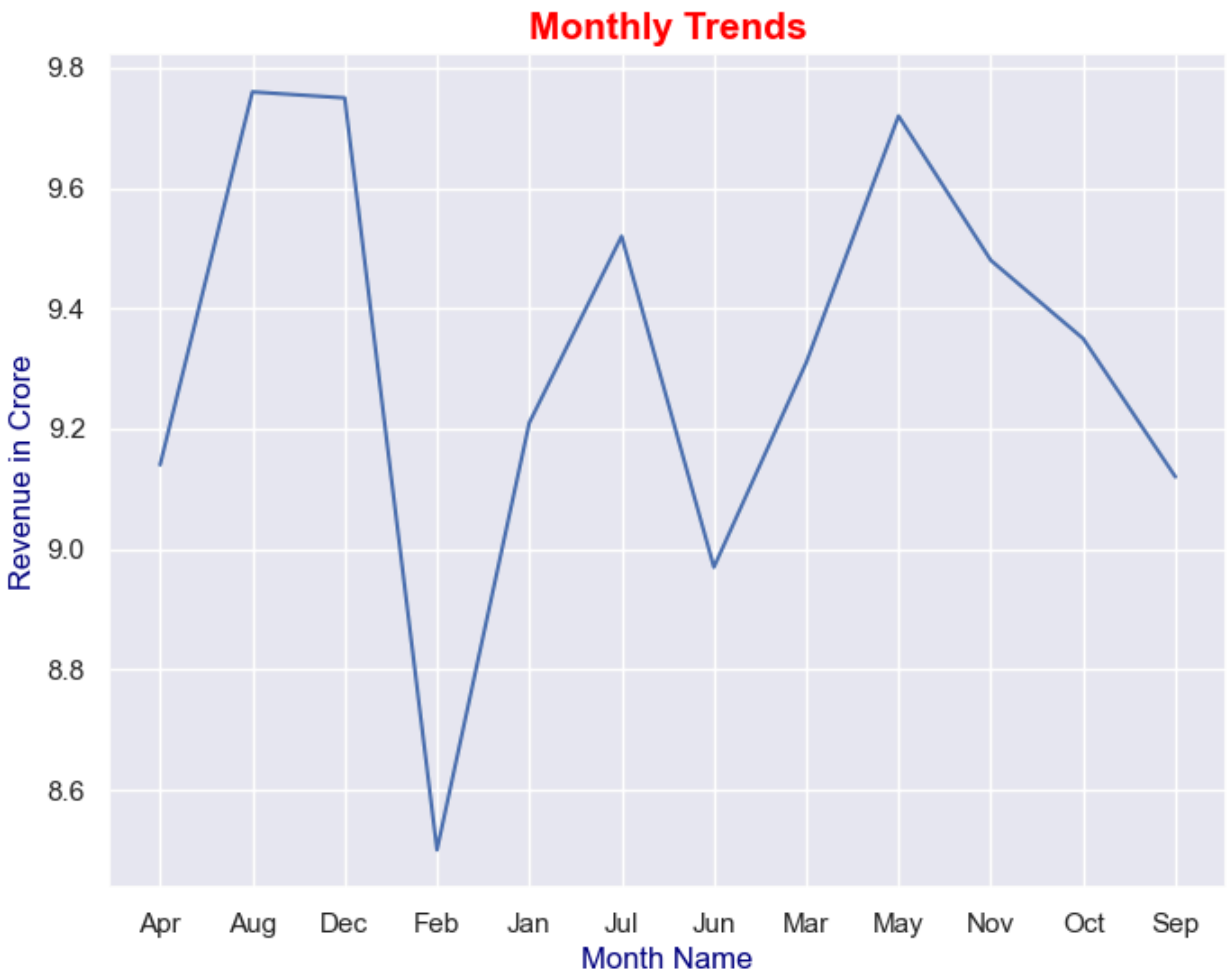
Sales Trends Analysis

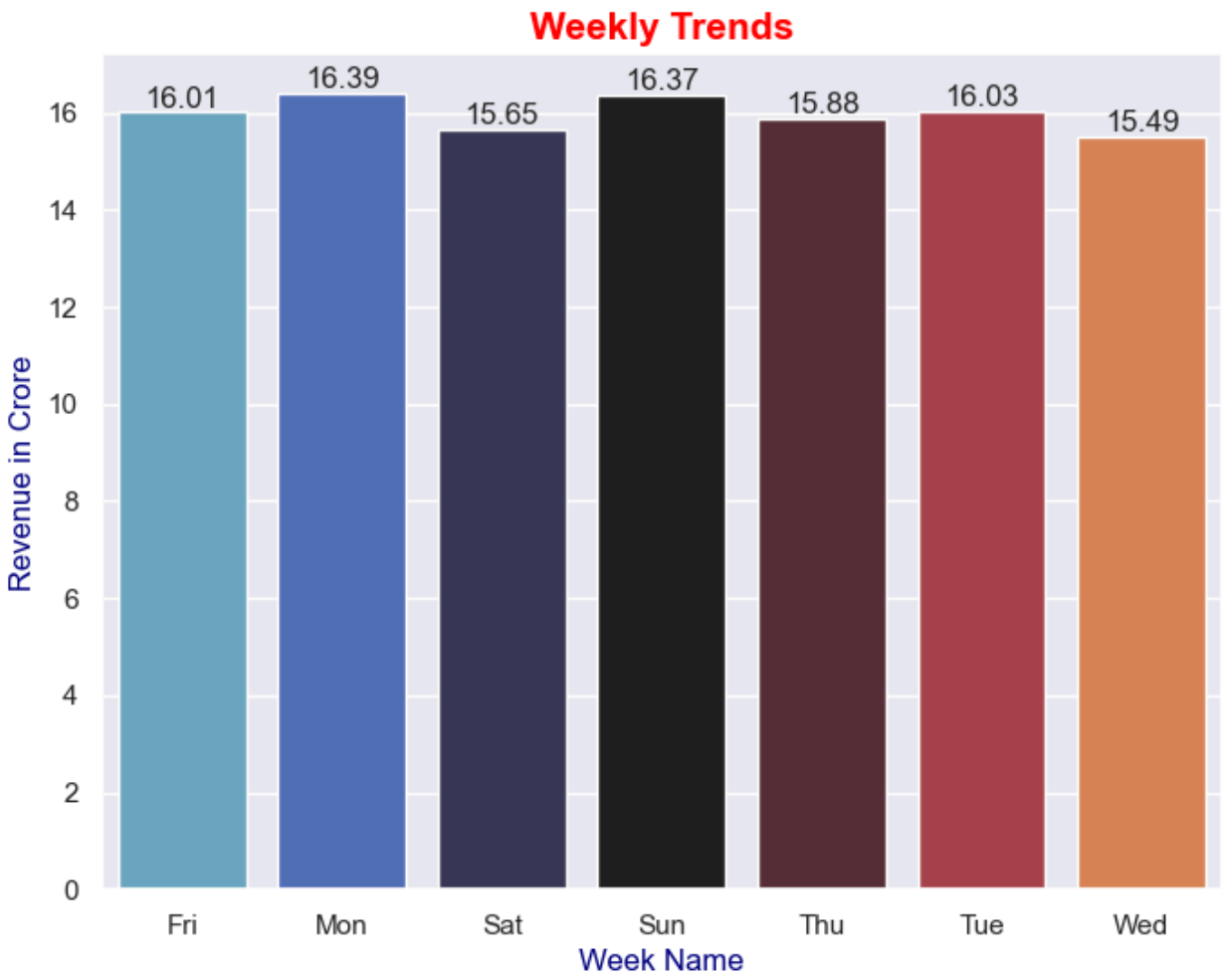
In [38]:

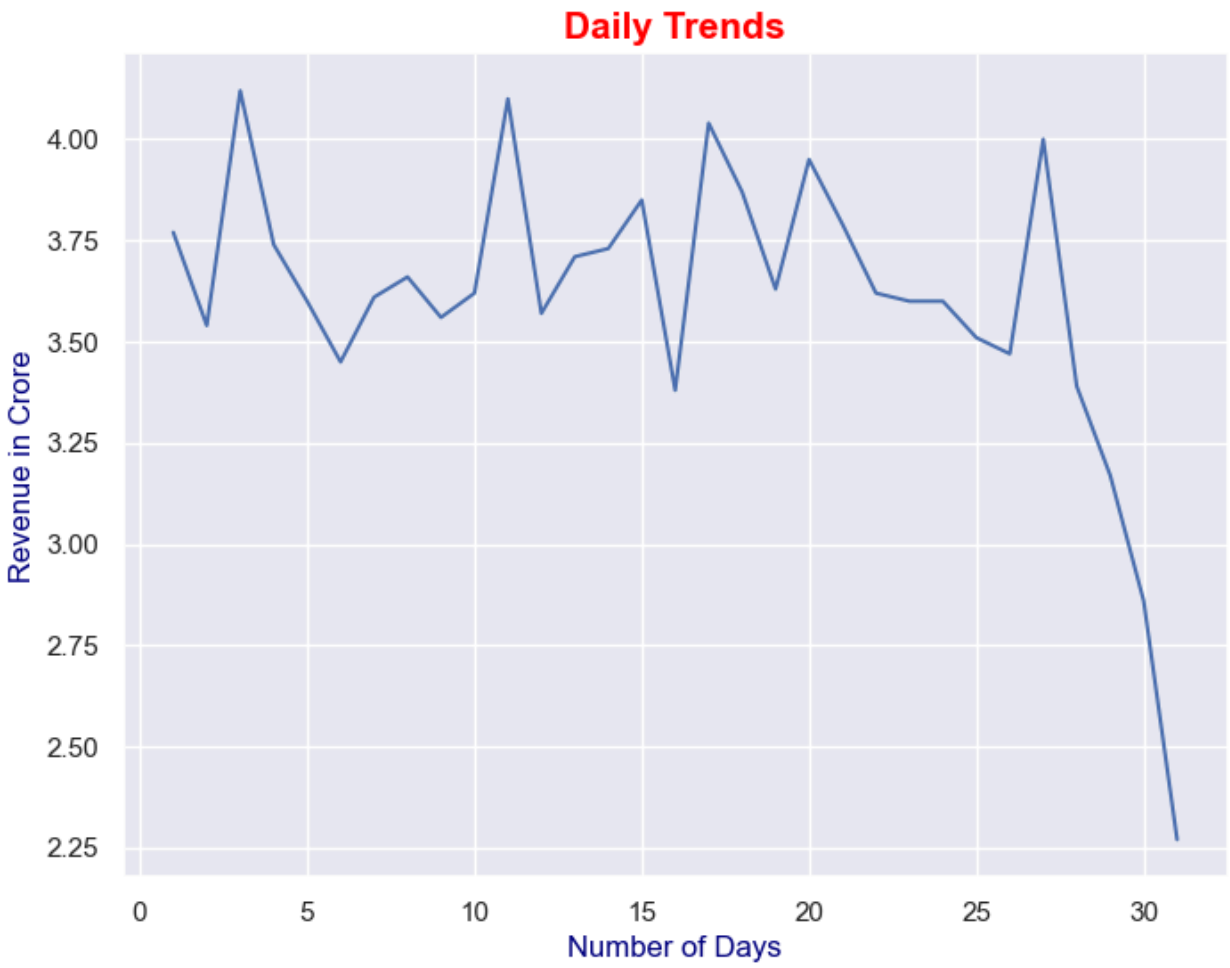
```
#MonthlyTrends
Monthly_Trends= df.groupby("Month")["Total_Sales_INR"].sum().reset_index()
# Convert in Crore
Monthly_Trends["Total_Sales_INR"]  = (Monthly_Trends["Total_Sales_INR"]/1e7).r
plt.figure(figsize= (8,6))
sns.lineplot(x = "Month", y = "Total_Sales_INR", data = Monthly_Trends)
plt.title("Monthly Trends", color="red",
size=15, fontweight="bold")
plt.xlabel("Month Name", size =12,
color="navy")
plt.ylabel("Revenue in Crore",size =12,
color="navy")
plt.show()

#Weekly Trends
Weekly_Trends= df.groupby("Week")["Total_Sales_INR"].sum().reset_index()
#Convery in Crore
Weekly_Trends["Total_Sales_INR"]  = (Weekly_Trends["Total_Sales_INR"]/1e7).rou
plt.figure(figsize=(8,6))
ax=sns.barplot(x = "Week",y="Total_Sales_INR",
data = Weekly_Trends, palette="icefire")
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Weekly Trends",size = 15,
color="red", fontweight="bold")
plt.xlabel("Week Name", size =12, color="navy")
plt.ylabel("Revenue in Crore", size =12, color="navy")
plt.show()

#Daily Trends
Daily_Trends= df.groupby("Days")["Total_Sales_INR"].sum().reset_index()
plt.figure(figsize= (8,6))
Daily_Trends["Total_Sales_INR"]  = (Daily_Trends["Total_Sales_INR"]/1e7).round
sns.lineplot(x = "Days", y = "Total_Sales_INR", data = Daily_Trends)
plt.title("Daily Trends", color="red",
size=15, fontweight="bold")
plt.xlabel("Number of Days", size =12,
color="navy")
plt.ylabel("Revenue in Crore",size =12,
color="navy")
plt.show()
```


Monthly Trends
Revenue in Crore
9.8
9.6
9.4
9.2
9.0
8.8
8.6
Apr
Aug
Dec
Feb
Jan
Jul
Jun
Mar
May
Nov
Oct
Sep
Month Name

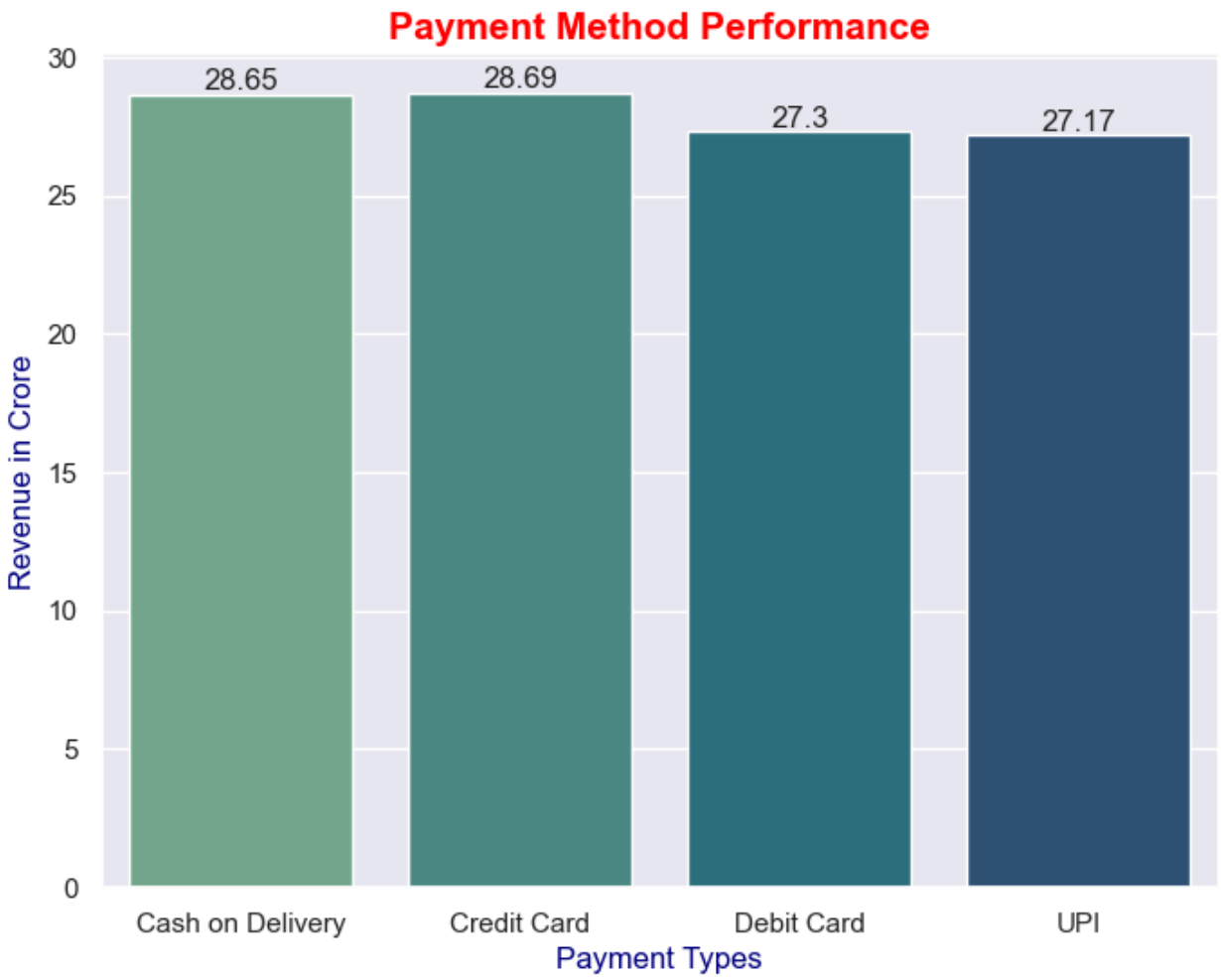

Weekly Trends
16.01
16.39
15.65
16.37
15.88
16.03
15.49
16
14
12
10
8
6
4
2
0
Revenue in Crore
Fri
Mon
Sat
Sun
Thu
Tue
Wed
Week Name



Payment Method Performance

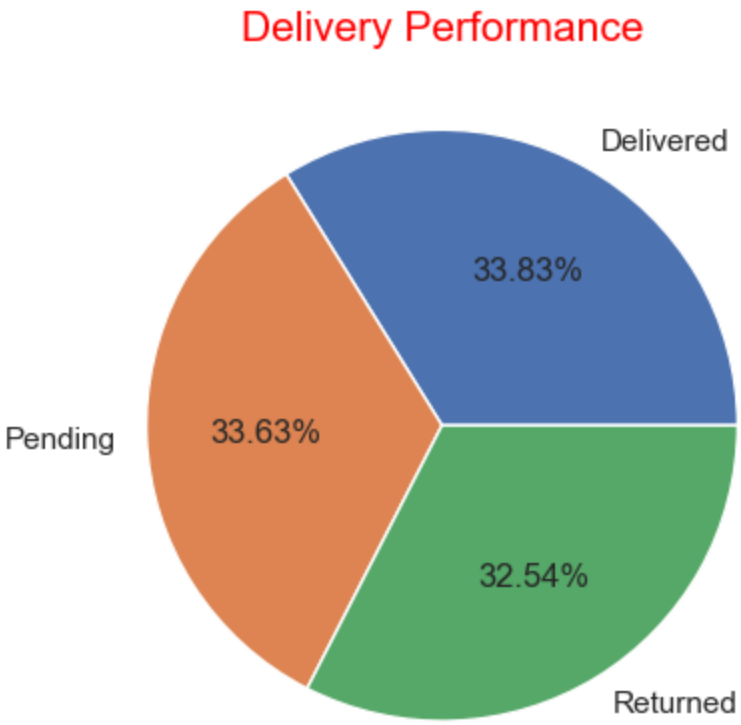
In [39]:

```
Payment= df.groupby("Payment_Method")["Total_Sales_INR"].sum().reset_index()
Payment["Total_Sales_INR"] = (Payment["Total_Sales_INR"]/1e7).round(2)
plt.figure(figsize=(8,6))
ax =sns.barplot(x = "Payment_Method", y ="Total_Sales_INR",
data = Payment, palette="crest")
for bars in ax.containers:
    ax.bar_label(bars)
plt.title("Payment Method Performance",
size =15, color="red", fontweight="bold")
plt.xlabel("Payment Types", size =12,
color="Navy")
plt.ylabel("Revenue in Crore",size =12,
color="navy")
plt.show()
```



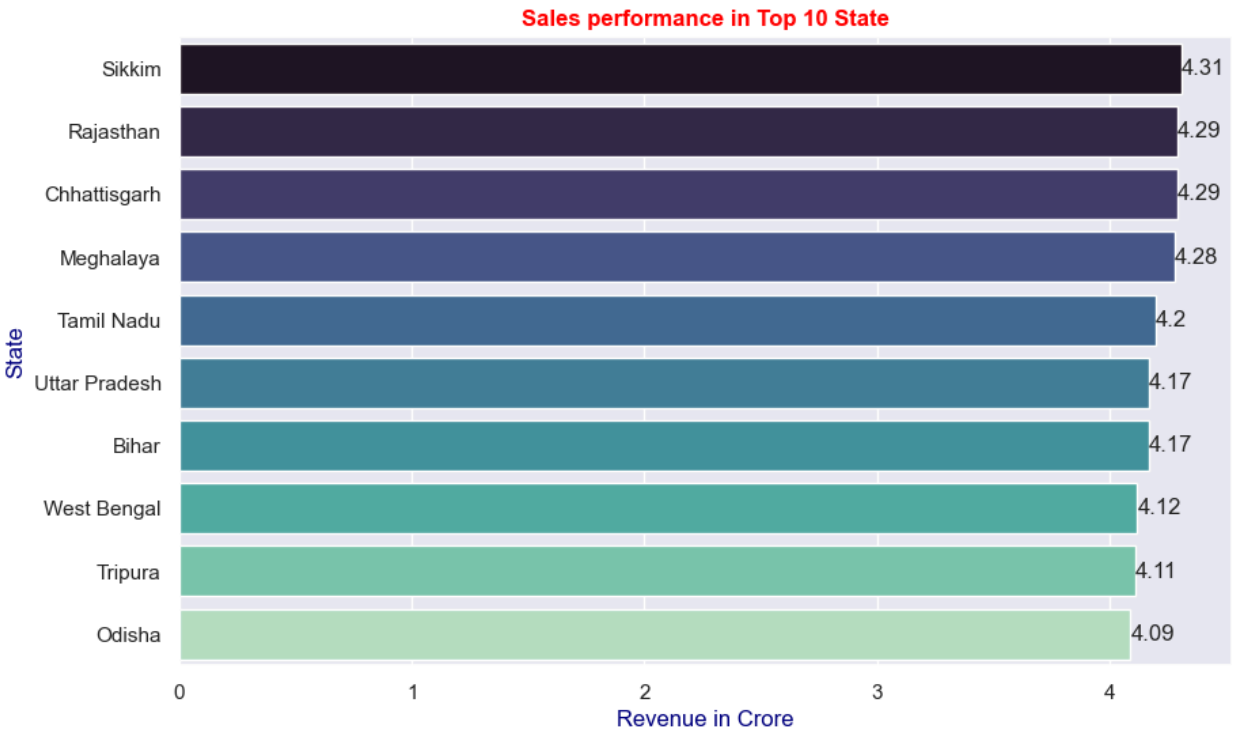
Order Fulfillment Status

In [ ]:
```
Status = df["Delivery_Status"].value_counts().reset_index()
x = Status["count"]
y = Status["Delivery_Status"]
plt.pie(x, labels =y,autopct="%0.2f%%")
plt.title("Delivery Performance", size =15,
color= "red")
plt.show()
```



Regional Sales Insights

In [42]:
```
State = df.groupby("State")["Total_Sales_INR"].sum().reset_index()
State = State.sort_values(by ="Total_Sales_INR", ascending=False)[:10]
State["Total_Sales_INR"] = (State["Total_Sales_INR"]/1e7).round(2)
plt.figure(figsize=(10,6))
ax =sns.barplot(x ="Total_Sales_INR",y= "State",
data = State, palette="mako")
for bars in ax.containers:
     ax.bar_label(bars)
plt.title("Sales performance in Top 10 State",
size =12, color="red", fontweight="bold")
plt.xlabel("Revenue in Crore", size =12,
color="navy")
plt.ylabel("State" ,size=12, color="navy")
plt.show()
```


Sales performance in Top 10 State
Sikkim
Rajasthan
Chhattisgarh
Meghalaya
Tamil Nadu
Uttar Pradesh
Bihar
West Bengal
Tripura
Odisha
4.31
4.29
4.29
4.28
4.2
4.17
4.17
4.12
4.11
4.09
State
0
1
2
3
4
Revenue in Crore