About Dataset

Analyzing Business Performance and Strategic Planning:

The online retail store has hired you as a consultant to analyse their data and provide insights to the CEO and CMO. The management wants to identify the major factors contributing to revenue and plan strategically for next year. They want to view metrics from both operations and marketing perspectives and seek guidance on areas performing well. They also want to view demographic-based metrics. The meeting with the CEO and CMO is scheduled for next month, and you need to provide analytics and insights to evaluate the current business performance and suggest metrics for expansion. As a consultant tasked with analyzing the data of the online retail store, your role is pivotal in understanding and optimizing the company's revenue generation. You'll be diving deep into the available data to uncover insights that will guide the company's strategic decisions for the upcoming year. The ultimate goal is to identify the key drivers of revenue growth, both from operational and marketing perspectives.

Insights and Recommendations:

In the meeting with the CEO and CMO, you'll present your findings and recommendations. You might highlight:

- 1. Top Performing Categories: Identify which product categories are driving the most revenue and suggest.
- 1. Geographic Insights: Showcase regions where revenue is particularly strong.
- 1. Top Country & Revenue: Anlysis The Best Selling Product on Top 10 Countries
- 1. Best Year Sale: Which Year is Best Year For Sale.
- 1. Best Time & Day For Sale : Best Time For Sale & Best Day For Sale
- 1. Best Selling Hour Country Wise : Which Hour is Best Selling Hour on Top 10 Growing Revenue Country
- 1. Best Day For Sale Country Wise : Which Day is Best For Selling On Country

Loading The Standered Libraries

In [102…
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
warnings.filterwarnings('ignore')
```

Loading the Dataset.

In [7]:
```
data = pd.read_csv("Online Retail Data Set.csv",encoding = 'cp1252')
data.sample(10)
```

Out[7]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
8825	537153	21696	SMALL SILVER TRELLIS CANDLEPOT	6	05-12-2010 13:03	2.95	16718.0	United Kingdom
193921	553546	21770	OPEN CLOSED METAL SIGN	36	17-05-2011 15:42	4.25	12415.0	Australia
58529	541250	21520	BOOZE & WOMEN GREETING CARD	12	16-01-2011 13:19	0.42	13700.0	United Kingdom
15883	537642	21218	RED SPOTTY BISCUIT TIN	2	07-12-2010 15:33	8.47	NaN	United Kingdom
268299	560399	82583	NaN	17	18-07-2011 14:18	0.00	NaN	United Kingdom
272478	560772	23173	REGENCY TEAPOT ROSES	1	20-07-2011 16:12	19.96	NaN	United Kingdom
180668	552333	21043	APRON MODERN VINTAGE COTTON	2	09-05-2011 10:38	1.95	15039.0	United Kingdom
541213	581494	23388	WOODLAND MINI BACKPACK	4	09-12-2011 10:13	4.15	12518.0	Germany
366797	568793	21668	RED STRIPE CERAMIC DRAWER KNOB	24	29-09-2011 09:55	1.45	16011.0	United Kingdom
355500	567939	84755	COLOUR GLASS T-LIGHT HOLDER HANGING	144	22-09-2011 18:15	0.55	16715.0	United Kingdom

Understanding The Dataset

In [8]:
```
data.shape
```

Out[8]:
```
(541909, 8)
```

In [10]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
 #   Column       Non-Null Count   Dtype  
---  ------       --------------   -----  
 0   InvoiceNo    541909 non-null  object 
 1   StockCode    541909 non-null  object 
 2   Description  540455 non-null  object 
 3   Quantity     541909 non-null  int64  
 4   InvoiceDate  541909 non-null  object 
 5   UnitPrice    541909 non-null  float64
 6   CustomerID   406829 non-null  float64
 7   Country      541909 non-null  object 
dtypes: float64(2), int64(1), object(5)
memory usage: 33.1+ MB
```

In [11]:
```
data.describe().T
```

Out[11]:

	count	mean	std	min	25%	50%	75%	max
Quantity	541909.0	9.552250	218.081158	-80995.00	1.00	3.00	10.00	80995.0
UnitPrice	541909.0	4.611114	96.759853	-11062.06	1.25	2.08	4.13	38970.0
CustomerID	406829.0	15287.690570	1713.600303	12346.00	13953.00	15152.00	16791.00	18287.0

Data Preprocessing

Let's Check the Missing And Duplicate's Values

In [13]:
```
data.isnull().sum()
```

Out[13]:
```
InvoiceNo           0
StockCode           0
Description      1454
Quantity            0
InvoiceDate         0
UnitPrice           0
CustomerID     135080
Country             0
dtype: int64
```

For Learning Perpose We'll Use Two Method to Filling Missing Value of Our Dataset

- 1. sklearn.impute ------>>> The First Method We'll Use Sklearn.Impute
- 1. dataset.fillna()----->>> And The Second One We Are Gonna to Use Which Is fillna() Method.

Note :- These Both Method Is Usefull For Filling Missing Value's Of Dataset

* 1. With Sklearn

In [21]:
```
from sklearn.impute import SimpleImputer
si = SimpleImputer(strategy = 'most_frequent')
si
```

Out[21]:
```
SimpleImputer(strategy='most_frequent')
```

In [22]:
```
data['Description'] = si.fit_transform(data[['Description']])
```

In [24]:
```
data['Description'].isnull().sum()
```

Out[24]:
```
0
```

* 2. With Fillna()

In [25]:
```
data['CustomerID'] = data['CustomerID'].fillna(0)
```

In [26]:
```
data['CustomerID'].isnull().sum()
```

Out[26]:
```
0
```

Changing Some Column DataType

In [30]:
```
data['Quantity'] = pd.to_numeric(data['Quantity'])
data['UnitPrice'] = pd.to_numeric(data['UnitPrice'])
data['InvoiceDate'] = pd.to_datetime(data['InvoiceDate'])
```

Adding Some More Column Which Help's To More Visualization About To Data

In [174…
```
data['Month'] = data['InvoiceDate'].dt.month_name()
data['Day_Name'] = data['InvoiceDate'].dt.day_name()
data['Year'] = data['InvoiceDate'].dt.year
data['Time'] = data['InvoiceDate'].dt.time
data['Hour'] = data['InvoiceDate'].dt.hour
```

Droping Some Un-Wanted Columns

In [157…
```
data.drop(['StockCode','CustomerID'],inplace = True,axis = 1)
```

In [175…
```
data.head()
```

Out[175]:

	InvoiceNo	Description	Quantity	InvoiceDate	UnitPrice	Country	Month	Year	Time	revenue	Day_Name	Hour
0	536365	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-01-12 08:26:00	2.55	United Kingdom	January	2010	08:26:00	15.30	Tuesday	8
1	536365	WHITE METAL LANTERN	6	2010-01-12 08:26:00	3.39	United Kingdom	January	2010	08:26:00	20.34	Tuesday	8
2	536365	CREAM CUPID HEARTS COAT HANGER	8	2010-01-12 08:26:00	2.75	United Kingdom	January	2010	08:26:00	22.00	Tuesday	8
3	536365	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-01-12 08:26:00	3.39	United Kingdom	January	2010	08:26:00	20.34	Tuesday	8
4	536365	RED WOOLLY HOTTIE WHITE HEART.	6	2010-01-12 08:26:00	3.39	United Kingdom	January	2010	08:26:00	20.34	Tuesday	8

Now Our Dataset Ready To Visualize

EXPLORATORY DATA ANALYSIS AND DATA VISULIZATION

Firstly We'll Analysis The Question Should Be Asked Frequently

----------->>>>> Top Performing Categories: Identify which product categories are driving the most revenue and suggest.<<<----------

In [48]:
```
data = data.drop_duplicates()
```

In [53]:
```
data['revenue'] = data['Quantity'] * data['UnitPrice']
```

In [59]:
```
product_sale = data.groupby('Description').sum()['revenue'].reset_index()
```

In [61]:
```
product_sale= product_sale[(product_sale['revenue']!=0)]
product_sale
```

Out[61]:

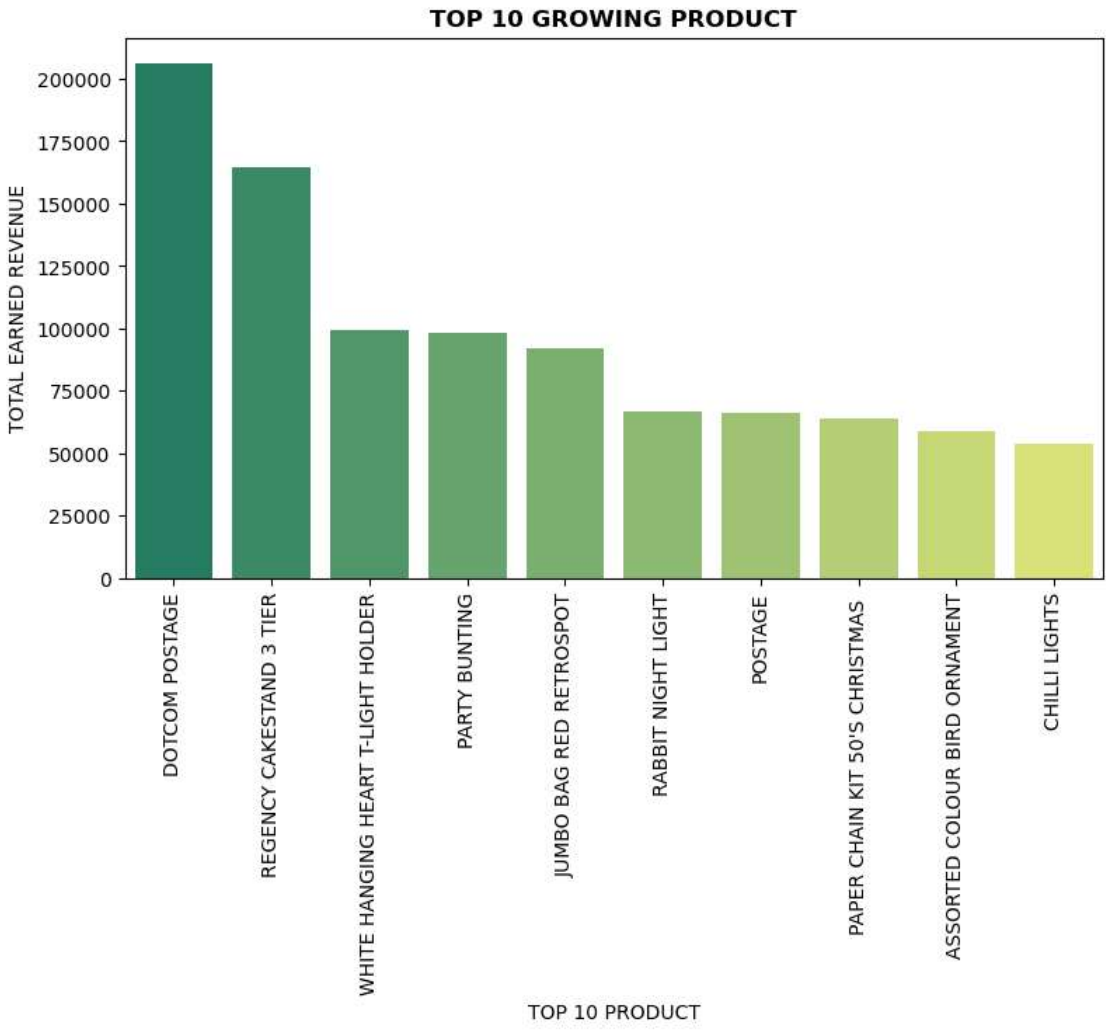
	Description	revenue
0	4 PURPLE FLOCK DINNER CANDLES	285.70
1	50'S CHRISTMAS GIFT BAG LARGE	2341.13
2	DOLLY GIRL BEAKER	2877.50
3	I LOVE LONDON MINI BACKPACK	1624.02
4	I LOVE LONDON MINI RUCKSACK	4.15
...	...	...
4099	ZINC T-LIGHT HOLDER STARS SMALL	4161.31
4100	ZINC TOP 2 DOOR WOODEN SHELF	193.33
4101	ZINC WILLIE WINKIE CANDLE STICK	2702.50
4102	ZINC WIRE KITCHEN ORGANISER	239.97
4103	ZINC WIRE SWEETHEART LETTER TRAY	275.62

4038 rows × 2 columns

Now We'll Check Top 10 Growing Product's As Per Requirement

In [80]:
```
top_10 = product_sale.nlargest(10, 'revenue')
display(top_10)
plt.figure(figsize = (9,5))
sns.barplot(data = top_10,x = 'Description',y = 'revenue',palette = 'summer')
plt.xlabel('TOP 10 PRODUCT')
plt.ylabel('TOTAL EARNED REVENUE')
plt.xticks(rotation=90)
plt.title('TOP 10 GROWING PRODUCT',fontweight = 'bold')
plt.show()
```

	Description	revenue
1098	DOTCOM POSTAGE	206245.48
2915	REGENCY CAKESTAND 3 TIER	164459.49
3918	WHITE HANGING HEART T-LIGHT HOLDER	99612.42
2471	PARTY BUNTING	98243.88
1866	JUMBO BAG RED RETROSPOT	92175.79
2803	RABBIT NIGHT LIGHT	66661.63
2753	POSTAGE	66230.64
2439	PAPER CHAIN KIT 50'S CHRISTMAS	63715.24
244	ASSORTED COLOUR BIRD ORNAMENT	58792.42
773	CHILLI LIGHTS	53746.66

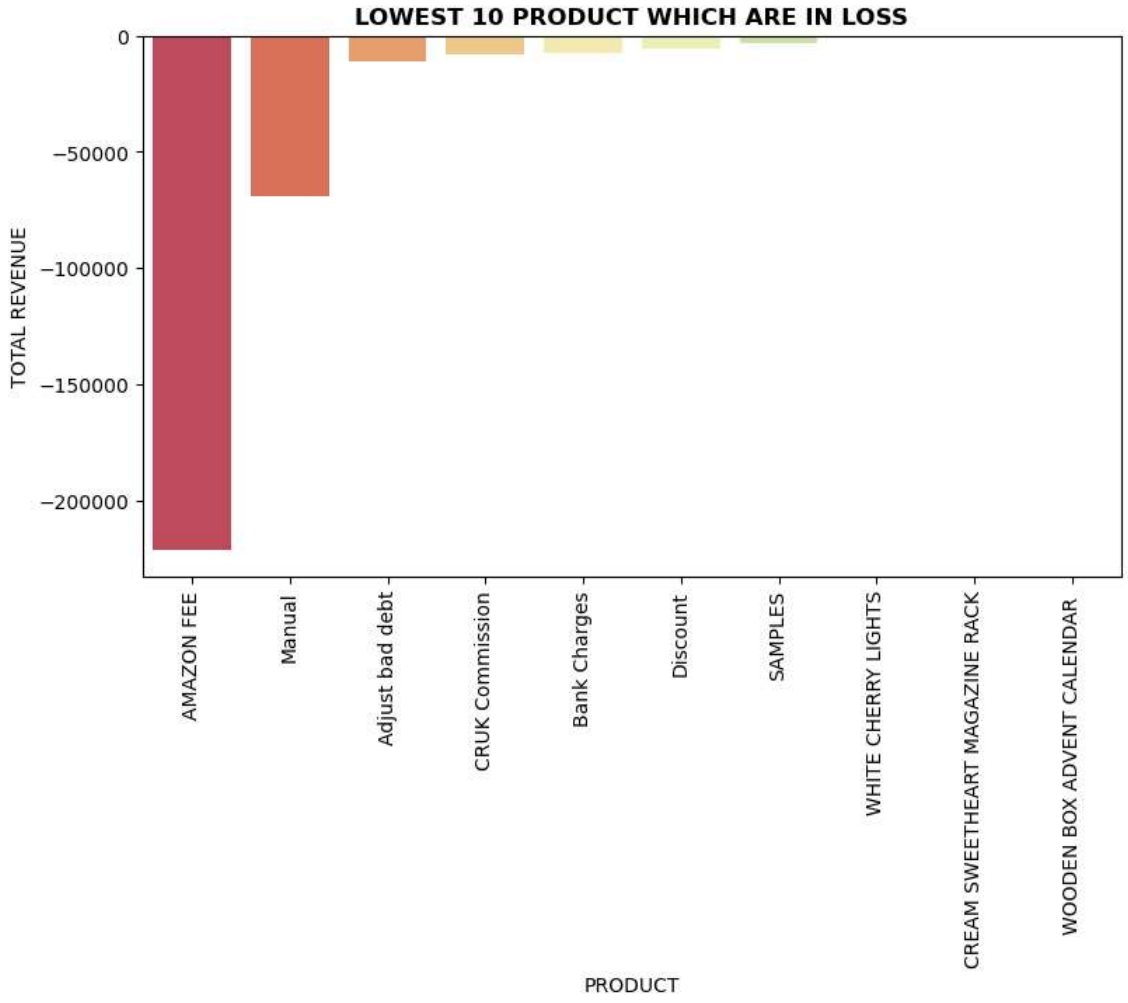


NOW WE'LL CHECK LOWEST 10 PRODUCT APART FROM WHICH REVENUES

In [101…

```
lowest_10 = product_sale.nsmallest(10,'revenue').reset_index()
display(lowest_10)
plt.figure(figsize = (9,5))
sns.barplot(data = lowest_10,x = lowest_10['Description'], y = 'revenue', palette = 'Spectral')
plt.xticks(rotation = 90)
plt.xlabel('PRODUCT')
plt.ylabel('TOTAL REVENUE')
plt.title('LOWEST 10 PRODUCT WHICH ARE IN LOSS' , fontweight= 'bold')
plt.show()
```

	index	Description	revenue
0	171	AMAZON FEE	-221520.500
1	2246	Manual	-69031.640
2	281	Adjust bad debt	-11062.060
3	934	CRUK Commission	-7933.430
4	615	Bank Charges	-7175.639
5	1126	Discount	-5696.220
6	3059	SAMPLES	-3039.650
7	3903	WHITE CHERRY LIGHTS	-54.000
8	922	CREAM SWEETHEART MAGAZINE RACK	-46.850
9	3977	WOODEN BOX ADVENT CALENDAR	-45.700



----------->>> Geographic Insights: Showcase regions where revenue is particularly strong. <<<-----------

In [245…

```
region_group = data.groupby('Country').sum()['revenue'].reset_index()
region_group.sort_values(by = 'revenue', ascending = False,ignore_index = True,inplace = True)
region_group
```

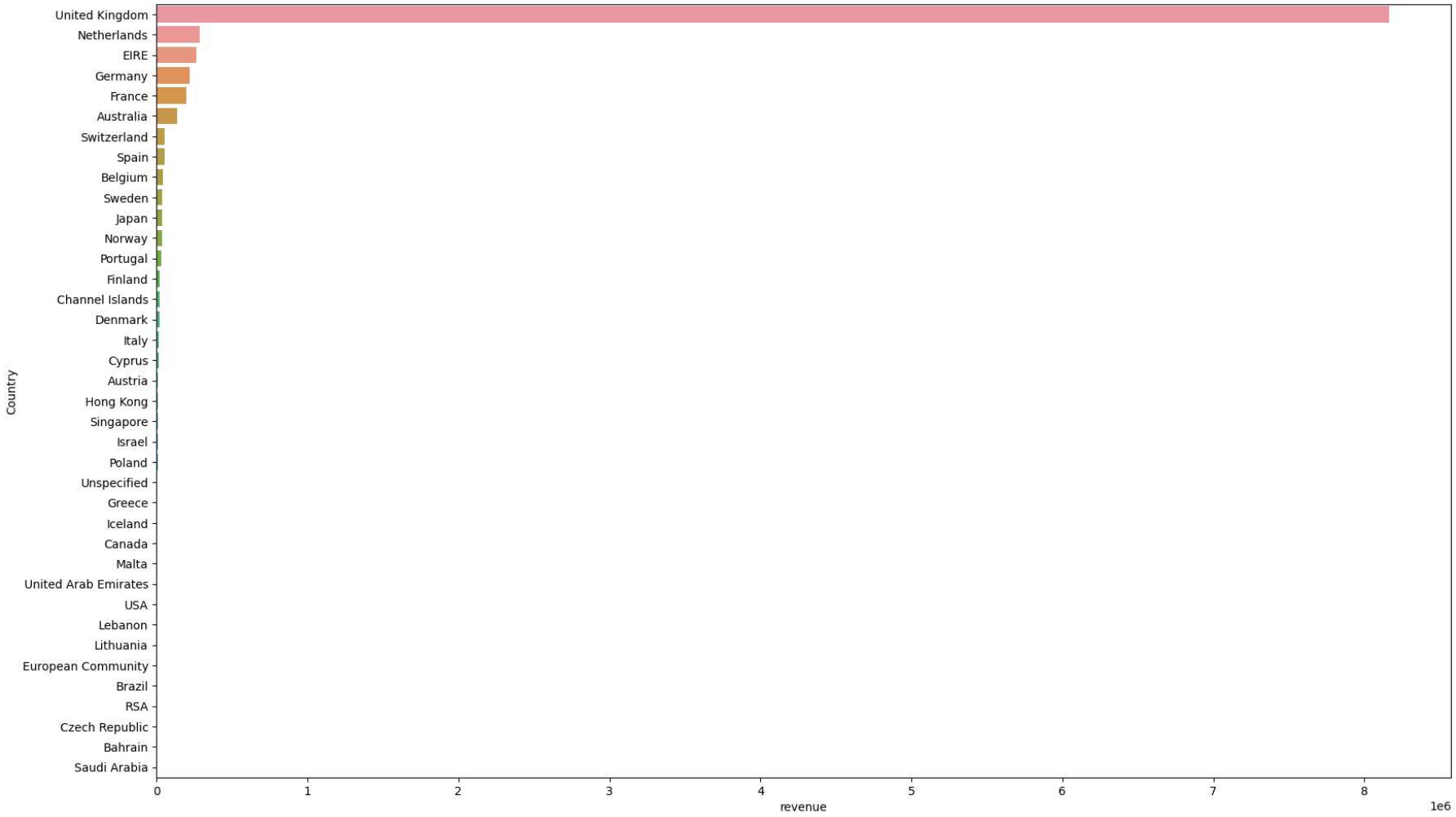
Out[245]:

	Country	revenue
0	United Kingdom	8166891.414
1	Netherlands	284661.540
2	EIRE	262993.380
3	Germany	221509.470
4	France	197317.110
5	Australia	137009.770
6	Switzerland	56363.050
7	Spain	54756.030
8	Belgium	40910.960
9	Sweden	36585.410
10	Japan	35340.620
11	Norway	35163.460
12	Portugal	29302.970
13	Finland	22326.740
14	Channel Islands	20076.390
15	Denmark	18768.140
16	Italy	16890.510
17	Cyprus	12850.360
18	Austria	10154.320
19	Hong Kong	9908.240
20	Singapore	9120.390
21	Israel	7891.970
22	Poland	7213.140
23	Unspecified	4740.940
24	Greece	4710.520
25	Iceland	4310.000
26	Canada	3666.380
27	Malta	2505.470
28	United Arab Emirates	1902.280
29	USA	1730.920
30	Lebanon	1693.880
31	Lithuania	1661.060
32	European Community	1291.750
33	Brazil	1143.600
34	RSA	1002.310
35	Czech Republic	707.720

	Country	revenue
36	Bahrain	548.400
37	Saudi Arabia	131.170

In [442…
```
plt.figure(figsize =(20,12))
sns.barplot(data = region_group, y = 'Country', x = 'revenue')
```

Out[442]:
```
<Axes: xlabel='revenue', ylabel='Country'>
```

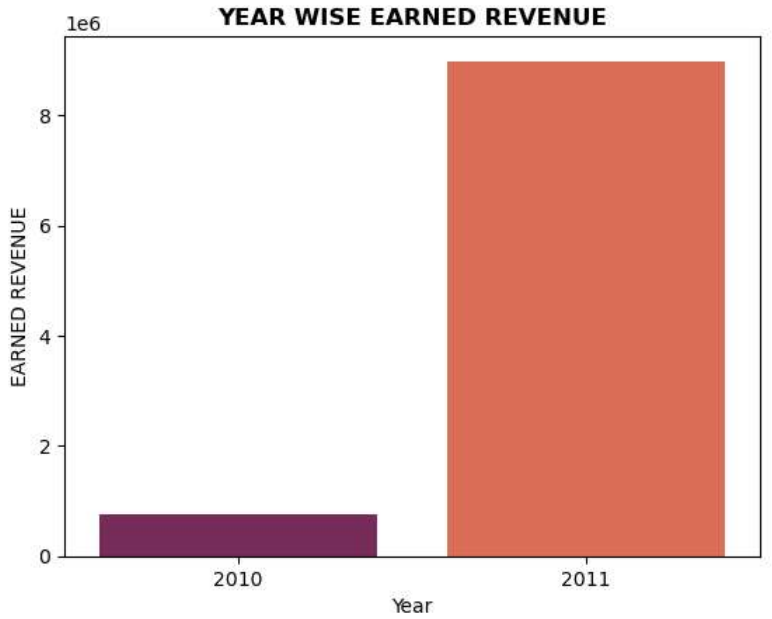


Best Year For Sale

In [149…
```
year_group = data.groupby('Year').sum()['revenue'].reset_index()
display(year_group)
sns.barplot(data = year_group, x = year_group['Year'],y = year_group['revenue'], palette = 'rocket')
plt.xlabel('Year')
plt.ylabel('EARNED REVENUE')
```

```
plt.title('YEAR WISE EARNED REVENUE',fontweight = 'bold')
plt.show()
```

	Year	revenue
0	2010	746659.940
1	2011	8979091.844

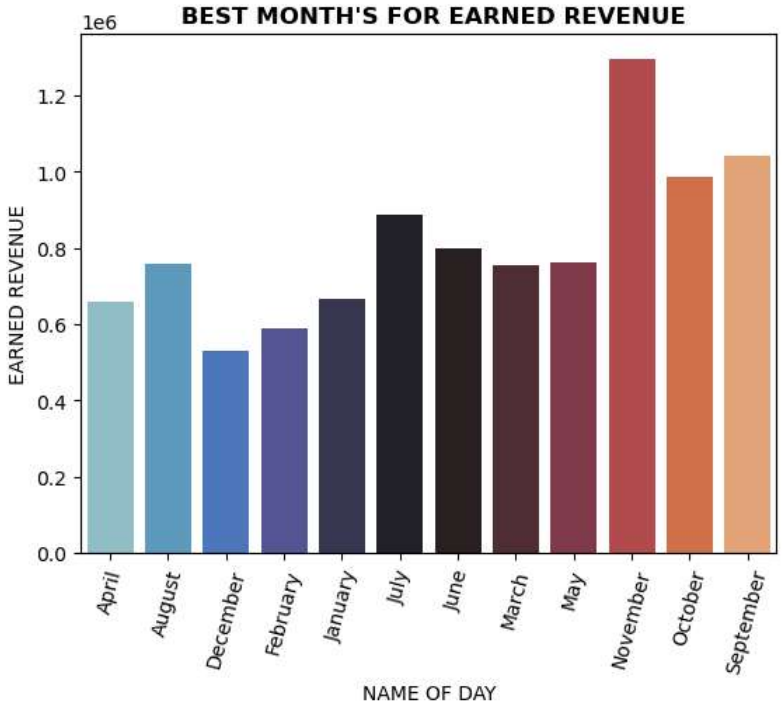


Best Day & Best Time For Sale

In [210…
```
def check(groupby , permeter,header,choose_color):
    temp = data.groupby(groupby).sum()[permeter].reset_index()
    display(temp)
    sns.barplot(data = temp , x = groupby, y = permeter,palette = choose_color)
    plt.xlabel('NAME OF DAY')
    plt.ylabel("EARNED REVENUE")
    plt.title(f" BEST {header} FOR EARNED REVENUE",fontweight = 'bold' )
    plt.xticks(rotation = 75)
    plt.figure(figsize= (9,5))
    plt.show()
    return ""
```

In [211…
```
display(check('Month','revenue',"MONTH'S" ,'icefire'))
```

	Month	revenue
0	April	658822.121
1	August	757923.200
2	December	528337.920
3	February	587104.620
4	January	664481.080
5	July	887404.191
6	June	797475.370
7	March	756089.290
8	May	762946.280
9	November	1295671.270
10	October	986240.770
11	September	1043255.672

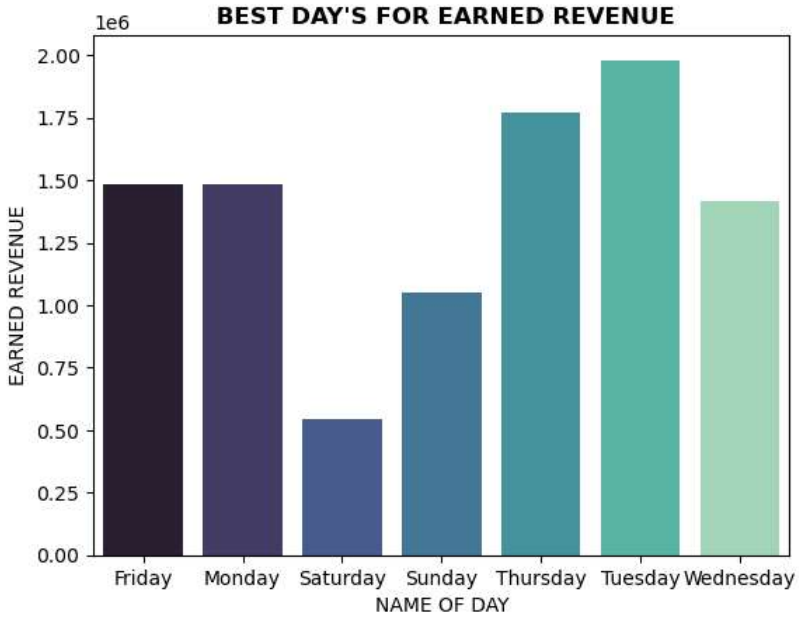



```
<Figure size 900x500 with 0 Axes>
''
```

In [207…

```
display(check('Day_Name','revenue',"DAY'S",'mako'))
```

	Day_Name	revenue
0	Friday	1482770.141
1	Monday	1484777.691
2	Saturday	543534.790
3	Sunday	1048394.261
4	Thursday	1770449.150
5	Tuesday	1980703.351
6	Wednesday	1415122.400

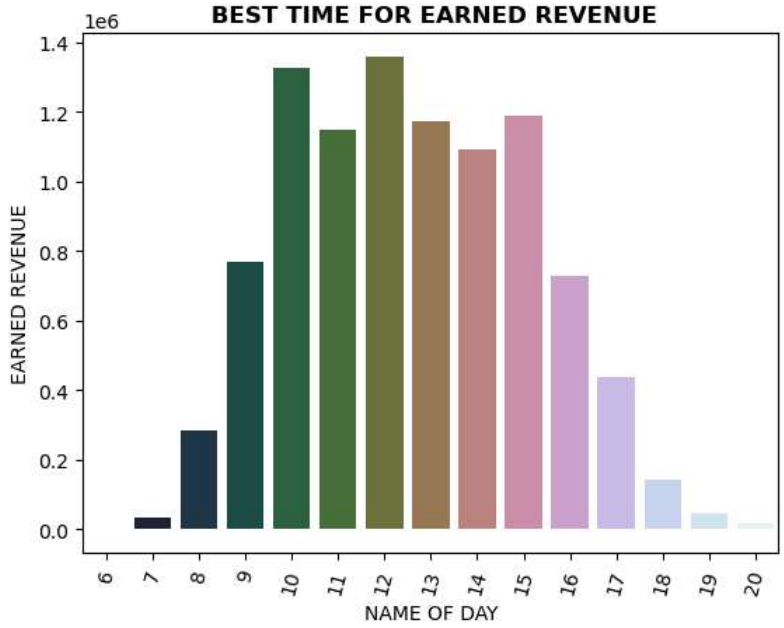



```
<Figure size 900x500 with 0 Axes>
''
```

In [212…

```
check('Hour','revenue','TIME','cubehelix')
```

	Hour	revenue
0	6	-497.350
1	7	31009.320
2	8	281713.020
3	9	766524.171
4	10	1327317.731
5	11	1146426.530
6	12	1357568.320
7	13	1172966.670
8	14	1091373.191
9	15	1186781.100
10	16	727621.110
11	17	434806.191
12	18	140359.260
13	19	45862.430
14	20	15920.090




```
<Figure size 900x500 with 0 Axes>
''
```

Out[212]:

BEST SELLING MONTH OF TOP GROWING COMPANY

In [251…
```
grouped_data = data.groupby(['Country','Month']).sum()['revenue'].reset_index()
grouped_data = grouped_data[(grouped_data['Country']!='Unspecified')]
grouped_data
```

Out[251]:

	Country	Month	revenue
0	Australia	April	-99.490
1	Australia	August	22560.940
2	Australia	December	1404.370
3	Australia	February	14862.710
4	Australia	January	4969.200
...	...	...	...
288	United Kingdom	March	642619.740
289	United Kingdom	May	624161.610
290	United Kingdom	November	1113228.800
291	United Kingdom	October	805432.460
292	United Kingdom	September	884916.522

293 rows × 3 columns

As From Our Previous Analysis We Know That Top Growing Company From (region_group) Data

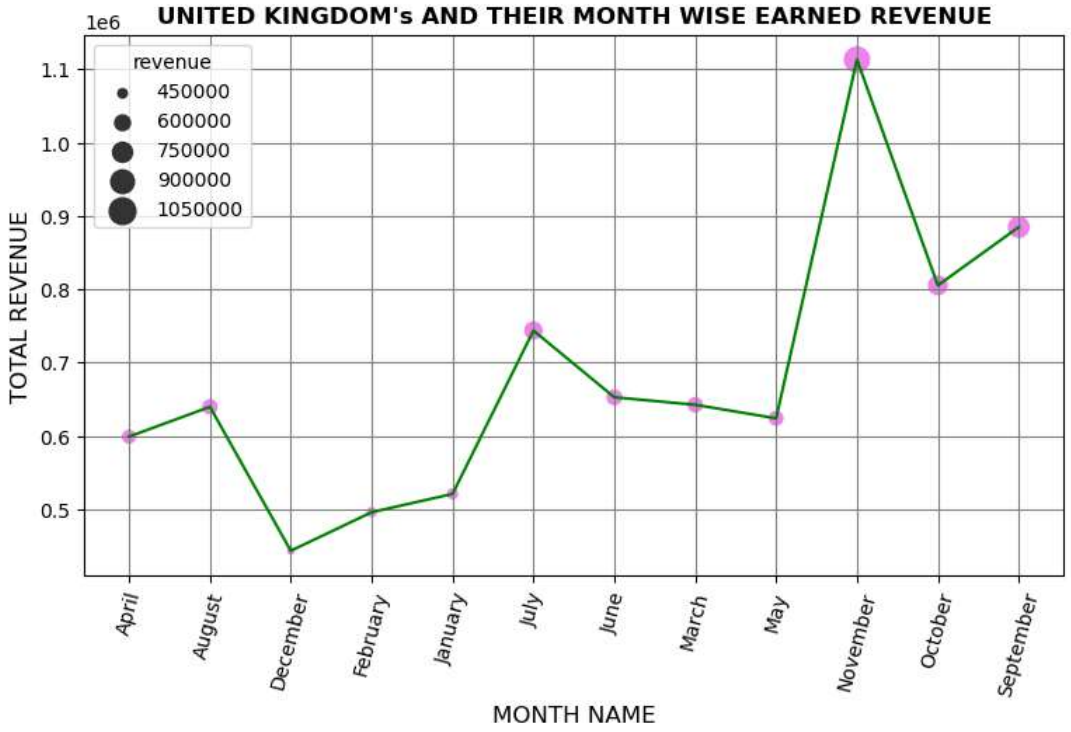
In [248…
```
cntry = region_group.Country[0:10]
list(cntry)
```

Out[248]:
```
['United Kingdom',
 'Netherlands',
 'EIRE',
 'Germany',
 'France',
 'Australia',
 'Switzerland',
 'Spain',
 'Belgium',
 'Sweden']
```

In [293…
```
for i in range(len(cntry)):
    temp = grouped_data[(grouped_data['Country']==cntry[i])]
    print(f"--------------------------------->>>> {cntry[i].upper()} AND THEIR REVENUE <<<<--------------------------------- ")
    display(temp[['Month','revenue']])
    plt.figure(figsize = (9,5))
    plt.grid(True, color='grey')
    sns.lineplot(data = temp,x = 'Month',y = 'revenue', color = 'g', )
    sns.scatterplot(data = temp, x = 'Month', y = 'revenue', size  = 'revenue',sizes = (20,200), color = 'violet')
    plt.xlabel('MONTH NAME',fontsize = 12)
    plt.ylabel('TOTAL REVENUE',fontsize = 12)
    plt.xticks(rotation = 75)
    plt.title(f"""{cntry[i].upper()+"'s"} AND THEIR MONTH WISE EARNED REVENUE""",fontweight='bold')
    plt.show()
```

```
--------------------------------->>>> UNITED KINGDOM AND THEIR REVENUE <<<<---------------------------------
```

	Month	revenue
281	April	599176.341
282	August	639829.640
283	December	443566.920
284	February	496181.030
285	January	521079.670
286	July	743826.581
287	June	652872.100
288	March	642619.740
289	May	624161.610
290	November	1113228.800
291	October	805432.460
292	September	884916.522

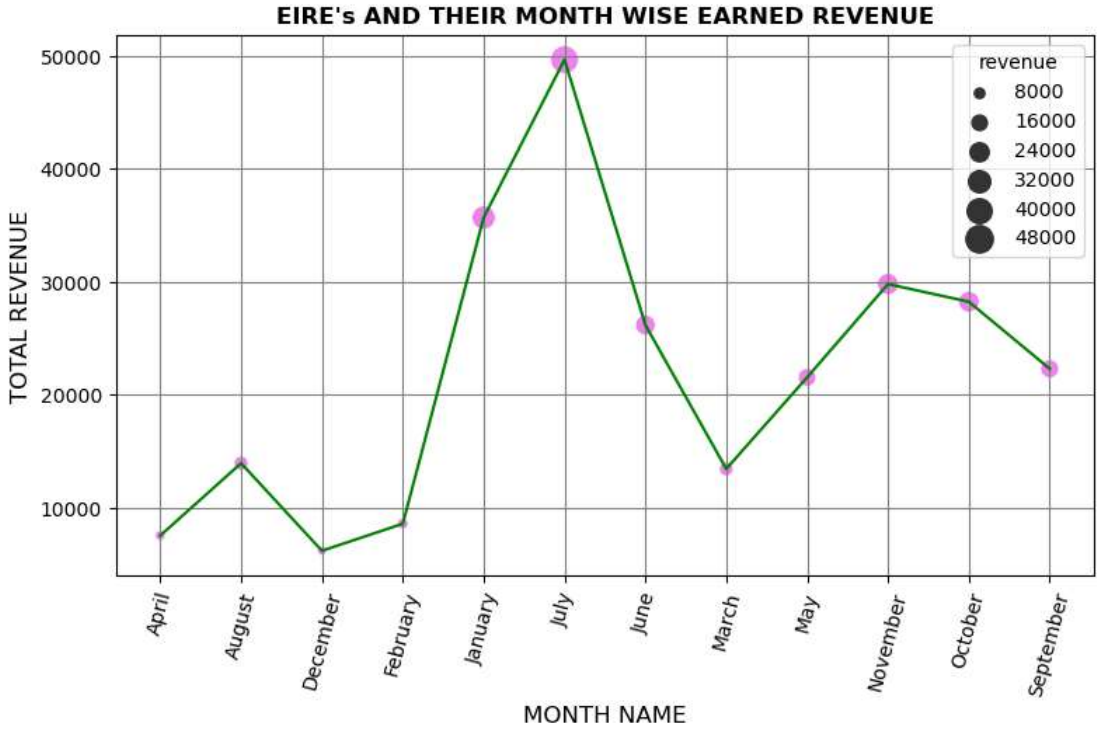


---------->>>> NETHERLANDS AND THEIR REVENUE <<<<----------

	Month	revenue
184	April	692.36
185	August	10639.40
186	December	25895.94
187	February	25978.09
188	January	32441.01
189	July	11675.37
190	June	21773.04
191	March	22356.69
192	May	14259.07
193	November	31521.04
194	October	49426.44
195	September	38003.09

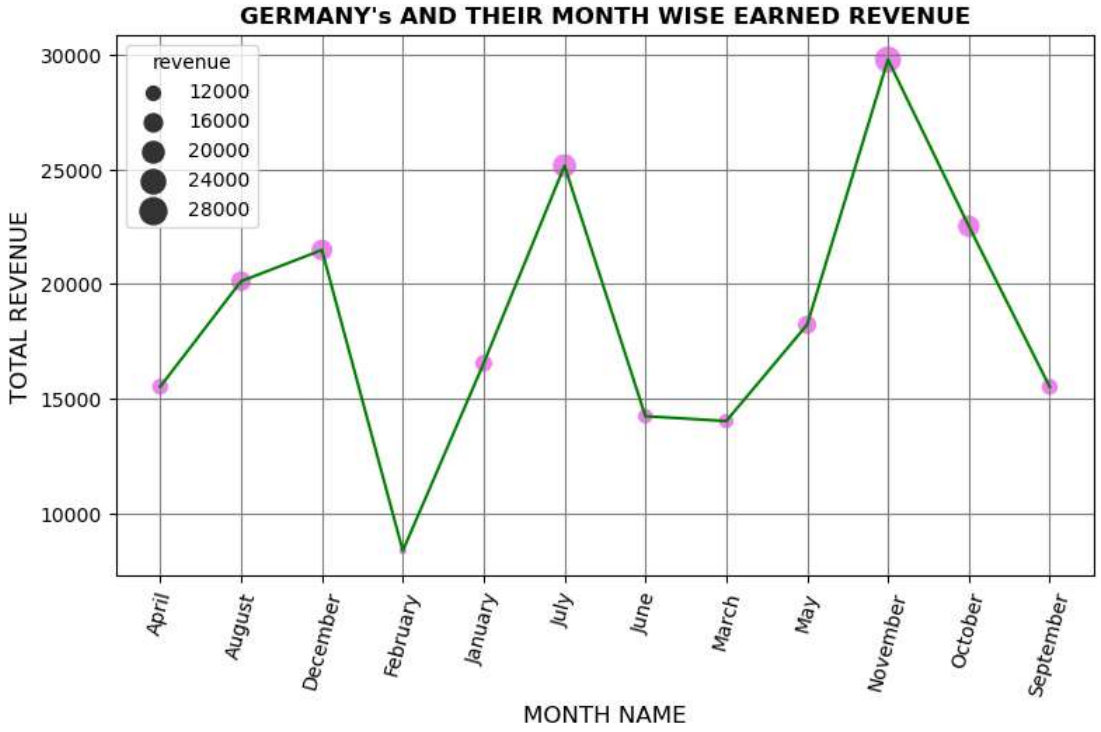
----------->>>> EIRE AND THEIR REVENUE <<<<-----------

	Month	revenue
74	April	7513.49
75	August	13928.25
76	December	6172.01
77	February	8569.04
78	January	35683.04
79	July	49678.19
80	June	26171.01
81	March	13423.69
82	May	21532.31
83	November	29796.42
84	October	28232.05
85	September	22293.88



----------->>>> GERMANY AND THEIR REVENUE <<<<-----------

	Month	revenue
115	April	15523.20
116	August	20122.26
117	December	21483.92
118	February	8366.98
119	January	16543.60
120	July	25158.48
121	June	14237.32
122	March	14024.24
123	May	18223.84
124	November	29787.21
125	October	22521.90
126	September	15516.52

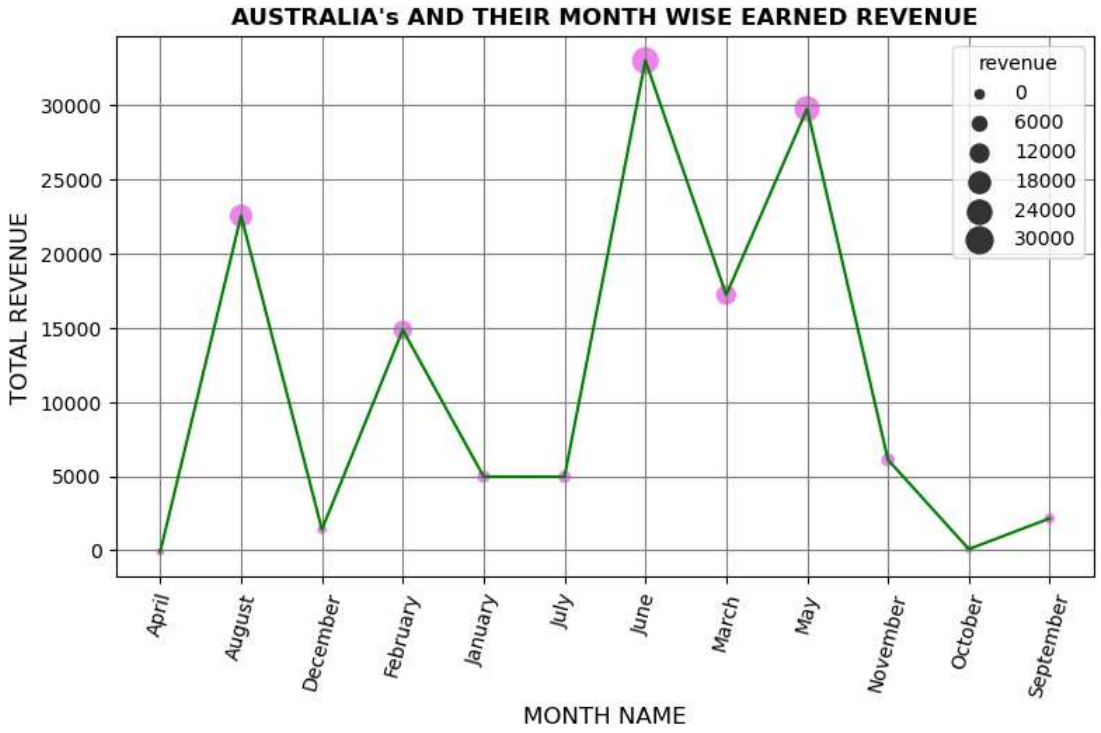


----------->>>> FRANCE AND THEIR REVENUE <<<<-----------

	Month	revenue
103	April	8673.15
104	August	17408.42
105	December	7277.90
106	February	8745.25
107	January	16162.13
108	July	12064.80
109	June	12717.63
110	March	17758.86
111	May	21854.72
112	November	31545.67
113	October	21799.09
114	September	21309.49

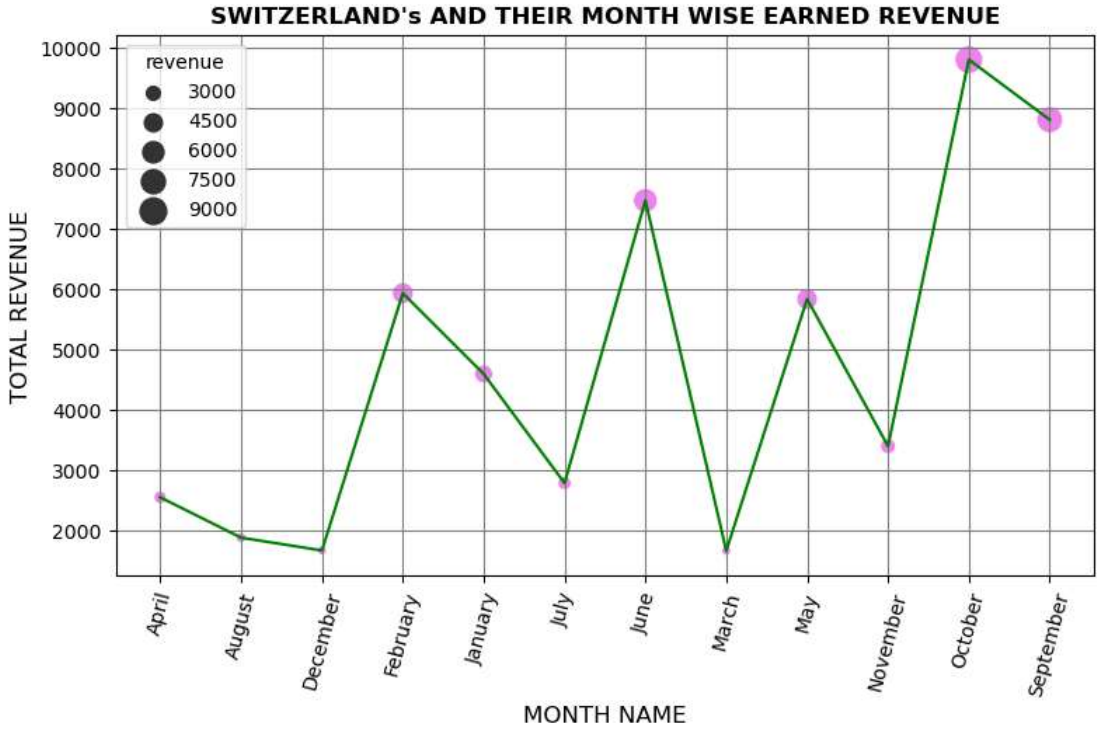
----------->>>> AUSTRALIA AND THEIR REVENUE <<<<-----------

	Month	revenue
0	April	-99.49
1	August	22560.94
2	December	1404.37
3	February	14862.71
4	January	4969.20
5	July	4959.57
6	June	33020.91
7	March	17219.63
8	May	29772.82
9	November	6104.30
10	October	81.60
11	September	2153.21



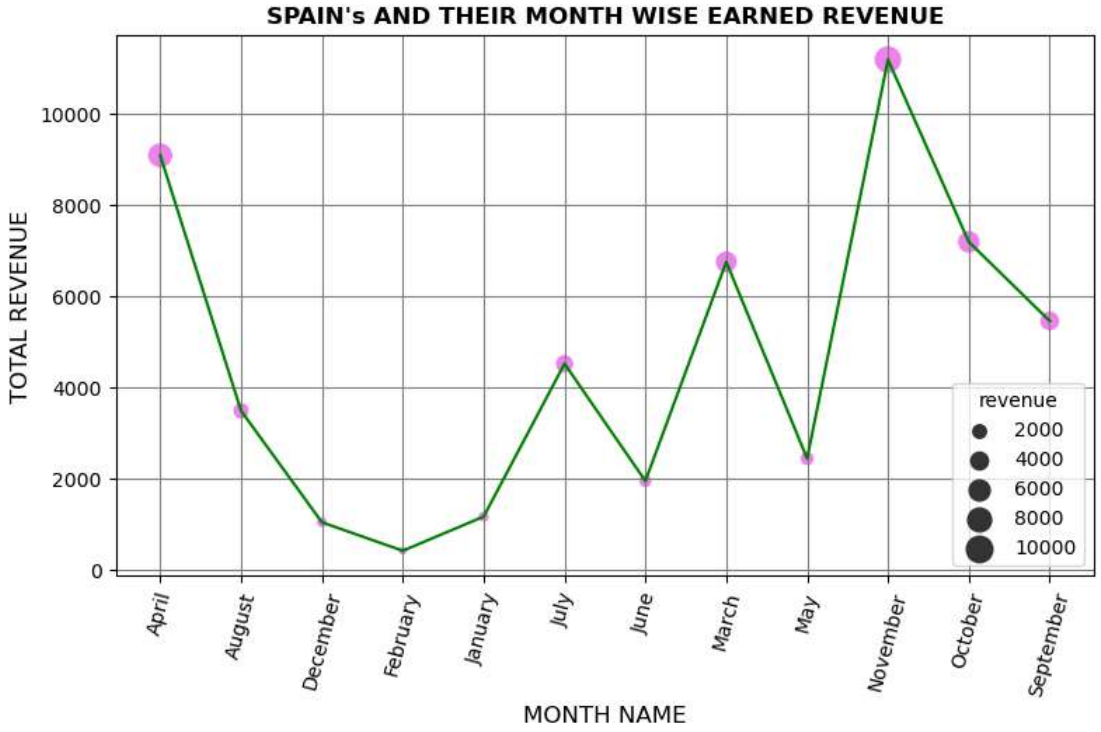
---------->>>> SWITZERLAND AND THEIR REVENUE <<<<----------

	Month	revenue
260	April	2545.37
261	August	1876.04
262	December	1663.44
263	February	5931.21
264	January	4592.03
265	July	2781.08
266	June	7470.11
267	March	1658.20
268	May	5834.27
269	November	3395.15
270	October	9806.61
271	September	8809.54



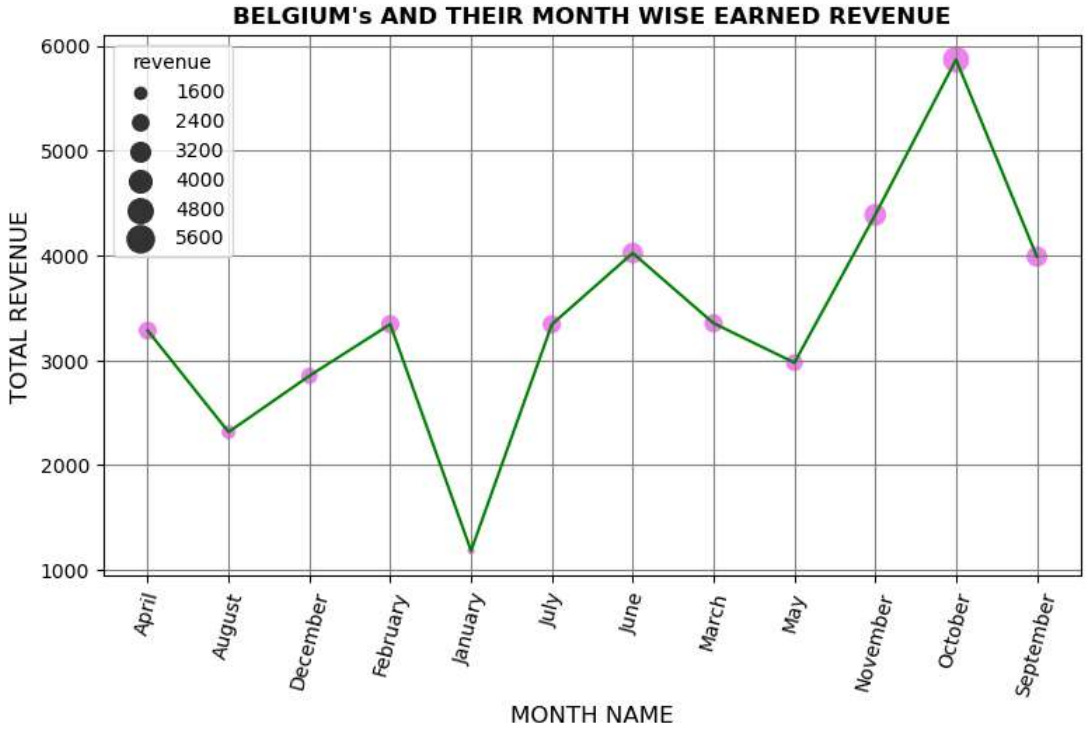
----------->>>> SPAIN AND THEIR REVENUE <<<<-----------

	Month	revenue
236	April	9090.05
237	August	3492.18
238	December	1049.01
239	February	425.36
240	January	1174.70
241	July	4524.68
242	June	1954.54
243	March	6755.75
244	May	2447.24
245	November	11187.99
246	October	7193.64
247	September	5460.89



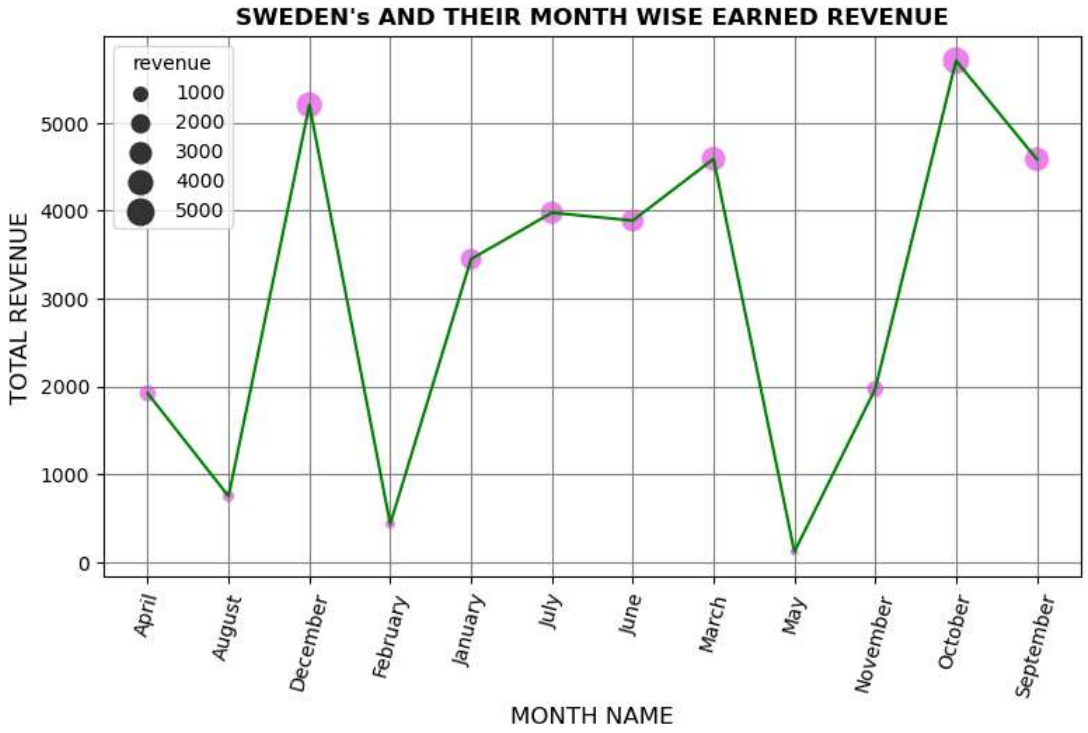
----------->>>> BELGIUM AND THEIR REVENUE <<<<-----------

	Month	revenue
26	April	3282.33
27	August	2314.89
28	December	2850.98
29	February	3343.66
30	January	1181.83
31	July	3342.49
32	June	4022.01
33	March	3352.86
34	May	2976.37
35	November	4386.84
36	October	5867.94
37	September	3988.76



----------->>>> SWEDEN AND THEIR REVENUE <<<<-----------

	Month	revenue
248	April	1920.16
249	August	747.61
250	December	5204.18
251	February	431.00
252	January	3448.26
253	July	3976.99
254	June	3887.29
255	March	4589.68
256	May	115.60
257	November	1970.37
258	October	5708.56
259	September	4585.71



NOW WE'LL ANALYSIS LESS SUCESSFUL REGION MONTH WISE

In [326…

```
cont = region_group.Country[-1:27:-1]
cont = list(cont)
cont
```

Out[326]:
```
['Saudi Arabia',
 'Bahrain',
 'Czech Republic',
 'RSA',
 'Brazil',
 'European Community',
 'Lithuania',
 'Lebanon',
 'USA',
 'United Arab Emirates']
```

In [360…
```
for i in range(len(cont)):
    temp = grouped_data[(grouped_data['Country']==cont[i])]
    temp = temp.sort_values(by='revenue',ascending = False)
    print(f"---------------------------------->>>> {cont[i].upper()} AND THEIR REVENUE <<<<---------------------------------- ")
    display(temp[['Month','revenue']])
```

```
---------------------------------->>>> SAUDI ARABIA AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
229	February	145.92
230	March	-14.75

```
---------------------------------->>>> BAHRAIN AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
25	September	459.40
23	December	205.74
24	May	-116.74

```
---------------------------------->>>> CZECH REPUBLIC AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
62	February	549.26
64	July	277.48
63	January	-57.51
65	November	-61.51

```
---------------------------------->>>> RSA AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
228	October	1002.31

```
---------------------------------->>>> BRAZIL AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
38	April	1143.6

```
---------------------------------->>>> EUROPEAN COMMUNITY AND THEIR REVENUE <<<<----------------------------------
```

	Month	revenue
87	July	676.80
89	May	387.05
86	April	191.40
88	June	45.00
90	October	-8.50

----------->>>> LITHUANIA AND THEIR REVENUE <<<<-----------

	Month	revenue
179	May	1598.06
178	August	63.00

----------->>>> LEBANON AND THEIR REVENUE <<<<-----------

	Month	revenue
177	January	1693.88

----------->>>> USA AND THEIR REVENUE <<<<-----------

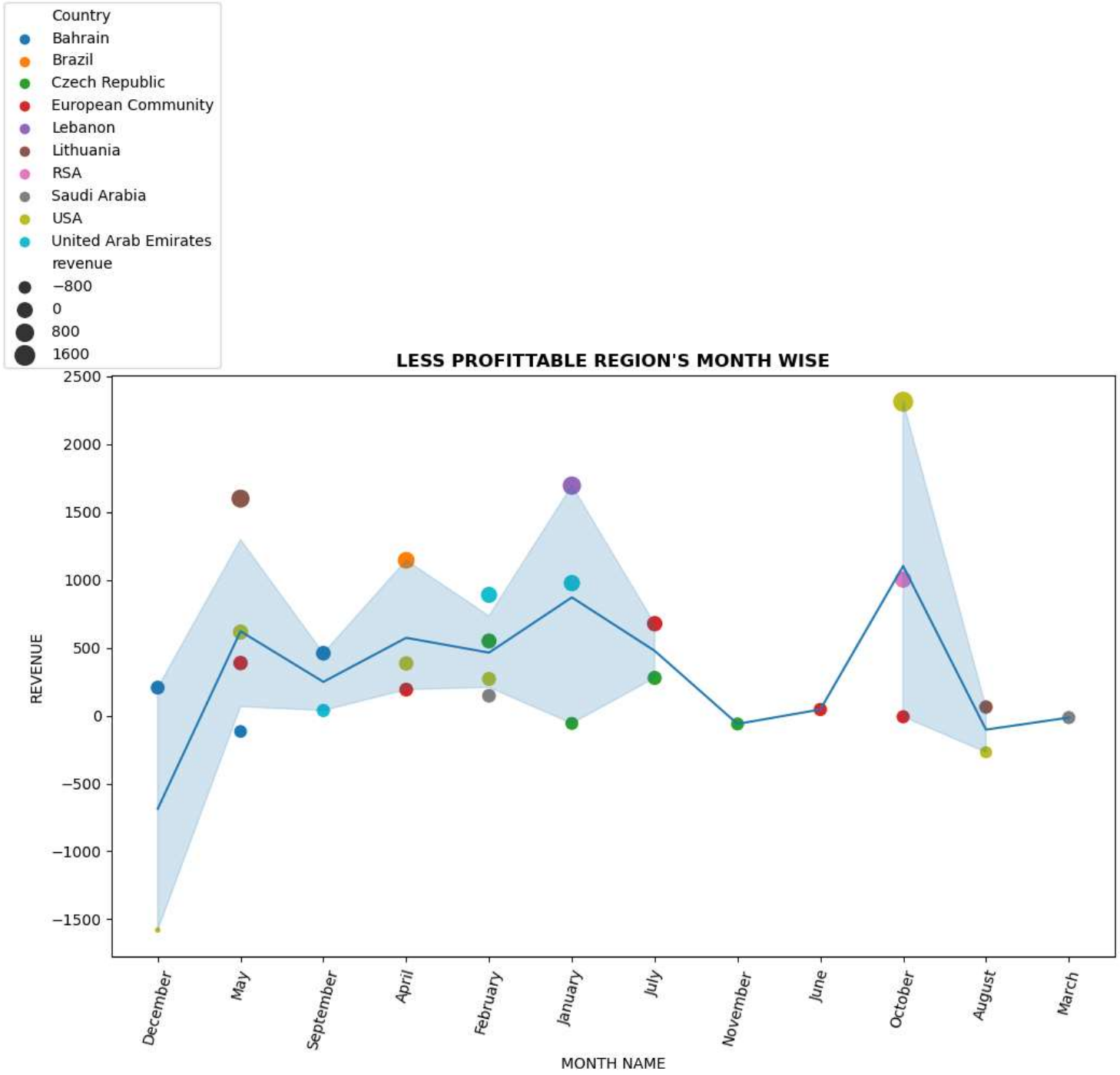
	Month	revenue
277	October	2311.20
276	May	615.28
272	April	383.95
275	February	269.96
273	August	-269.96
274	December	-1579.51

----------->>>> UNITED ARAB EMIRATES AND THEIR REVENUE <<<<-----------

	Month	revenue
279	January	975.54
278	February	889.24
280	September	37.50

In [377…

```
test = grouped_data[(grouped_data['Country'].isin(cont))]
plt.figure(figsize = (12,7))
sns.scatterplot(data = test, x = 'Month', y = 'revenue', hue = 'Country', size = 'revenue', sizes = (20,200),s = 5)
sns.lineplot(data = test , x = 'Month', y = 'revenue')
plt.legend(loc='lower center', bbox_to_anchor=(0.0, 1))
plt.xticks(rotation = 75)
plt.xlabel('MONTH NAME')
plt.ylabel('REVENUE')
plt.title("LESS PROFITTABLE REGION'S MONTH WISE",fontweight = 'bold')
plt.show()
```



Top Growing Country Coustomer Wise in Each Month

In [418…
```
print("------------------->>> TOP 10 GROWING COUNTRY COUSTOMER WISE <<<-------------------")
check = data.groupby('Country').count()['InvoiceNo'].reset_index()
check = check.sort_values(by = 'InvoiceNo' , ascending = False)
check = check.nlargest(10,'InvoiceNo')
display(check)
```

```
------------------->>> TOP 10 GROWING COUNTRY COUSTOMER WISE <<<-------------------
```

	Country	InvoiceNo
36	United Kingdom	490232
14	Germany	9480
13	France	8541
10	EIRE	8184
31	Spain	2528
24	Netherlands	2371
3	Belgium	2069
33	Switzerland	1994
27	Portugal	1510
0	Australia	1258

Now We'll Check Growing Country Coustomer Wise in Each Month

In [420…
```
cont_name = check['Country']
cont_name = list(cont_name)
cont_name
```

Out[420]:
```
['United Kingdom',
 'Germany',
 'France',
 'EIRE',
 'Spain',
 'Netherlands',
 'Belgium',
 'Switzerland',
 'Portugal',
 'Australia']
```

In [433…
```
data_grouped_2 = data.groupby(['Country','Month']).count()['InvoiceNo'].reset_index()
data_grouped_2.sort_values(by= 'InvoiceNo',ascending = False)
color = ["dark:salmon_r","YlOrBr","Blues","YlOrBr","icefire","Spectral","coolwarm","Spectral","rocket","cubehelix"]
```

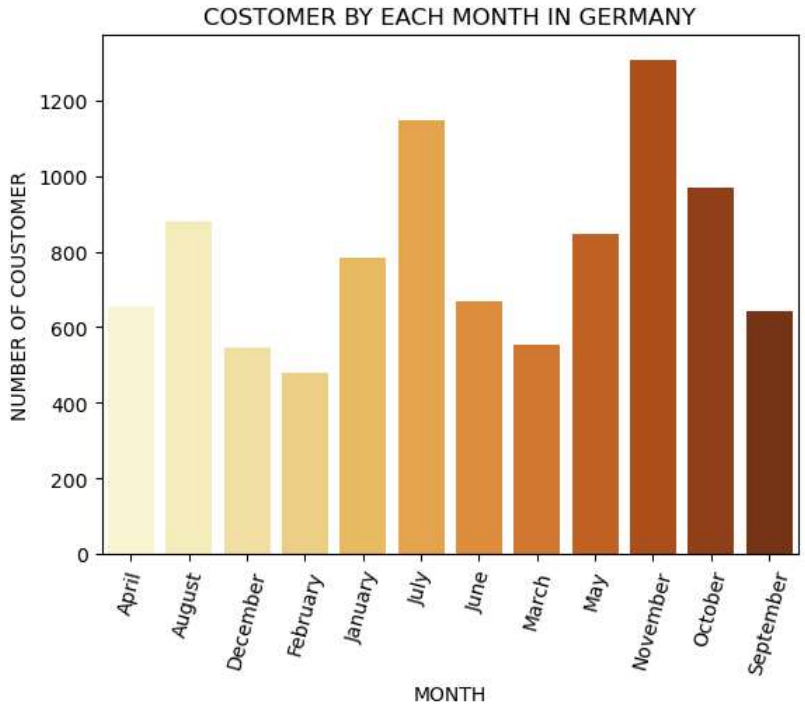
In [434…
```
for i in range(len(cont_name)):
    temp = data_grouped_2[(data_grouped_2['Country']==cont_name[i])]
    print(f"------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN {cont_name[i].upper()} <<<<------------------")
    display(temp[['Month','InvoiceNo']])
    sns.barplot(data = temp , x = 'Month', y = 'InvoiceNo', palette = color[i])
    plt.xlabel('MONTH')
    plt.ylabel('NUMBER OF COUSTOMER')
    plt.xticks(rotation = 75)
    plt.title(f'COSTOMER BY EACH MONTH IN {cont_name[i].upper()}')
    plt.show()
```

```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN UNITED KINGDOM <<<<------------------
```

	Month	InvoiceNo
281	April	34319
282	August	40319
283	December	27214
284	February	30826
285	January	35890
286	July	42140
287	June	41687
288	March	35269
289	May	38242
290	November	66606
291	October	51269
292	September	46451

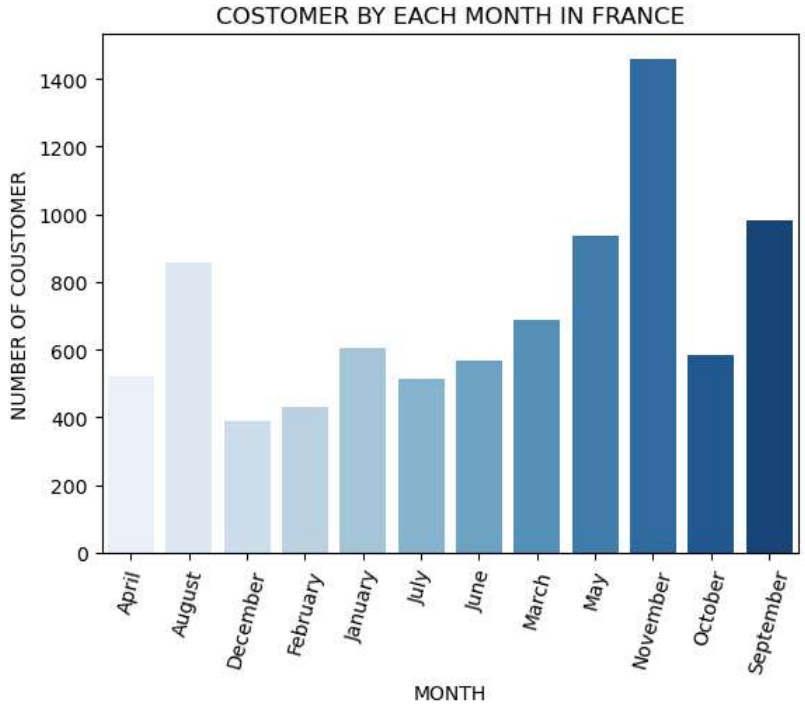

```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN GERMANY <<<<------------------
```

	Month	InvoiceNo
115	April	653
116	August	880
117	December	546
118	February	478
119	January	785
120	July	1149
121	June	669
122	March	554
123	May	847
124	November	1308
125	October	970
126	September	641



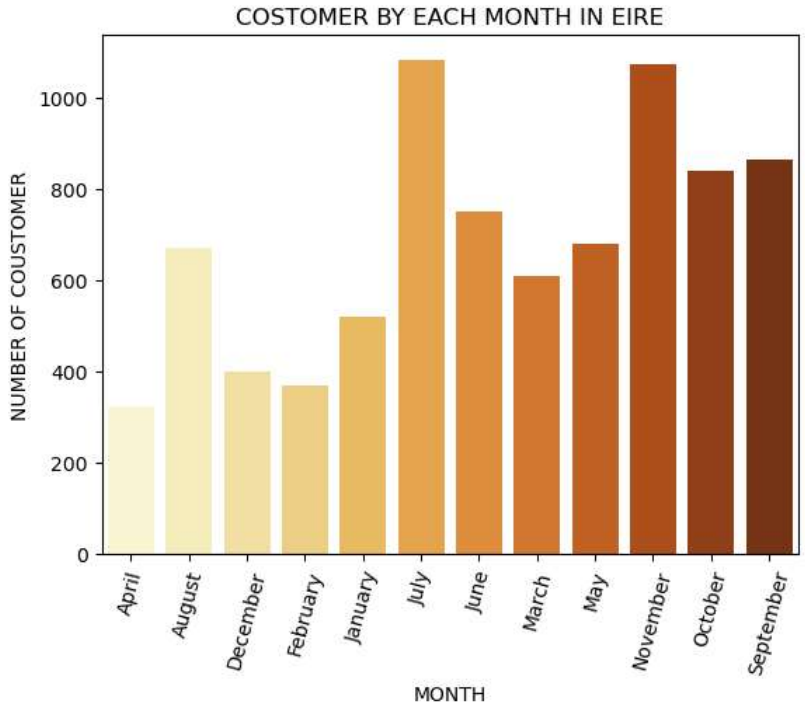

```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN FRANCE <<<<------------------
```

	Month	InvoiceNo
103	April	522
104	August	856
105	December	391
106	February	432
107	January	604
108	July	515
109	June	566
110	March	689
111	May	938
112	November	1459
113	October	585
114	September	984



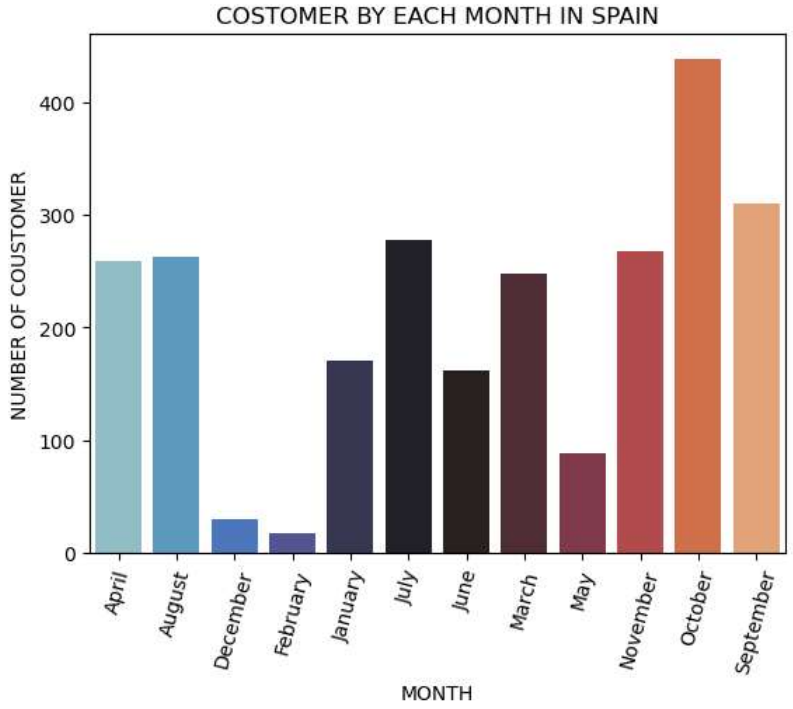
----------->>>> NUMBER OF COUSTOMER BY EACH MONTH IN EIRE <<<<-----------

	Month	InvoiceNo
74	April	324
75	August	670
76	December	401
77	February	368
78	January	521
79	July	1083
80	June	750
81	March	610
82	May	680
83	November	1073
84	October	841
85	September	863



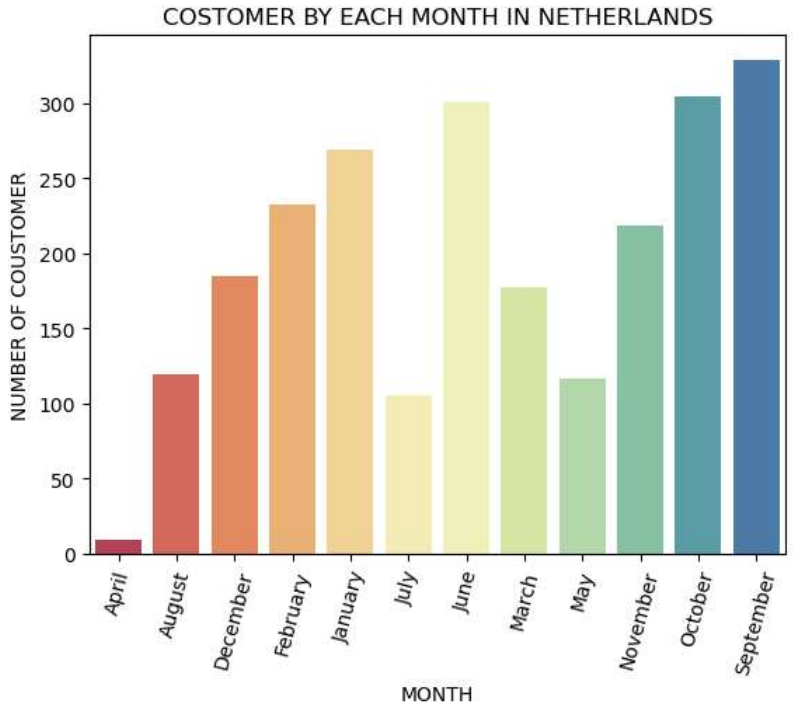
----------->>>> NUMBER OF COUSTOMER BY EACH MONTH IN SPAIN <<<<----------

	Month	InvoiceNo
236	April	259
237	August	263
238	December	30
239	February	17
240	January	170
241	July	277
242	June	162
243	March	247
244	May	88
245	November	267
246	October	438
247	September	310



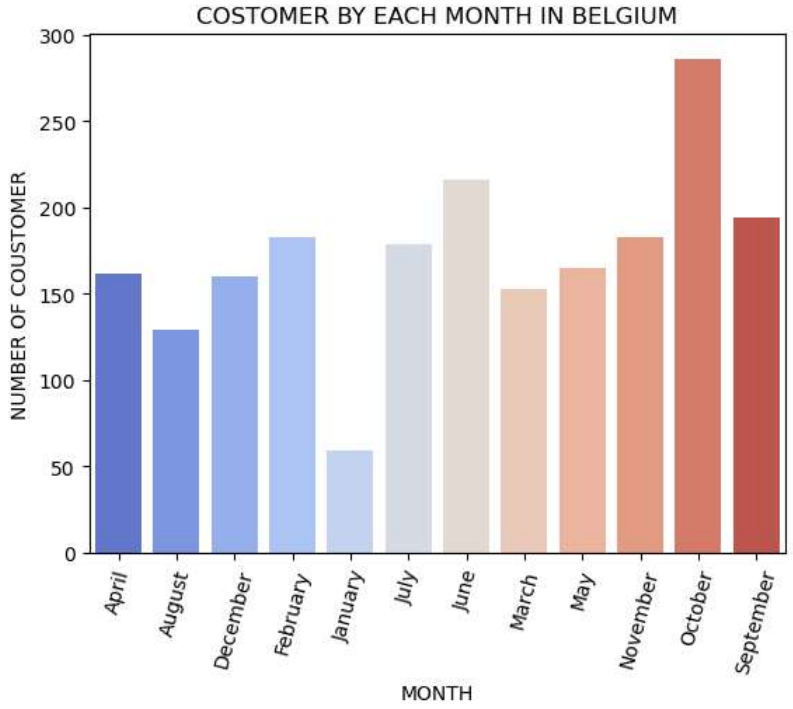

```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN NETHERLANDS <<<<------------------
```

	Month	InvoiceNo
184	April	9
185	August	120
186	December	185
187	February	233
188	January	269
189	July	106
190	June	301
191	March	178
192	May	117
193	November	219
194	October	305
195	September	329



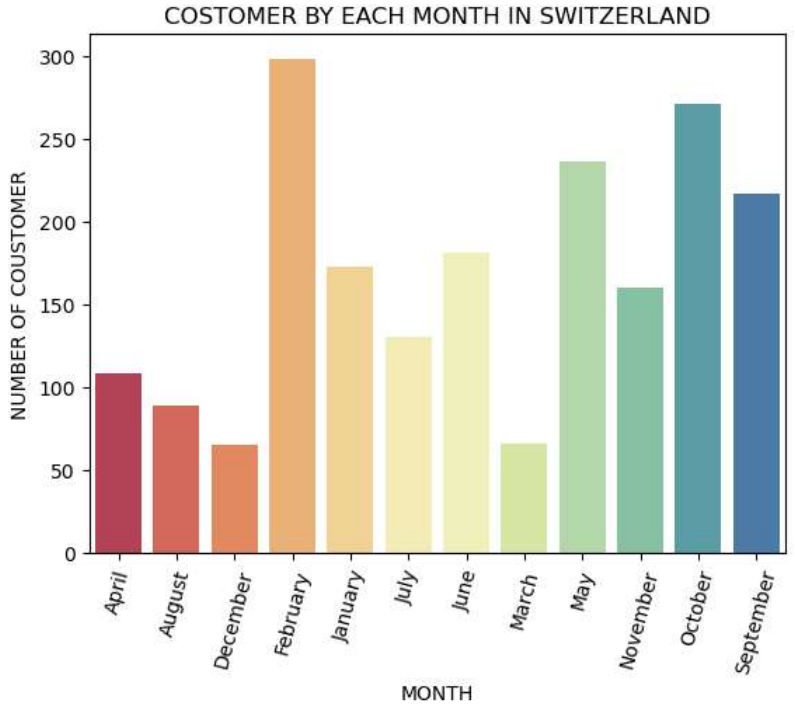
----------->>>> NUMBER OF COUSTOMER BY EACH MONTH IN BELGIUM <<<<-----------

	Month	InvoiceNo
26	April	162
27	August	129
28	December	160
29	February	183
30	January	59
31	July	179
32	June	216
33	March	153
34	May	165
35	November	183
36	October	286
37	September	194



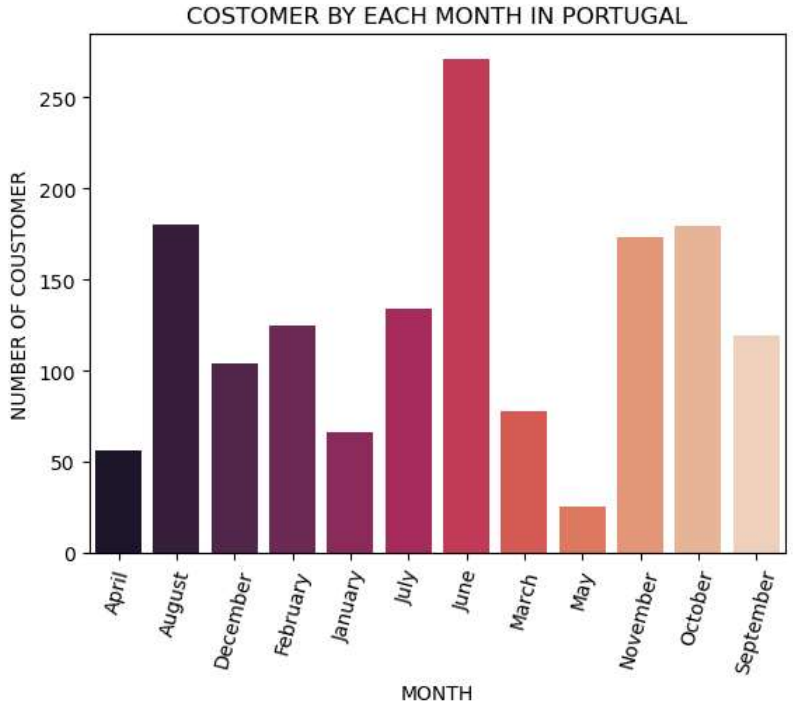
----------->>>> NUMBER OF COUSTOMER BY EACH MONTH IN SWITZERLAND <<<<-----------

	Month	InvoiceNo
260	April	108
261	August	89
262	December	65
263	February	298
264	January	173
265	July	130
266	June	181
267	March	66
268	May	236
269	November	160
270	October	271
271	September	217



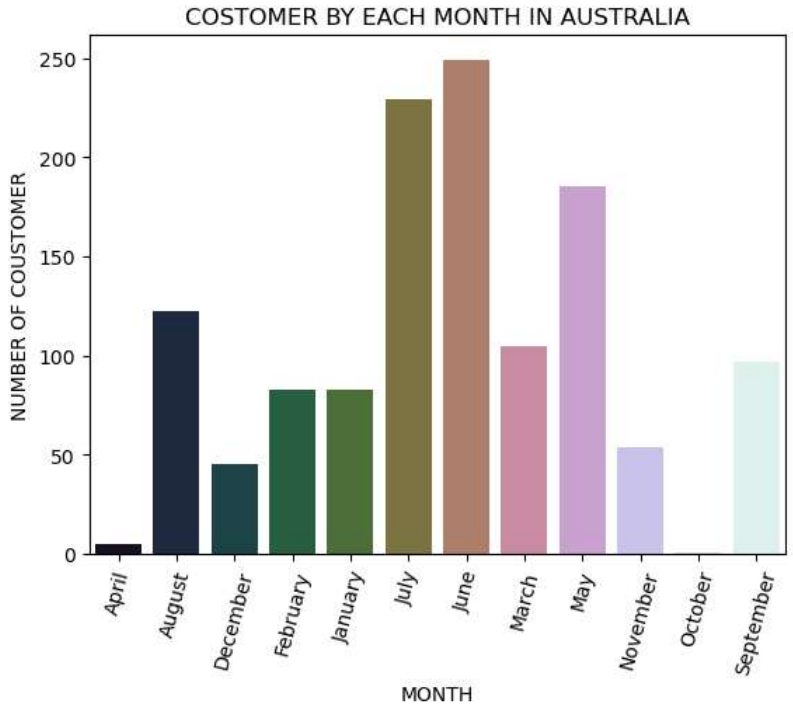

```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN PORTUGAL <<<<------------------
```

	Month	InvoiceNo
216	April	56
217	August	180
218	December	104
219	February	125
220	January	66
221	July	134
222	June	271
223	March	78
224	May	25
225	November	173
226	October	179
227	September	119




```
------------------>>>> NUMBER OF COUSTOMER BY EACH MONTH IN AUSTRALIA <<<<------------------
```

	Month	InvoiceNo
0	April	5
1	August	122
2	December	45
3	February	83
4	January	83
5	July	229
6	June	249
7	March	105
8	May	185
9	November	54
10	October	1
11	September	97



Thank You!