Online Food Orders Analysis with Python

In this project, we aim to analyze a dataset titled "Online Food Orders Analysis with Python." The dataset contains information about online food orders, including details such as order ID, customer ID, order date, order time, items ordered, quantity, price, and delivery information. We will use Python and various data analysis libraries to explore, visualize, and derive insights from the dataset.

Import Library

In [1]:
```
import pandas as pd
```

In [2]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns
```

```
C:\Users\Syed Arif\anaconda3\lib\site-packages\scipy\__init__.py:146: UserWar
ning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of Sc
iPy (detected version 1.25.1
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

Uploading Csv fle

In [3]:
```
df = pd.read_csv(r"C:\Users\Syed Arif\Desktop\onlinefoods.csv")
```

Data Preprocessing

.head()

head is used show to the By default = 5 rows in the dataset

In [4]:
```
df.head()
```

Out[4]:

	Age	Gender	Marital Status	Occupation	Monthly Income	Educational Qualifications	Family size	latitude	longitude	Pi cod
0	20	Female	Single	Student	No Income	Post Graduate	4	12.9766	77.5993	56000
1	24	Female	Single	Student	Below Rs.10000	Graduate	3	12.9770	77.5773	56000
2	22	Male	Single	Student	Below Rs.10000	Post Graduate	3	12.9551	77.6593	56001
3	22	Female	Single	Student	No Income	Graduate	6	12.9473	77.5616	56001
4	22	Male	Single	Student	Below Rs.10000	Post Graduate	4	12.9850	77.5533	56001

.tail()

tail is used to show rows by Descending order

In [5]:
```
df.tail()
```

Out[5]:

	Age	Gender	Marital Status	Occupation	Monthly Income	Educational Qualifications	Family size	latitude	longitude	co
383	23	Female	Single	Student	No Income	Post Graduate	2	12.9766	77.5993	560
384	23	Female	Single	Student	No Income	Post Graduate	4	12.9854	77.7081	560
385	22	Female	Single	Student	No Income	Post Graduate	5	12.9850	77.5533	560
386	23	Male	Single	Student	Below Rs.10000	Post Graduate	2	12.9770	77.5773	560
387	23	Male	Single	Student	No Income	Post Graduate	5	12.8988	77.5764	560

.shape

It show the total no of rows & Column in the dataset

In [6]:
```
df.shape
```

Out[6]:
```
(388, 13)
```

.Columns

It show the no of each Column

In [7]:
```
df.columns
```

Out[7]:
```
Index(['Age', 'Gender', 'Marital Status', 'Occupation', 'Monthly Income',
       'Educational Qualifications', 'Family size', 'latitude', 'longitude',
       'Pin code', 'Output', 'Feedback', 'Unnamed: 12'],
      dtype='object')
```

.dtypes

This Attribute show the data type of each column

In [8]:
```
df.dtypes
```

Out[8]:
```
Age                             int64
Gender                         object
Marital Status                 object
Occupation                     object
Monthly Income                 object
Educational Qualifications     object
Family size                     int64
latitude                      float64
longitude                     float64
Pin code                        int64
Output                         object
Feedback                       object
Unnamed: 12                    object
dtype: object
```

.unique()

In a column, It show the unique value of specific column.

In [10]:
```
df["Monthly Income"].unique()
```

Out[10]:
```
array(['No Income', 'Below Rs.10000', 'More than 50000', '10001 to 25000',
       '25001 to 50000'], dtype=object)
```

.nuique()

It will show the total no of unque value from whole data frame

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

Out[11]:
```
Age                             16
Gender                           2
Marital Status                   3
Occupation                       4
Monthly Income                   5
Educational Qualifications       5
Family size                      6
latitude                        77
longitude                       76
Pin code                        77
Output                           2
Feedback                         2
Unnamed: 12                      2
dtype: int64
```

.describe()

It show the Count, mean , median etc

In [12]:
```
df.describe()
```

Out[12]:

	Age	Family size	latitude	longitude	Pin code
count	388.000000	388.000000	388.000000	388.000000	388.000000
mean	24.628866	3.280928	12.972058	77.600160	560040.113402
std	2.975593	1.351025	0.044489	0.051354	31.399609
min	18.000000	1.000000	12.865200	77.484200	560001.000000
25%	23.000000	2.000000	12.936900	77.565275	560010.750000
50%	24.000000	3.000000	12.977000	77.592100	560033.500000
75%	26.000000	4.000000	12.997025	77.630900	560068.000000
max	33.000000	6.000000	13.102000	77.758200	560109.000000

.value_counts

It Shows all the unique values with their count

In [13]:
```
df["Monthly Income"].value_counts()
```

Out[13]:
```
No Income          187
25001 to 50000      69
More than 50000     62
10001 to 25000      45
Below Rs.10000      25
Name: Monthly Income, dtype: int64
```

.isnull()

It shows the how many null values

In [14]:
```
df.isnull()
```

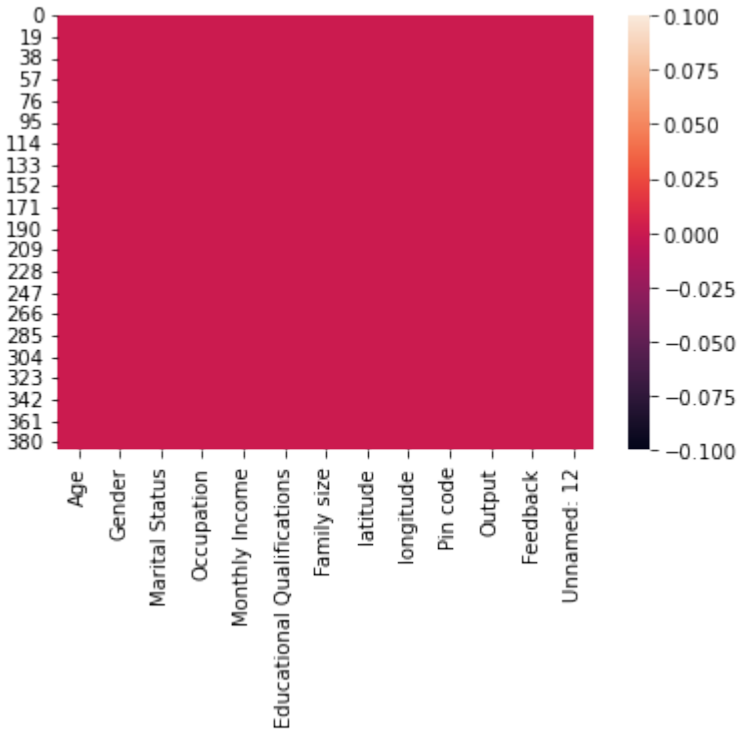
Out[14]:

	Age	Gender	Marital Status	Occupation	Monthly Income	Educational Qualifications	Family size	latitude	longitude	Pi cod
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
...	...	...	...	...	...	...	...	...	...	.
383	False	False	False	False	False	False	False	False	False	Fals
384	False	False	False	False	False	False	False	False	False	Fals
385	False	False	False	False	False	False	False	False	False	Fals
386	False	False	False	False	False	False	False	False	False	Fals
387	False	False	False	False	False	False	False	False	False	Fals

388 rows × 13 columns

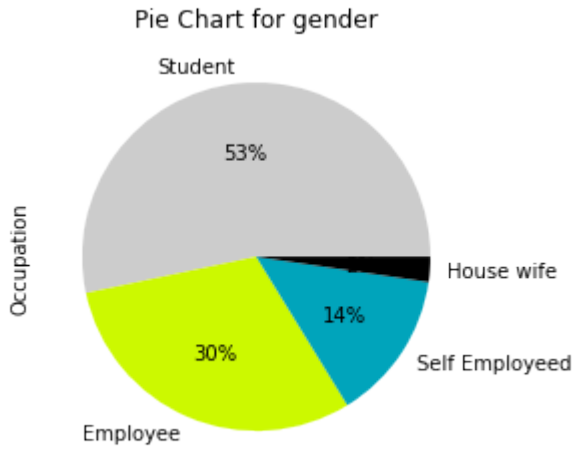
In [15]:
```
sns.heatmap(df.isnull())
```

Out[15]: <AxesSubplot:>

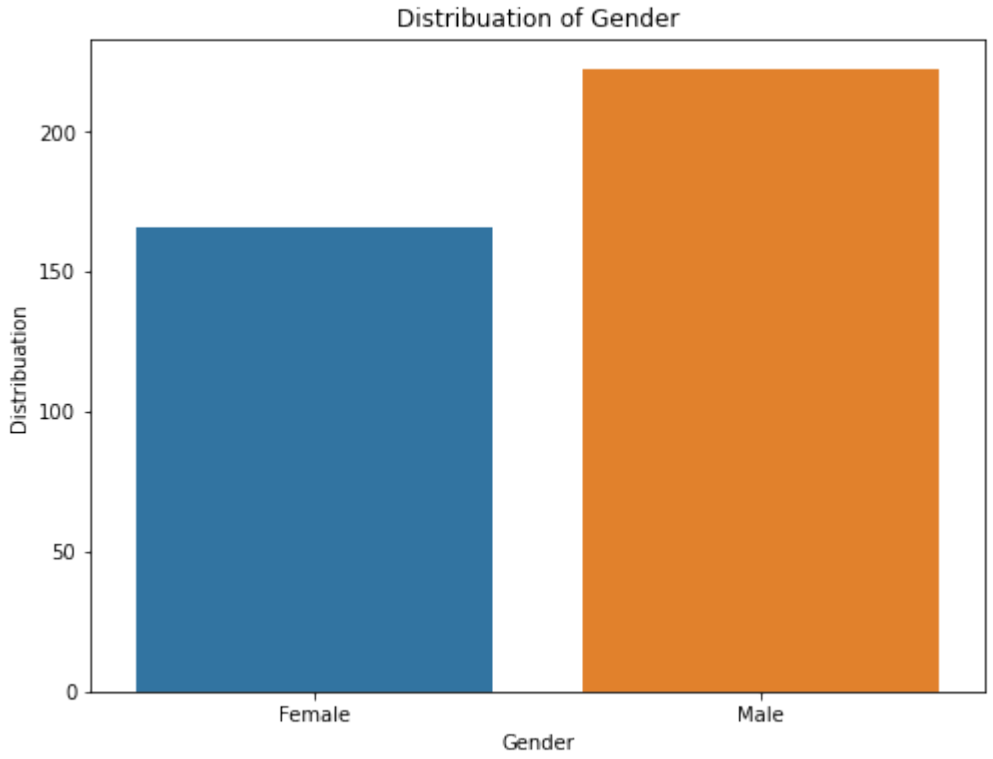


In [17]:
```
df['Occupation'].value_counts().plot(kind = 'pie' , title = 'Pie Chart for gen
                                     autopct="%.0f%%", colormap='nipy_spectral_r')
```

Out[17]: <AxesSubplot:title={'center':'Pie Chart for gender'}, ylabel='Occupation'>

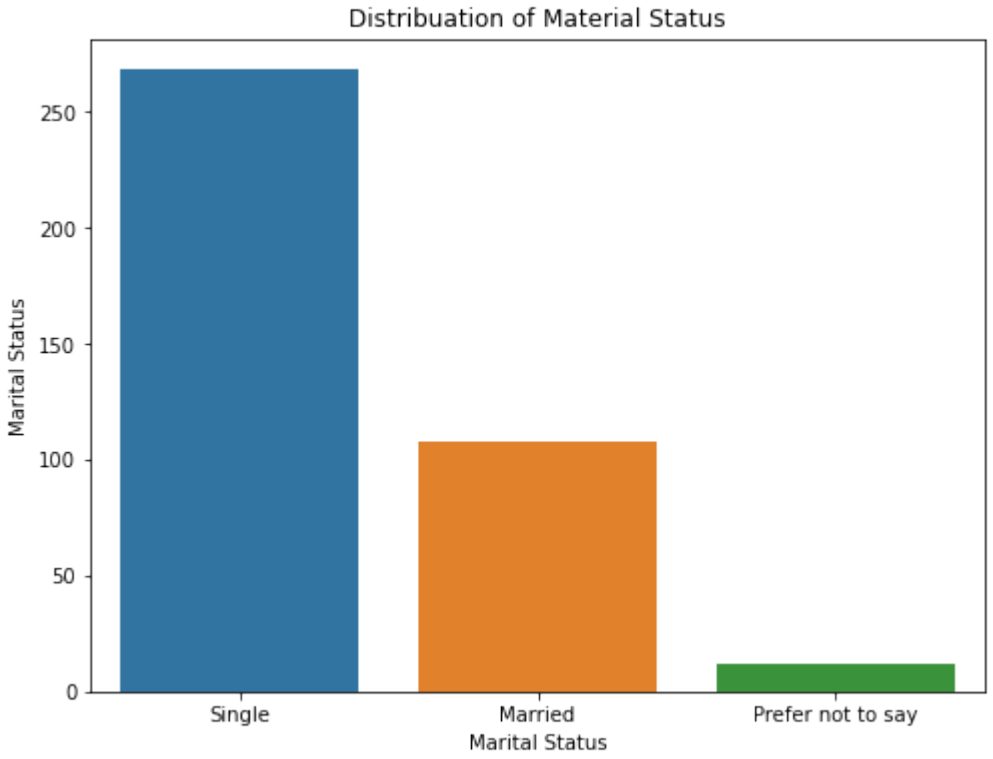


In [19]:
```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Gender')
plt.xlabel('Gender')
plt.ylabel('Distribuation')
plt.title('Distribuation of Gender')
plt.show()
```



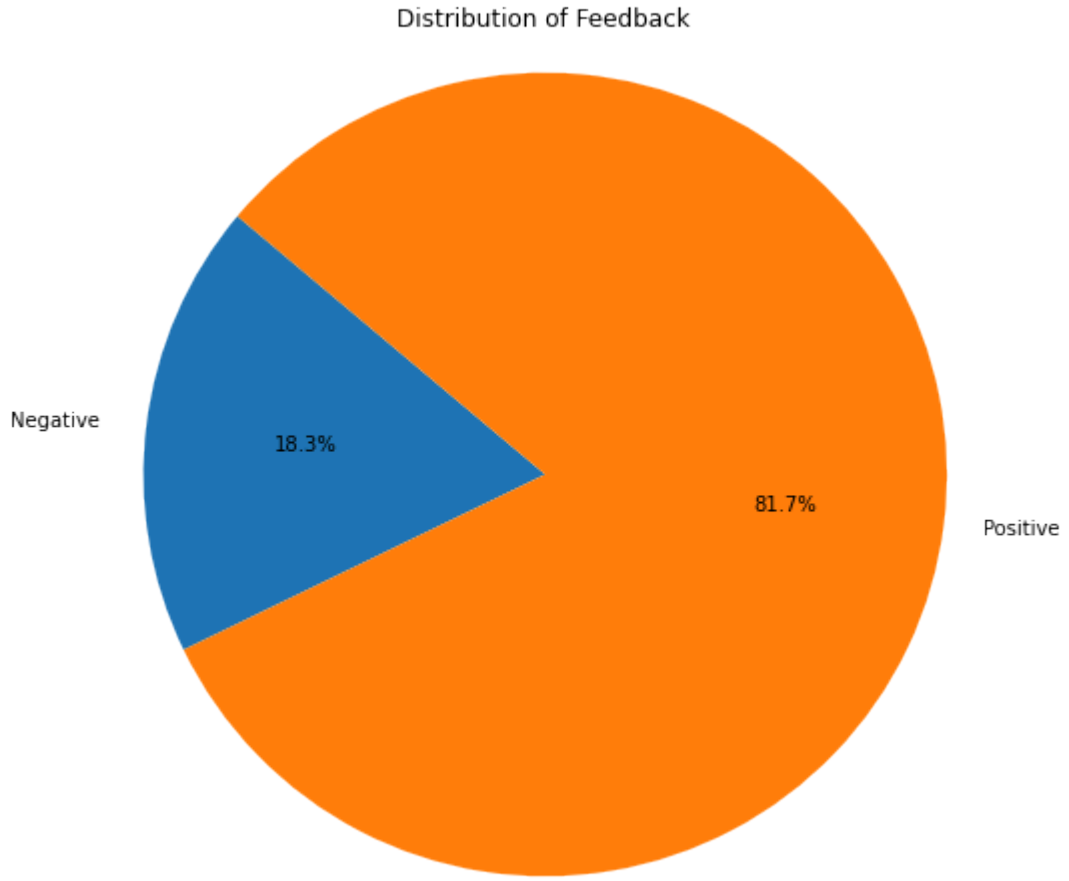
In [20]:
```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Gender', hue ="Occupation")
plt.xlabel('Gender')
plt.ylabel('Occupation')
plt.title('Gender Vise Occupation')
plt.show()
```

In [21]:
```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Marital Status')
plt.xlabel('Marital Status')
plt.ylabel('Marital Status')
plt.title('Distribuation of Material Status')
plt.show()
```

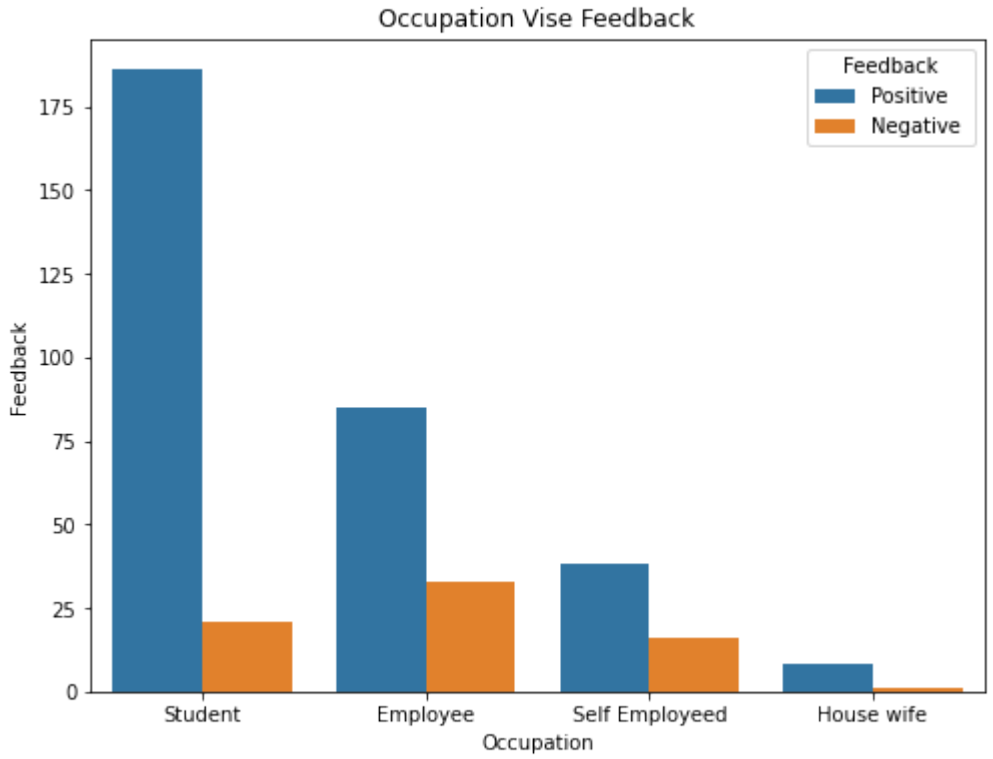


In [22]:
```
# Group the data by Feedback and calculate the count of each category
feedback_counts = df.groupby('Feedback').size()

# Create a pie chart
plt.figure(figsize=(8, 8))
plt.pie(feedback_counts, labels=feedback_counts.index, autopct='%1.1f%%', start
plt.title('Distribution of Feedback')
plt.axis('equal')  # Equal aspect ratio ensures that pie is drawn as a circle.
plt.show()
```



In [23]:
```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Occupation', hue ="Feedback")
plt.xlabel('Occupation')
plt.ylabel('Feedback')
plt.title('Occupation Vise Feedback')
plt.show()
```



In [25]:
```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='Monthly Income')
plt.xlabel('Monthly Income')
plt.ylabel('Educational Qualifications')
plt.title('Distribuation of Monthly Income')
plt.show()
```