In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
```

In [2]:

```
airline_data = pd.read_csv("airline_passenger_satisfaction.csv")
```

In [3]:

```
airline_data.head()
```

Out[3]:

	ID	Gender	Age	Customer Type	Type of Travel	Class	Flight Distance	Departure Delay	Arrival Delay	Departure and Arrival Time Convenience	.
0	1	Male	48	First-time	Business	Business	821	2	5.0	3	.
1	2	Female	35	Returning	Business	Business	821	26	39.0	2	.
2	3	Male	41	Returning	Business	Business	853	0	0.0	4	.
3	4	Male	50	Returning	Business	Business	1905	0	0.0	2	.
4	5	Female	49	Returning	Business	Business	3470	0	1.0	3	.

5 rows × 24 columns

In [4]:

```
airline_data.tail()
```

Out[4]:

	ID	Gender	Age	Customer Type	Type of Travel	Class	Flight Distance	Departure Delay	Arrival Delay	De an Conv
129875	129876	Male	28	Returning	Personal	Economy Plus	447	2	3.0	
129876	129877	Male	41	Returning	Personal	Economy Plus	308	0	0.0	
129877	129878	Male	42	Returning	Personal	Economy Plus	337	6	14.0	
129878	129879	Male	50	Returning	Personal	Economy Plus	337	31	22.0	
129879	129880	Female	20	Returning	Personal	Economy Plus	337	0	0.0	

5 rows × 24 columns

In [5]:

```
airline_data.shape
```

Out[5]:

```
(129880, 24)
```

In [6]:

```
airline_data.columns
```

Out[6]:

```
Index(['ID', 'Gender', 'Age', 'Customer Type', 'Type of Travel', 'Class',
       'Flight Distance', 'Departure Delay', 'Arrival Delay',
       'Departure and Arrival Time Convenience', 'Ease of Online Booking',
       'Check-in Service', 'Online Boarding', 'Gate Location',
       'On-board Service', 'Seat Comfort', 'Leg Room Service', 'Cleanlines
s',
       'Food and Drink', 'In-flight Service', 'In-flight Wifi Service',
       'In-flight Entertainment', 'Baggage Handling', 'Satisfaction'],
      dtype='object')
```

In [7]:

```
airline_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 129880 entries, 0 to 129879
Data columns (total 24 columns):
 #   Column                                  Non-Null Count   Dtype  
---  ------                                  --------------   -----  
 0   ID                                      129880 non-null  int64  
 1   Gender                                  129880 non-null  object 
 2   Age                                     129880 non-null  int64  
 3   Customer Type                           129880 non-null  object 
 4   Type of Travel                          129880 non-null  object 
 5   Class                                   129880 non-null  object 
 6   Flight Distance                         129880 non-null  int64  
 7   Departure Delay                         129880 non-null  int64  
 8   Arrival Delay                           129487 non-null  float64
 9   Departure and Arrival Time Convenience  129880 non-null  int64  
 10  Ease of Online Booking                  129880 non-null  int64  
 11  Check-in Service                        129880 non-null  int64  
 12  Online Boarding                         129880 non-null  int64  
 13  Gate Location                           129880 non-null  int64  
 14  On-board Service                        129880 non-null  int64  
 15  Seat Comfort                            129880 non-null  int64  
 16  Leg Room Service                        129880 non-null  int64  
 17  Cleanliness                             129880 non-null  int64  
 18  Food and Drink                          129880 non-null  int64  
 19  In-flight Service                       129880 non-null  int64  
 20  In-flight Wifi Service                  129880 non-null  int64  
 21  In-flight Entertainment                 129880 non-null  int64  
 22  Baggage Handling                        129880 non-null  int64  
 23  Satisfaction                            129880 non-null  object 
dtypes: float64(1), int64(18), object(5)
memory usage: 23.8+ MB
```

In [8]:

```
airline_data.describe()
```

Out[8]:

	ID	Age	Flight Distance	Departure Delay	Arrival Delay	Departur Arrival Conven
count	129880.000000	129880.000000	129880.000000	129880.000000	129487.000000	129880.0C
mean	64940.500000	39.427957	1190.316392	14.713713	15.091129	3.05
std	37493.270818	15.119360	997.452477	38.071126	38.465650	1.52
min	1.000000	7.000000	31.000000	0.000000	0.000000	0.0C
25%	32470.750000	27.000000	414.000000	0.000000	0.000000	2.0C
50%	64940.500000	40.000000	844.000000	0.000000	0.000000	3.0C
75%	97410.250000	51.000000	1744.000000	12.000000	13.000000	4.0C
max	129880.000000	85.000000	4983.000000	1592.000000	1584.000000	5.0C

In [9]:

```
airline_data.isnull().sum()
```

Out[9]:

```
ID                                        0
Gender                                    0
Age                                       0
Customer Type                             0
Type of Travel                            0
Class                                     0
Flight Distance                           0
Departure Delay                           0
Arrival Delay                           393
Departure and Arrival Time Convenience    0
Ease of Online Booking                    0
Check-in Service                          0
Online Boarding                           0
Gate Location                             0
On-board Service                          0
Seat Comfort                              0
Leg Room Service                          0
Cleanliness                               0
Food and Drink                            0
In-flight Service                         0
In-flight Wifi Service                    0
In-flight Entertainment                   0
Baggage Handling                          0
Satisfaction                              0
dtype: int64
```

In [10]:

```
airline_data = airline_data.dropna()
```

In [11]:

```
airline_data.shape
```

Out[11]:

```
(129487, 24)
```

In [12]:

```
airline_data.nunique()
```

Out[12]:

```
ID                                      129487
Gender                                       2
Age                                         75
Customer Type                                2
Type of Travel                               2
Class                                        3
Flight Distance                           3821
Departure Delay                            464
Arrival Delay                              472
Departure and Arrival Time Convenience       6
Ease of Online Booking                       6
Check-in Service                             6
Online Boarding                              6
Gate Location                                6
On-board Service                             6
Seat Comfort                                 6
Leg Room Service                             6
Cleanliness                                  6
Food and Drink                               6
In-flight Service                            6
In-flight Wifi Service                       6
In-flight Entertainment                      6
Baggage Handling                             5
Satisfaction                                 2
dtype: int64
```

In [13]:

```
airline_data.columns
```

Out[13]:

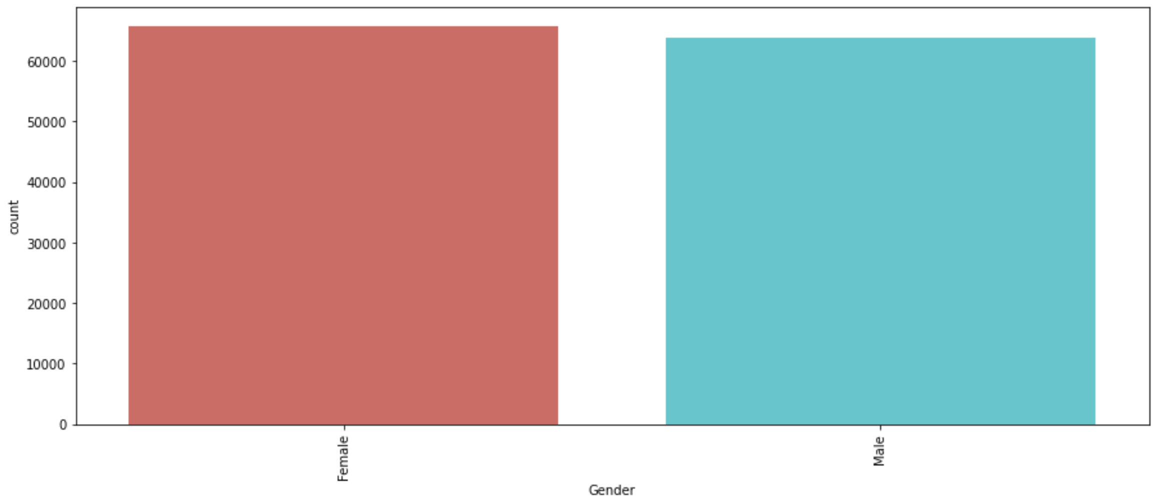
```
Index(['ID', 'Gender', 'Age', 'Customer Type', 'Type of Travel', 'Class',
       'Flight Distance', 'Departure Delay', 'Arrival Delay',
       'Departure and Arrival Time Convenience', 'Ease of Online Booking',
       'Check-in Service', 'Online Boarding', 'Gate Location',
       'On-board Service', 'Seat Comfort', 'Leg Room Service', 'Cleanlines
s',
       'Food and Drink', 'In-flight Service', 'In-flight Wifi Service',
       'In-flight Entertainment', 'Baggage Handling', 'Satisfaction'],
      dtype='object')
```

In [14]:

```
airline_data1 = airline_data[['Gender', 'Age', 'Customer Type', 'Type of Travel',
                              'Class','Departure and Arrival Time Convenience',
                              'Ease of Online Booking','Check-in Service',
                              'Online Boarding', 'Gate Location',
                              'On-board Service', 'Seat Comfort', 'Leg Room Service',
                              'Cleanliness', 'Food and Drink', 'In-flight Service',
                              'In-flight Wifi Service','In-flight Entertainment',
                              'Baggage Handling', 'Satisfaction']]
```

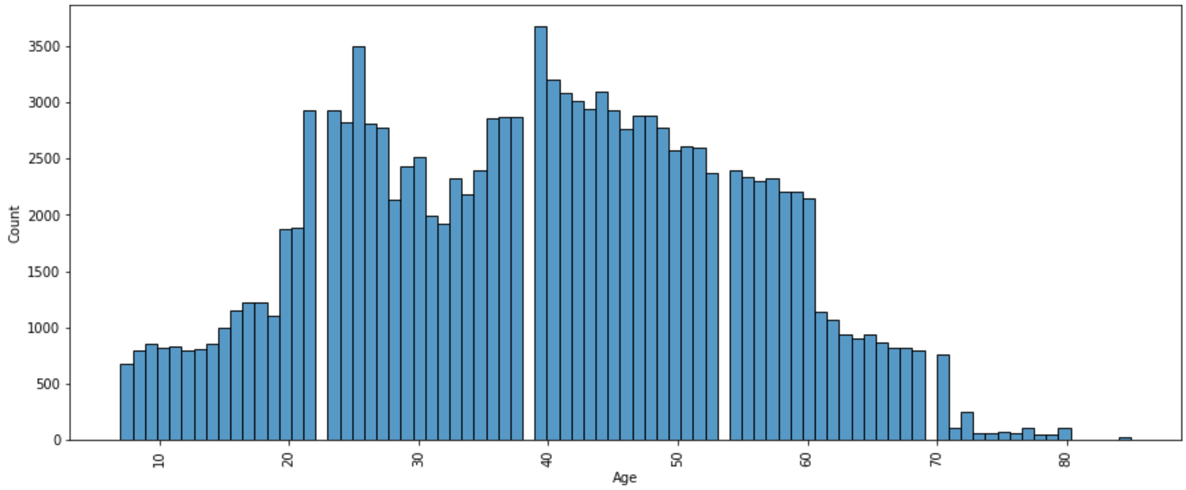
In [21]:

```
for i in airline_data1.columns:
    plt.figure(figsize=(15,6))
    sns.countplot(x = airline_data1[i][1:],data=airline_data1.iloc[1:],
                  order=airline_data1[i][1:].value_counts().index,
                  palette='hls')
    plt.xticks(rotation = 90)
    plt.show()
```



In [23]:

```
plt.figure(figsize=(15,6))
sns.histplot(airline_data1['Age'], palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [24]:

```
airline_data1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 129487 entries, 0 to 129879
Data columns (total 20 columns):
 #   Column                                 Non-Null Count   Dtype 
---  ------                                 --------------   ----- 
 0   Gender                                 129487 non-null  object
 1   Age                                    129487 non-null  int64 
 2   Customer Type                          129487 non-null  object
 3   Type of Travel                         129487 non-null  object
 4   Class                                  129487 non-null  object
 5   Departure and Arrival Time Convenience 129487 non-null  int64 
 6   Ease of Online Booking                 129487 non-null  int64 
 7   Check-in Service                       129487 non-null  int64 
 8   Online Boarding                        129487 non-null  int64 
 9   Gate Location                          129487 non-null  int64 
 10  On-board Service                       129487 non-null  int64 
 11  Seat Comfort                           129487 non-null  int64 
 12  Leg Room Service                       129487 non-null  int64 
 13  Cleanliness                            129487 non-null  int64 
 14  Food and Drink                         129487 non-null  int64 
 15  In-flight Service                      129487 non-null  int64 
 16  In-flight Wifi Service                 129487 non-null  int64 
 17  In-flight Entertainment                129487 non-null  int64 
 18  Baggage Handling                       129487 non-null  int64 
 19  Satisfaction                           129487 non-null  object
dtypes: int64(15), object(5)
memory usage: 24.8+ MB
```

In [26]:

```
airline_data1.columns
```

Out[26]:

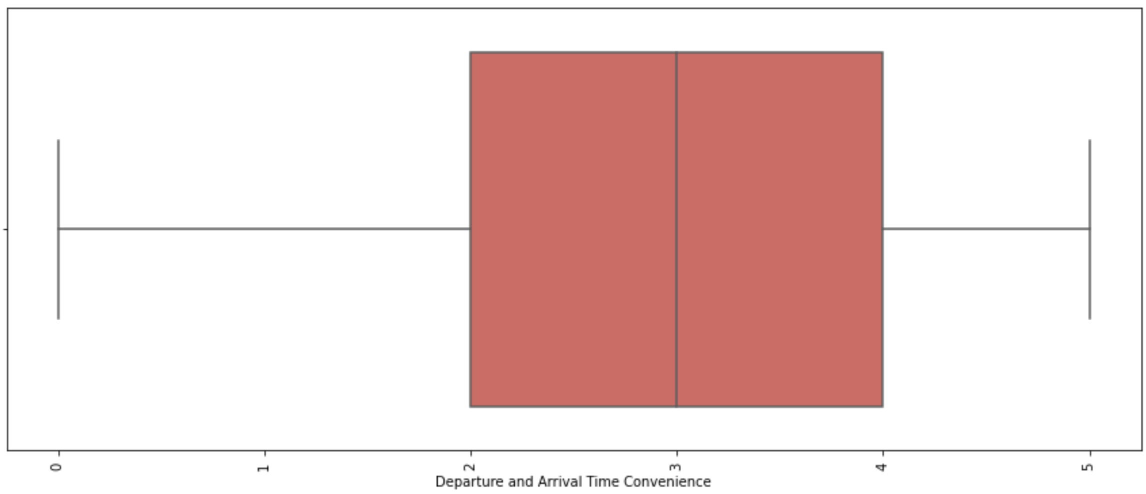
```
Index(['Gender', 'Age', 'Customer Type', 'Type of Travel', 'Class',
       'Departure and Arrival Time Convenience', 'Ease of Online Booking',
       'Check-in Service', 'Online Boarding', 'Gate Location',
       'On-board Service', 'Seat Comfort', 'Leg Room Service', 'Cleanlines
s',
       'Food and Drink', 'In-flight Service', 'In-flight Wifi Service',
       'In-flight Entertainment', 'Baggage Handling', 'Satisfaction'],
      dtype='object')
```

In [27]:

```
airline_data2 = airline_data1[['Departure and Arrival Time Convenience',
                               'Ease of Online Booking',
       'Check-in Service', 'Online Boarding', 'Gate Location',
       'On-board Service', 'Seat Comfort', 'Leg Room Service', 'Cleanliness',
       'Food and Drink', 'In-flight Service', 'In-flight Wifi Service',
       'In-flight Entertainment', 'Baggage Handling']]
```

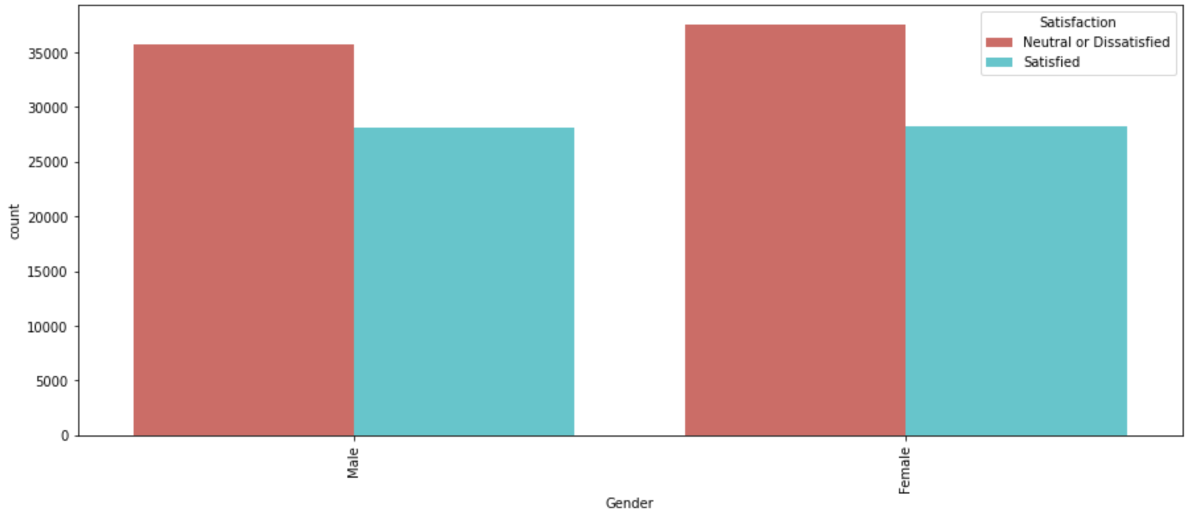
In [30]:

```
for j in airline_data2.columns:
    plt.figure(figsize=(15,6))
    sns.boxplot(x = airline_data2[j][1:],data=airline_data2.iloc[1:],
                order=airline_data2[j][1:].value_counts().index,
                palette='hls')
    plt.xticks(rotation = 90)
    plt.show()
```



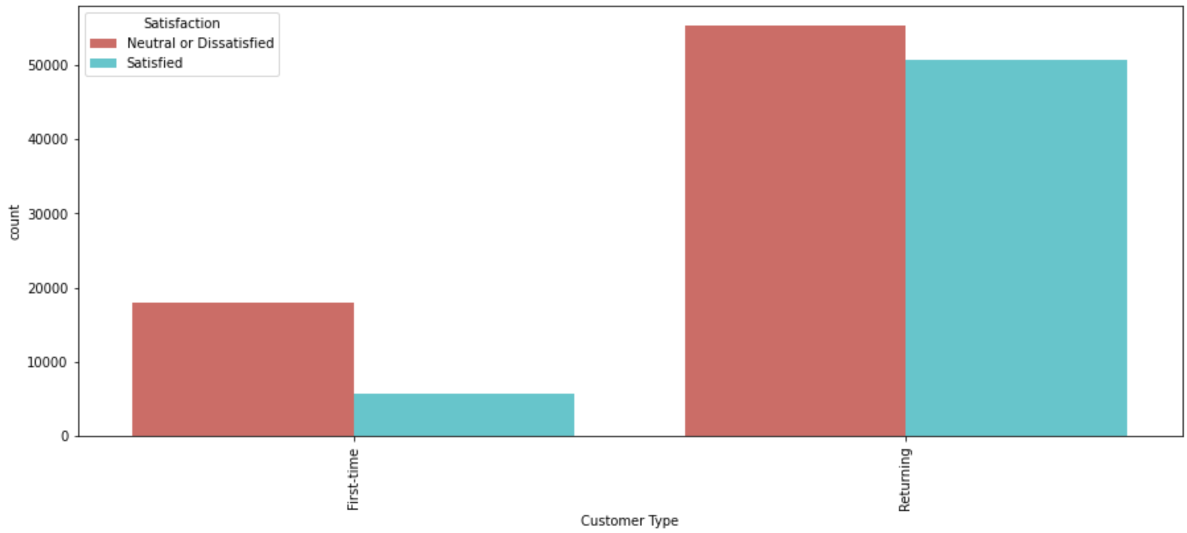
In [31]:

```
plt.figure(figsize=(15,6))
sns.countplot('Gender', hue = 'Satisfaction',
              data = airline_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



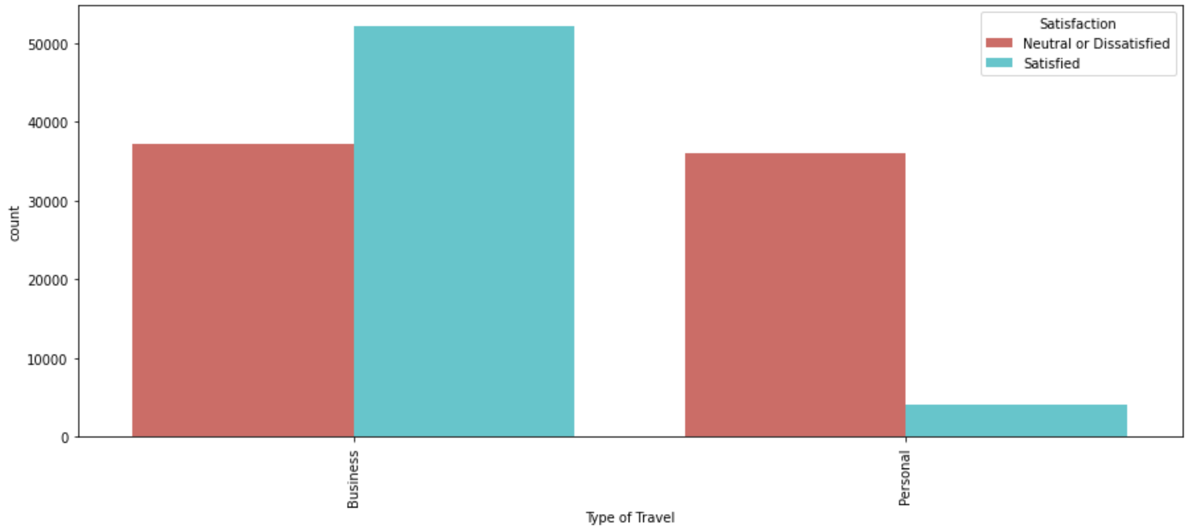
In [32]:

```
plt.figure(figsize=(15,6))
sns.countplot('Customer Type', hue = 'Satisfaction',
              data = airline_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



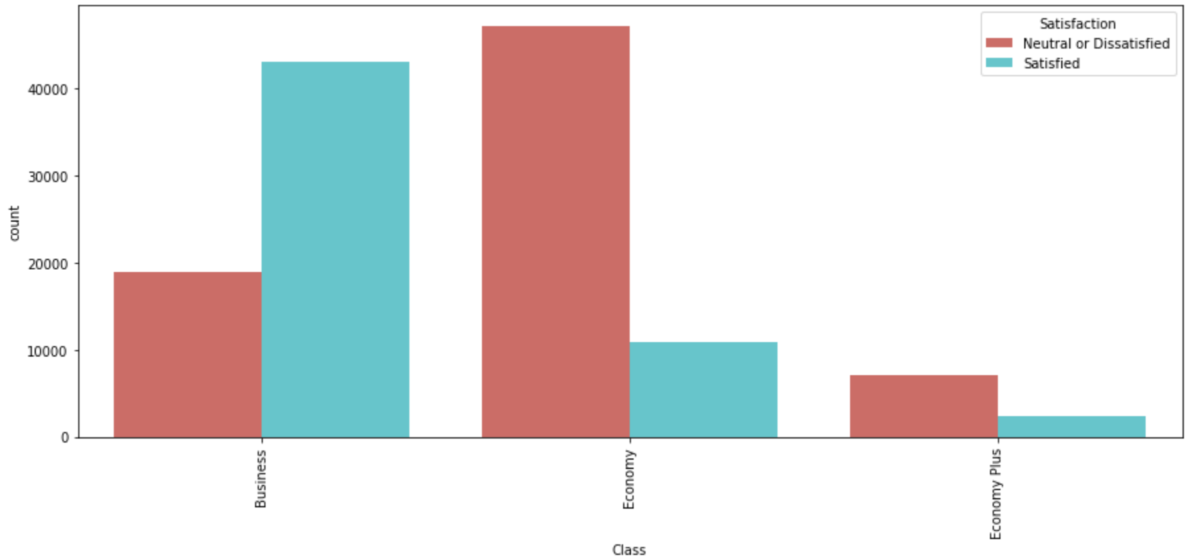
In [33]:

```
plt.figure(figsize=(15,6))
sns.countplot('Type of Travel', hue = 'Satisfaction',
              data = airline_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [34]:

```
plt.figure(figsize=(15,6))
sns.countplot('Class', hue = 'Satisfaction',
              data = airline_data, palette='hls')
plt.xticks(rotation = 90)
plt.show()
```



In [36]:

```
airline_data3 = airline_data[['Gender', 'Customer Type', 'Type of Travel',
                              'Class', 'Satisfaction']]
```

In [35]:

```
from sklearn import preprocessing
label_encoder = preprocessing.LabelEncoder()
```

In [37]:

```
airline_data3.loc[:,:] = airline_data3.loc[:,:].apply(label_encoder.fit_transform)
```

In [38]:

```
airline_data3.head()
```

Out[38]:

	Gender	Customer Type	Type of Travel	Class	Satisfaction
0	1	0	0	0	0
1	0	1	0	0	1
2	1	1	0	0	1
3	1	1	0	0	1
4	0	1	0	0	1

In [40]:

```
airline_data[list(airline_data3.columns)] = airline_data3
airline_data = airline_data.apply(pd.to_numeric, errors='coerce')
airline_data.head()
```

Out[40]:

	ID	Gender	Age	Customer Type	Type of Travel	Class	Flight Distance	Departure Delay	Arrival Delay	Departure and Arrival Time Convenience	...	bo Serv
0	1	1	48	0	0	0	821	2	5.0	3	...	
1	2	0	35	1	0	0	821	26	39.0	2	...	
2	3	1	41	1	0	0	853	0	0.0	4	...	
3	4	1	50	1	0	0	1905	0	0.0	2	...	
4	5	0	49	1	0	0	3470	0	1.0	3	...	

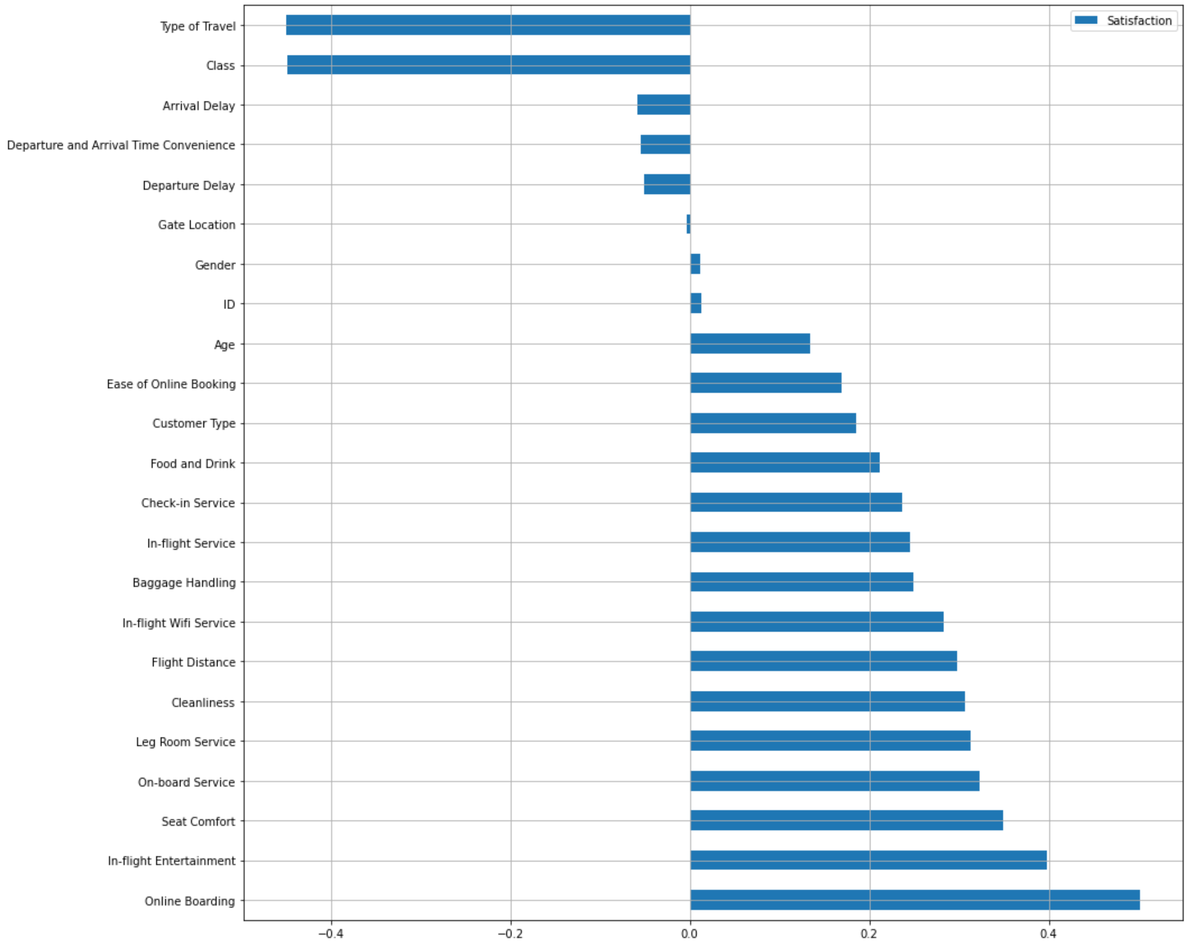
5 rows × 24 columns

In [41]:

```
corr = pd.DataFrame(airline_data.corr()['Satisfaction']).drop('Satisfaction',axis=0).sor
corr.plot(kind='barh',grid=True,figsize=(15,15))
```

Out[41]:

<AxesSubplot:>



In [42]:

```
feature_columns = []

for x in corr.index:
    if corr.loc[x].values < -0.4:
        feature_columns.append(x)
    elif corr.loc[x].values > 0.2:
        feature_columns.append(x)
```

In [43]:

```
print(feature_columns)
```

```
['Online Boarding', 'In-flight Entertainment', 'Seat Comfort', 'On-board S
ervice', 'Leg Room Service', 'Cleanliness', 'Flight Distance', 'In-flight
Wifi Service', 'Baggage Handling', 'In-flight Service', 'Check-in Servic
e', 'Food and Drink', 'Class', 'Type of Travel']
```

In [44]:

```
X = airline_data[feature_columns]
y = airline_data['Satisfaction']
```

In [45]:

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
```

In [46]:

```
X = pd.DataFrame(scaler.fit_transform(X),columns=X.columns)
```

In [47]:

```
X.shape
```

Out[47]:

```
(129487, 14)
```

In [49]:

```
y.shape
```

Out[49]:

```
(129487,)
```

In [50]:

```
x_train = X.iloc[:100001]
```

In [51]:

```
y_train = y.iloc[:100001]
```

In [52]:

```
x_test = X.iloc[100001:129488]
```

In [53]:

```
y_test = y.iloc[100001:129488]
```

In [54]:

```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [55]:

```
lr = LogisticRegression()
lr.fit(x_train,y_train)
```

Out[55]:

```
LogisticRegression()
```

In [56]:

```
lr_pred = lr.predict(x_test)
```

In [57]:

```
print("Training Accuracy :", lr.score(x_train, y_train))
print("Testing Accuracy :", lr.score(x_test, y_test))
```

```
Training Accuracy : 0.8501414985850142
Testing Accuracy : 0.8721087973953741
```

In [58]:

```
print('Logistic Regression Accuracy Score:',
      accuracy_score(lr_pred,y_test))
```

```
Logistic Regression Accuracy Score: 0.8721087973953741
```

In [59]:

```
from sklearn.tree import DecisionTreeClassifier
dt = DecisionTreeClassifier()
```

In [60]:

```
dt.fit(x_train,y_train)
dt_pred = dt.predict(x_test)
```

In [61]:

```
print("Training Accuracy :", dt.score(x_train, y_train))
print("Testing Accuracy :", dt.score(x_test, y_test))
```

```
Training Accuracy : 0.999980000199998
Testing Accuracy : 0.9404463135047141
```

In [62]:

```
print('Decision Tree Accuracy Score:',
      accuracy_score(dt_pred,y_test))
```

```
Decision Tree Accuracy Score: 0.9404463135047141
```

In [63]:

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier()
```

In [65]:

```
rf.fit(x_train,y_train)
rf_pred = rf.predict(x_test)
```

In [66]:

```
print("Training Accuracy :", rf.score(x_train, y_train))
print("Testing Accuracy :", rf.score(x_test, y_test))
```

```
Training Accuracy : 0.999970000299997
Testing Accuracy : 0.9600488367360781
```

In [67]:

```
print('Random Forest Accuracy Score :', rf.score(x_test,y_test))
```

```
Random Forest Accuracy Score : 0.9600488367360781
```

In [68]:

```
from xgboost import XGBClassifier
xgb = XGBClassifier()
```

In [69]:

```
xgb.fit(x_train, y_train)
xgb_pred = xgb.predict(x_test)
```

In [70]:

```
print("Training Accuracy :", xgb.score(x_train, y_train))
print("Testing Accuracy :", xgb.score(x_test, y_test))
```

```
Training Accuracy : 0.959990400095999
Testing Accuracy : 0.9597096927355355
```

In [71]:

```
print('XGB Accuracy Score:', accuracy_score(xgb_pred,y_test))
```

```
XGB Accuracy Score: 0.9597096927355355
```