Project Name - Customer Satisfaction Prediction (ML _ FA _ DA projects)(Part 1)

Project Type - Data Analysis

Industry - Unified Mentor

Contribution - Individual

Member Name - Hare Krishana Mishra

Task - 1

Project Summary -

Project Description:

The Customer Support Analysis project involves exploring and analyzing a customer support ticket dataset to identify common issues, track support trends, and segment customers based on ticket characteristics. The dataset includes details such as customer demographics, product purchased, type of support request, resolution status, priority levels, and satisfaction ratings. Through data visualization and segmentation, the project helps uncover key patterns in customer service operations, providing valuable insights for process improvement and decision-making.

Objective:

The objective of this project is to analyze customer support tickets to improve service quality and efficiency. Specifically, the project aims to:

- Identify the most frequent support issues.
- Track ticket trends over time.
- Understand customer demographics and behavior.
- Segment customers based on ticket types and satisfaction ratings.
- Provide insights for optimizing ticket resolution processes.

Key Project Details:

Domain: Data Analytics / Exploratory Data Analysis (EDA)

Difficulty Level: Advanced

Tools & Technologies: Python, Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, Jupyter Notebook, VS Code, SQL, Excel

Dataset: Customer Support Ticket Dataset containing fields like Ticket ID, Customer Age, Gender, Product Purchased, Ticket Type, Status, Resolution, Priority, Channel, First Response Time, Time to Resolution, and Customer Satisfaction Rating.

Key Steps:

- Data Preprocessing and Cleaning
- Exploratory Data Analysis (EDA)
- Visualizing Ticket Trends and Common Issues
- Segmentation by Ticket Type and Satisfaction Rating
- Analysis of Demographics and Support Channels

Use Cases: Identifying key service pain points, monitoring ticket trends, improving resource allocation, and enhancing the customer experience.

Let's Begin:-

Data Preprocessing

In [ ]:
```
# Importing necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusior
```

In [ ]:
```
# Load the dataset
data = pd.read_csv('/content/customer_support_tickets.csv')
data
```

Out[ ]:

	Ticket ID	Customer Name	Customer Email	Customer Age	Customer Gender	F Pur
0	1	Marisa Obrien	carrollallison@example.com	32	Other	GoI
1	2	Jessica Rios	clarkeashley@example.com	42	Female	L
2	3	Christopher Robbins	gonzalestracy@example.com	48	Other	I
3	4	Christina Dillon	bradleyolson@example.org	27	Female	M
4	5	Alexander Carroll	bradleymark@example.com	67	Female	A A
...	...	...	...	...	...	
8464	8465	David Todd	adam28@example.net	22	Female	L
8465	8466	Lori Davis	russell68@example.com	27	Female	So
8466	8467	Michelle Kelley	ashley83@example.org	57	Female	
8467	8468	Steven Rodriguez	fpowell@example.org	54	Male	Play
8468	8469	Steven Davis MD	lori20@example.net	53	Other	Phi

8469 rows × 17 columns

In [ ]:

```
# Display basic info about the dataset
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8469 entries, 0 to 8468
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Ticket ID                     8469 non-null   int64  
 1   Customer Name                 8469 non-null   object 
 2   Customer Email                8469 non-null   object 
 3   Customer Age                  8469 non-null   int64  
 4   Customer Gender               8469 non-null   object 
 5   Product Purchased             8469 non-null   object 
 6   Date of Purchase              8469 non-null   object 
 7   Ticket Type                   8469 non-null   object 
 8   Ticket Subject                8469 non-null   object 
 9   Ticket Description            8469 non-null   object 
 10  Ticket Status                 8469 non-null   object 
 11  Resolution                    2769 non-null   object 
 12  Ticket Priority               8469 non-null   object 
 13  Ticket Channel                8469 non-null   object 
 14  First Response Time           5650 non-null   object 
 15  Time to Resolution            2769 non-null   object 
 16  Customer Satisfaction Rating  2769 non-null   float64
dtypes: float64(1), int64(2), object(14)
memory usage: 1.1+ MB
None
```

In [ ]:

```python
# Data Preprocessing
# Handling missing values
data = data.dropna()
```

In [ ]:

```python
# Encoding categorical variables
label_encoders = {}
for column in data.select_dtypes(include=['object']).columns:
  label_encoders[column] = LabelEncoder()
  data[column] = label_encoders[column].fit_transform(data[column])
```

In [ ]:

```python
# Define features and target variable
X = data.drop(['Customer Email', 'Customer Satisfaction Rating'], axis=1)
y = data['Customer Satisfaction Rating']
```

In [ ]:

```python
# Splitting the dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,ranc
```

In [ ]:

```python
from sklearn.preprocessing import LabelEncoder, StandardScaler

# Make a copy to avoid altering the original data
df = data.copy()

# Drop columns that are not useful for prediction (IDs, names, emails, free-
df = df.drop(['Ticket ID', 'Customer Name', 'Customer Email', 'Ticket Subjec

# Encode categorical columns
label_encoders = {}
for col in df.select_dtypes(include=['object']).columns:
    label_encoders[col] = LabelEncoder()
```

```python
    df[col] = label_encoders[col].fit_transform(df[col].astype(str))

# Define features and target
X = df.drop('Customer Satisfaction Rating', axis=1)
y = df['Customer Satisfaction Rating']

# Train-test split
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rar

# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [ ]:
```python
# Remove rows where target is NaN
df = df.dropna(subset=['Customer Satisfaction Rating'])

# Now split again
X = df.drop('Customer Satisfaction Rating', axis=1)
y = df['Customer Satisfaction Rating']

from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rar

# Scale features
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Train the model
from sklearn.ensemble import RandomForestClassifier
rfc = RandomForestClassifier(random_state=42)
rfc.fit(X_train, y_train)
```

Out[ ]:
```
▼ RandomForestClassifier
RandomForestClassifier(random_state=42)
```

In [ ]:
```python
# Predict on the test set
y_pred = rfc.predict(X_test)
```

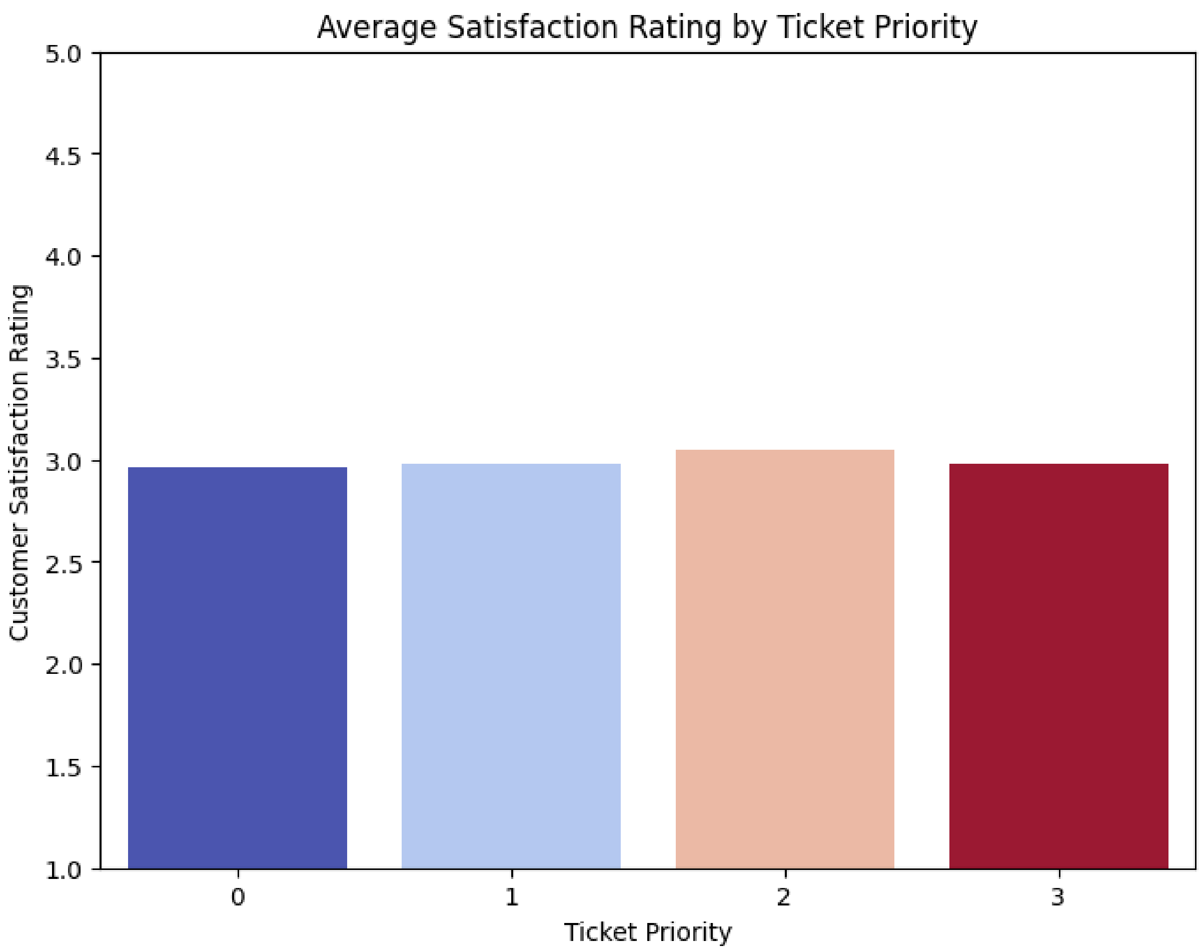
In [ ]:
```python
# Model Evaluation
print("Accuracy:", accuracy_score(y_test, y_pred))
print("Classification Report:\n", classification_report(y_test, y_pred))
```

```
Accuracy: 0.2069795427196149
Classification Report:
               precision    recall  f1-score   support

         1.0       0.21      0.19      0.20       168
         2.0       0.20      0.19      0.19       174
         3.0       0.23      0.25      0.24       175
         4.0       0.18      0.17      0.17       162
         5.0       0.21      0.24      0.22       152

    accuracy                           0.21       831
   macro avg       0.21      0.21      0.21       831
weighted avg       0.21      0.21      0.21       831
```

Exploratory Data Analysis (EDA)

Average Customer Satisfaction Score Across Ticket Priority Levels

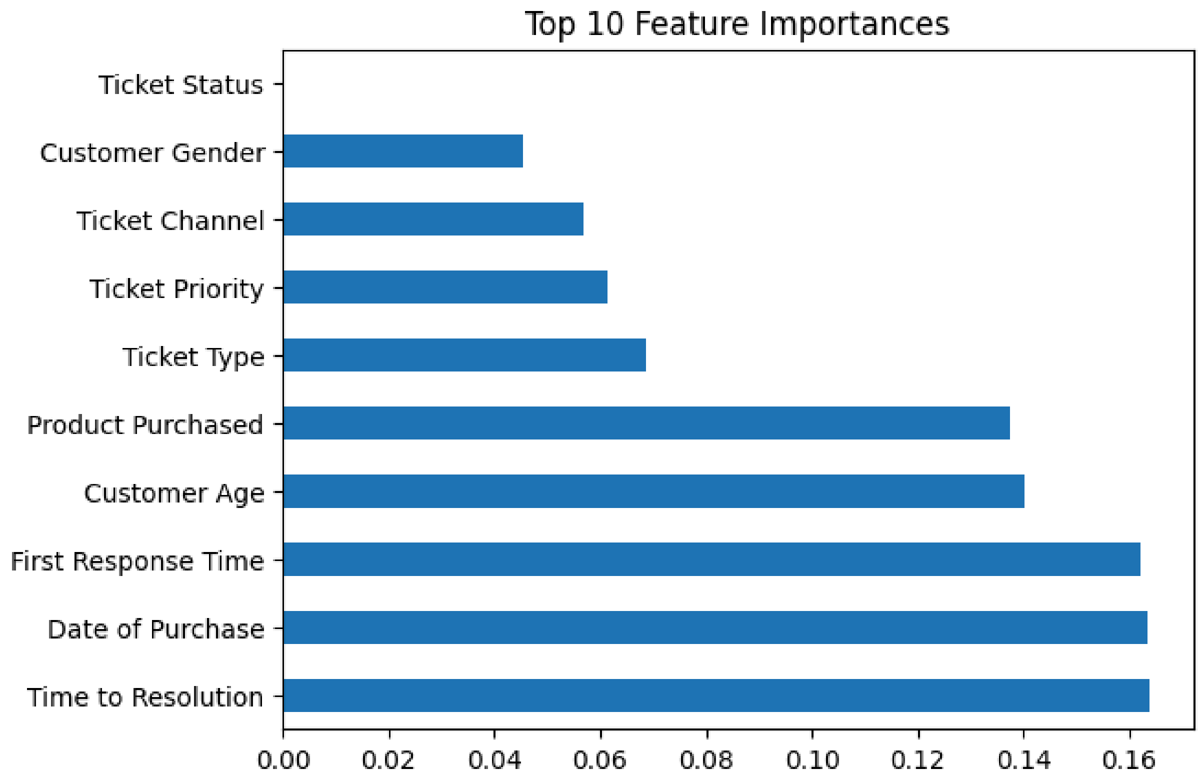
In [ ]:
```
plt.figure(figsize=(8,6))
avg_satisfaction_priority = df.groupby('Ticket Priority')['Customer Satisfac

sns.barplot(
    x='Ticket Priority',
    y='Customer Satisfaction Rating',
    hue='Ticket Priority',  # same as x
    data=avg_satisfaction_priority,
    palette='coolwarm',
    legend=False  # hides redundant legend
)

plt.title('Average Satisfaction Rating by Ticket Priority')
plt.ylim(1,5)
plt.show()
```



Top 10 Most Influential Features Driving Customer Satisfaction

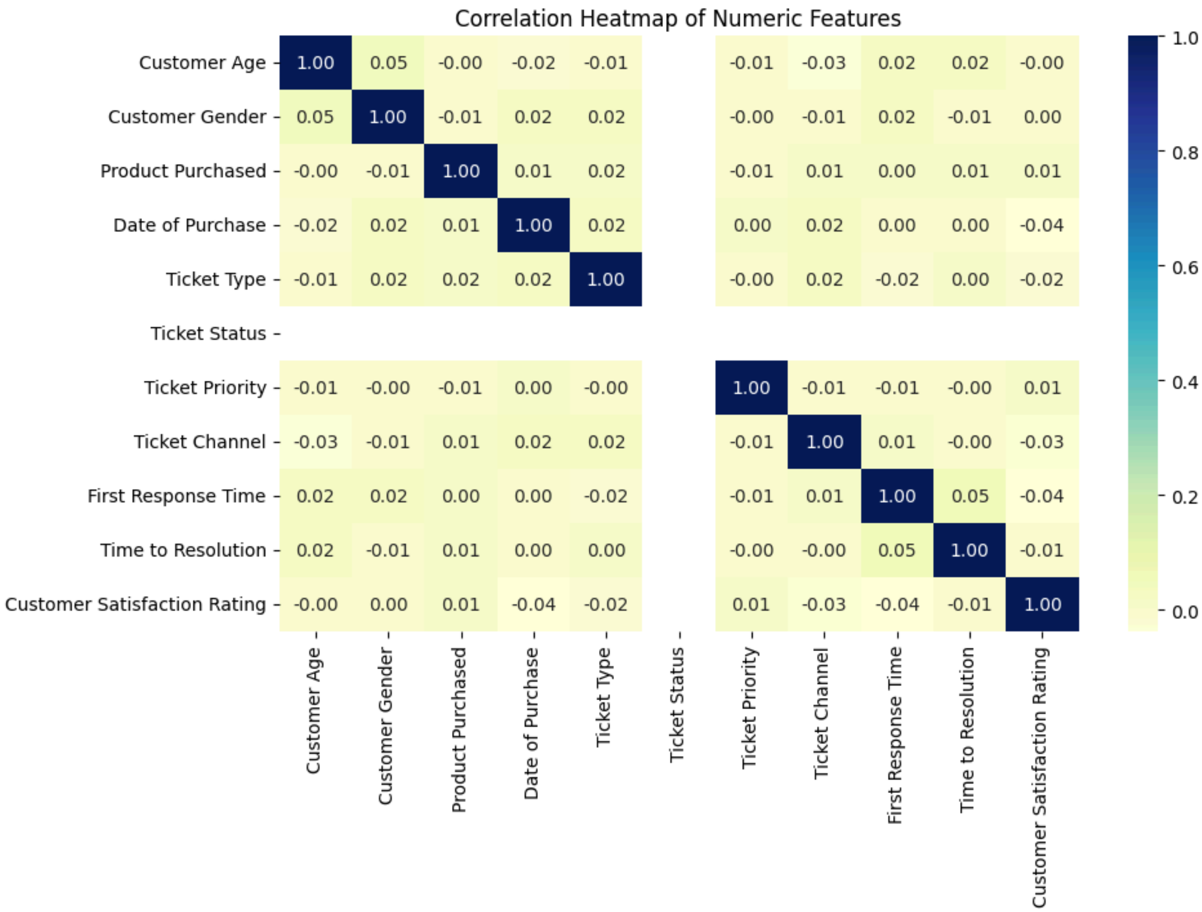
In [ ]:
```
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
# Visualization of Results
# Feature Importance
feature_importances = pd.Series(rfc.feature_importances_, index=X.columns)
feature_importances.nlargest(10).plot(kind='barh')
plt.title('Top 10 Feature Importances')
plt.show()
```

```
Confusion Matrix:
 [[32 34 40 33 29]
 [36 33 47 28 30]
 [36 31 44 30 34]
 [29 35 31 27 40]
 [21 33 32 30 36]]
```



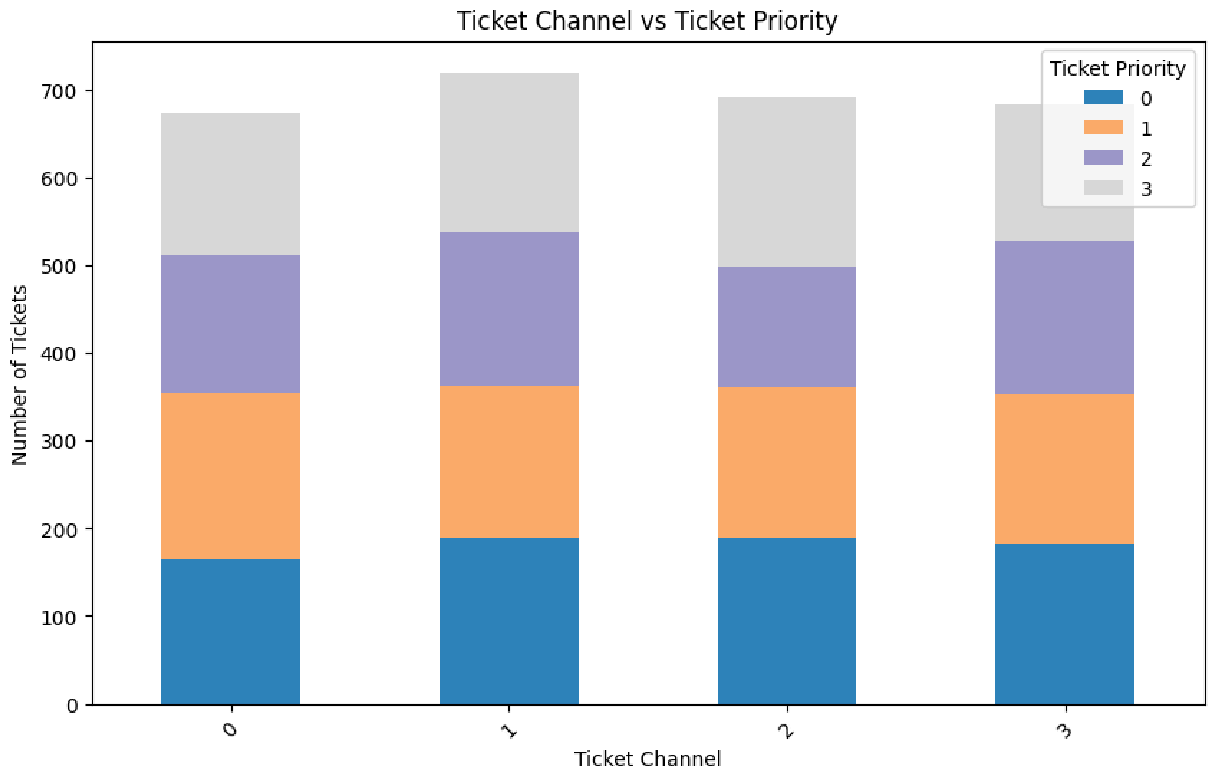
Correlation Heatmap: Relationships Between Numeric Features

In [ ]:
```
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(), annot=True, cmap='YlGnBu', fmt=".2f")
plt.title('Correlation Heatmap of Numeric Features')
plt.show()
```



Ticket Volume by Channel and Priority Level

In [ ]:
```
channel_priority = df.groupby(['Ticket Channel','Ticket Priority']).size().

channel_priority.plot(kind='bar', stacked=True, figsize=(10,6), colormap='ta
plt.title('Ticket Channel vs Ticket Priority')
plt.ylabel('Number of Tickets')
plt.xticks(rotation=45)
plt.show()
```



Ticket Channel vs Ticket Priority
Ticket Priority
0
1
2
3
Number of Tickets
0
100
200
300
400
500
600
700
0
1
2
3
Ticket Channel

Project Name - Customer Satisfaction Prediction (ML _ FA _ DA projects)(Part 2)

Project Type - Data Analysis

Industry - Unified Mentor

Contribution - Individual

Member Name - Hare Krishana Mishra

Task - 2

Project Summary -

Project Description:

Project Description The Customer Satisfaction Prediction project focuses on analyzing customer support ticket data to forecast customer satisfaction ratings. The dataset contains detailed information about customer demographics, purchased products, ticket types, support channels, priorities, resolution times, and satisfaction scores. Using data analysis and machine learning techniques, the project aims to uncover patterns in customer interactions, identify key factors affecting satisfaction, and generate actionable insights to enhance service quality and operational efficiency.

Objective:

The main objective of this project is to predict customer satisfaction levels using historical support ticket data. In addition, the project seeks to:

- Identify the most influential features impacting customer satisfaction.
- Provide insights to improve customer support processes.
- Enable proactive measures to address customer concerns.
- Support decision-making for product and service enhancements.

Key Project Details:

Domain: Data Science / Machine Learning

Difficulty Level: Advanced

Tools & Technologies: Python, Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, Jupyter Notebook, VS Code, SQL, Excel

Dataset: Customer Support Ticket Dataset containing fields like Ticket ID, Customer Age, Gender, Product Purchased, Ticket Type, Status, Resolution, Priority, Channel, First Response Time, Time to Resolution, and Customer Satisfaction Rating.

Key Steps:

- Data Preprocessing and Cleaning
- Exploratory Data Analysis (EDA)
- Feature Engineering
- Model Building using Machine Learning Algorithms
- Model Evaluation and Performance Metrics
- Visualization of Insights

Use Cases: Customer service performance tracking, satisfaction prediction, ticket resolution time forecasting, customer segmentation, and product feedback analysis.

Let's Begin:-

Data Preprocessing

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.cluster import KMeans
```

In [2]:
```
# Load the dataset
data =pd.read_csv("/content/customer_support_tickets.csv")
```

In [3]:
```
# Display the first few rows of the dataset
data.head()
```

Out[3]:

	Ticket ID	Customer Name	Customer Email	Customer Age	Customer Gender	Prod Purcha
0	1	Marisa Obrien	carrollallison@example.com	32	Other	GoPro H
1	2	Jessica Rios	clarkeashley@example.com	42	Female	LG Sn
2	3	Christopher Robbins	gonzalestracy@example.com	48	Other	Dell
3	4	Christina Dillon	bradleyolson@example.org	27	Female	Micro Of
4	5	Alexander Carroll	bradleymark@example.com	67	Female	Autoc Auto(

Exploratory Data Analysis (EDA

In [4]:

```
# Perform initial exploratory data analysis (EDA)
print(data.info())
data.describe()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8469 entries, 0 to 8468
Data columns (total 17 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   Ticket ID                     8469 non-null   int64  
 1   Customer Name                 8469 non-null   object 
 2   Customer Email                8469 non-null   object 
 3   Customer Age                  8469 non-null   int64  
 4   Customer Gender               8469 non-null   object 
 5   Product Purchased             8469 non-null   object 
 6   Date of Purchase              8469 non-null   object 
 7   Ticket Type                   8469 non-null   object 
 8   Ticket Subject                8469 non-null   object 
 9   Ticket Description            8469 non-null   object 
 10  Ticket Status                 8469 non-null   object 
 11  Resolution                    2769 non-null   object 
 12  Ticket Priority               8469 non-null   object 
 13  Ticket Channel                8469 non-null   object 
 14  First Response Time           5650 non-null   object 
 15  Time to Resolution            2769 non-null   object 
 16  Customer Satisfaction Rating  2769 non-null   float64
dtypes: float64(1), int64(2), object(14)
memory usage: 1.1+ MB
None
```

Out[4]:

	Ticket ID	Customer Age	Customer Satisfaction Rating
count	8469.000000	8469.000000	2769.000000
mean	4235.000000	44.026804	2.991333
std	2444.934048	15.296112	1.407016
min	1.000000	18.000000	1.000000
25%	2118.000000	31.000000	2.000000
50%	4235.000000	44.000000	3.000000
75%	6352.000000	57.000000	4.000000
max	8469.000000	70.000000	5.000000

In [5]:

```python
# Print column names
print(data.columns)
```

```
Index(['Ticket ID', 'Customer Name', 'Customer Email', 'Customer Age',
       'Customer Gender', 'Product Purchased', 'Date of Purchase',
       'Ticket Type', 'Ticket Subject', 'Ticket Description', 'Ticket Statu
s',
       'Resolution', 'Ticket Priority', 'Ticket Channel',
       'First Response Time', 'Time to Resolution',
       'Customer Satisfaction Rating'],
      dtype='object')
```

In [6]:

```python
#Analyze customer support ticket trends
# Identify common issues
```

```python
common_issues = data['Ticket Subject'].value_counts().head(10)
print("Top 10 Common Issues:")
print(common_issues)
```

```
Top 10 Common Issues:
Ticket Subject
Refund request           576
Software bug             574
Product compatibility    567
Delivery problem         561
Hardware issue           547
Battery life             542
Network problem          539
Installation support     530
Product setup            529
Payment issue            526
Name: count, dtype: int64
```

In [7]:

```python
# Plotting ticket trends over time
data['Date of Purchase'] = pd.to_datetime(data['Date of Purchase'])
data['YearMonth'] = data['Date of Purchase'].dt.to_period('M')
ticket_trends = data.groupby('YearMonth').size()
```

Visualization

In [8]:

```python
# Segment customers
# Segment based on ticket types
ticket_type_segmentation = data.groupby('Ticket Type').size()
print("\nSegmentation based on Ticket Types:")
print(ticket_type_segmentation)
```

```
Segmentation based on Ticket Types:
Ticket Type
Billing inquiry         1634
Cancellation request    1695
Product inquiry         1641
Refund request          1752
Technical issue         1747
dtype: int64
```

In [9]:

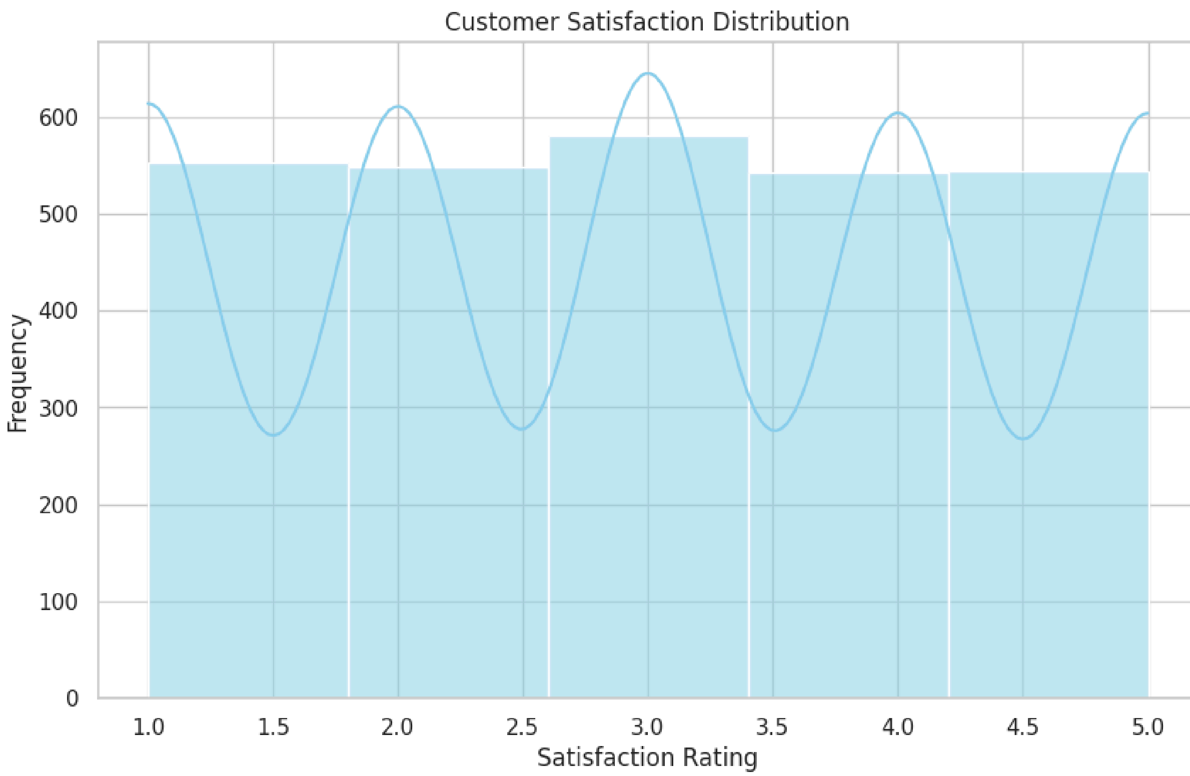
```python
# Segment based on satisfaction levels
satisfaction_segmentation = data.groupby('Customer Satisfaction Rating').siz
print("\nSegmentation based on Customer Satisfaction Levels:")
print(satisfaction_segmentation)
```

```
Segmentation based on Customer Satisfaction Levels:
Customer Satisfaction Rating
1.0    553
2.0    549
3.0    580
4.0    543
5.0    544
dtype: int64
```

In [10]:
```python
# Set up the plotting aesthetics
sns.set(style="whitegrid")
```

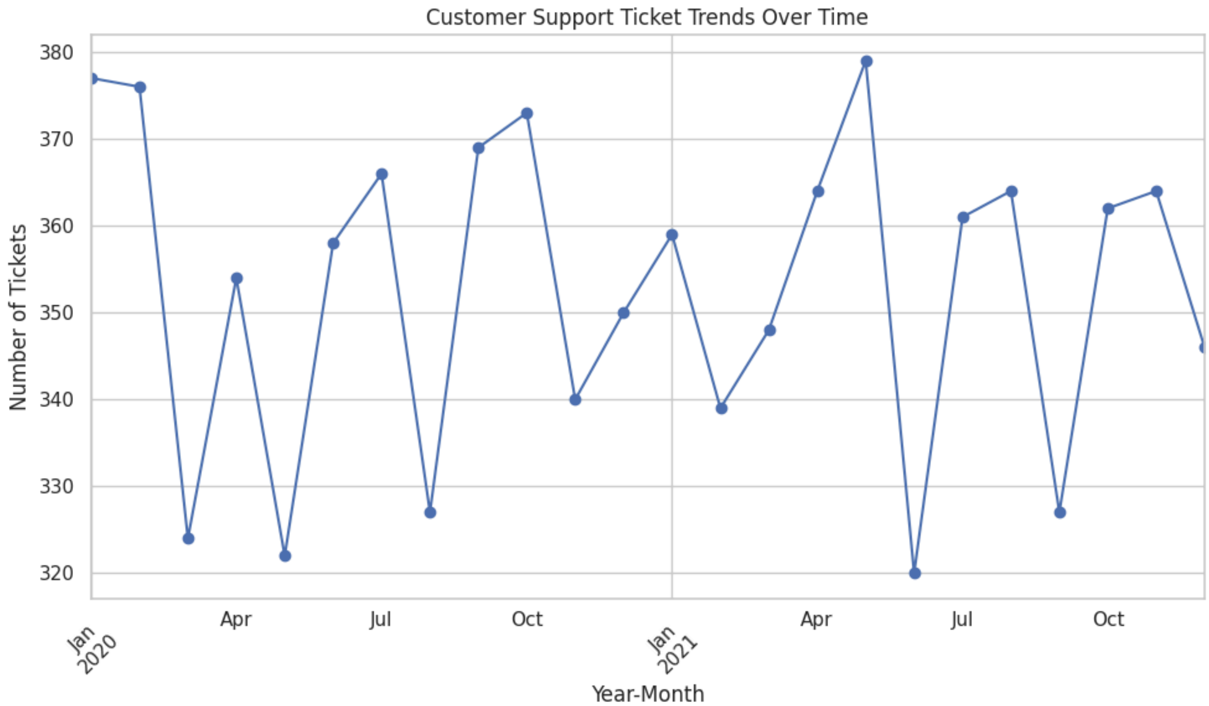
Distribution of Customer Satisfaction Ratings

In [11]:
```python
#Customer Satisfaction Distribution
plt.figure(figsize=(10, 6))
sns.histplot(data['Customer Satisfaction Rating'], bins=5,
kde=True, color='skyblue')
plt.title('Customer Satisfaction Distribution')
plt.xlabel('Satisfaction Rating')
plt.ylabel('Frequency')
plt.show()
```



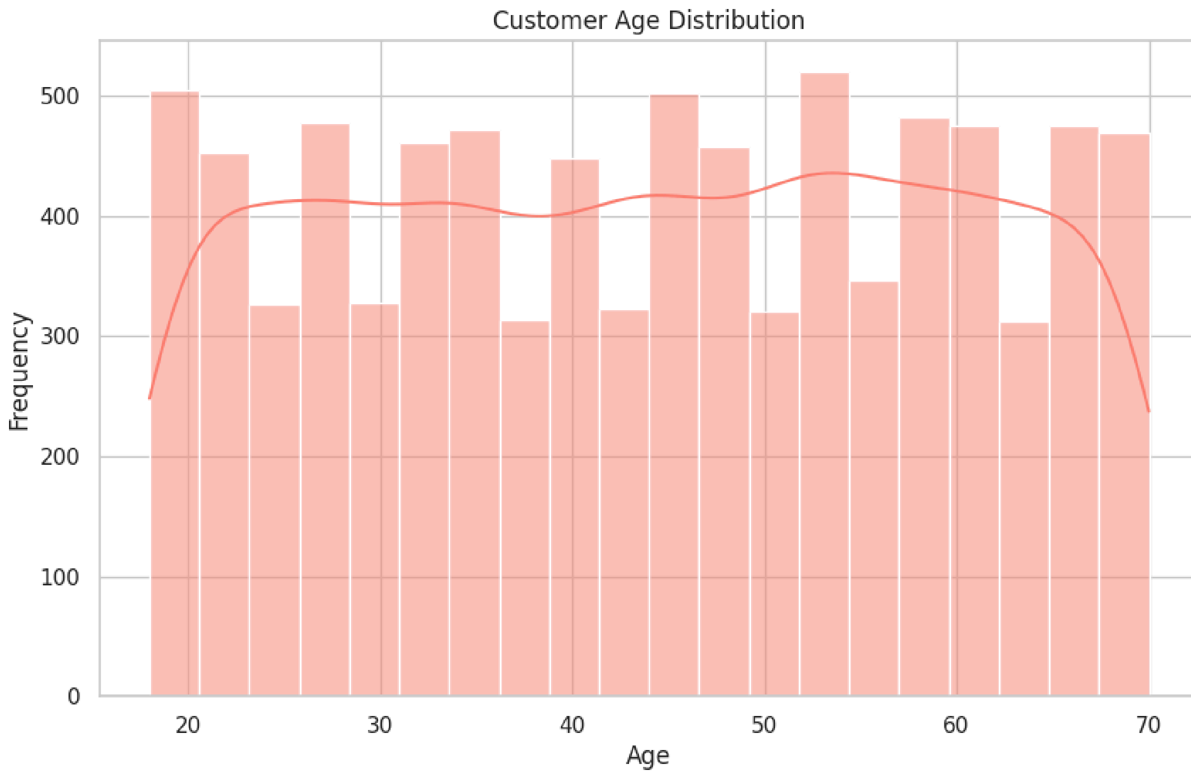
Monthly Trends in Customer Support Ticket Volume

In [12]:
```python
plt.figure(figsize=(10, 6))
ticket_trends.plot(kind='line', marker='o')
plt.title('Customer Support Ticket Trends Over Time')
plt.xlabel('Year-Month')
plt.ylabel('Number of Tickets')
plt.grid(True)
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



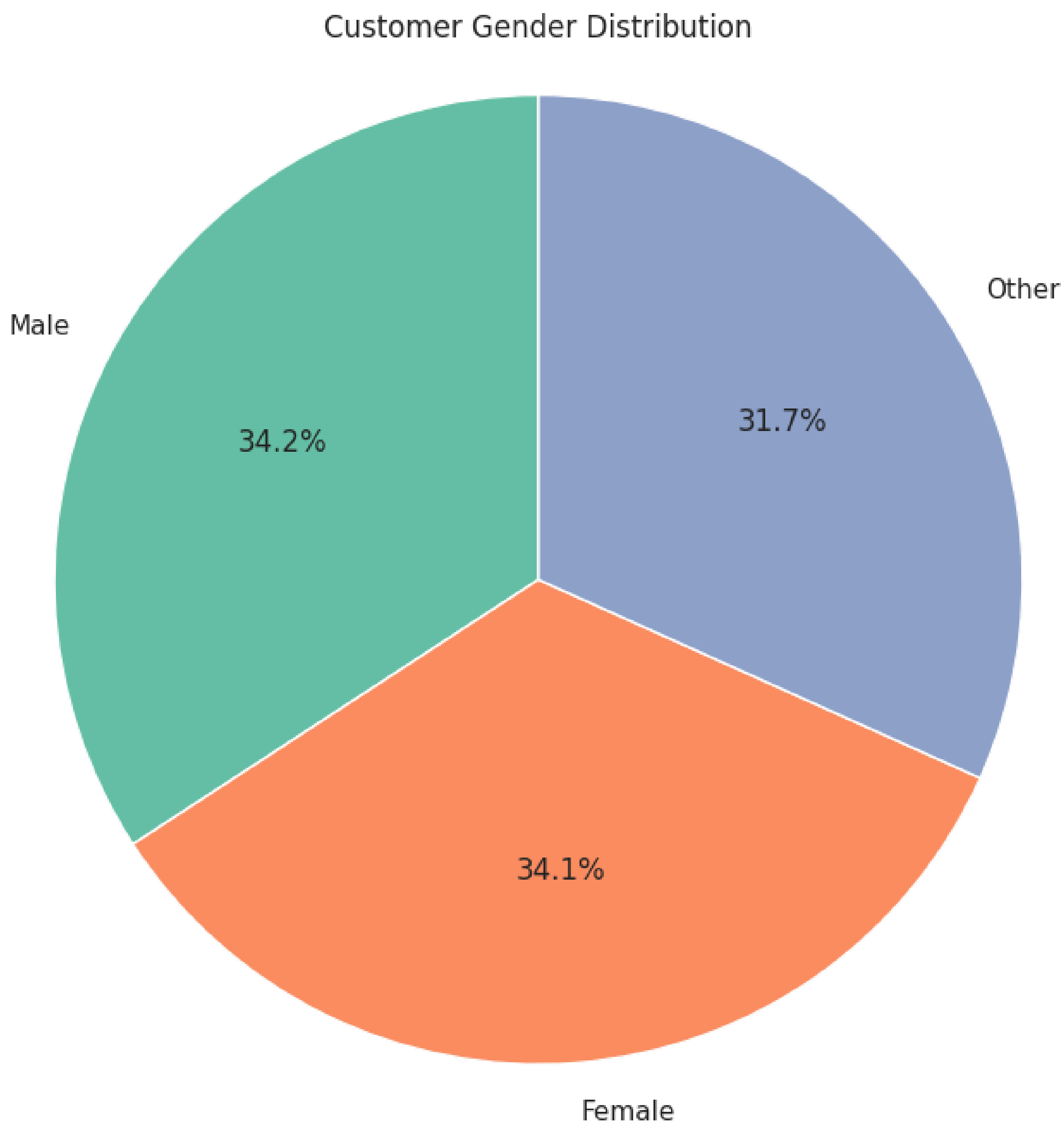
Distribution of Customer Ages

In [13]:
```
#Customer Age Distribution
plt.figure(figsize=(10, 6))
sns.histplot(data['Customer Age'], bins=20, kde=True,
color='salmon')
plt.title('Customer Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```



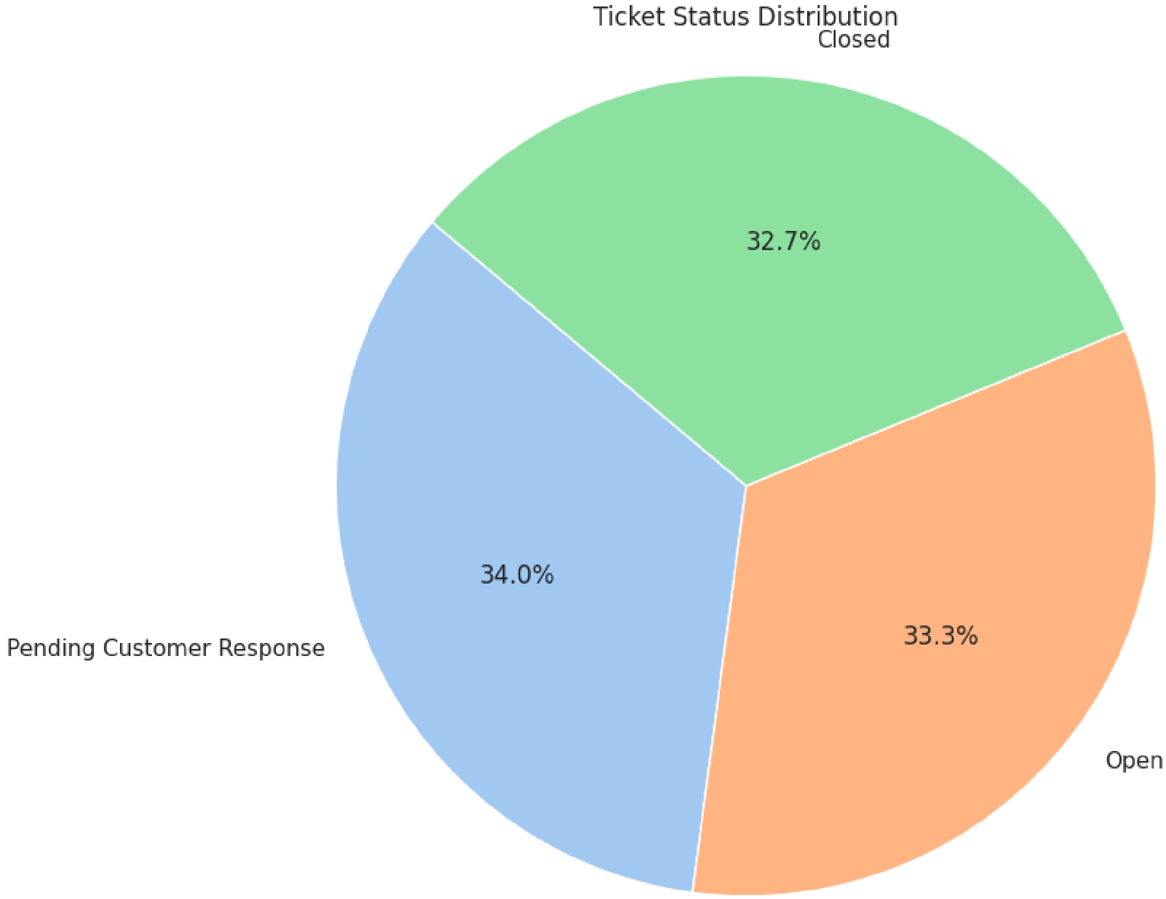
Gender Distribution of Customers

In [14]:
```
#Customer Gender Distribution
customer_gender_distribution = data['Customer Gender'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(customer_gender_distribution,
labels=customer_gender_distribution.index, autopct='%1.1f%%',
colors=sns.color_palette('Set2'), startangle=90)
plt.title('Customer Gender Distribution')
plt.axis('equal')
plt.show()
```



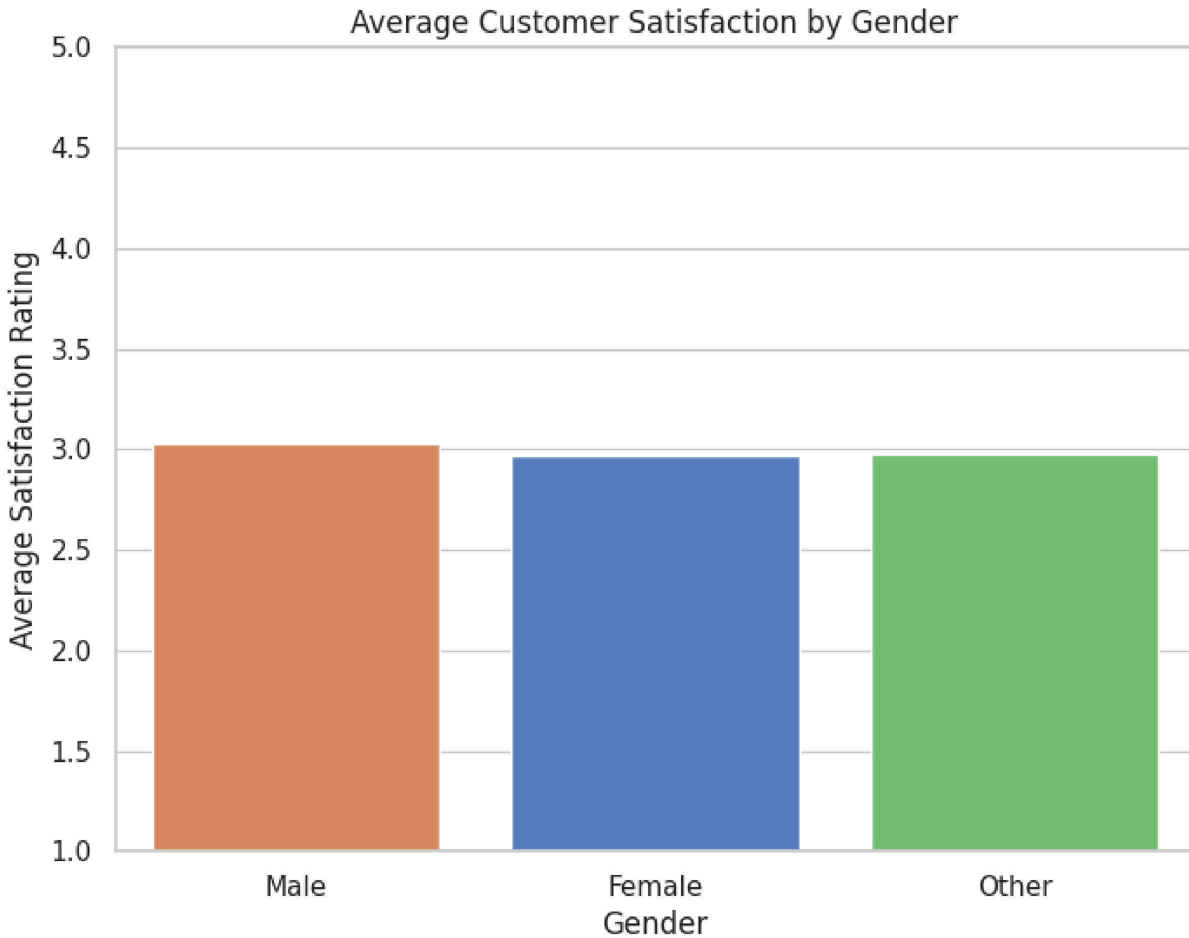
Distribution of Ticket Statuses

In [15]:
```
#Ticket Status Distribution
ticket_status_distribution = data['Ticket Status'].value_counts()
plt.figure(figsize=(8, 8))
plt.pie(ticket_status_distribution,
labels=ticket_status_distribution.index, autopct='%1.1f%%',
colors=sns.color_palette('pastel'), startangle=140)
plt.title('Ticket Status Distribution')
plt.axis('equal')
plt.show()
```



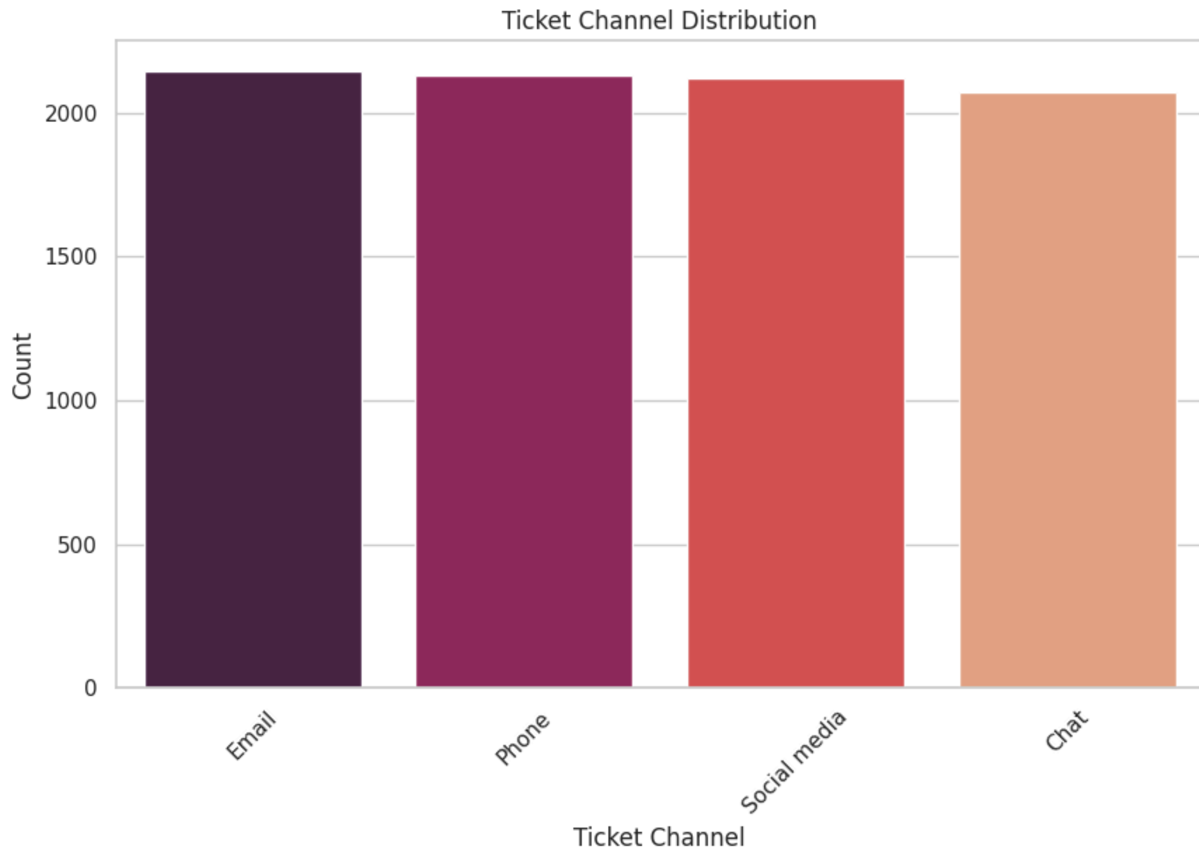
Average Customer Satisfaction Ratings by Gender

In [16]:
```
# Chart 1: Average Customer Satisfaction by Gender (Bar Plot)
average_satisfaction = data.groupby('Customer Gender')['Customer Satisfactio

plt.figure(figsize=(8, 6))
sns.barplot(
    x='Customer Gender',
    y='Customer Satisfaction Rating',
    hue='Customer Gender',  # same as x
    data=average_satisfaction,
    palette='muted',
    order=['Male', 'Female', 'Other'],
    legend=False  # hides redundant legend
)

plt.title('Average Customer Satisfaction by Gender')
plt.xlabel('Gender')
plt.ylabel('Average Satisfaction Rating')
plt.ylim(1, 5)  # Adjust y-axis limit if needed
plt.show()
```



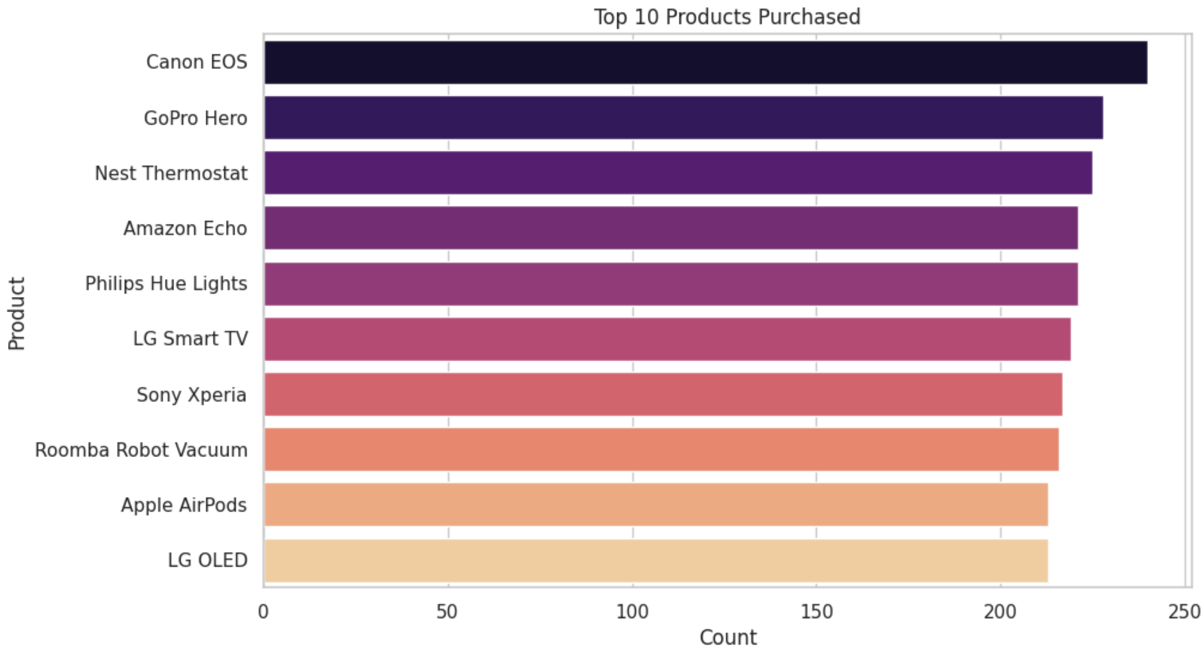
Distribution of Tickets by Channel

In [17]:
```
# Ticket Channel Distribution
plt.figure(figsize=(10, 6))
ticket_channel_distribution = data['Ticket Channel'].value_counts().reset_ir
ticket_channel_distribution.columns = ['Ticket Channel', 'Count']

sns.barplot(
    x='Ticket Channel',
    y='Count',
    hue='Ticket Channel',  # same as x
    data=ticket_channel_distribution,
    palette='rocket',
    legend=False  # hide duplicate legend
)

plt.title('Ticket Channel Distribution')
plt.xlabel('Ticket Channel')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```



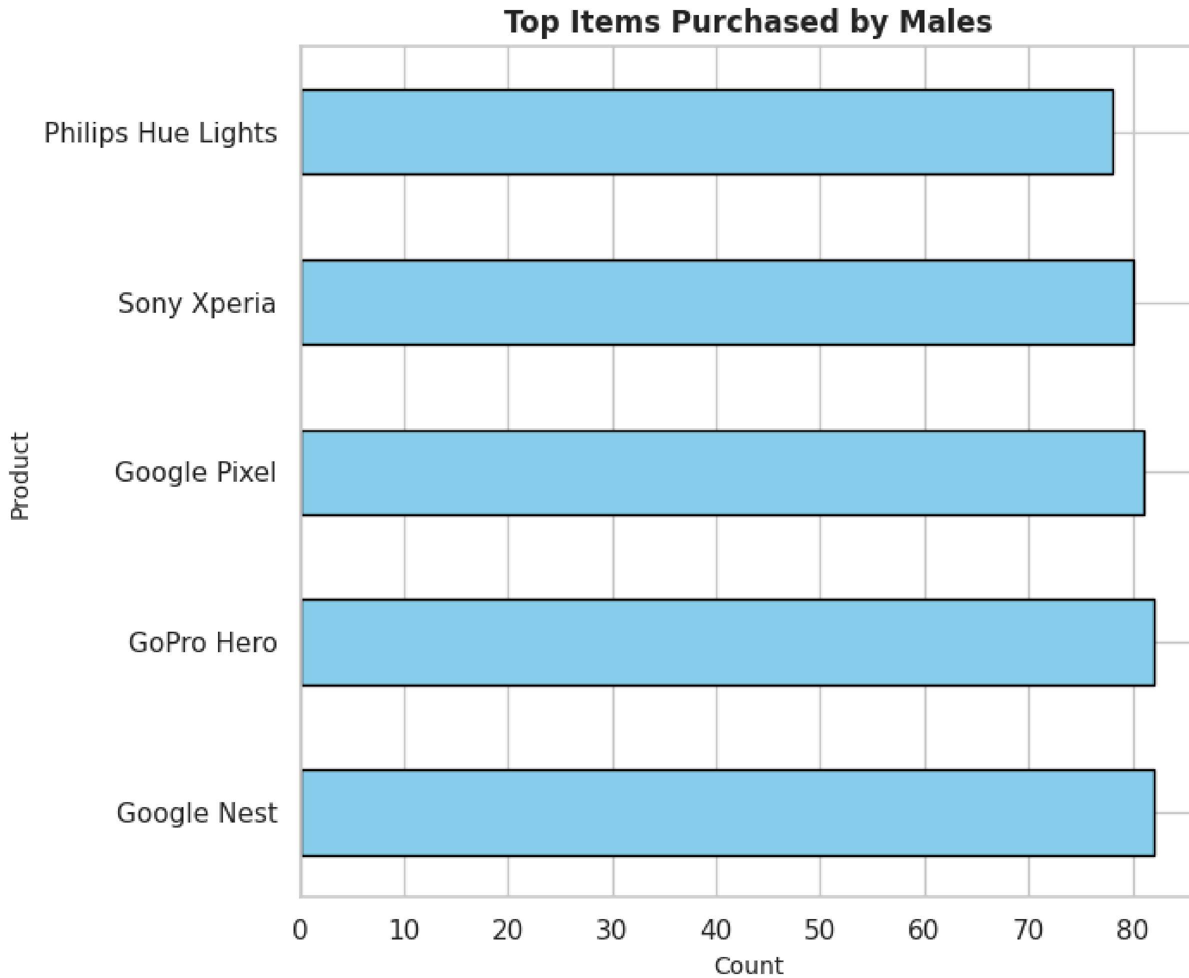
Top 10 Most Purchased Products

In [18]:
```
# Product Purchased Distribution - Top 10
plt.figure(figsize=(10, 6))

# Prepare data
product_purchased_distribution = (
    data['Product Purchased']
    .value_counts()
    .head(10)
    .reset_index()
)
product_purchased_distribution.columns = ['Product Purchased', 'Count']

# Plot with hue same as y to avoid warning
sns.barplot(
    y='Product Purchased',
    x='Count',
    hue='Product Purchased',  # same as y
    data=product_purchased_distribution,
    palette='magma',
    legend=False
)

plt.title('Top 10 Products Purchased')
plt.xlabel('Count')
plt.ylabel('Product')
plt.show()
```



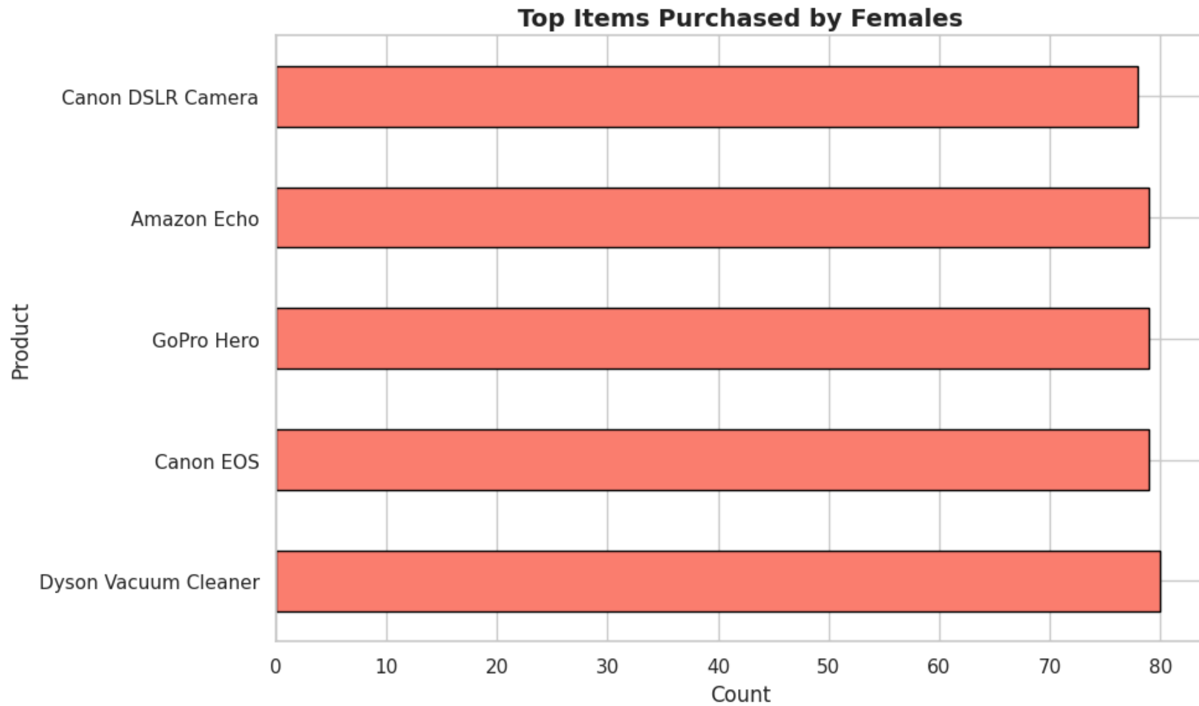
Most Popular Products Among Male Customers

In [19]:
```
# Chart 2: Top Items Purchased by Gender (Horizontal Bar Chart)
plt.figure(figsize=(18, 6))  # wider overall figure

# Top Items Purchased by Males
plt.subplot(1, 3, 1)
top_items_male = (
    data[data['Customer Gender'] == 'Male']['Product Purchased']
    .value_counts()
    .head(5)
)
top_items_male.plot(kind='barh', color='skyblue', edgecolor='black')
plt.title('Top Items Purchased by Males', fontsize=12, fontweight='bold')
plt.xlabel('Count', fontsize=10)
plt.ylabel('Product', fontsize=10)

plt.tight_layout()
plt.show()
```



Most Popular Products Among Female Customers

In [20]:
```
# Top Items Purchased by Females
plt.figure(figsize=(10, 6))  # wider chart

top_items_female = (
    data[data['Customer Gender'] == 'Female']['Product Purchased']
    .value_counts()
    .head(5)
)

top_items_female.plot(kind='barh', color='salmon', edgecolor='black')

plt.title('Top Items Purchased by Females', fontsize=14, fontweight='bold')
plt.xlabel('Count', fontsize=12)
plt.ylabel('Product', fontsize=12)

plt.tight_layout()
plt.show()
```

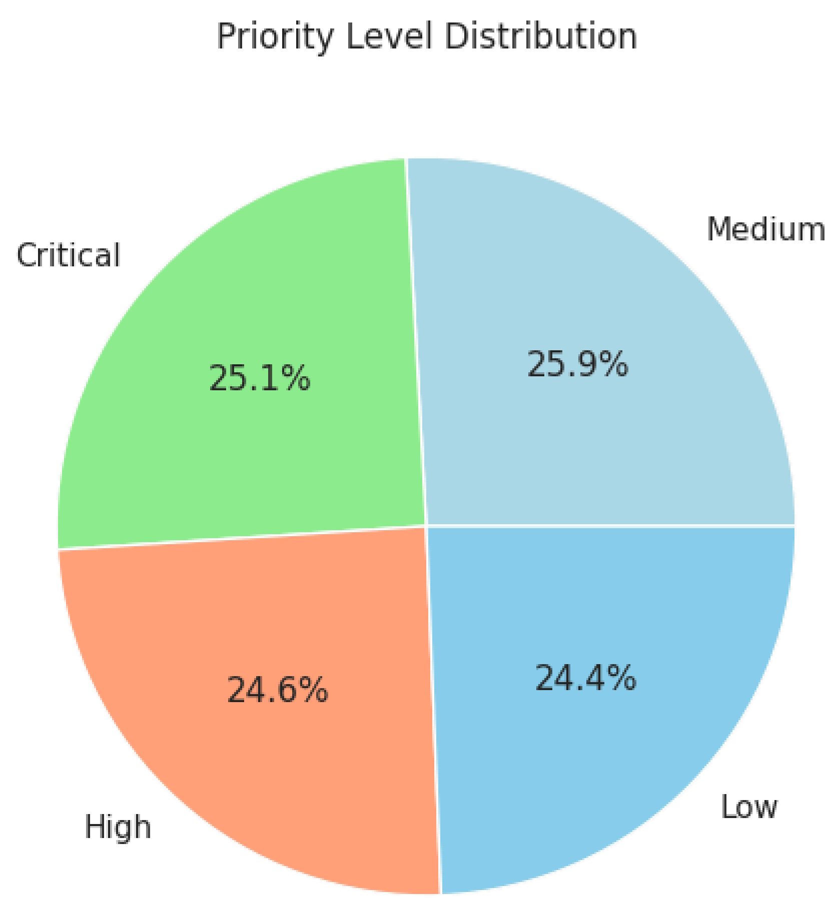


Top 5 Products Purchased by Other Gender Customers

In [21]:
```
# Top Items Purchased by Other Gender
plt.figure(figsize=(10, 6))  # wider chart

top_items_other = (
    data[data['Customer Gender'] == 'Other']['Product Purchased']
    .value_counts()
    .head(5)
)

top_items_other.plot(kind='barh', color='lightgreen', edgecolor='black')

plt.title('Top Items Purchased by Other Genders', fontsize=14, fontweight='b
plt.xlabel('Count', fontsize=12)
plt.ylabel('Product', fontsize=12)

plt.tight_layout()
plt.show()
```

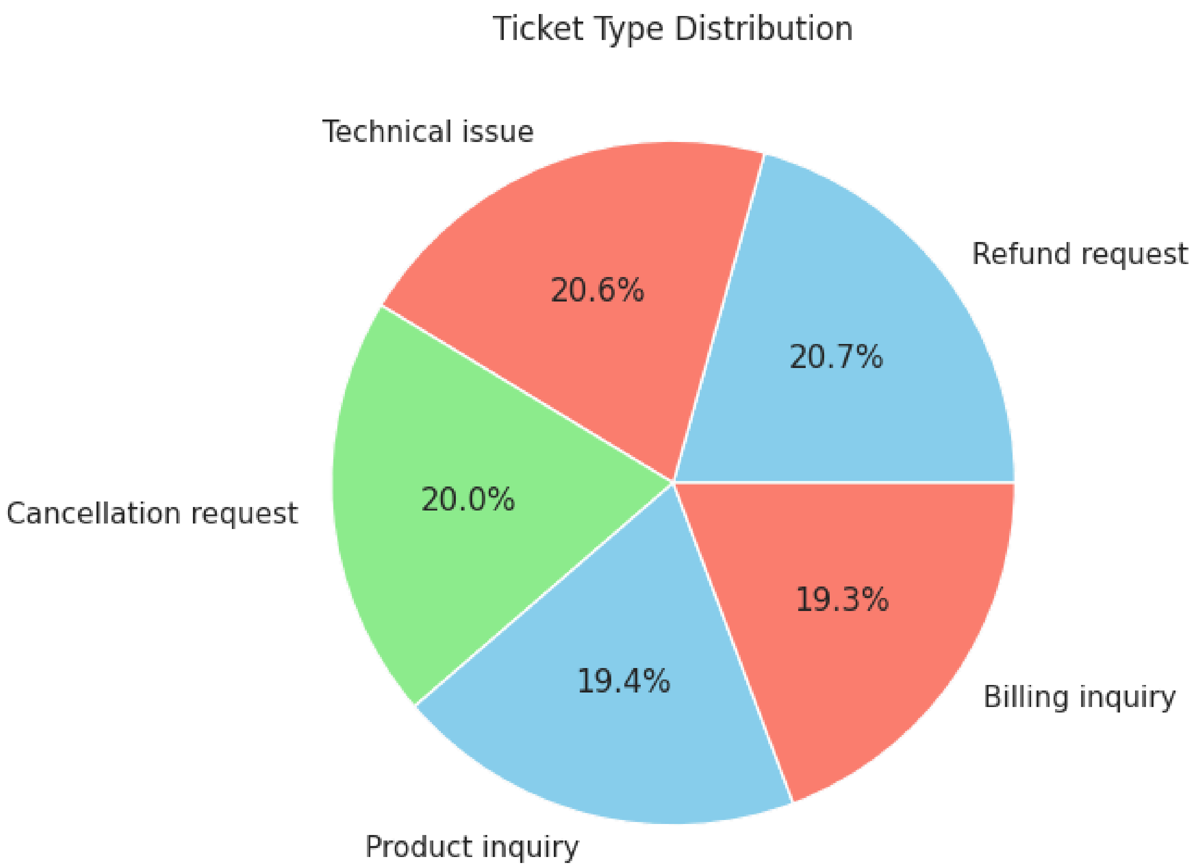
Proportion of Tickets by Priority Level

In [22]:
```
# Count ticket priorities
priority_distribution = data['Ticket Priority'].value_counts()
# Plot
plt.figure(figsize=(8, 6))
priority_distribution.plot(kind='pie', autopct='%1.1f%%',
colors=['lightblue', 'lightgreen', 'lightsalmon', 'skyblue'])
plt.title('Priority Level Distribution')
plt.ylabel('')
plt.show()
```



Distribution of Ticket Types

In [23]:
```
# Count ticket types
ticket_type_distribution = data['Ticket Type'].value_counts()
# Plot
plt.figure(figsize=(8, 6))
ticket_type_distribution.plot(kind='pie', autopct='%1.1f%%',
colors=['skyblue', 'salmon', 'lightgreen'])
plt.title('Ticket Type Distribution')
plt.ylabel('')
plt.show()
```



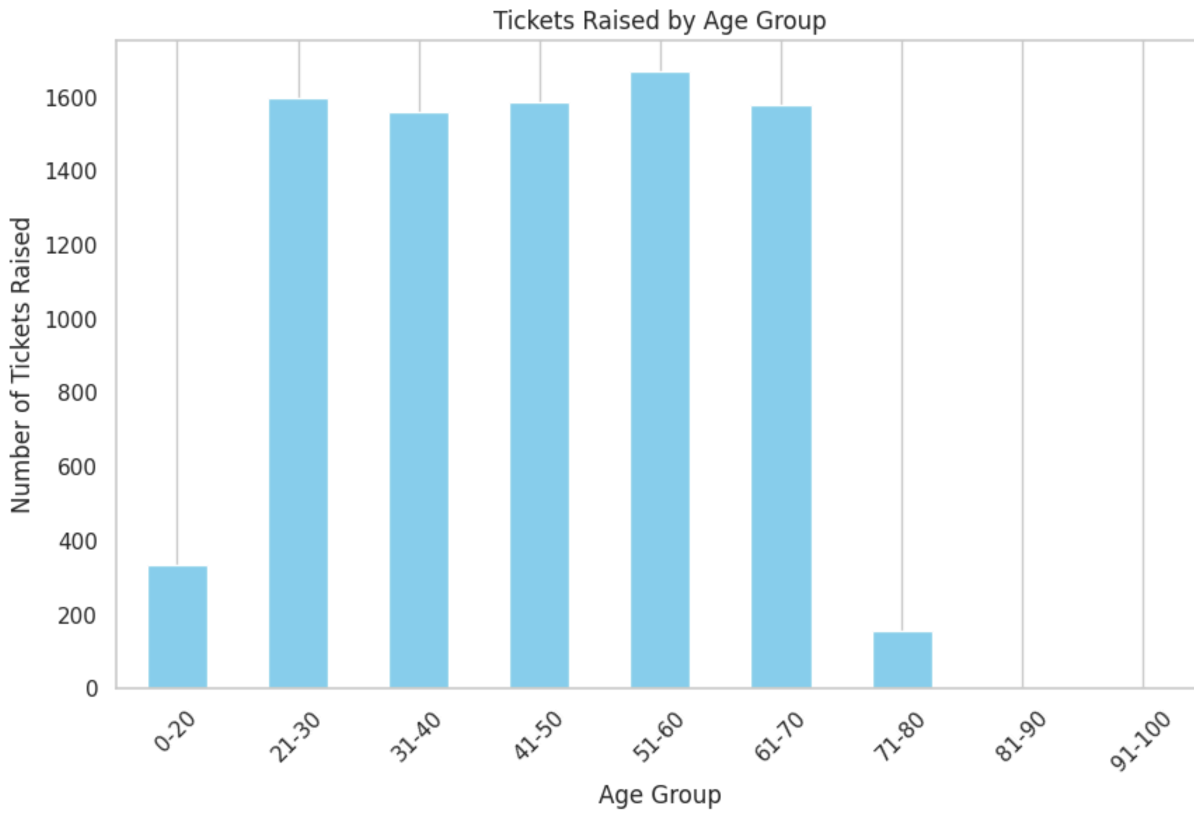
In [24]:
```
# Define age groups
bins = [0, 20, 30, 40, 50, 60, 70, 80, 90, 100]
labels = ['0-20', '21-30', '31-40', '41-50', '51-60', '61-70','71-80', '81-9
```

In [25]:
```
# Categorize customers into age groups
data['Age Group'] = pd.cut(data['Customer Age'], bins=bins,labels=labels, ri
```

Number of Tickets Raised by Age Group

In [26]:
```
# Calculate number of tickets raised by each age group
tickets_by_age_group = data.groupby('Age Group').size()
# Plot
plt.figure(figsize=(10, 6))
tickets_by_age_group.plot(kind='bar', color='skyblue')
plt.title('Tickets Raised by Age Group')
plt.xlabel('Age Group')
plt.ylabel('Number of Tickets Raised')
plt.xticks(rotation=45)
plt.grid(axis='y')
plt.show()
```

```
/tmp/ipython-input-1556793647.py:2: FutureWarning: The default of observed=F
alse is deprecated and will be changed to True in a future version of panda
s. Pass observed=False to retain current behavior or observed=True to adopt
the future default and silence this warning.
  tickets_by_age_group = data.groupby('Age Group').size()
```



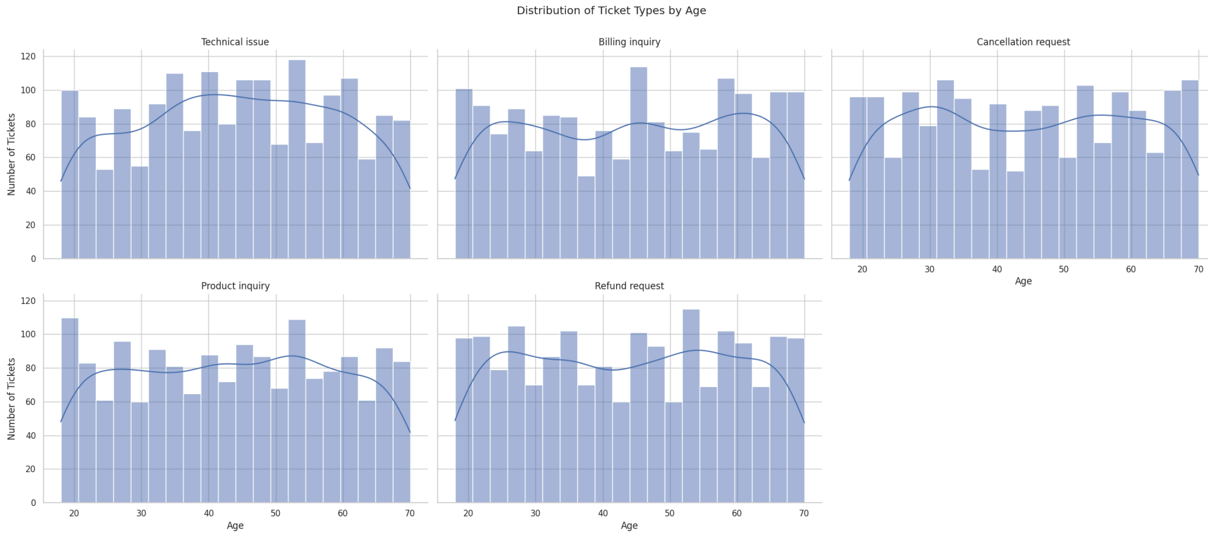
In [27]:

```
# Replace inf values with NaN
data.replace([np.inf, -np.inf], np.nan, inplace=True)
```

Age Distribution for Each Ticket Type

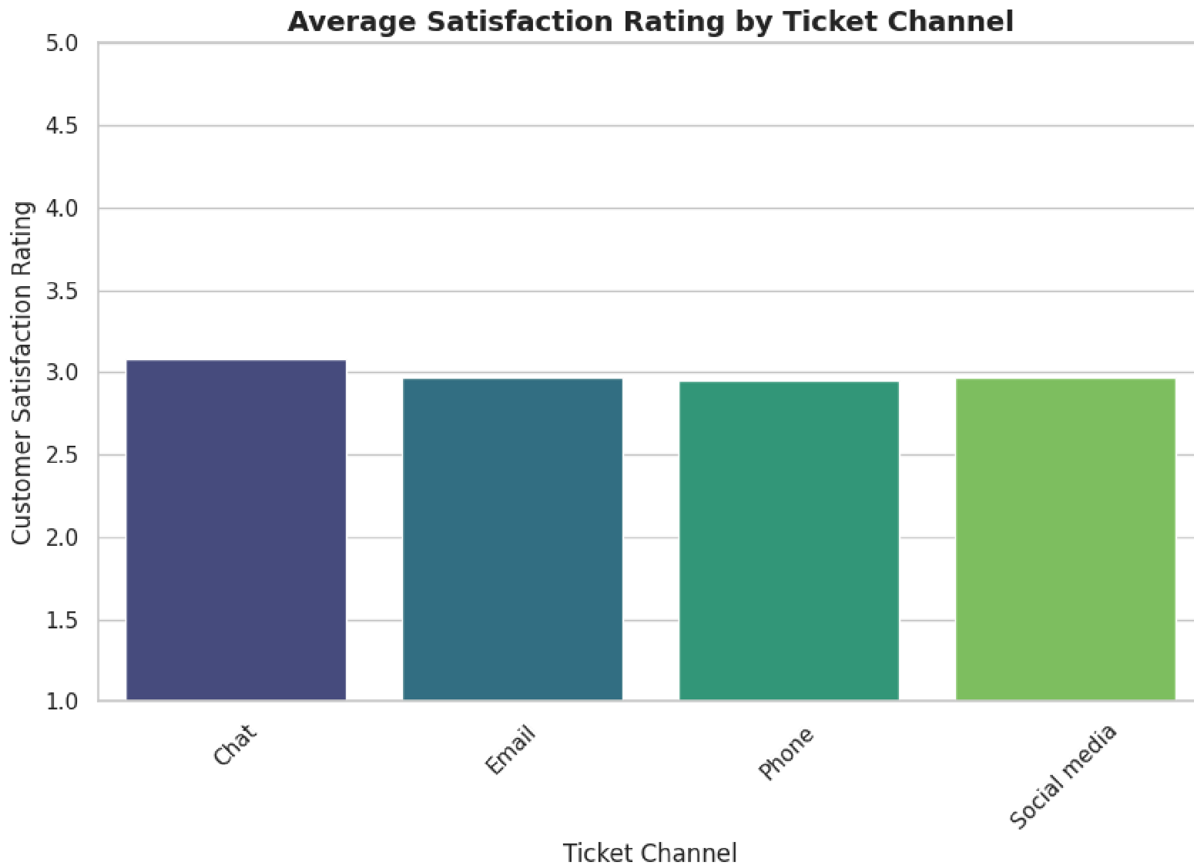
In [28]:

```
# Create a facet grid for each ticket type
g = sns.FacetGrid(data, col='Ticket Type', col_wrap=3,height=5, aspect=1.5)
g.map(sns.histplot, 'Customer Age', bins=20, kde=True)

# Set titles and labels
g.set_titles('{col_name}')
g.set_axis_labels('Age', 'Number of Tickets')

# Adjust layout
plt.subplots_adjust(top=0.9)
g.fig.suptitle('Distribution of Ticket Types by Age')
# Show plot
plt.show()
```



Average Satisfaction by Ticket Channel

In [29]:
```
plt.figure(figsize=(10,6))
avg_satisfaction_channel = data.groupby('Ticket Channel')['Customer Satisfac
sns.barplot(
    x='Ticket Channel',
    y='Customer Satisfaction Rating',
    hue='Ticket Channel',
    data=avg_satisfaction_channel,
    palette='viridis',
    legend=False
)
plt.title('Average Satisfaction Rating by Ticket Channel', fontsize=14, font
plt.ylim(1, 5)
plt.xticks(rotation=45)
plt.show()
```

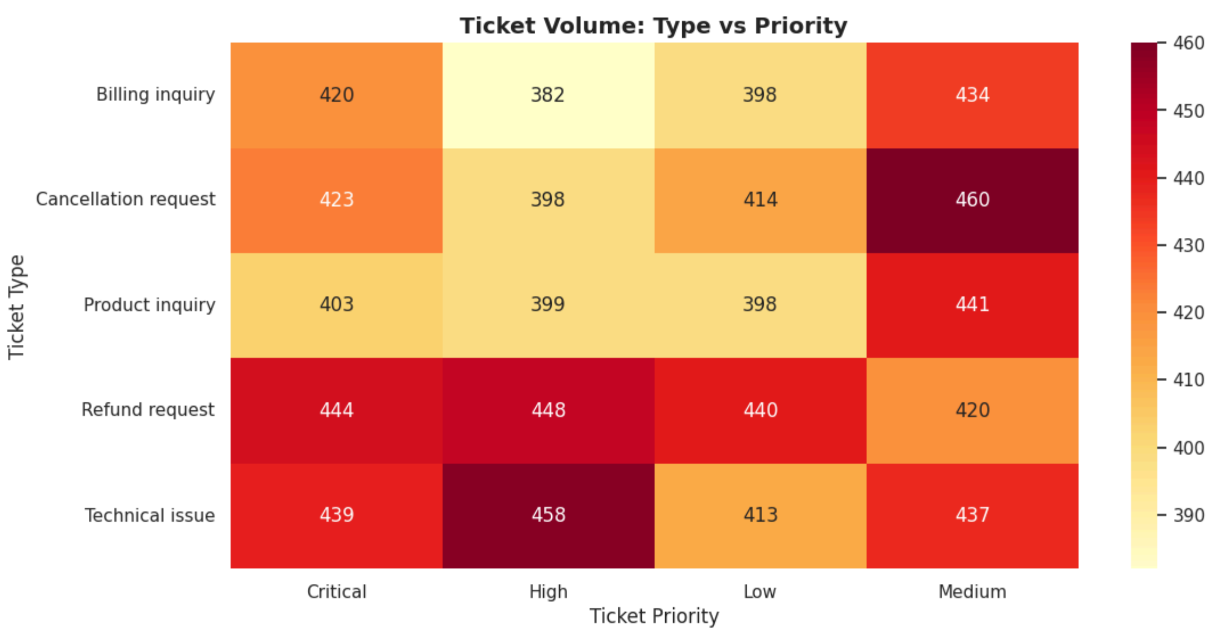


Resolution Time vs. Satisfaction

In [30]:
```
data['First Response Time'] = pd.to_datetime(data['First Response Time'], er
data['Time to Resolution'] = pd.to_datetime(data['Time to Resolution'], erro
data['Resolution Hours'] = (data['Time to Resolution'] - data['First Respons

plt.figure(figsize=(10,6))
sns.scatterplot(
    x='Resolution Hours',
    y='Customer Satisfaction Rating',
    data=data,
    alpha=0.5,
    color='purple'
)
plt.title('Resolution Time vs. Customer Satisfaction', fontsize=14, fontweig
plt.xlabel('Resolution Time (Hours)')
plt.ylabel('Satisfaction Rating')
plt.show()
```

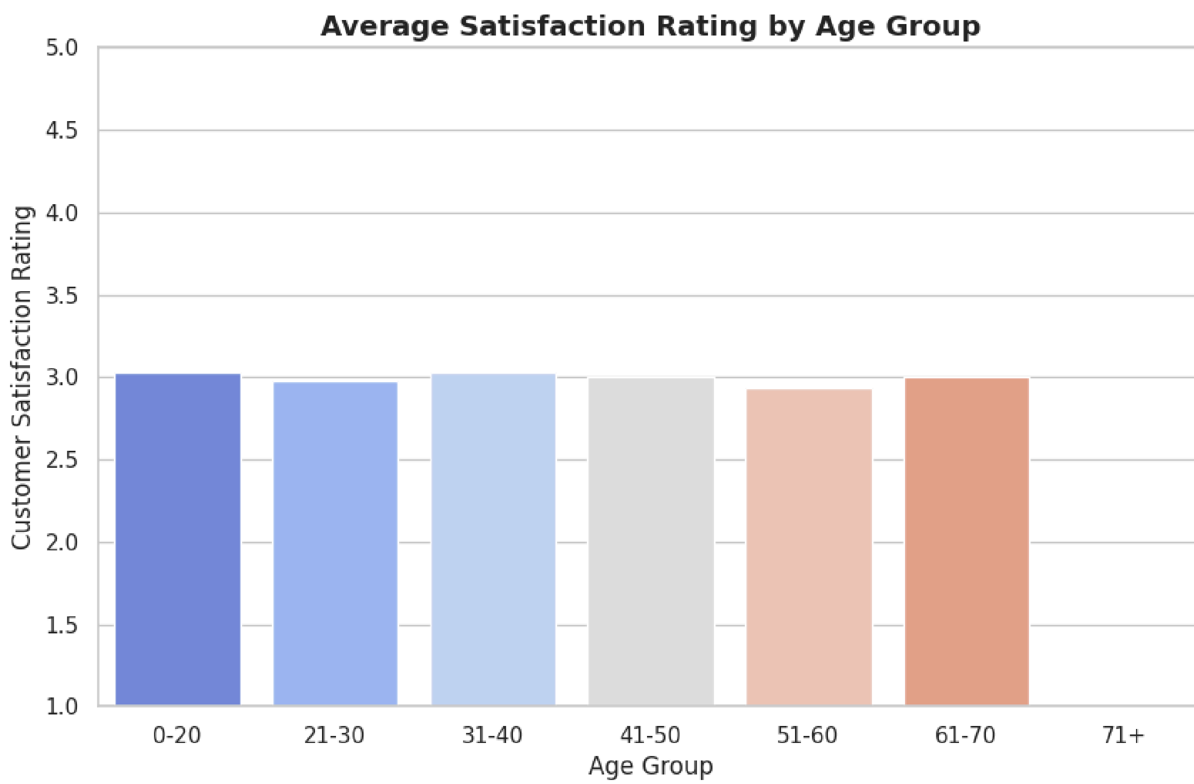
Heatmap: Ticket Type vs Ticket Priority

In [31]:
```
heatmap_data = data.groupby(['Ticket Type', 'Ticket Priority']).size().unsta

plt.figure(figsize=(12,6))
sns.heatmap(heatmap_data, annot=True, fmt='d', cmap='YlOrRd')
plt.title('Ticket Volume: Type vs Priority', fontsize=14, fontweight='bold')
plt.xlabel('Ticket Priority')
plt.ylabel('Ticket Type')
plt.show()
```



Average Satisfaction by Age Group

```
In [32]: bins = [0, 20, 30, 40, 50, 60, 70, 100]
labels = ['0-20','21-30','31-40','41-50','51-60','61-70','71+']
data['Age Group'] = pd.cut(data['Customer Age'], bins=bins, labels=labels)

# Explicitly set observed=False to silence the warning
avg_satisfaction_age = (
    data.groupby('Age Group', observed=False)['Customer Satisfaction Rating'
    .mean()
    .reset_index()
)

plt.figure(figsize=(10,6))
sns.barplot(
    x='Age Group',
    y='Customer Satisfaction Rating',
    hue='Age Group',    # same as x to apply palette
    data=avg_satisfaction_age,
    palette='coolwarm',
    legend=False
)
plt.title('Average Satisfaction Rating by Age Group', fontsize=14, fontweigh
plt.ylim(1, 5)
plt.show()
```



Ticket Volume by Month

```
In [33]: import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Ensure Date of Purchase is datetime
data['Date of Purchase'] = pd.to_datetime(data['Date of Purchase'], errors='

# Group by month period
monthly_tickets = data.groupby(data['Date of Purchase'].dt.to_period('M')).s

# Convert the period back to a timestamp for plotting
monthly_tickets['Date of Purchase'] = monthly_tickets['Date of Purchase'].dt

# Plot
plt.figure(figsize=(12, 6))
sns.lineplot(x='Date of Purchase', y='Count', data=monthly_tickets, marker='
plt.title('Monthly Ticket Volume Trend', fontsize=14, fontweight='bold')
plt.xlabel('Month')
plt.ylabel('Number of Tickets')
plt.grid(True, linestyle='--', alpha=0.6)
plt.show()
```