



```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
# Load the dataset
flight_df=pd.read_csv('/content/Clean_Dataset (6).csv')
```

flight_df.head()




	Unnamed: 0	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
0	0	SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
1	1	SpiceJet	SG-8157	Delhi	Early_Morning	zero	Morning	Mumbai	Economy	2.33	1	5953
2	2	AirAsia	I5-764	Delhi	Early_Morning	zero	Early_Morning	Mumbai	Economy	2.17	1	5956
3	3	Vistara	UK-995	Delhi	Morning	zero	Afternoon	Mumbai	Economy	2.25	1	5955
4	4	Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955

flight_df.tail()



	Unnamed: 0	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
300148	300148	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	69265
300149	300149	Vistara	UK-826	Chennai	Afternoon	one	Night	Hyderabad	Business	10.42	49	77105
300150	300150	Vistara	UK-832	Chennai	Early_Morning	one	Night	Hyderabad	Business	13.83	49	79099
300151	300151	Vistara	UK-828	Chennai	Early_Morning	one	Evening	Hyderabad	Business	10.00	49	81585
300152	300152	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	81585


```
# Print first 5 rows of data
print("First 5 rows of data:")
print(flight_df.head())
```



First 5 rows of data:

	Unnamed: 0	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
0	0	SpiceJet	SG-8709	Delhi	Evening	zero	Night	Mumbai	Economy	2.17	1	5953
1	1	SpiceJet	SG-8157	Delhi	Early_Morning	zero	Morning	Mumbai	Economy	2.33	1	5953
2	2	AirAsia	I5-764	Delhi	Early_Morning	zero	Early_Morning	Mumbai	Economy	2.17	1	5956
3	3	Vistara	UK-995	Delhi	Morning	zero	Afternoon	Mumbai	Economy	2.25	1	5955
4	4	Vistara	UK-963	Delhi	Morning	zero	Morning	Mumbai	Economy	2.33	1	5955

```
# Print last 5 rows of data
print("\nLast 5 rows of data:")
print(flight_df.tail())
```




Last 5 rows of data:

	Unnamed: 0	airline	flight	source_city	departure_time	stops	arrival_time	destination_city	class	duration	days_left	price
300148	300148	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	69265
300149	300149	Vistara	UK-826	Chennai	Afternoon	one	Night	Hyderabad	Business	10.42	49	77105
300150	300150	Vistara	UK-832	Chennai	Early_Morning	one	Night	Hyderabad	Business	13.83	49	79099
300151	300151	Vistara	UK-828	Chennai	Early_Morning	one	Evening	Hyderabad	Business	10.00	49	81585
300152	300152	Vistara	UK-822	Chennai	Morning	one	Evening	Hyderabad	Business	10.08	49	81585

```
# Cleaning the data for missing values, null values etc.
flight_df.dropna(inplace=True)
```

```
# Remove IN-DATA index column if it exists
if 'index' in flight_df.columns:
    flight_df.drop(columns=['index'], inplace=True)
```

```
# Get some info about the data
print("\nInfo about the data:")
print(flight_df.info())
```



Info about the data:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 300153 entries, 0 to 300152

Data columns (total 12 columns):

#	Column	Non-Null Count	Dtype
0	Unnamed: 0	300153 non-null	int64
1	airline	300153 non-null	object
2	flight	300153 non-null	object
3	source_city	300153 non-null	object
4	departure_time	300153 non-null	object
5	stops	300153 non-null	object
6	arrival_time	300153 non-null	object
7	destination_city	300153 non-null	object
8	class	300153 non-null	object
9	duration	300153 non-null	float64

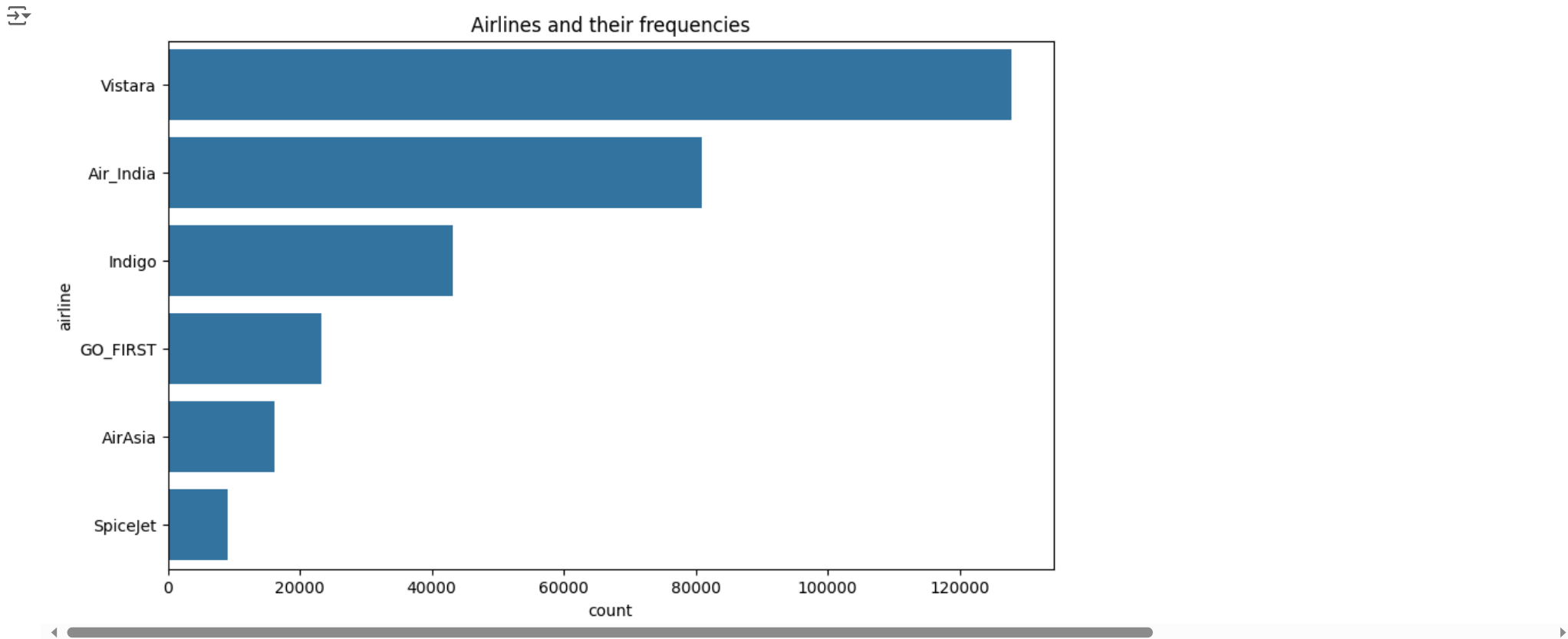
```
10  days_left      300153 non-null  int64
11  price          300153 non-null  int64
dtypes: float64(1), int64(3), object(8)
memory usage: 27.5+ MB
None
```

```
# Get some description about the data
print("\nDescription of the data:")
print(flight_df.describe())
```

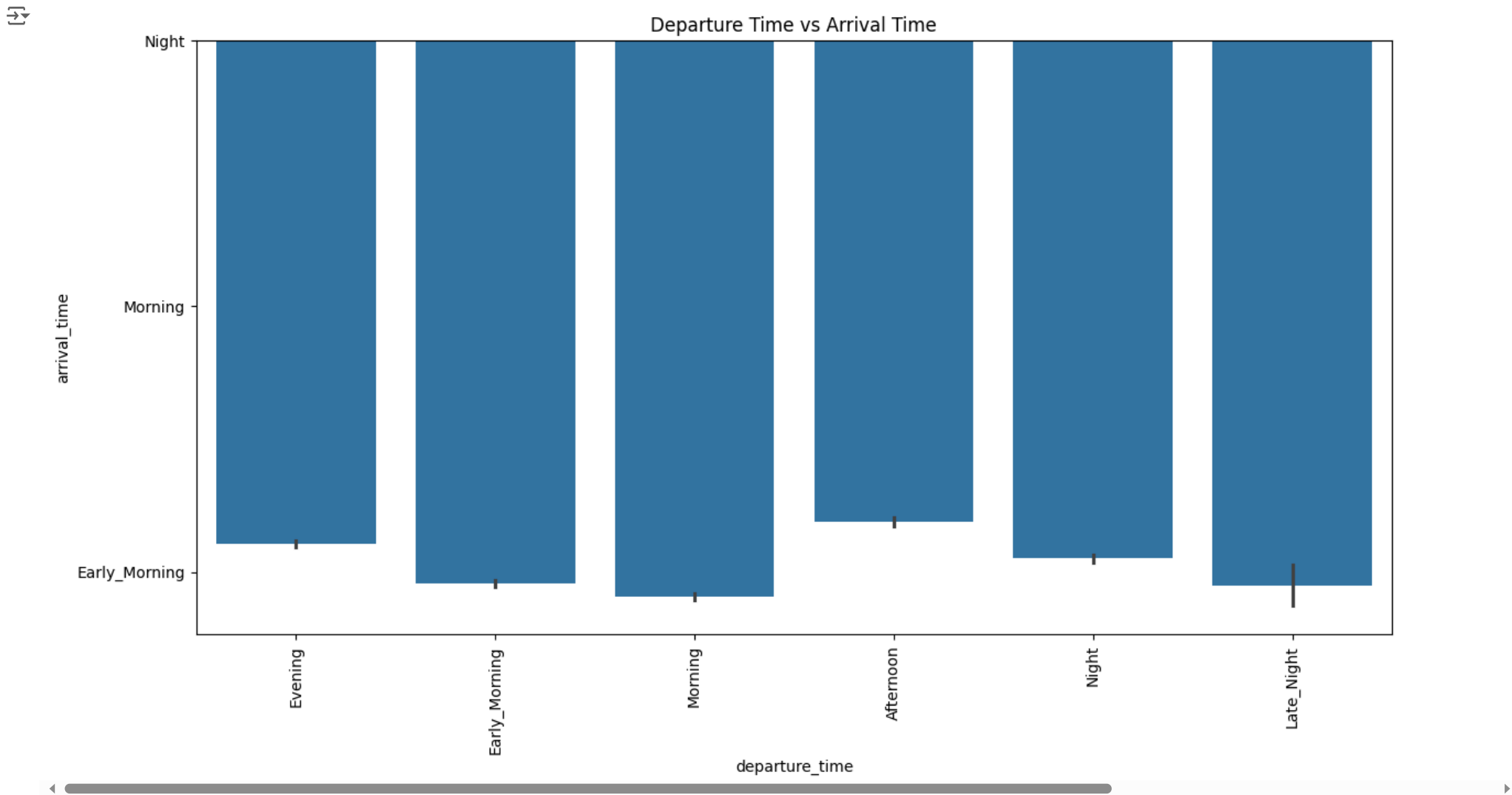
↻

Description of the data:				
	Unnamed: 0	duration	days_left	price
count	300153.000000	300153.000000	300153.000000	300153.000000
mean	150076.000000	12.221021	26.004751	20889.660523
std	86646.852011	7.191997	13.561004	22697.767366
min	0.000000	0.830000	1.000000	1105.000000
25%	75038.000000	6.830000	15.000000	4783.000000
50%	150076.000000	11.250000	26.000000	7425.000000
75%	225114.000000	16.170000	38.000000	42521.000000
max	300152.000000	49.830000	49.000000	123071.000000

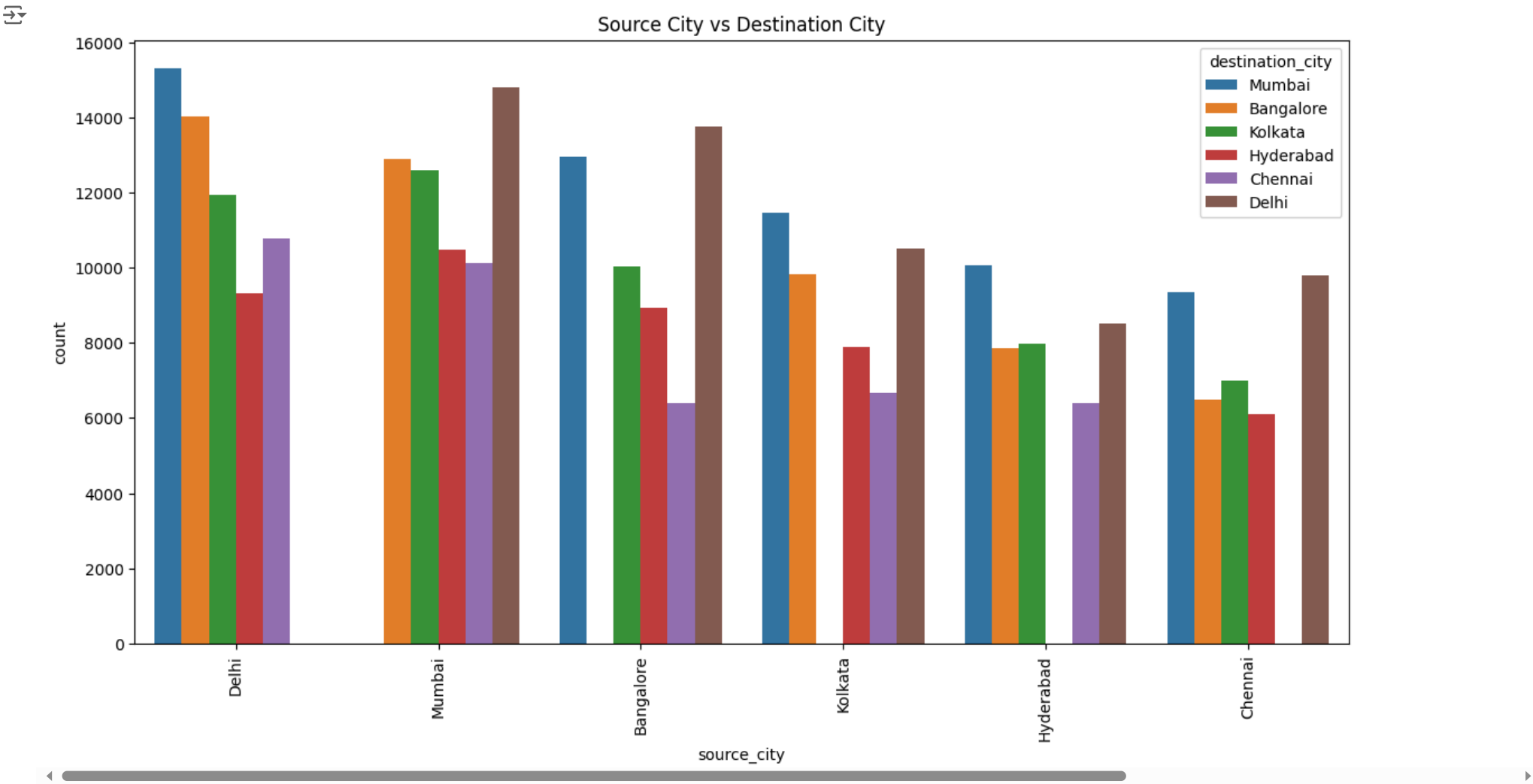
```
# Visualization 1: Airlines and their frequencies
plt.figure(figsize=(10, 6))
sns.countplot(y='airline', data=flight_df, order=flight_df['airline'].value_counts().index)
plt.title('Airlines and their frequencies')
plt.show()
```



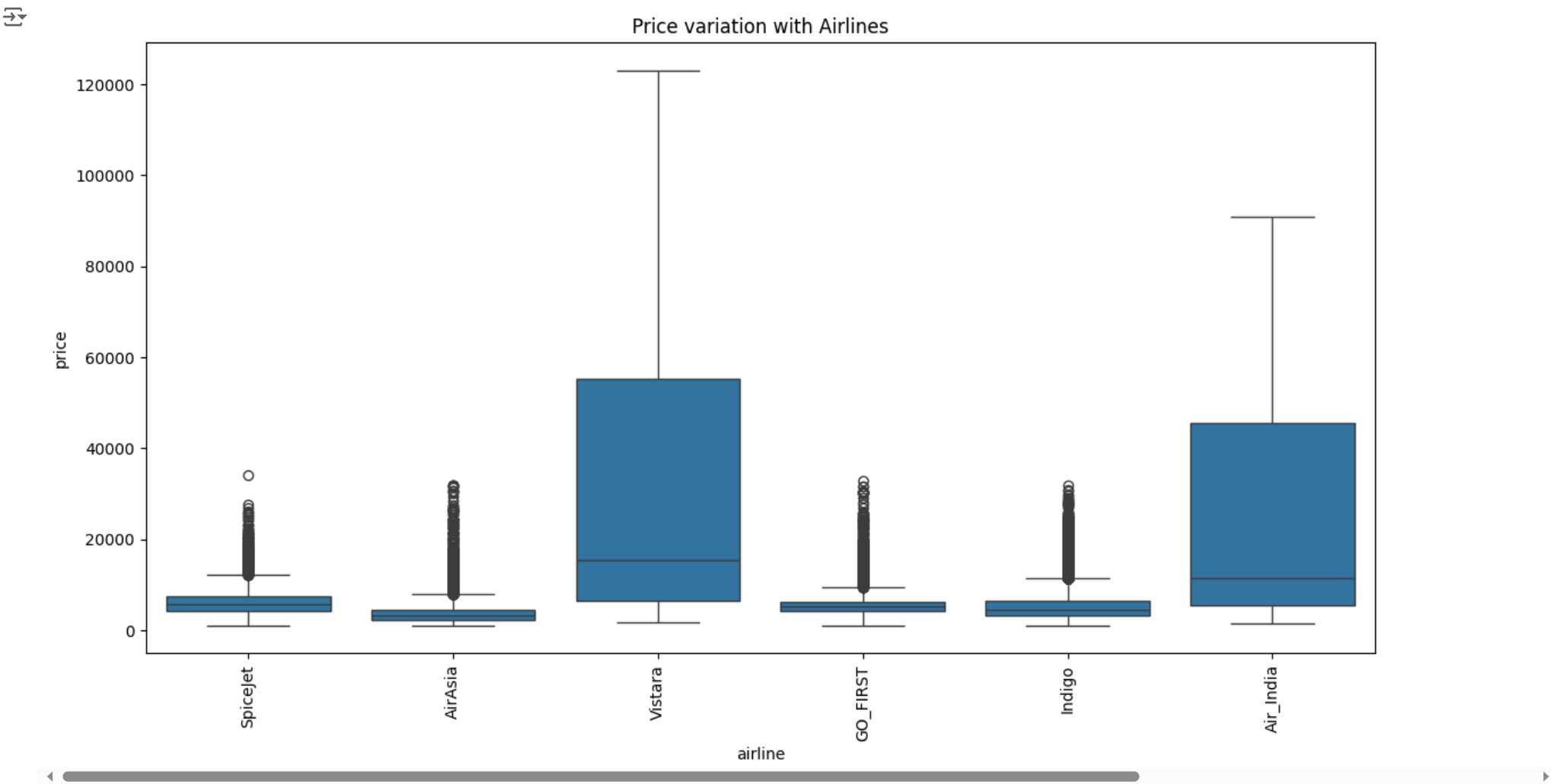
```
# Visualization 2: Departure time against Arrival time using barplot
plt.figure(figsize=(14, 7))
sns.barplot(x='departure_time', y='arrival_time', data=flight_df)
plt.title('Departure Time vs Arrival Time')
plt.xticks(rotation=90)
plt.show()
```



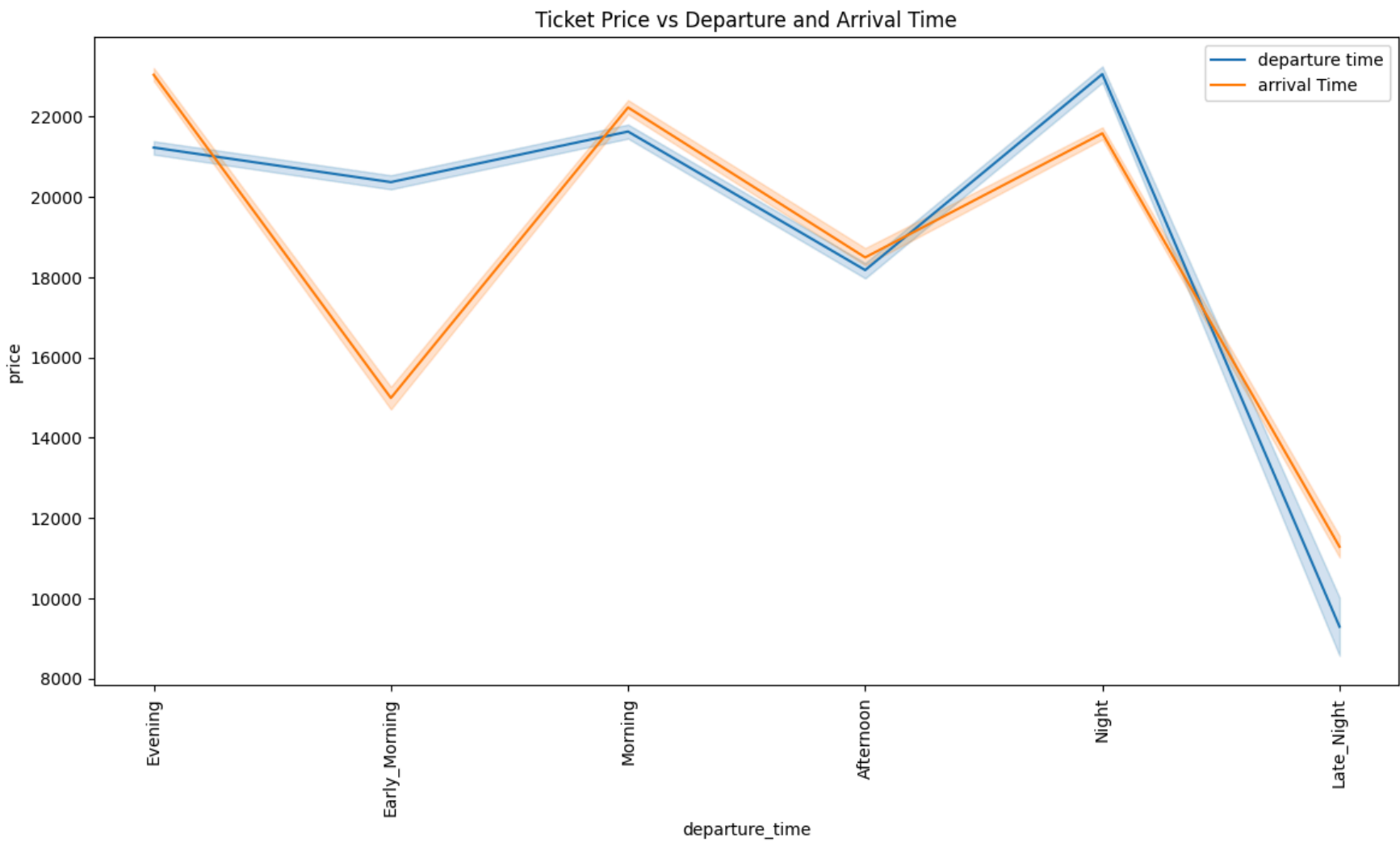
```
# Visualization 3: Source city against Destination city
plt.figure(figsize=(14, 7))
sns.countplot(x='source_city', hue='destination_city', data=flight_df)
plt.title('Source City vs Destination City')
plt.xticks(rotation=90)
plt.show()
```



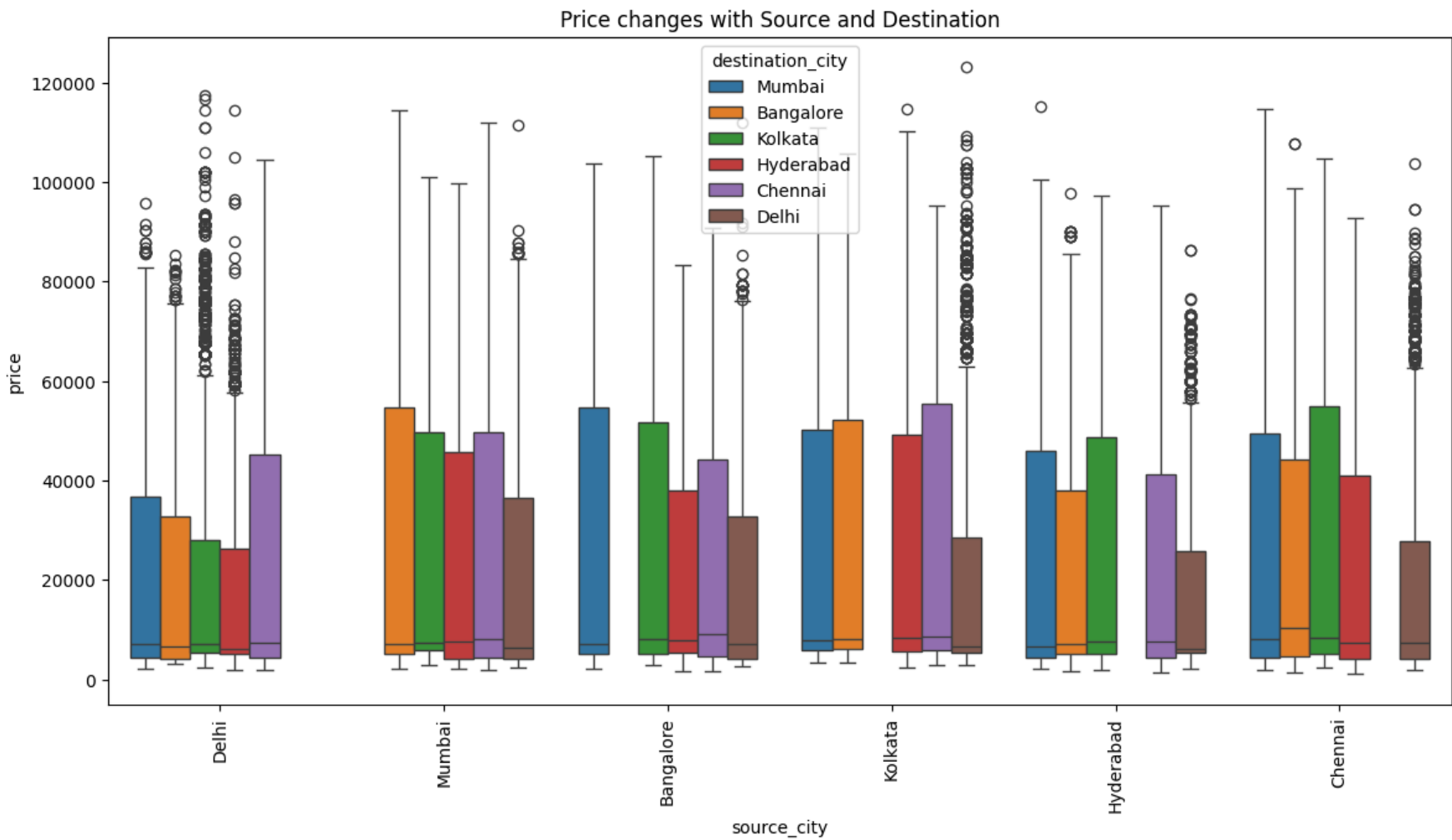
```
# Visualization 4: Does price vary with Airlines?
plt.figure(figsize=(14, 7))
sns.boxplot(x='airline', y='price', data=flight_df)
plt.title('Price variation with Airlines')
plt.xticks(rotation=90)
plt.show()
```



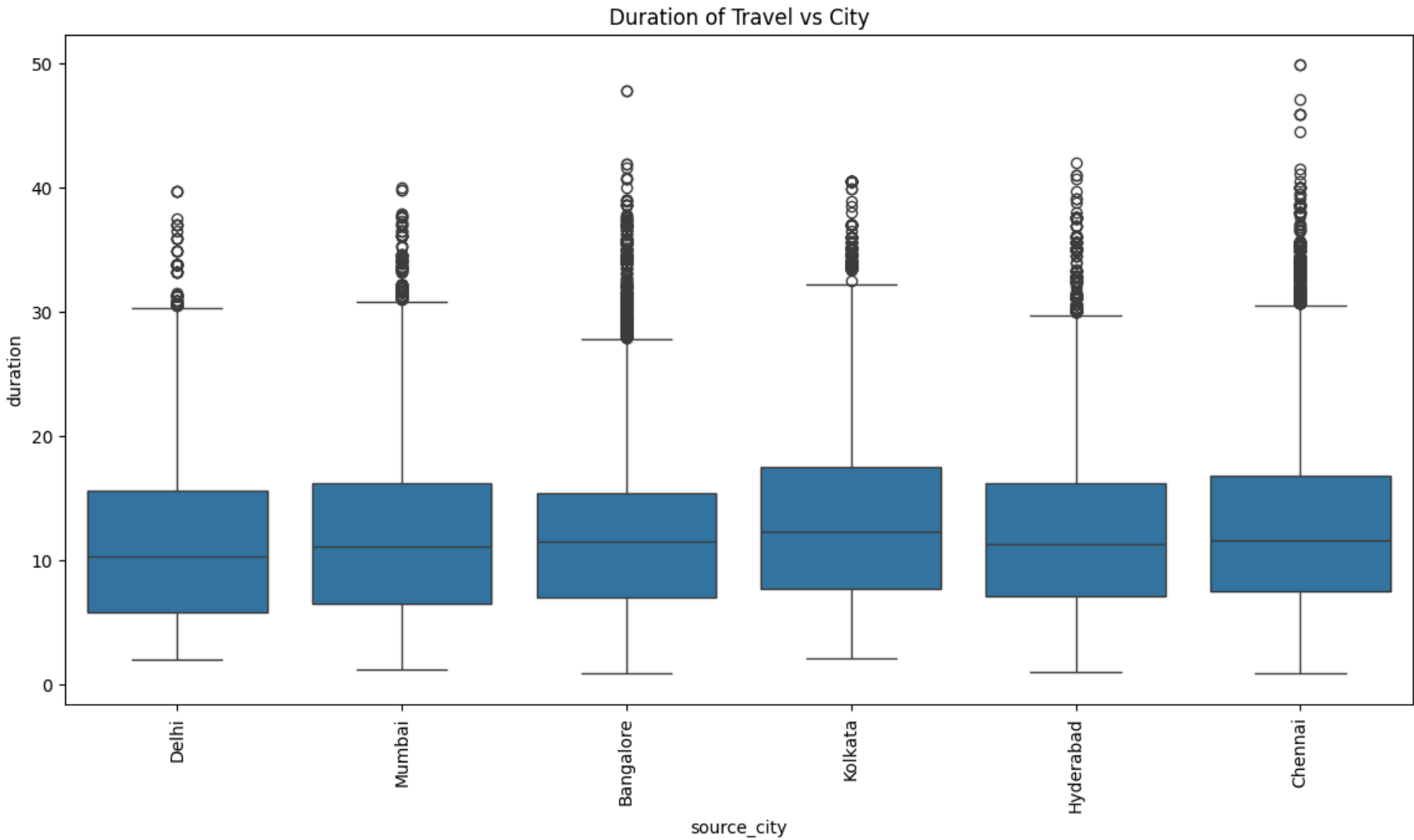
```
# Visualization 5: Ticket price change based on departure and arrival time using line plot
plt.figure(figsize=(14, 7))
sns.lineplot(x='departure_time', y='price', data=flight_df, label='departure time')
sns.lineplot(x='arrival_time', y='price', data=flight_df, label='arrival Time')
plt.title('Ticket Price vs Departure and Arrival Time')
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



```
# Visualization 6: Price changes with Source and Destination
plt.figure(figsize=(14, 7))
sns.boxplot(x='source_city', y='price', hue='destination_city', data=flight_df)
plt.title('Price changes with Source and Destination')
plt.xticks(rotation=90)
plt.show()
```



```
# Visualization 7: Duration of travel vs city
plt.figure(figsize=(14, 7))
sns.boxplot(x='source_city', y='duration', data=flight_df)
plt.title('Duration of Travel vs City')
plt.xticks(rotation=90)
plt.show()
```



```
# Visualization 8: High price with class type for city
plt.figure(figsize=(14, 7))
sns.boxplot(x='source_city', y='price', hue='class', data=flight_df)
plt.title('High Price with Class Type for City')
plt.xticks(rotation=90)
plt.show()
```

