In [13]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load the data
file_path = "C:/Users/NTC/Downloads/Data Tasks (1)/tasks/task_2/corrupted_file.csv"
data = pd.read_csv(file_path)

# Display the first few rows of the data
data.head()
```

Out[13]:

	ad_id	reporting_start	reporting_end	campaign_id	fb_campaign_id	age	gender	interest1	interest2	inte
0	708746	17/08/2017	17/08/2017	916	103916	30-34	M	15	17	
1	708749	17/08/2017	17/08/2017	916	103917	30-34	M	16	19	
2	708771	17/08/2017	17/08/2017	916	103920	30-34	M	20	25	
3	708815	30/08/2017	30/08/2017	916	103928	30-34	M	28	32	
4	708818	17/08/2017	17/08/2017	916	103928	30-34	M	28	33	

In [14]:
```
data.tail()
```

Out[14]:

	ad_id	reporting_start	reporting_end	campaign_id	fb_campaign_id	age	gender	interest1	interest2	int
94	735032	19/08/2017	19/08/2017	936	108791	35-39	F	32	33	
95	735033	18/08/2017	18/08/2017	936	108792	35-39	F	36	40	
96	735043	18/08/2017	18/08/2017	936	108793	35-39	F	63	65	
97	735048	18/08/2017	18/08/2017	936	108794	35-39	F	64	65	
98	735065	19/08/2017	19/08/2017	936	108797	40-44	F	7	10	

In [15]:
```
data.describe()
```

Out[15]:

	ad_id	campaign_id	fb_campaign_id	interest1	interest2	interest3	impressions	clic
count	99.000000	99.000000	99.000000	99.000000	99.000000	99.000000	99.000000	99.0000
mean	721147.818182	925.090909	106207.818182	24.161616	27.696970	81.515152	9041.848485	2.1061
std	12418.705190	10.009272	2316.736248	13.137194	13.194351	536.999893	14714.985879	3.5544
min	708746.000000	916.000000	103916.000000	7.000000	8.000000	8.000000	1.000000	0.0000
25%	709391.500000	916.000000	104023.500000	16.000000	19.000000	20.000000	1375.000000	0.0000
50%	711623.000000	916.000000	104396.000000	20.000000	25.000000	24.000000	4259.000000	1.0000
75%	734663.000000	936.000000	108729.500000	29.000000	32.500000	32.000000	11501.000000	3.0000
max	735065.000000	936.000000	108797.000000	65.000000	70.000000	5369.000000	104648.000000	24.0000

In [16]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 15 columns):
 #   Column               Non-Null Count  Dtype  
---  ------               --------------  -----  
 0   ad_id                99 non-null     int64  
 1   reporting_start      99 non-null     object 
 2   reporting_end        99 non-null     object 
 3   campaign_id          99 non-null     int64  
 4   fb_campaign_id       99 non-null     int64  
 5   age                  99 non-null     object 
 6   gender               99 non-null     object 
 7   interest1            99 non-null     int64  
 8   interest2            99 non-null     int64  
 9   interest3            99 non-null     int64  
 10  impressions          99 non-null     int64  
 11  clicks               99 non-null     float64
 12  spent                99 non-null     float64
 13  total_conversion     99 non-null     int64  
 14  approved_conversion  95 non-null     float64
dtypes: float64(3), int64(8), object(4)
memory usage: 11.7+ KB
```

In [17]:
```
data.isna().sum()
```

Out[17]:
```
ad_id                  0
reporting_start        0
reporting_end          0
campaign_id            0
fb_campaign_id         0
age                    0
gender                 0
interest1              0
interest2              0
interest3              0
impressions            0
clicks                 0
spent                  0
total_conversion       0
approved_conversion    4
dtype: int64
```

In [18]:
```
data.isnull().mean()*100
```

Out[18]:
```
ad_id                  0.000000
reporting_start        0.000000
reporting_end          0.000000
campaign_id            0.000000
fb_campaign_id         0.000000
age                    0.000000
gender                 0.000000
interest1              0.000000
interest2              0.000000
interest3              0.000000
impressions            0.000000
clicks                 0.000000
spent                  0.000000
total_conversion       0.000000
approved_conversion    4.040404
dtype: float64
```

In [19]:
```
data.describe().transpose()
```

Out[19]:

	count	mean	std	min	25%	50%	75%	max
ad_id	99.0	721147.818182	12418.705190	708746.0	709391.5	711623.00	734663.00	735065.00

campaign_id	99.0	925.090909	10.009272	916.0	916.0	916.00	936.00	936.00
fb_campaign_id	99.0	106207.818182	2316.736248	103916.0	104023.5	104396.00	108729.50	108797.00
interest1	99.0	24.161616	13.137194	7.0	16.0	20.00	29.00	65.00
interest2	99.0	27.696970	13.194351	8.0	19.0	25.00	32.50	70.00
interest3	99.0	81.515152	536.999893	8.0	20.0	24.00	32.00	5369.00
impressions	99.0	9041.848485	14714.985879	1.0	1375.0	4259.00	11501.00	104648.00
clicks	99.0	2.106162	3.554423	0.0	0.0	1.00	3.00	24.00
spent	99.0	2.839192	4.928773	0.0	0.0	1.29	3.59	33.33
total_conversion	99.0	1.121212	0.539704	0.0	1.0	1.00	1.00	4.00
approved_conversion	95.0	0.431579	0.497924	0.0	0.0	0.00	1.00	1.00

In [20]:
```
# Identify rows with any missing data
corrupted_rows = data.isnull().any(axis=1)

# Count the number of corrupted rows
num_corrupted_rows = corrupted_rows.sum()

num_corrupted_rows
```

Out[20]:
```
4
```

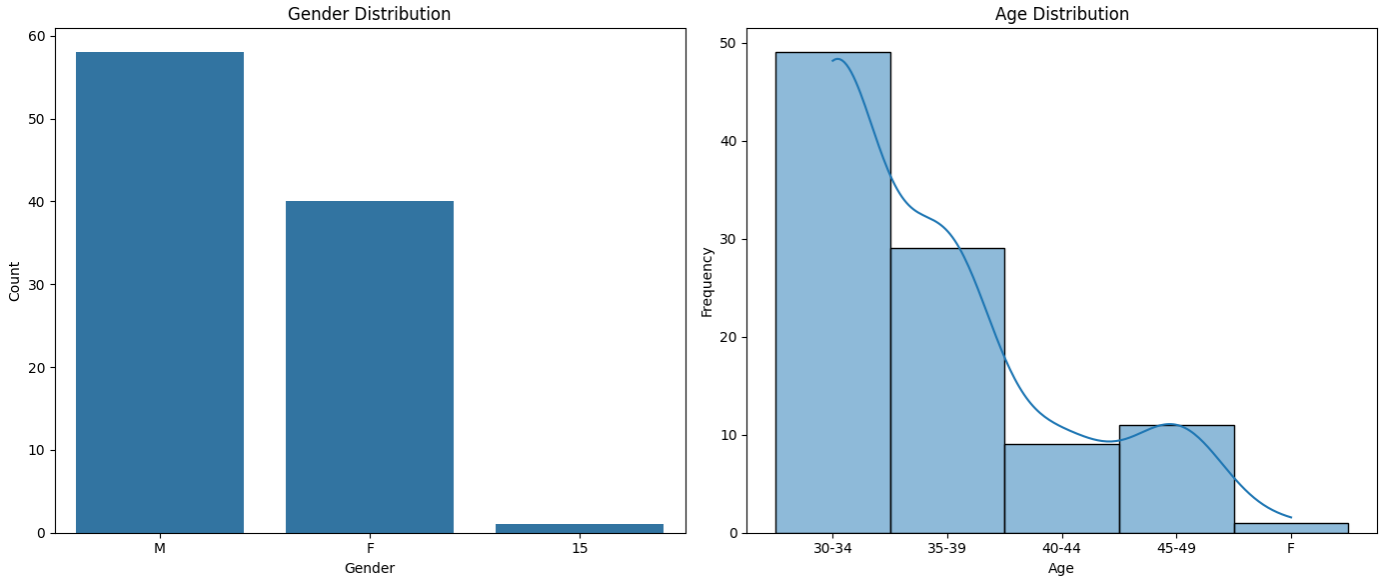
The code identified that there are 4 corrupted records in the dataset in the Approved coversations column

In [21]:
```
# Initialize a dictionary to store row indices and their corresponding corrupted columns
corrupted_columns = {}

# Iterate over the DataFrame to find which columns have missing values in corrupted rows
for idx in data[corrupted_rows].index:
    # Identify the columns with missing values in the current row
    missing_cols = data.columns[data.loc[idx].isnull()].tolist()
    # Store the row index and the corresponding corrupted columns
    corrupted_columns[idx] = missing_cols

corrupted_columns
```

Out[21]:
```
{6: ['approved_conversion'],
 42: ['approved_conversion'],
 83: ['approved_conversion'],
 90: ['approved_conversion']}
```

In [23]:
```
# Step 1: Identify corrupted records in 'gender' and 'age' columns
gender_age_corrupted_rows = data[data[['gender', 'age']].isnull().any(axis=1)]
num_gender_age_corrupted_rows = gender_age_corrupted_rows.shape[0]

# Step 2: Plot the distribution of 'gender' and 'age'
plt.figure(figsize=(14, 6))

# Plot distribution of 'gender'
plt.subplot(1, 2, 1)
sns.countplot(data=data, x='gender')
plt.title('Gender Distribution')
plt.xlabel('Gender')
plt.ylabel('Count')

# Plot distribution of 'age'
```

```
plt.subplot(1, 2, 2)
sns.histplot(data=data, x='age', bins=20, kde=True)
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')

plt.tight_layout()
plt.show()

num_gender_age_corrupted_rows
```



Out[23]: 0

The visualizations display the distribution of 'gender' and 'age' in the dataset. Additionally, the analysis reveals that thereno missing values in these columns in the 'gender' and 'age' columns but

The visualizations display the distribution of 'gender' and 'age' in the dataset, revealing incorrect categorical data in both columns: specifically, the values '15' and 'F'.