Task 1: Data Exploration

1. Load the dataset into your preferred data analysis tool (e.g., Python, R, Excel).
2. Check for missing values in the dataset and handle them appropriately.
3. Provide summary statistics for key numeric columns (e.g., Charge, TalkTime).
4. Visualize the distribution of call charges using a histogram. Task 2: Call Center Performance Metrics
5. Calculate the average TalkTime for different call activities (Activity column).
6. Determine the most common source of calls (Source column)
7. Calculate the total earnings and spending for both users and masters.
8. Analyze the relationship between TalkTime and Charge.

Task 3: Call Handling Analysis

9. Calculate the average time it takes for calls to be connected (ConnectTime - DialTime).
10. Identify the most common reason for call disconnection (UnconnectTime - ConnectTime).
11. Analyze the HangUpTime patterns and identify any trends.

Task 4: Order and Refund Analysis

12. Determine the order status distribution for calls.
13. Calculate the total refund amount and identify the refund status distribution.

Task 5: Insights and Recommendations

14. Based on your analysis, provide insights into call center performance and any actionable recommendations for improving call handling, user satisfaction, or earnings.

Task 6: Data Visualization

7-aAdditional Visualizations ● Create additional visualizations to showcase your data visualization skills. ● Please create a line chart showing the trend in call charges over time. ● Create a scatter plot to analyze the relationship between TalkTime and UserSpend.

In [77]:

```
import pandas as pd

# Load the dataset
file_path = '1Dec to 3Jan Consultations.csv'
data = pd.read_csv(file_path)

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

```
C:\Users\Aravi\AppData\Local\Temp\ipykernel_16684\3957137881.py:5: DtypeWarning: Columns (36,43,44) have mixed type
s. Specify dtype option on import or set low_memory=False.
  data = pd.read_csv(file_path)
```

Out[77]:

	_id	user	chatStatus	guru	guruName	gid	uid	consultationType	website
0	656922f2919324acb6e87139	65691b0c919324acb6e7e046	incomplete	65054786f5f203225bfcdd0d	Astro Sanjeev	95	30129	Chat	gurucool
1	65692413919324acb6e87441	65691b0c919324acb6e7e046	incomplete	65054786f5f203225bfcdd0d	Astro Sanjeev	95	30129	Chat	gurucool
2	65692612919324acb6e87738	656528dcee2b1ab52f019119	incomplete	65054786f5f203225bfcdd0d	Astro Sanjeev	95	28743	Chat	gurucool
3	656927c9919324acb6e87b2f	6568ee9f919324acb6e77e53	incomplete	65054786f5f203225bfcdd0d	Astro Sanjeev	95	30120	Chat	gurucool
4	65692d75919324acb6e882dd	65629cfc4232eb704d712d98	failed	65054786f5f203225bfcdd0d	Astro Sanjeev	95	27790	Chat	gurucool

5 rows × 45 columns

In [81]:

```
# Summary statistics for key numeric columns
print('Task 1: Data Exploration')
print('Summary Statistics')
key_numeric_columns = ['amount', 'timeDuration', 'callSeconds']
summary_statistics = data[key_numeric_columns].describe()
summary_statistics
```

```
Task 1: Data Exploration
Summary Statistics
```

Out[81]:

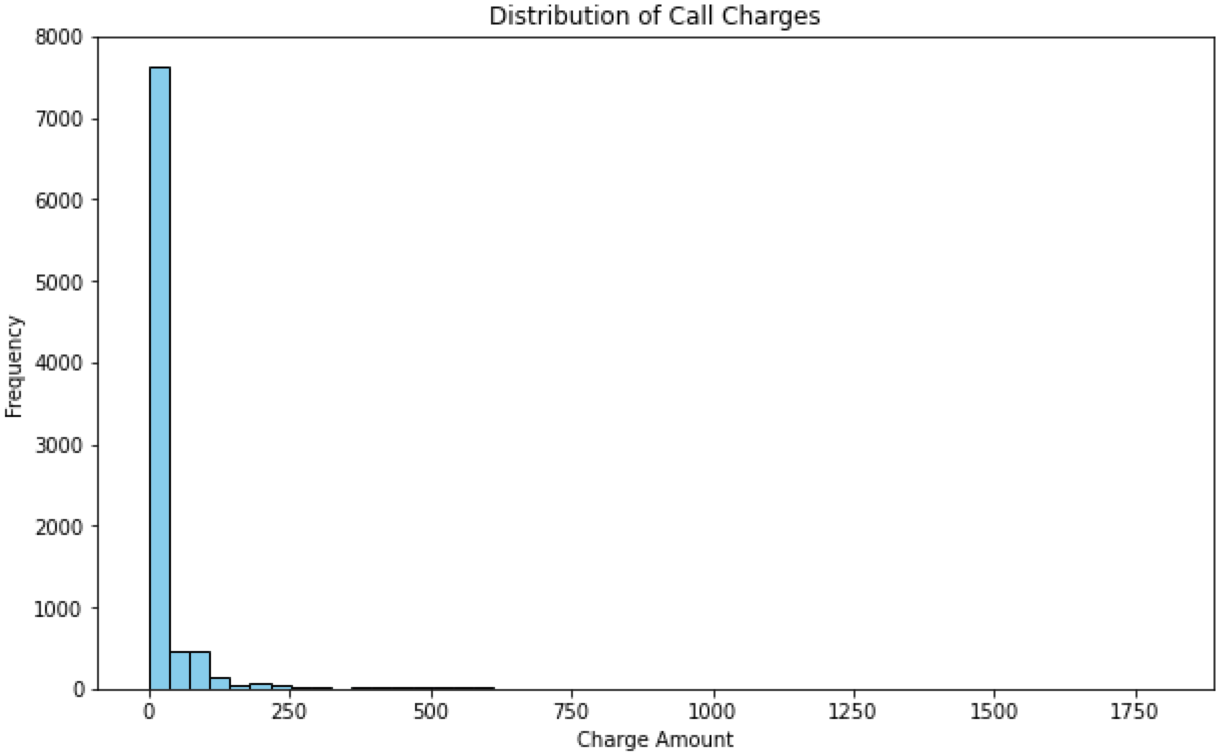
	amount	timeDuration	callSeconds
count	9016.000000	21398.000000	2.000000
mean	23.742890	1.977168	33000.000000
std	87.134998	3.340467	38183.766184
min	0.000000	-0.166667	6000.000000
25%	0.000000	0.000000	19500.000000
50%	0.000000	0.100000	33000.000000
75%	0.000000	4.356962	46500.000000
max	1800.000000	67.000000	60000.000000

In [82]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
file_path = '1Dec to 3Jan Consultations.csv'
data = pd.read_csv(file_path)

# Display the column names for debugging
print('Column Names')
print(data.columns)

# Title and Introduction
print('Call Center Performance Analysis')
print('Analysis of call center data from 1 Dec to 3 Jan.')


print('Histogram of Call Charges')
plt.figure(figsize=(10, 6))
plt.hist(data['amount'].dropna(), bins=50, color='skyblue', edgecolor='black')
plt.title('Distribution of Call Charges')
plt.xlabel('Charge Amount')
plt.ylabel('Frequency')
plt.show()
```

```
C:\Users\Aravi\AppData\Local\Temp\ipykernel_16684\662060699.py:6: DtypeWarning: Columns (36,43,44) have mixed types.
Specify dtype option on import or set low_memory=False.
  data = pd.read_csv(file_path)
```

```
Column Names
Index(['_id', 'user', 'chatStatus', 'guru', 'guruName', 'gid', 'uid',
       'consultationType', 'website', 'refundStatus', 'isWhiteListUser',
       'chatSeconds', 'queue', 'freeCall', 'freeChat', 'createdAt',
       'updatedAt', '__v', 'statementEntryId', 'chatStartTime', 'chatEndTime',
       'timeDuration', 'callChannel', 'callIvrType', 'callStatus', 'CallSid',
       'amount', 'astrologerCallStatus', 'astrologerOnCallDuration',
       'astrologersEarnings', 'netAmount', 'region', 'userCallStatus',
       'userOnCallDuration', 'RecordingUrl', 'feedback', 'hideHistory',
       'cardPosition', 'Remedies', 'offer', 'refunds[0]', 'source',
       'callSeconds', 'complementaryCallOrganiser', 'disconnectedBy'],
      dtype='object')
Call Center Performance Analysis
Analysis of call center data from 1 Dec to 3 Jan.
Histogram of Call Charges
```



In [43]:

```
# Task 2: Call Center Performance Metrics
print('## Task 2: Call Center Performance Metrics')
print('### Average TalkTime for Different Call Activities')
average_talktime_by_activity = data.groupby('consultationType')['timeDuration'].mean().reset_index()
average_talktime_by_activity.columns = ['Activity', 'AverageTalkTime']
print(average_talktime_by_activity)

print('Most Common Source of Calls')
most_common_source = data['source'].value_counts().idxmax()
print(f'The most common source of calls is: {most_common_source}')

print('Total Earnings and Spending')
total_earnings_users = data['amount'].sum()
total_spending_masters = data['astrologersEarnings'].sum()
print(f'Total earnings for users: {total_earnings_users}')
print(f'Total spending for masters: {total_spending_masters}')

# highest_earning_user = data.loc[data['astrologersEarnings'].idxmax()]
# print(f'highest_earning_user: {highest_earning_user}')
highest_earning_user = data.loc[data['astrologersEarnings'].idxmax(), ['gid', 'guruName', 'amount', 'consultationType'
print(f'Highest Earning User:{highest_earning_user}')

print('Relationship Between TalkTime and Charge')
correlation = data['timeDuration'].corr(data['amount'])
print(f'The correlation between TalkTime and Charge is: {correlation}')
```

```
## Task 2: Call Center Performance Metrics
### Average TalkTime for Different Call Activities
           Activity  AverageTalkTime
0              Call         1.979025
1              Chat         1.975428
2     Complementary         5.000000
3  public_live_Call         2.529850
Most Common Source of Calls
The most common source of calls is: Production
Total Earnings and Spending
Total earnings for users: 214065.89866666665
Total spending for masters: 99146.57095000001
Highest Earning User:gid                            19
guruName             Dr Balkrisna
amount                     1800.0
consultationType             Call
timeDuration                 30.0
Name: 24992, dtype: object
Relationship Between TalkTime and Charge
The correlation between TalkTime and Charge is: 0.6522799350153664
```

In [ ]:

```

```

In [ ]:

```

```

In [73]:

```
# Task 3: Call Handling Analysis
print('## Task 3: Call Handling Analysis')
print('Average Connection Time')
if 'connectTime' in data.columns and 'dialTime' in data.columns:
    data['connectTime'] = pd.to_datetime(data['connectTime'])
    data['dialTime'] = pd.to_datetime(data['dialTime'])
    data['connectionTime'] = (data['connectTime'] - data['dialTime']).dt.total_seconds()
    average_connection_time = data['connectionTime'].mean()
    print(f'The average connection time is: {average_connection_time} seconds')
else:
    print('connectTime or dialTime column is missing.')


print('Most Common Reason for Call Disconnection')

if 'unconnectTime' in data.columns and 'connectTime' in data.columns:
    data['unconnectTime'] = pd.to_datetime(data['unconnectTime'])
    data['disconnectionTime'] = (data['unconnectTime'] - data['connectTime']).dt.total_seconds()
    most_common_disconnection_reason = data['disconnectedBy'].value_counts().idxmax()
    print(f'The most common reason for call disconnection is: {most_common_disconnection_reason}')
else:
    print('unconnectTime or connectTime column is missing.')

print('#HangUpTime Patterns')
if 'disconnectedBy' in data.columns:
    hangup_patterns = data['disconnectedBy'].value_counts()
    print(hangup_patterns)
else:
    print('hangUpTime column is missing.')

# astrologerCallStatus
# print('### Total calls ')
# if 'astrologerCallStatus' in data.columns:
#     callStatus = data['astrologerCallStatus'].value_counts()
#     print(f'Total astrologerCallStatus: {callStatus}')

# else:
#     print('astrologerCallStatus column is missing.')
```

```
print('### Total calls based on astrologerCallStatus')

if 'astrologerCallStatus' in data.columns:
    astrologerCallStatus = data['astrologerCallStatus'].value_counts()
    print(astrologerCallStatus)

    # If callStatus details are required (assuming callStatus is another variable/column)
#     if 'callStatus' in locals() or 'callStatus' in globals():
#         print(f'Details of callStatus: {callStatus}')
#     else:
#         print('callStatus details are not available.')

else:
    print('astrologerCallStatus column is missing.')



# plt.savefig('chart_name.png')
```

```
## Task 3: Call Handling Analysis
Average Connection Time
connectTime or dialTime column is missing.
Most Common Reason for Call Disconnection
unconnectTime or connectTime column is missing.
#HangUpTime Patterns
user            6
astro_system    1
user_system     1
astrologer      1
Name: disconnectedBy, dtype: int64
### Total calls based on astrologerCallStatus
completed    9172
no-answer    5367
busy         1342
failed        377
Name: astrologerCallStatus, dtype: int64
```

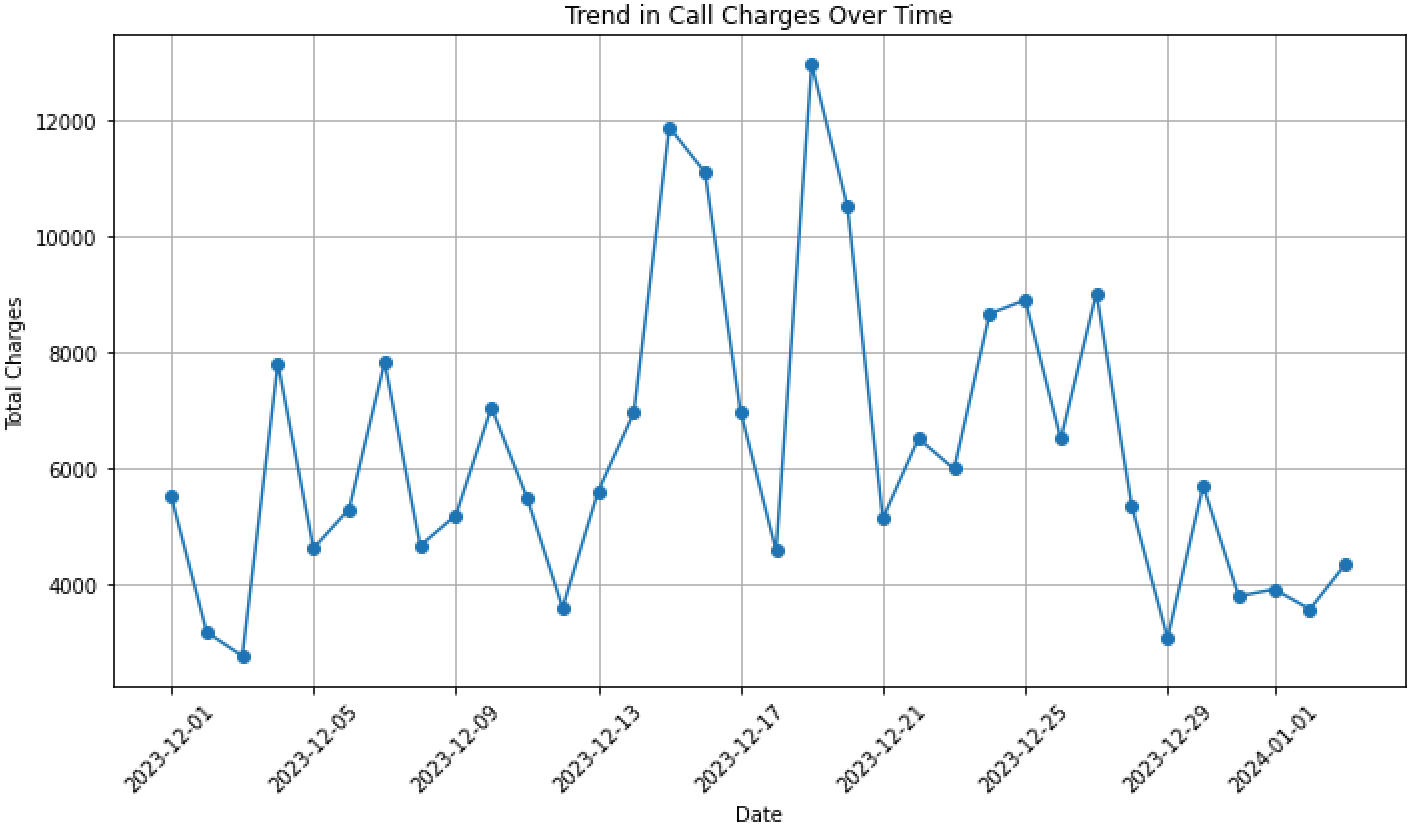
In [32]:

```
# Task 4: Order and Refund Analysis
print('## Task 4: Order and Refund Analysis')
print('### Order Status Distribution')
if 'orderStatus' in data.columns:
    order_status_distribution = data['orderStatus'].value_counts()
    print(order_status_distribution)
else:
    print('orderStatus column is missing.')

print('### Total Refund Amount and Refund Status Distribution')
if 'amount' in data.columns and 'refundStatus' in data.columns:
    total_refund_amount = data['amount'].sum()
    refund_status_distribution = data['refundStatus'].value_counts()
    print(f'Total refund amount: {total_refund_amount}')
    print(refund_status_distribution)
else:
    print('refundAmount or refundStatus column is missing.')
```

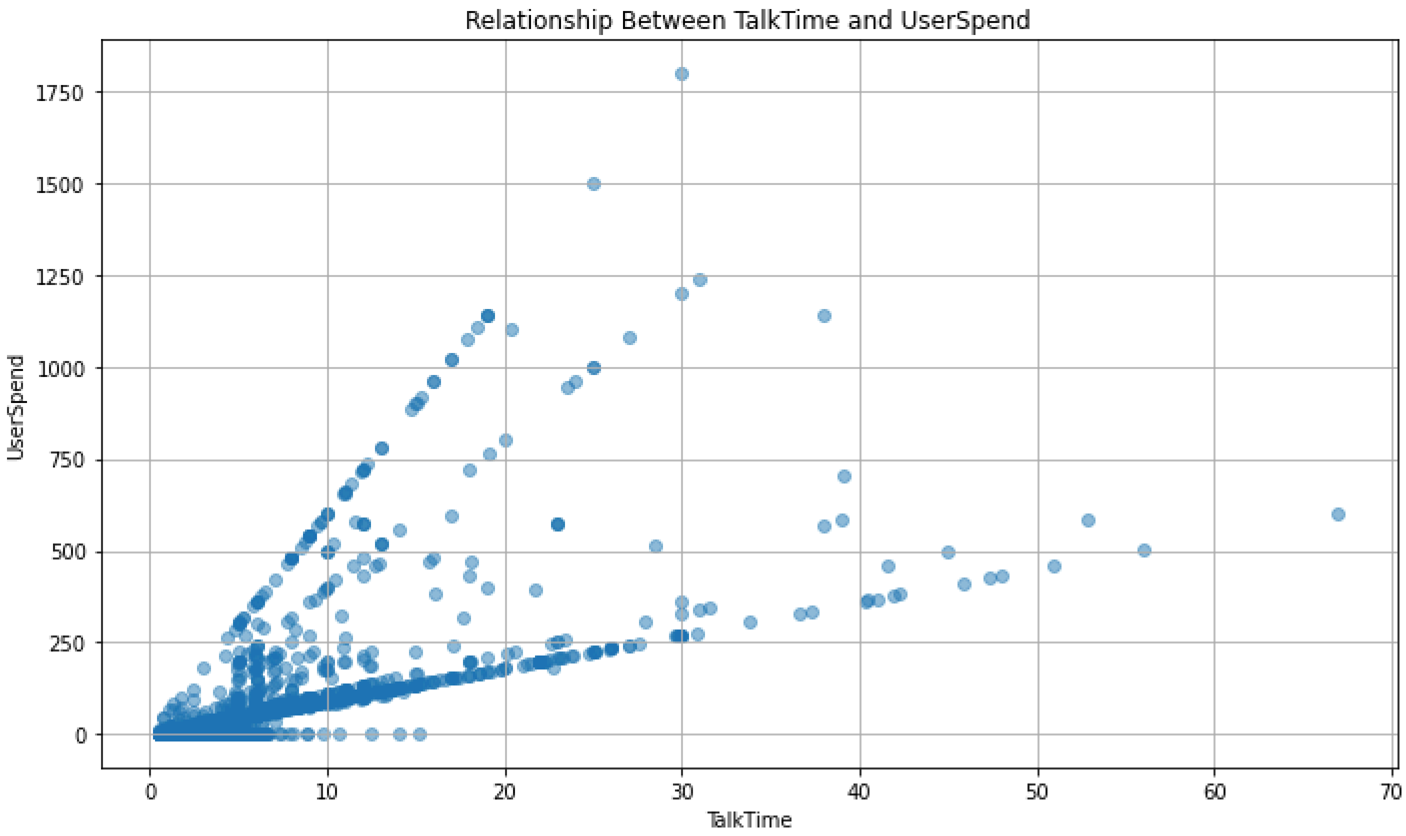
```
## Task 4: Order and Refund Analysis
### Order Status Distribution
orderStatus column is missing.
### Total Refund Amount and Refund Status Distribution
Total refund amount: 214065.89866666665
no-refund     28022
Approved          3
Rejected          1
Processing        1
Name: refundStatus, dtype: int64
```

In [74]:

```
# Task 6: Additional Visualizations

# Convert 'createdAt' column to datetime format
data['createdAt'] = pd.to_datetime(data['createdAt'])

# Group by date and sum the 'amount' column to get total charges per day
daily_charges = data.groupby(data['createdAt'].dt.date)['amount'].sum()

# Plotting the line chart
plt.figure(figsize=(10, 6))
plt.plot(daily_charges.index, daily_charges.values, marker='o', linestyle='-')
plt.title('Trend in Call Charges Over Time')
plt.xlabel('Date')
plt.ylabel('Total Charges')
plt.xticks(rotation=45)
plt.grid(True)
plt.tight_layout()
plt.show()
```

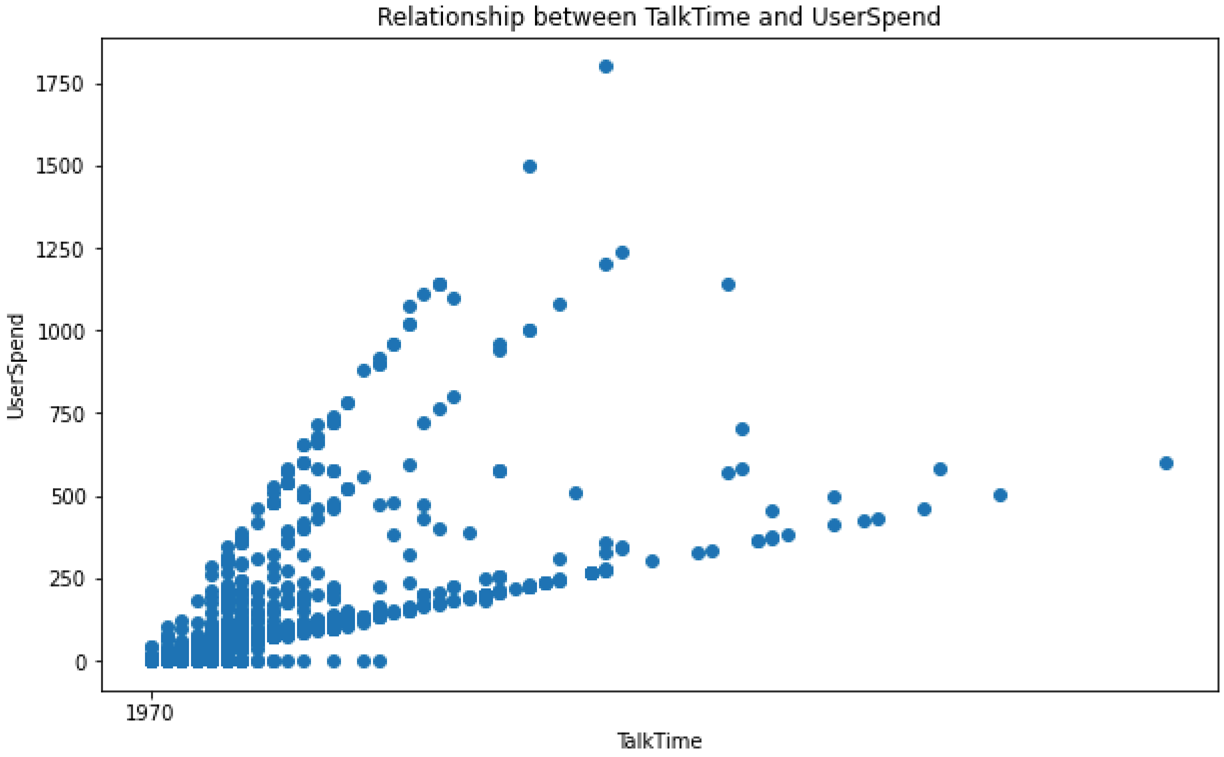

Trend in Call Charges Over Time
Total Charges
12000
10000
8000
6000
4000
Date
2023-12-01
2023-12-05
2023-12-09
2023-12-13
2023-12-17
2023-12-21
2023-12-25
2023-12-29
2024-01-01

In [75]:
```python
plt.figure(figsize=(10, 6))
plt.scatter(data['timeDuration'], data['amount'], alpha=0.5)
plt.title('Relationship Between TalkTime and UserSpend')
plt.xlabel('TalkTime')
plt.ylabel('UserSpend')
plt.grid(True)
plt.tight_layout()
plt.show()
```



In [35]:

```
st.write('### Relationship Between TalkTime and UserSpend')
if 'timeDuration' in data.columns and 'amount' in data.columns:
    plt.figure(figsize=(10, 6))
    plt.scatter(data['timeDuration'], data['amount'])
    plt.title('Relationship between TalkTime and UserSpend')
    plt.xlabel('TalkTime')
    plt.ylabel('UserSpend')
    st.pyplot(plt)
else:
    st.write('timeDuration or amount column is missing.')
```


Relationship between TalkTime and UserSpend
UserSpend
TalkTime