Data Analysis Step 1 : Ask Buisness task is How do annual members and casual riders use Cyclistic bikes differently?

In [2]:
```
# by importing the python libraries
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
```

Data Analysis Step 2 : Prepare the data

In [3]:
```
df=pd.read_csv(r"E:\New folder (2)\202310-divvy-tripdata.csv")
df.head()
```

Out[3]:

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_st
0	4449097279F8BBE7	classic_bike	2023-10-08 10:36:26	2023-10-08 10:49:19	Orleans St & Chestnut St (NEXT Apts)	620	She
1	9CF060543CA7B439	electric_bike	2023-10-11 17:23:59	2023-10-11 17:36:08	Desplaines St & Kinzie St	TA1306000003	She
2	667F21F4D6BDE69C	electric_bike	2023-10-12 07:02:33	2023-10-12 07:06:53	Orleans St & Chestnut St (NEXT Apts)	620	Frankl
3	F92714CC6B019B96	classic_bike	2023-10-24 19:13:03	2023-10-24 19:18:29	Desplaines St & Kinzie St	TA1306000003	Frankl
4	5E34BA5DE945A9CC	classic_bike	2023-10-09 18:19:26	2023-10-09 18:30:56	Desplaines St & Kinzie St	TA1306000003	Frankl

In [4]:
```
# Convert date columns to datetime
df['started_at'] = pd.to_datetime(df['started_at'])
df['ended_at'] = pd.to_datetime(df['ended_at'])

# Calculate new column ride length
df['ride_length'] = (df['ended_at'] - df['started_at']).dt.total_seconds()

# Calculate new column day of the week
df['day_of_week'] = df['started_at'].dt.dayofweek + 1

# Explore the prepare data
df.head()
```

Out[4]:

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_st
0	4449097279F8BBE7	classic_bike	2023-10-08 10:36:26	2023-10-08 10:49:19	Orleans St & Chestnut St (NEXT Apts)	620	She
1	9CF060543CA7B439	electric_bike	2023-10-11 17:23:59	2023-10-11 17:36:08	Desplaines St & Kinzie St	TA1306000003	She
2	667F21F4D6BDE69C	electric_bike	2023-10-12 07:02:33	2023-10-12 07:06:53	Orleans St & Chestnut St (NEXT Apts)	620	Frankl
3	F92714CC6B019B96	classic_bike	2023-10-24 19:13:03	2023-10-24 19:18:29	Desplaines St & Kinzie St	TA1306000003	Frankl
4	5E34BA5DE945A9CC	classic_bike	2023-10-09 18:19:26	2023-10-09 18:30:56	Desplaines St & Kinzie St	TA1306000003	Frankl

Step 3 : Process(cleaning and manipulation of data)

In [5]:
```
# Convert day of week numarical to monday,sunday
df['day_of_week'] = df['started_at'].dt.day_name()
df.head()
```

Out[5]:

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_st
0	4449097279F8BBE7	classic_bike	2023-10-08 10:36:26	2023-10-08 10:49:19	Orleans St & Chestnut St (NEXT Apts)	620	She
1	9CF060543CA7B439	electric_bike	2023-10-11 17:23:59	2023-10-11 17:36:08	Desplaines St & Kinzie St	TA1306000003	She
2	667F21F4D6BDE69C	electric_bike	2023-10-12 07:02:33	2023-10-12 07:06:53	Orleans St & Chestnut St (NEXT Apts)	620	Frankl
3	F92714CC6B019B96	classic_bike	2023-10-24 19:13:03	2023-10-24 19:18:29	Desplaines St & Kinzie St	TA1306000003	Frankl
4	5E34BA5DE945A9CC	classic_bike	2023-10-09 18:19:26	2023-10-09 18:30:56	Desplaines St & Kinzie St	TA1306000003	Frankl

In [6]:
```
df['ride_length'].unique()
```

Out[6]:
```
array([  773.,   729.,   260., ...,  9891.,  6377., 20319.])
```

In [7]:
```
# Is there any missing values in data?
df.isnull().sum()
```

Out[7]:
```
ride_id                 0
rideable_type           0
started_at              0
ended_at                0
start_station_name    84412
start_station_id      84412
end_station_name      89253
end_station_id        89253
start_lat               0
start_lng               0
end_lat               592
end_lng               592
member_casual           0
ride_length             0
day_of_week             0
dtype: int64
```

In [8]:
```
# drop  missing values in dataset
df.dropna(subset=['end_lat','end_lng','start_station_name','start_station_id','end_sta
df.head()
```

Out[8]:

	ride_id	rideable_type	started_at	ended_at	start_station_name	start_station_id	end_st
0	4449097279F8BBE7	classic_bike	2023-10-08 10:36:26	2023-10-08 10:49:19	Orleans St & Chestnut St (NEXT Apts)	620	She
1	9CF060543CA7B439	electric_bike	2023-10-11 17:23:59	2023-10-11 17:36:08	Desplaines St & Kinzie St	TA1306000003	She
2	667F21F4D6BDE69C	electric_bike	2023-10-12 07:02:33	2023-10-12 07:06:53	Orleans St & Chestnut St (NEXT Apts)	620	Frankl
3	F92714CC6B019B96	classic_bike	2023-10-24 19:13:03	2023-10-24 19:18:29	Desplaines St & Kinzie St	TA1306000003	Frankl
4	5E34BA5DE945A9CC	classic_bike	2023-10-09 18:19:26	2023-10-09 18:30:56	Desplaines St & Kinzie St	TA1306000003	Frankl

Step 4 : Analyze

In [9]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 403781 entries, 0 to 537112
Data columns (total 15 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   ride_id             403781 non-null  object
 1   rideable_type       403781 non-null  object
 2   started_at          403781 non-null  datetime64[ns]
 3   ended_at            403781 non-null  datetime64[ns]
 4   start_station_name  403781 non-null  object
 5   start_station_id    403781 non-null  object
 6   end_station_name    403781 non-null  object
 7   end_station_id      403781 non-null  object
 8   start_lat           403781 non-null  float64
 9   start_lng           403781 non-null  float64
 10  end_lat             403781 non-null  float64
 11  end_lng             403781 non-null  float64
 12  member_casual       403781 non-null  object
 13  ride_length         403781 non-null  float64
 14  day_of_week         403781 non-null  object
dtypes: datetime64[ns](2), float64(5), object(8)
memory usage: 49.3+ MB
```

In [10]:
```
# statstical summary of data
df.describe()
```

Out[10]:

	started_at	ended_at	start_lat	start_lng	end_lat	end_
count	403781	403781	403781.000000	403781.000000	403781.000000	403781.000
mean	2023-10-15 01:47:27.261933824	2023-10-15 02:02:14.567235840	41.897146	-87.643589	41.897592	-87.643
min	2023-10-01 00:00:05	2023-10-01 00:02:02	41.648596	-87.836798	41.648501	-87.841
25%	2023-10-06 19:01:55	2023-10-06 19:15:32	41.877864	-87.656952	41.878119	-87.658
50%	2023-10-14 22:45:14	2023-10-14 22:59:09	41.894345	-87.641238	41.894503	-87.641
75%	2023-10-22 17:47:32	2023-10-22 18:07:14	41.925566	-87.627050	41.925602	-87.627
max	2023-10-31 23:59:57	2023-11-01 15:36:02	42.064854	-87.528232	42.064854	-87.528
std	NaN	NaN	0.046077	0.025406	0.046252	0.025

In [11]:
```
# mode of day of week
df['day_of_week'].mode()
```

Out[11]:
```
0    Tuesday
Name: day_of_week, dtype: object
```

In [12]:
```
df['day_of_week'].value_counts()
```

Out[12]:
```
day_of_week
Tuesday      72476
Monday       64348
Sunday       62565
Wednesday    59261
Thursday     52374
Friday       47397
Saturday     45360
Name: count, dtype: int64
```

In [13]:
```
df['ride_length'].unique()
```

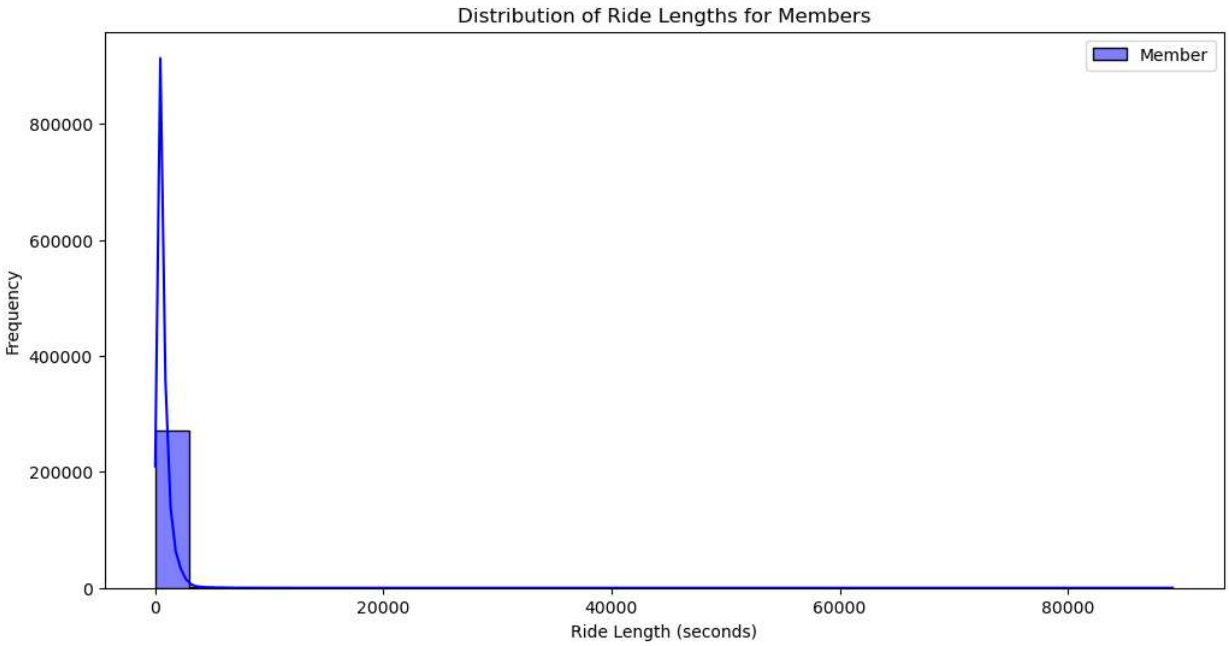
Out[13]:
```
array([  773.,   729.,   260., ...,  6377., 20319.,  7054.])
```

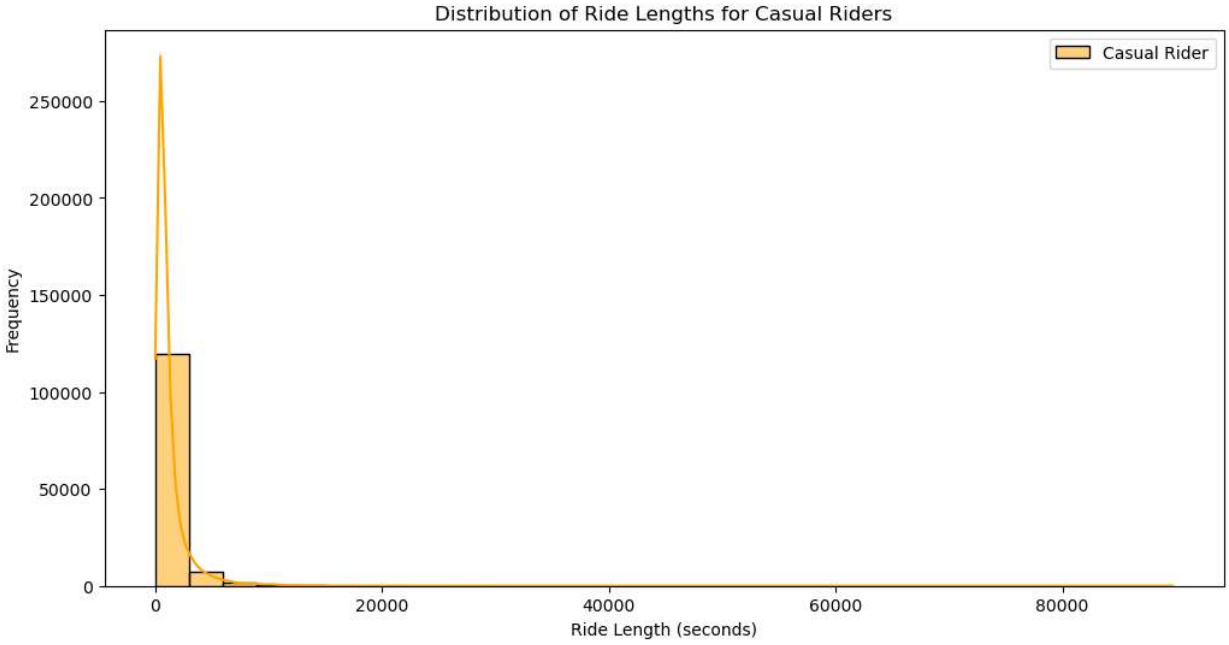
In [23]:
```
import scipy.stats as stats

# Hypothesis Testing: Check for significant differences in ride lengths between member
member_rides = df[df['member_casual'] == 'member']['ride_length']
casual_rides = df[df['member_casual'] == 'casual']['ride_length']

# Perform an independent t-test
t_stat, p_value = stats.ttest_ind(member_rides, casual_rides, equal_var=False)

# Print the results
print(f'T-test statistic: {t_stat}')
print(f'P-value: {p_value}')
```

```
T-test statistic: -67.17808304829691
P-value: 0.0
```

The negative t-test statistic indicates that, on average, the ride lengths of members are significantly shorter than those of casual riders.

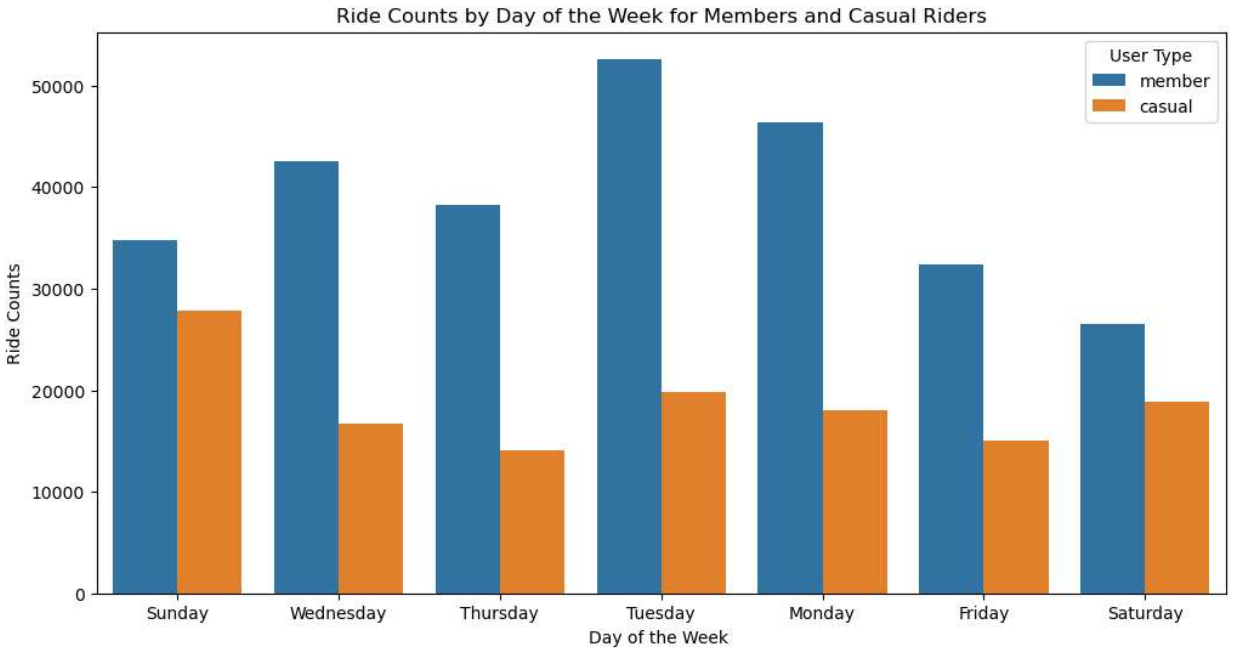
In [14]:
```
#  distribution of ride lengths for members
plt.figure(figsize=(12, 6))
sns.histplot(data=df[df['member_casual'] == 'member'], x='ride_length', bins=30, kde=T
plt.title('Distribution of Ride Lengths for Members')
plt.xlabel('Ride Length (seconds)')
plt.ylabel('Frequency')
plt.legend()
plt.show()

# the distribution of ride lengths for casual riders
plt.figure(figsize=(12, 6))
sns.histplot(data=df[df['member_casual'] == 'casual'], x='ride_length', bins=30, kde=T
plt.title('Distribution of Ride Lengths for Casual Riders')
plt.xlabel('Ride Length (seconds)')
plt.ylabel('Frequency')
plt.legend()
plt.show()
```



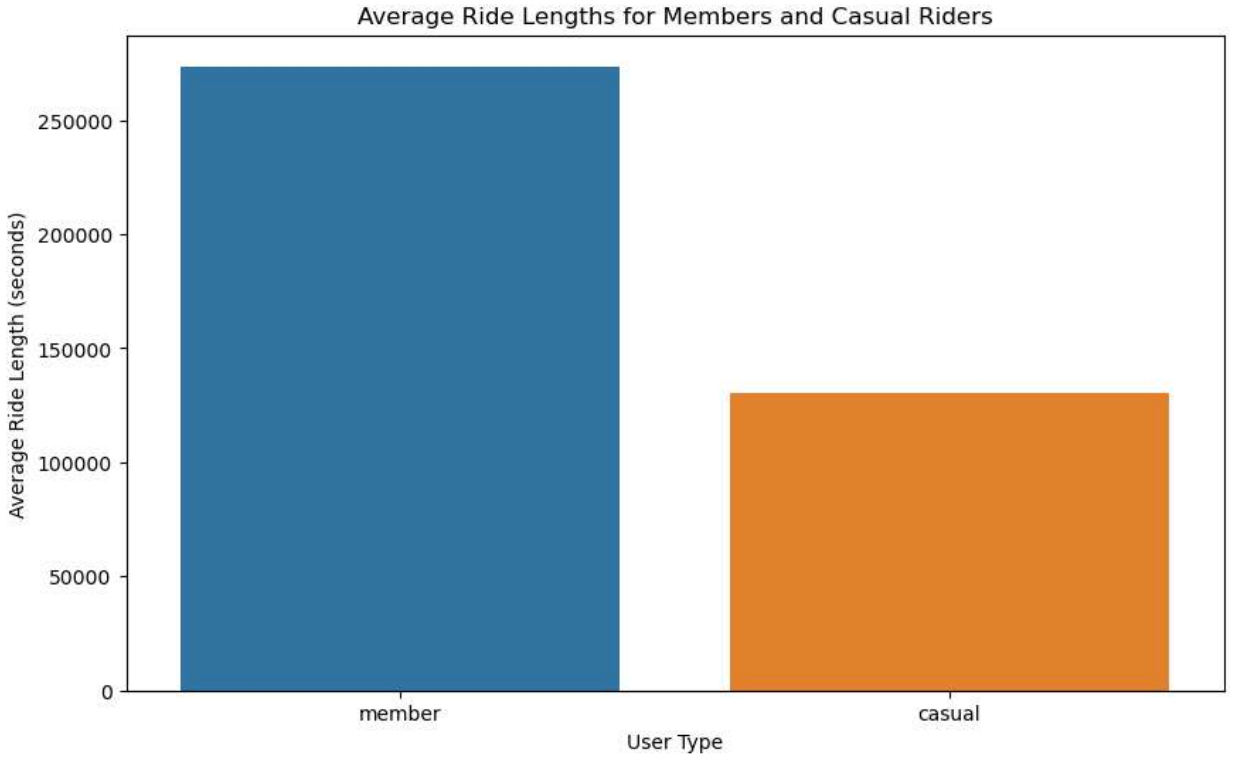


Step 6 : Share

In [15]:
```
# Number of rides by day of week for members and casual
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='day_of_week', hue='member_casual')
plt.title('Ride Counts by Day of the Week for Members and Casual Riders')
plt.xlabel('Day of the Week ')
plt.ylabel('Ride Counts')
plt.legend(title='User Type')
plt.show()
```

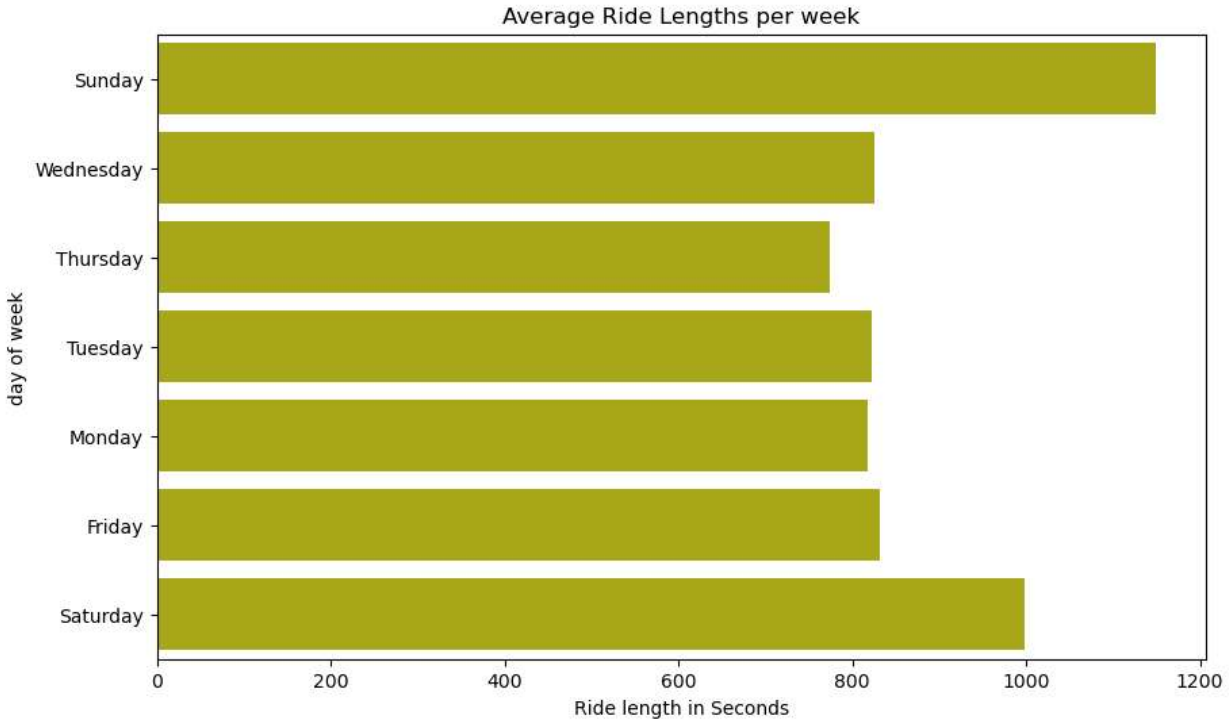


In [16]:
```
# ride length for members and casual
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='member_casual')
plt.title('Average Ride Lengths for Members and Casual Riders')
plt.xlabel('User Type')
plt.ylabel('Average Ride Length (seconds)')
plt.show()
```



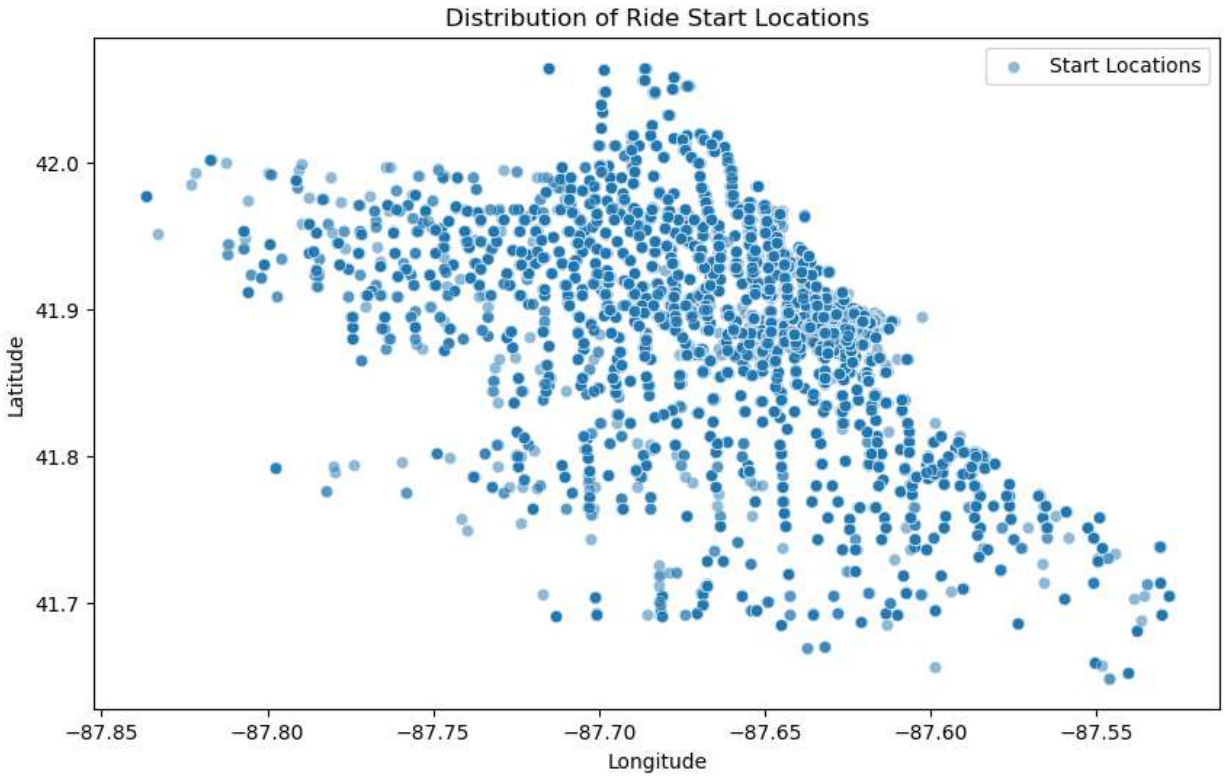
In [17]:
```
# average ride length per week
plt.figure(figsize=(10, 6))
sns.barplot(data=df, x='ride_length', y='day_of_week',errorbar=None,color='y')
plt.title('Average Ride Lengths per week')
plt.xlabel('Ride length in Seconds')
```

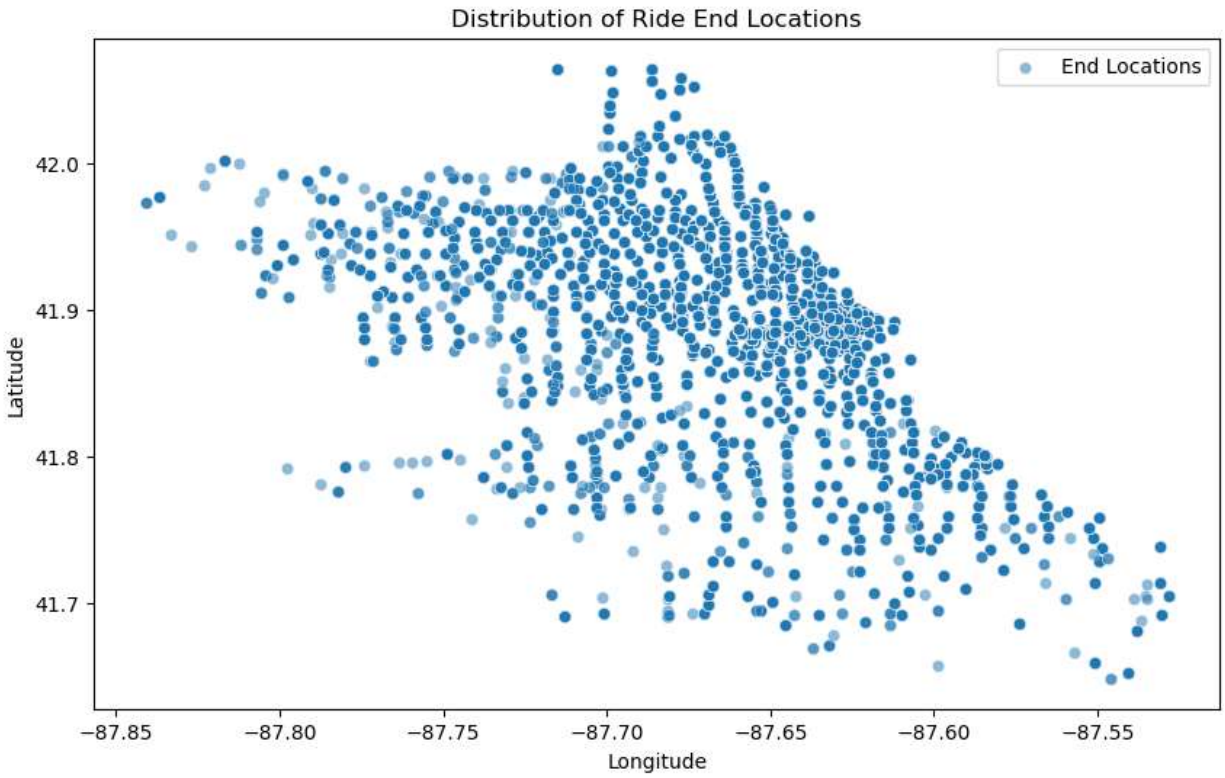
```
plt.ylabel('day of week')
plt.show()
```



In [18]:

```
start_locations = df[['start_lat', 'start_lng']]
end_locations = df[['end_lat', 'end_lng']]

# Scatter plot for start locations
plt.figure(figsize=(10, 6))
sns.scatterplot(x='start_lng', y='start_lat', data=start_locations, alpha=0.5, label='
plt.title('Distribution of Ride Start Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.legend()
plt.show()

# Scatter plot for end locations
plt.figure(figsize=(10, 6))
sns.scatterplot(x='end_lng', y='end_lat', data=end_locations, alpha=0.5, label='End Lo
plt.title('Distribution of Ride End Locations')
plt.xlabel('Longitude')
plt.ylabel('Latitude')
plt.legend()
plt.show()
```





In [20]:
```
df['hour_of_day'] = df['started_at'].dt.hour

# Analyze ride patterns during different times of the day
#By  Creating  a bar plot
plt.figure(figsize=(12, 6))
sns.countplot(data=df, x='hour_of_day', hue='member_casual')
plt.title('Ride Counts by Hour of the Day for Members and Casual Riders')
plt.xlabel('Hour of the Day')
plt.ylabel('Ride Counts')
```

```
plt.legend(title='User Type')
plt.show()
```