step : Imports & Reading Data

In [1]:

```
pip install pandas sqlite3 matplotlib seaborn
```

```
Requirement already satisfied: pandas in c:₩users₩seoin₩anaconda₩lib₩site-packages
(2.2.2)
Note: you may need to restart the kernel to use updated packages.
```

```
ERROR: Could not find a version that satisfies the requirement sqlite3 (from version
s: none)
ERROR: No matching distribution found for sqlite3
```

In [2]:

```
import pandas as pd
import sqlite3
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from scipy import stats
pd.set_option('display.max_columns', None)
```

```
C:₩Users₩seoin₩anaconda₩Lib₩site-packages₩pandas₩core₩arrays₩masked.py:60: UserWarni
ng: Pandas requires version '1.3.6' or newer of 'bottleneck' (version '1.3.5' curren
tly installed).
  from pandas.core import (
```

In [3]:

```
# Connect to SQLite database
conn = sqlite3.connect('cruise_data.db')
cursor = conn.cursor()

# Load CSV data into a DataFrame
df = pd.read_csv('C:/Users/seoin/Desktop/Tui/task_data/data.csv')

# Create a table in the SQLite database
df.to_sql('cruise_data', conn, if_exists='replace', index=False)

# Verify the data is loaded
query = "SELECT * FROM cruise_data"
df_sql = pd.read_sql_query(query, conn)
print(df_sql.head())
```

```
           Start Time             End Time Vessel Name  Power Galley 1 (MW)  ₩
0  2023-01-01T00:00:00  2023-01-01T00:05:00    Vessel 1               0.0946
1  2023-01-01T00:05:00  2023-01-01T00:10:00    Vessel 1               0.0540
2  2023-01-01T00:10:00  2023-01-01T00:15:00    Vessel 1               0.0439
3  2023-01-01T00:15:00  2023-01-01T00:20:00    Vessel 1               0.0733
4  2023-01-01T00:20:00  2023-01-01T00:25:00    Vessel 1               0.0780

   Power Galley 2 (MW)  Power Service (MW)  HVAC Chiller 1 Power (MW)  ₩
0               0.1384              5.4654                     0.5074
1               0.1370              5.4387                     0.5158
2               0.1785              5.5265                     0.5117
3               0.1725              5.5257                     0.5177
4               0.1397              5.4634                     0.5169

   HVAC Chiller 2 Power (MW)  HVAC Chiller 3 Power (MW)  Scrubber Power (MW)  ₩
0                        0.0                     0.4979               0.4191
1                        0.0                     0.4982               0.4204
2                        0.0                     0.5032               0.4199
3                        0.0                     0.5103               0.4188
4                        0.0                     0.5100               0.4203

   Sea Temperature (Celsius)  Boiler 1 Fuel Flow Rate (L/h)  ₩
0                    27.3000                         0.0000
1                    27.3000                        47.7695
2                    27.3000                        77.2034
3                    27.3076                        60.6369
4                    27.3518                        55.2184

   Boiler 2 Fuel Flow Rate (L/h)  Incinerator 1 Fuel Flow Rate (L/h)  ₩
0                            0.0                             19.0090
1                            0.0                            216.3180
2                            0.0                            439.4300
3                            0.0                            218.2797
4                            0.0                              0.0000

   Diesel Generator 1 Power (MW)  Diesel Generator 2 Power (MW)  ₩
0                            0.0                            0.0
1                            0.0                            0.0
2                            0.0                            0.0
3                            0.0                            0.0
4                            0.0                            0.0

   Diesel Generator 3 Power (MW)  Diesel Generator 4 Power (MW)  ₩
0                            0.0                         7.3349
1                            0.0                         7.3011
2                            0.0                         7.3299
3                            0.0                         7.3712
4                            0.0                         7.3032

   Latitude (Degrees)  Longitude (Degrees)  Relative Wind Angle (Degrees)  ₩
0            17.72523            -65.45738                         8.4428
1            17.73088            -65.44803                        41.3100
2            17.73655            -65.43887                        23.9997
3            17.74202            -65.42980                        14.5540
4            17.74713            -65.42042                        14.5632

   True Wind Angle (Degrees)  Depth (m)  Relative Wind Direction (Degrees)  ₩
0                    10.9049        NaN                            64.3112
1                    78.7817        NaN                            62.8161
2                    33.6216        NaN                            80.7356
3                    20.0348        NaN                            75.9723
4                    20.0328        NaN                            74.6509

   True Wind Direction (Degrees)  Draft (m)  Speed Over Ground (knots)  ₩
```

```
0                    66.7735     7.8721                        7.6300
1                    64.3452     7.8713                        7.5800
2                    90.3574     7.8718                        7.4379
3                    81.4529     7.8710                        7.3979
4                    80.1204     7.8707                        7.4343

   True Wind Speed (knots)  Relative Wind Speed (knots)  ₩
0                  19.5050                      27.0579
1                  19.2968                      26.8067
2                  19.4491                      25.8380
3                  20.6231                      27.6498
4                  20.4554                      27.5341

   Speed Through Water (knots)  Local Time (h)  Trim (m)  ₩
0                       7.8881        19.67367   -0.1425
1                       7.7438        19.75763   -0.1405
2                       7.6320        19.84158   -0.1450
3                       7.5080        19.92551   -0.1308
4                       7.5521        20.00947   -0.1269

   Propulsion Power (MW)  Port Side Propulsion Power (MW)  ₩
0                 1.8691                           0.8854
1                 1.8622                           0.8737
2                 1.8036                           0.8441
3                 1.8457                           0.8543
4                 1.8399                           0.8467

   Starboard Side Propulsion Power (MW)  Bow Thruster 1 Power (MW)  ₩
0                                0.9837                        0.0
1                                0.9885                        0.0
2                                0.9595                        0.0
3                                0.9914                        0.0
4                                0.9932                        0.0

   Bow Thruster 2 Power (MW)  Bow Thruster 3 Power (MW)  ₩
0                        0.0                        0.0
1                        0.0                        0.0
2                        0.0                        0.0
3                        0.0                        0.0
4                        0.0                        0.0

   Stern Thruster 1 Power (MW)  Stern Thruster 2 Power (MW)  ₩
0                          0.0                          0.0
1                          0.0                          0.0
2                          0.0                          0.0
3                          0.0                          0.0
4                          0.0                          0.0

   Main Engine 1 Fuel Flow Rate (kg/h)  Main Engine 2 Fuel Flow Rate (kg/h)  ₩
0                                  0.0                                  0.0
1                                  0.0                                  0.0
2                                  0.0                                  0.0
3                                  0.0                                  0.0
4                                  0.0                                  0.0

   Main Engine 3 Fuel Flow Rate (kg/h)  Main Engine 4 Fuel Flow Rate (kg/h)
0                                  0.0                           1645.82000
1                                  0.0                           1643.78999
2                                  0.0                           1642.07000
3                                  0.0                           1650.71000
4                                  0.0                           1644.54000
```

Step : Data Understanding

In [4]:
```
df.shape
```

Out[4]:
```
(210240, 44)
```

In [5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 210240 entries, 0 to 210239
Data columns (total 44 columns):
 #   Column                                Non-Null Count   Dtype
---  ------                                --------------   -----
 0   Start Time                            210240 non-null  object
 1   End Time                              210240 non-null  object
 2   Vessel Name                           210240 non-null  object
 3   Power Galley 1 (MW)                   210224 non-null  float64
 4   Power Galley 2 (MW)                   210224 non-null  float64
 5   Power Service (MW)                    210222 non-null  float64
 6   HVAC Chiller 1 Power (MW)             210033 non-null  float64
 7   HVAC Chiller 2 Power (MW)             210033 non-null  float64
 8   HVAC Chiller 3 Power (MW)             210033 non-null  float64
 9   Scrubber Power (MW)                   210224 non-null  float64
 10  Sea Temperature (Celsius)             210224 non-null  float64
 11  Boiler 1 Fuel Flow Rate (L/h)         210224 non-null  float64
 12  Boiler 2 Fuel Flow Rate (L/h)         210224 non-null  float64
 13  Incinerator 1 Fuel Flow Rate (L/h)    210224 non-null  float64
 14  Diesel Generator 1 Power (MW)         210224 non-null  float64
 15  Diesel Generator 2 Power (MW)         210224 non-null  float64
 16  Diesel Generator 3 Power (MW)         210224 non-null  float64
 17  Diesel Generator 4 Power (MW)         210224 non-null  float64
 18  Latitude (Degrees)                    209900 non-null  float64
 19  Longitude (Degrees)                   209900 non-null  float64
 20  Relative Wind Angle (Degrees)         210226 non-null  float64
 21  True Wind Angle (Degrees)             210166 non-null  float64
 22  Depth (m)                             152746 non-null  float64
 23  Relative Wind Direction (Degrees)     210185 non-null  float64
 24  True Wind Direction (Degrees)         210166 non-null  float64
 25  Draft (m)                             209097 non-null  float64
 26  Speed Over Ground (knots)             209340 non-null  float64
 27  True Wind Speed (knots)               210166 non-null  float64
 28  Relative Wind Speed (knots)           210226 non-null  float64
 29  Speed Through Water (knots)           209299 non-null  float64
 30  Local Time (h)                        209900 non-null  float64
 31  Trim (m)                              209161 non-null  float64
 32  Propulsion Power (MW)                 210224 non-null  float64
 33  Port Side Propulsion Power (MW)       210224 non-null  float64
 34  Starboard Side Propulsion Power (MW)  210224 non-null  float64
 35  Bow Thruster 1 Power (MW)             210224 non-null  float64
 36  Bow Thruster 2 Power (MW)             210224 non-null  float64
 37  Bow Thruster 3 Power (MW)             210224 non-null  float64
 38  Stern Thruster 1 Power (MW)           210224 non-null  float64
 39  Stern Thruster 2 Power (MW)           210224 non-null  float64
 40  Main Engine 1 Fuel Flow Rate (kg/h)   210224 non-null  float64
 41  Main Engine 2 Fuel Flow Rate (kg/h)   210224 non-null  float64
 42  Main Engine 3 Fuel Flow Rate (kg/h)   210224 non-null  float64
 43  Main Engine 4 Fuel Flow Rate (kg/h)   210224 non-null  float64
dtypes: float64(41), object(3)
memory usage: 70.6+ MB
```

In [6]:
```
df.head()
```

Out[6]:

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
0	2023-01-01T00:00:00	2023-01-01T00:05:00	Vessel 1	0.0946	0.1384	5.4654	0.5074	0.0	0.4979	0.4191	
1	2023-01-01T00:05:00	2023-01-01T00:10:00	Vessel 1	0.0540	0.1370	5.4387	0.5158	0.0	0.4982	0.4204	
2	2023-01-01T00:10:00	2023-01-01T00:15:00	Vessel 1	0.0439	0.1785	5.5265	0.5117	0.0	0.5032	0.4199	
3	2023-01-01T00:15:00	2023-01-01T00:20:00	Vessel 1	0.0733	0.1725	5.5257	0.5177	0.0	0.5103	0.4188	
4	2023-01-01T00:20:00	2023-01-01T00:25:00	Vessel 1	0.0780	0.1397	5.4634	0.5169	0.0	0.5100	0.4203	

In [7]:

```
df.columns
```

Out[7]:

```
Index(['Start Time', 'End Time', 'Vessel Name', 'Power Galley 1 (MW)',
       'Power Galley 2 (MW)', 'Power Service (MW)',
       'HVAC Chiller 1 Power (MW)', 'HVAC Chiller 2 Power (MW)',
       'HVAC Chiller 3 Power (MW)', 'Scrubber Power (MW)',
       'Sea Temperature (Celsius)', 'Boiler 1 Fuel Flow Rate (L/h)',
       'Boiler 2 Fuel Flow Rate (L/h)', 'Incinerator 1 Fuel Flow Rate (L/h)',
       'Diesel Generator 1 Power (MW)', 'Diesel Generator 2 Power (MW)',
       'Diesel Generator 3 Power (MW)', 'Diesel Generator 4 Power (MW)',
       'Latitude (Degrees)', 'Longitude (Degrees)',
       'Relative Wind Angle (Degrees)', 'True Wind Angle (Degrees)',
       'Depth (m)', 'Relative Wind Direction (Degrees)',
       'True Wind Direction (Degrees)', 'Draft (m)',
       'Speed Over Ground (knots)', 'True Wind Speed (knots)',
       'Relative Wind Speed (knots)', 'Speed Through Water (knots)',
       'Local Time (h)', 'Trim (m)', 'Propulsion Power (MW)',
       'Port Side Propulsion Power (MW)',
       'Starboard Side Propulsion Power (MW)', 'Bow Thruster 1 Power (MW)',
       'Bow Thruster 2 Power (MW)', 'Bow Thruster 3 Power (MW)',
       'Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)',
       'Main Engine 1 Fuel Flow Rate (kg/h)',
       'Main Engine 2 Fuel Flow Rate (kg/h)',
       'Main Engine 3 Fuel Flow Rate (kg/h)',
       'Main Engine 4 Fuel Flow Rate (kg/h)'],
      dtype='object')
```

Step : Data Preperation

In [8]:

```
# Convert time columns to datetime dtype
df['Start Time'] = pd.to_datetime(df['Start Time'])
df['End Time'] = pd.to_datetime(df['End Time'])

# Display info for only 'Start Time' and 'End Time' columns
df[['Start Time', 'End Time']].info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 210240 entries, 0 to 210239
Data columns (total 2 columns):
 #   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   Start Time  210240 non-null  datetime64[ns]
 1   End Time    210240 non-null  datetime64[ns]
dtypes: datetime64[ns](2)
memory usage: 3.2 MB
```

Missing value check

In [9]:

```
df.isnull().sum()
```

Out[9]:

```
Start Time                                   0
End Time                                     0
Vessel Name                                  0
Power Galley 1 (MW)                         16
Power Galley 2 (MW)                         16
Power Service (MW)                          18
HVAC Chiller 1 Power (MW)                  207
HVAC Chiller 2 Power (MW)                  207
HVAC Chiller 3 Power (MW)                  207
Scrubber Power (MW)                         16
Sea Temperature (Celsius)                   16
Boiler 1 Fuel Flow Rate (L/h)               16
Boiler 2 Fuel Flow Rate (L/h)               16
Incinerator 1 Fuel Flow Rate (L/h)          16
Diesel Generator 1 Power (MW)               16
Diesel Generator 2 Power (MW)               16
Diesel Generator 3 Power (MW)               16
Diesel Generator 4 Power (MW)               16
Latitude (Degrees)                         340
Longitude (Degrees)                        340
Relative Wind Angle (Degrees)               14
True Wind Angle (Degrees)                   74
Depth (m)                                57494
Relative Wind Direction (Degrees)           55
True Wind Direction (Degrees)               74
Draft (m)                                 1143
Speed Over Ground (knots)                  900
True Wind Speed (knots)                     74
Relative Wind Speed (knots)                 14
Speed Through Water (knots)                941
Local Time (h)                             340
Trim (m)                                  1079
Propulsion Power (MW)                       16
Port Side Propulsion Power (MW)             16
Starboard Side Propulsion Power (MW)        16
Bow Thruster 1 Power (MW)                   16
Bow Thruster 2 Power (MW)                   16
Bow Thruster 3 Power (MW)                   16
Stern Thruster 1 Power (MW)                 16
Stern Thruster 2 Power (MW)                 16
Main Engine 1 Fuel Flow Rate (kg/h)         16
Main Engine 2 Fuel Flow Rate (kg/h)         16
Main Engine 3 Fuel Flow Rate (kg/h)         16
Main Engine 4 Fuel Flow Rate (kg/h)         16
dtype: int64
```

In [10]:

```
df.duplicated().sum()
```

Out[10]:

```
0
```

In [11]:

```
df[df['Power Galley 1 (MW)'].isna()]
```

Out[11]:

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
88578	2023-11-04 13:30:00	2023-11-04 13:35:00	Vessel 1	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
140167	2023-05-02 16:35:00	2023-05-02 16:40:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
145233	2023-05-20 06:45:00	2023-05-20 06:50:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
161071	2023-07-14 06:35:00	2023-07-14 06:40:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
163498	2023-07-22 16:50:00	2023-07-22 16:55:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
163499	2023-07-22 16:55:00	2023-07-22 17:00:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
163500	2023-07-22 17:00:00	2023-07-22 17:05:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
165191	2023-07-28 13:55:00	2023-07-28 14:00:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
165767	2023-07-30 13:55:00	2023-07-30 14:00:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
165768	2023-07-30 14:00:00	2023-07-30 14:05:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
168077	2023-08-07 14:25:00	2023-08-07 14:30:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
168078	2023-08-07 14:30:00	2023-08-07 14:35:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
179568	2023-09-16 12:00:00	2023-09-16 12:05:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
179569	2023-09-16 12:05:00	2023-09-16 12:10:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
179572	2023-09-16 12:20:00	2023-09-16 12:25:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
209699	2023-12-30 02:55:00	2023-12-30 03:00:00	Vessel 2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	

In [12]:
```
# Drop rows where 'Power Galley 1 (MW)' has missing values becasue most of the data
df = df.dropna(subset=['Power Galley 1 (MW)'])
```

In [13]:
```
df[df['HVAC Chiller 1 Power (MW)'].isna()]
```

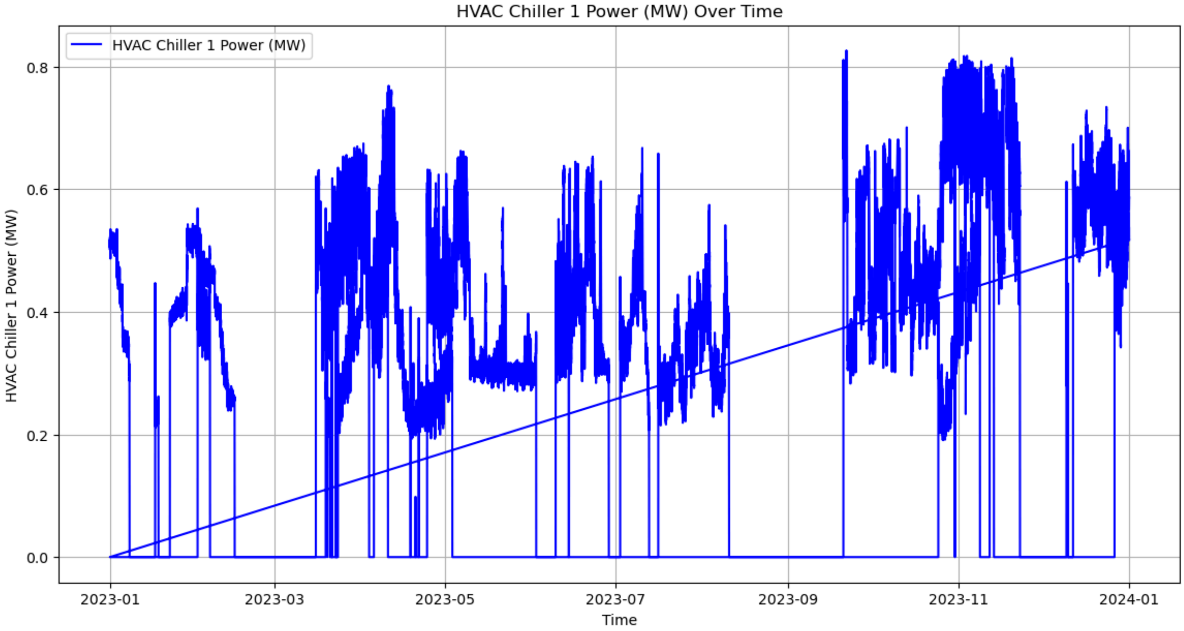
Out[13]:

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
197377	2023-11-17 08:05:00	2023-11-17 08:10:00	Vessel 2	0.0000	0.0918	4.4755	NaN	NaN	NaN	0.1543	
197378	2023-11-17 08:10:00	2023-11-17 08:15:00	Vessel 2	0.0000	0.0716	4.5178	NaN	NaN	NaN	0.1534	
197379	2023-11-17 08:15:00	2023-11-17 08:20:00	Vessel 2	0.0000	0.0638	4.4713	NaN	NaN	NaN	0.1530	
197380	2023-11-17 08:20:00	2023-11-17 08:25:00	Vessel 2	0.0071	0.0620	4.5521	NaN	NaN	NaN	0.1545	
197381	2023-11-17 08:25:00	2023-11-17 08:30:00	Vessel 2	0.0012	0.0420	4.4380	NaN	NaN	NaN	0.1539	
...	...	...	...	...	...	...	...	...	...	...	
197563	2023-11-17 23:35:00	2023-11-17 23:40:00	Vessel 2	0.0218	0.1772	5.5094	NaN	NaN	NaN	0.1577	
197564	2023-11-17 23:40:00	2023-11-17 23:45:00	Vessel 2	0.0487	0.1671	5.5004	NaN	NaN	NaN	0.1569	
197565	2023-11-17 23:45:00	2023-11-17 23:50:00	Vessel 2	0.0308	0.1678	5.5444	NaN	NaN	NaN	0.1581	
197566	2023-11-17 23:50:00	2023-11-17 23:55:00	Vessel 2	0.0497	0.1516	5.5213	NaN	NaN	NaN	0.1584	
197567	2023-11-17 23:55:00	2023-11-18 00:00:00	Vessel 2	0.0697	0.2012	5.6888	NaN	NaN	NaN	0.1585	

191 rows × 44 columns

In [14]:

```
# Ensure 'Start Time' is in datetime format
df['Start Time'] = pd.to_datetime(df['Start Time'])

# Plot the time series for 'HVAC Chiller 1 Power (MW)'
plt.figure(figsize=(14, 7))
plt.plot(df['Start Time'], df['HVAC Chiller 1 Power (MW)'], label='HVAC Chiller 1 Po
plt.xlabel('Time')
plt.ylabel('HVAC Chiller 1 Power (MW)')
plt.title('HVAC Chiller 1 Power (MW) Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



In [15]:

```
# Sort the dataframe by 'Start Time' to ensure proper filling
df = df.sort_values(by='Start Time')

# Set 'Start Time' as the index
df.set_index('Start Time', inplace=True)

# Interpolate to fill missing values for chiller columns
##interpolation is a process of determining the unknown values that lie in between th
df['HVAC Chiller 1 Power (MW)'] = df['HVAC Chiller 1 Power (MW)'].interpolate()
df['HVAC Chiller 2 Power (MW)'] = df['HVAC Chiller 2 Power (MW)'].interpolate()
df['HVAC Chiller 3 Power (MW)'] = df['HVAC Chiller 3 Power (MW)'].interpolate()

# If you need to reset the index back to columns
df.reset_index(inplace=True)
```

In [16]:

```
df[df['Power Service (MW)'].isna()]
```

Out[16]:

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
148890	2023-09-16 12:15:00	2023-09-16 12:20:00	Vessel 2	0.03	0.080	NaN	0.000	0.384	0.0	0.773	
209145	2023-12-30 03:00:00	2023-12-30 03:05:00	Vessel 2	0.03	0.087	NaN	0.508	0.528	0.0	0.346	

In [17]:

```
# First forward fill, then backward fill
# # the next available value after the missing data point replaces the missing value
df['Power Service (MW)'] = df['Power Service (MW)'].ffill().bfill()
```

In [18]:

```
df[df['Speed Over Ground (knots)'].isna()]
```

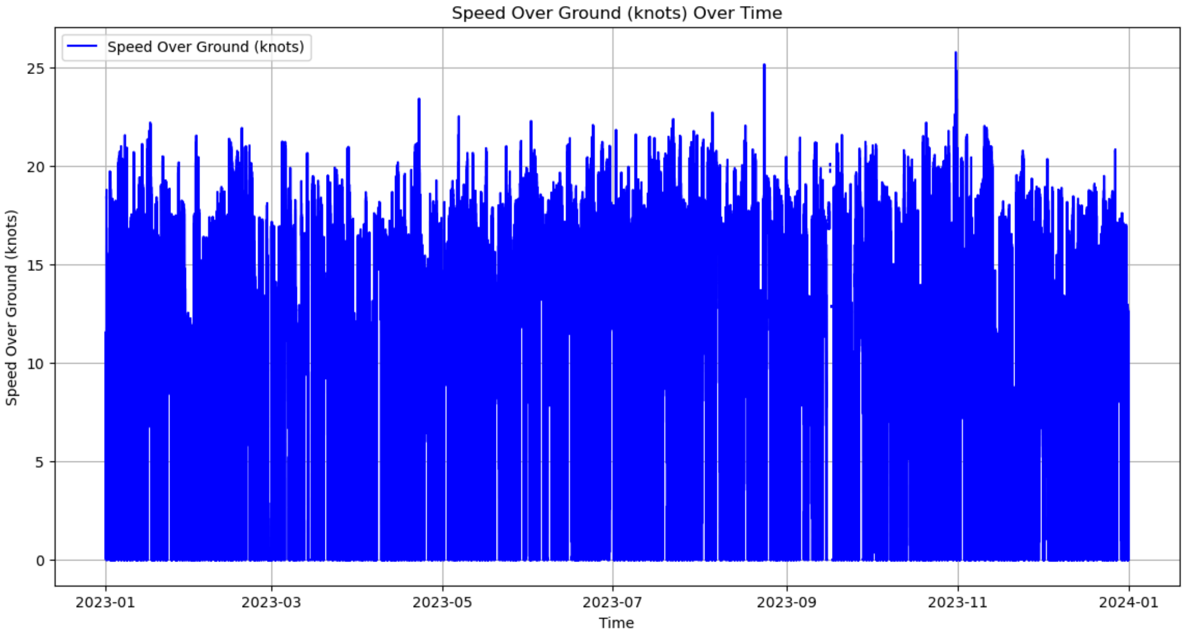
Out[18]:

	Start Time	End Time	Vessel Name	Power Galley 1 (MW)	Power Galley 2 (MW)	Power Service (MW)	HVAC Chiller 1 Power (MW)	HVAC Chiller 2 Power (MW)	HVAC Chiller 3 Power (MW)	Scrubber Power (MW)	Tem
137881	2023-08-28 09:30:00	2023-08-28 09:35:00	Vessel 2	0.1025	0.1235	-0.0400	0.0000	0.0	0.4086	0.0000	
137884	2023-08-28 09:35:00	2023-08-28 09:40:00	Vessel 2	0.1113	0.1194	-0.0400	0.0000	0.0	0.4061	0.0000	
137885	2023-08-28 09:40:00	2023-08-28 09:45:00	Vessel 2	0.0895	0.0986	-0.0400	0.0000	0.0	0.4006	0.0000	
137887	2023-08-28 09:45:00	2023-08-28 09:50:00	Vessel 2	0.1265	0.1295	-0.0400	0.0000	0.0	0.3959	0.0000	
137889	2023-08-28 09:50:00	2023-08-28 09:55:00	Vessel 2	0.1095	0.1345	-0.0400	0.0000	0.0	0.3969	0.0000	
...	...	...	...	...	...	...	...	...	...	...	
153941	2023-09-25 06:45:00	2023-09-25 06:50:00	Vessel 2	0.0281	0.1347	5.8139	0.4105	0.0	0.0000	0.7854	
153943	2023-09-25 06:50:00	2023-09-25 06:55:00	Vessel 2	0.0195	0.1565	5.8844	0.4183	0.0	0.0000	0.7832	
153945	2023-09-25 06:55:00	2023-09-25 07:00:00	Vessel 2	0.0317	0.1817	5.8343	0.4083	0.0	0.0000	0.7865	
153947	2023-09-25 07:00:00	2023-09-25 07:05:00	Vessel 2	0.0166	0.2210	5.9644	0.4213	0.0	0.0000	0.7857	
153949	2023-09-25 07:05:00	2023-09-25 07:10:00	Vessel 2	0.0486	0.2097	5.9657	0.4175	0.0	0.0000	0.7886	

886 rows × 44 columns

In [19]:

```
# Plot the time series for 'Speed Over Ground (knots)'
plt.figure(figsize=(14, 7))
plt.plot(df['Start Time'], df['Speed Over Ground (knots)'], label='Speed Over Ground
plt.xlabel('Time')
plt.ylabel('Speed Over Ground (knots)')
plt.title('Speed Over Ground (knots) Over Time')
plt.legend()
plt.grid(True)
plt.show()
```

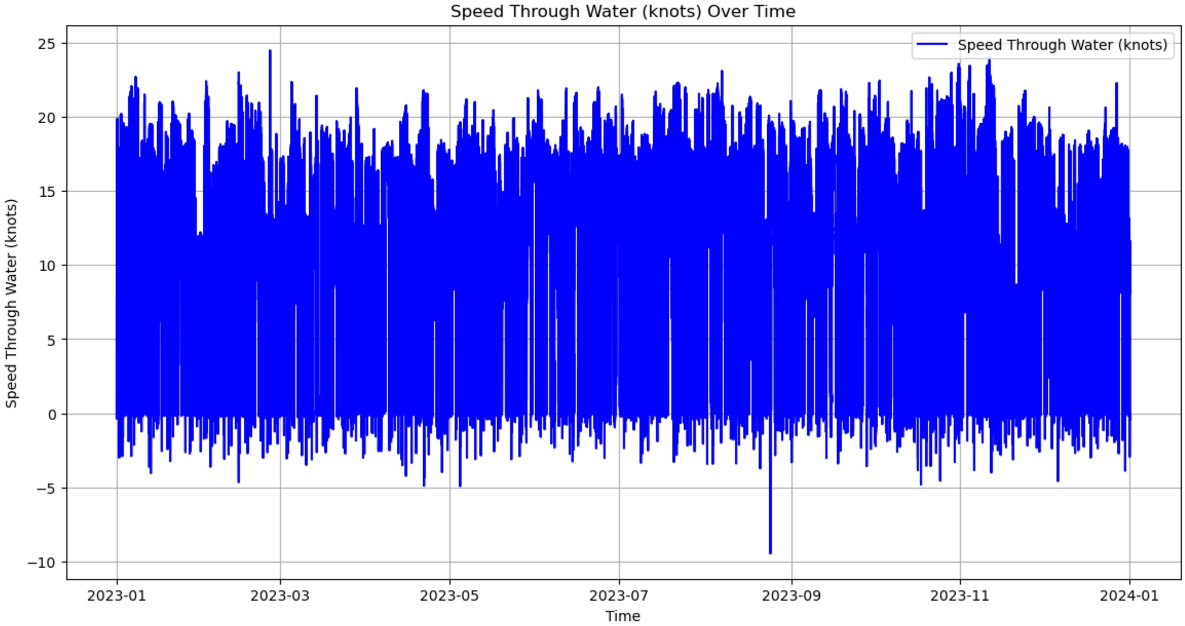


In [20]:

```
# Sort the dataframe by 'Start Time' to ensure proper filling
df = df.sort_values(by='Start Time')

# Set 'Start Time' as the index
df.set_index('Start Time', inplace=True)

# Interpolate to fill missing values
df['Speed Over Ground (knots)'] = df['Speed Over Ground (knots)'].interpolate()

# If you need to reset the index back to columns
df.reset_index(inplace=True)
```

In [21]:

```
# Set 'Start Time' as the index
df.set_index('Start Time', inplace=True)

# Interpolate to fill missing values
df['Speed Through Water (knots)'] = df['Speed Through Water (knots)'].interpolate()

# If you need to reset the index back to columns
df.reset_index(inplace=True)

# Plot the time series for 'Speed Through Water (knots)'
plt.figure(figsize=(14, 7))
plt.plot(df['Start Time'], df['Speed Through Water (knots)'], label='Speed Through W
plt.xlabel('Time')
plt.ylabel('Speed Through Water (knots)')
plt.title('Speed Through Water (knots) Over Time')
plt.legend()
plt.grid(True)
plt.show()
```



In [22]:

```
# Set 'Start Time' as the index
df.set_index('Start Time', inplace=True)

# Interpolate to fill missing values
df['Speed Through Water (knots)'] = df['Speed Through Water (knots)'].interpolate()

# If you need to reset the index back to columns
df.reset_index(inplace=True)
```

In [23]:

```
# # missing value check
df.isnull().sum()
```

Out[23]:

```
Start Time                                  0
End Time                                    0
Vessel Name                                 0
Power Galley 1 (MW)                         0
Power Galley 2 (MW)                         0
Power Service (MW)                          0
HVAC Chiller 1 Power (MW)                   0
HVAC Chiller 2 Power (MW)                   0
HVAC Chiller 3 Power (MW)                   0
Scrubber Power (MW)                         0
Sea Temperature (Celsius)                   0
Boiler 1 Fuel Flow Rate (L/h)               0
Boiler 2 Fuel Flow Rate (L/h)               0
Incinerator 1 Fuel Flow Rate (L/h)          0
Diesel Generator 1 Power (MW)               0
Diesel Generator 2 Power (MW)               0
Diesel Generator 3 Power (MW)               0
Diesel Generator 4 Power (MW)               0
Latitude (Degrees)                        326
Longitude (Degrees)                       326
Relative Wind Angle (Degrees)               0
True Wind Angle (Degrees)                  60
Depth (m)                               57479
Relative Wind Direction (Degrees)          41
True Wind Direction (Degrees)              60
Draft (m)                                1127
Speed Over Ground (knots)                   0
True Wind Speed (knots)                    60
Relative Wind Speed (knots)                 0
Speed Through Water (knots)                 0
Local Time (h)                            326
Trim (m)                                 1063
Propulsion Power (MW)                       0
Port Side Propulsion Power (MW)             0
Starboard Side Propulsion Power (MW)        0
Bow Thruster 1 Power (MW)                   0
Bow Thruster 2 Power (MW)                   0
Bow Thruster 3 Power (MW)                   0
Stern Thruster 1 Power (MW)                 0
Stern Thruster 2 Power (MW)                 0
Main Engine 1 Fuel Flow Rate (kg/h)         0
Main Engine 2 Fuel Flow Rate (kg/h)         0
Main Engine 3 Fuel Flow Rate (kg/h)         0
Main Engine 4 Fuel Flow Rate (kg/h)         0
dtype: int64
```

Sorting Vessels data to Vessel 1 & Vessel 2

In [24]:

```
# Assuming 'Vessel Name' is the column indicating the vessel
vessel_1_data = df[df['Vessel Name'] == 'Vessel 1']
vessel_2_data = df[df['Vessel Name'] == 'Vessel 2']

# Verify the split
print(vessel_1_data.shape)
print(vessel_2_data.shape)
```

```
(105119, 44)
(105105, 44)
```

In [25]:

```
df.columns
```

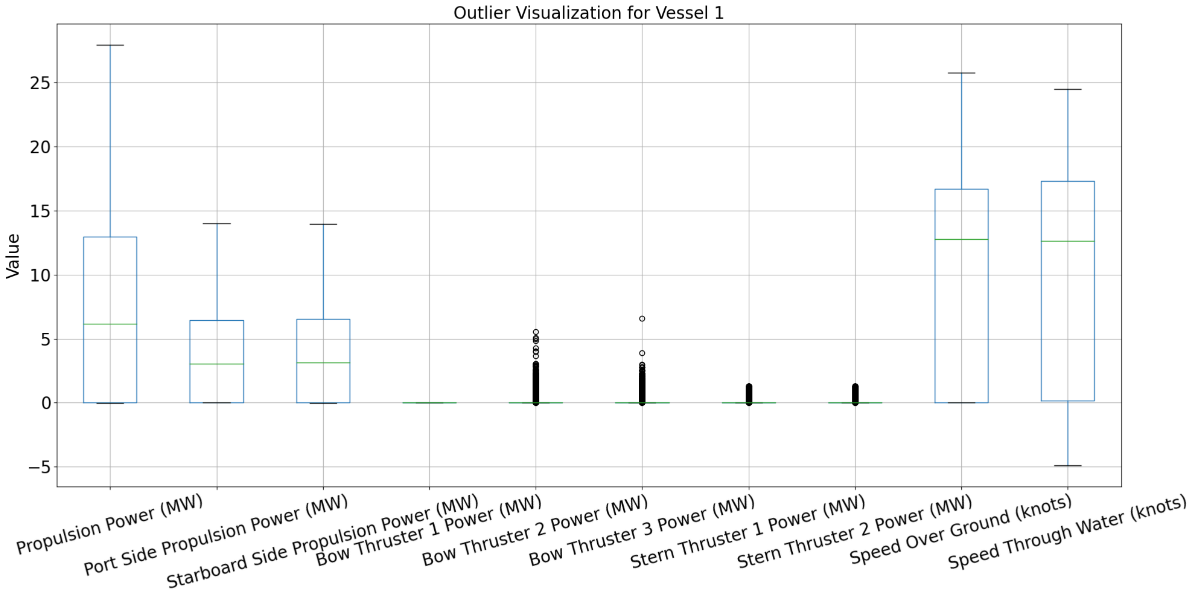
Out[25]:

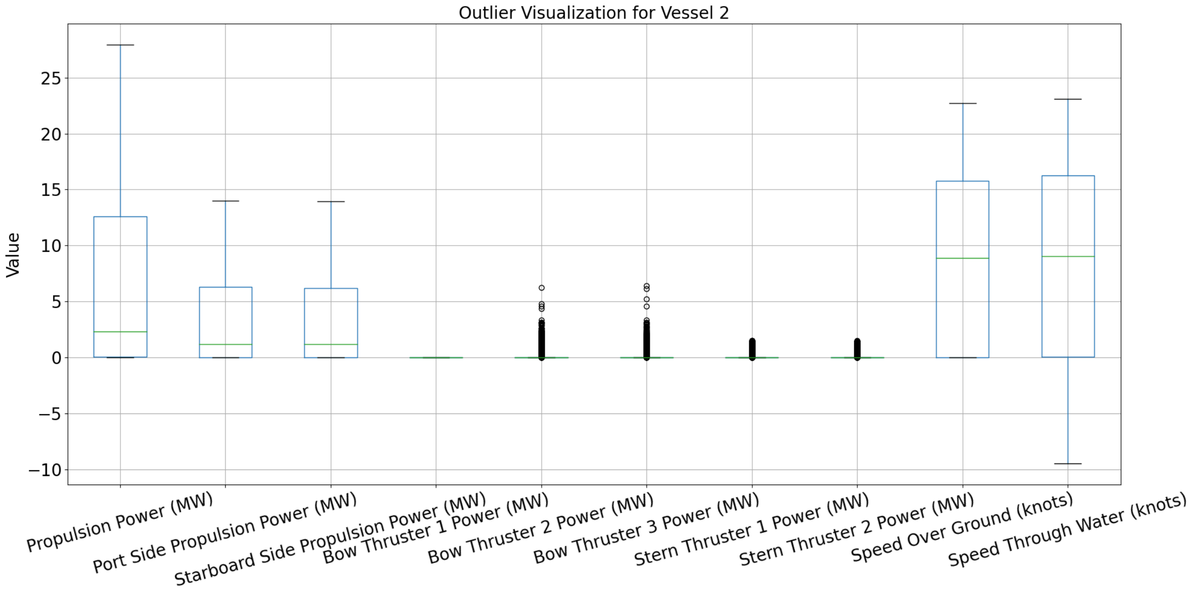
```
Index(['Start Time', 'End Time', 'Vessel Name', 'Power Galley 1 (MW)',
       'Power Galley 2 (MW)', 'Power Service (MW)',
       'HVAC Chiller 1 Power (MW)', 'HVAC Chiller 2 Power (MW)',
       'HVAC Chiller 3 Power (MW)', 'Scrubber Power (MW)',
       'Sea Temperature (Celsius)', 'Boiler 1 Fuel Flow Rate (L/h)',
       'Boiler 2 Fuel Flow Rate (L/h)', 'Incinerator 1 Fuel Flow Rate (L/h)',
       'Diesel Generator 1 Power (MW)', 'Diesel Generator 2 Power (MW)',
       'Diesel Generator 3 Power (MW)', 'Diesel Generator 4 Power (MW)',
       'Latitude (Degrees)', 'Longitude (Degrees)',
       'Relative Wind Angle (Degrees)', 'True Wind Angle (Degrees)',
       'Depth (m)', 'Relative Wind Direction (Degrees)',
       'True Wind Direction (Degrees)', 'Draft (m)',
       'Speed Over Ground (knots)', 'True Wind Speed (knots)',
       'Relative Wind Speed (knots)', 'Speed Through Water (knots)',
       'Local Time (h)', 'Trim (m)', 'Propulsion Power (MW)',
       'Port Side Propulsion Power (MW)',
       'Starboard Side Propulsion Power (MW)', 'Bow Thruster 1 Power (MW)',
       'Bow Thruster 2 Power (MW)', 'Bow Thruster 3 Power (MW)',
       'Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)',
       'Main Engine 1 Fuel Flow Rate (kg/h)',
       'Main Engine 2 Fuel Flow Rate (kg/h)',
       'Main Engine 3 Fuel Flow Rate (kg/h)',
       'Main Engine 4 Fuel Flow Rate (kg/h)'],
      dtype='object')
```

Outlier check of Propulsion Analysis

In [26]:

```python
# Define efficiency columns
propulsion_columns = [
     'Propulsion Power (MW)', 'Port Side Propulsion Power (MW)',
    'Starboard Side Propulsion Power (MW)', 'Bow Thruster 1 Power (MW)',
    'Bow Thruster 2 Power (MW)', 'Bow Thruster 3 Power (MW)',
    'Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)',
    'Speed Over Ground (knots)', 'Speed Through Water (knots)'
]

# Assuming 'Vessel Name' is the column indicating the vessel
vessel_1_data = df[df['Vessel Name'] == 'Vessel 1'].copy()
vessel_2_data = df[df['Vessel Name'] == 'Vessel 2'].copy()

# Function to create box plots for outliers visualization
def plot_outliers(df, columns, vessel_name):
    plt.figure(figsize=(20, 10))
    df[columns].boxplot()
    plt.title(f'Outlier Visualization for {vessel_name}', fontsize=20)
    plt.ylabel('Value', fontsize=20)
    plt.xticks(rotation=15, fontsize=20)  # Change rotation to 0 for horizontal labe
    plt.yticks(fontsize=20)
    plt.grid(True)
    plt.tight_layout()  # Adjust layout to make room for the labels
    plt.show()

# Plot outliers for Vessel 1
plot_outliers(vessel_1_data, propulsion_columns, 'Vessel 1')

# Plot outliers for Vessel 2
plot_outliers(vessel_2_data, propulsion_columns, 'Vessel 2')
```

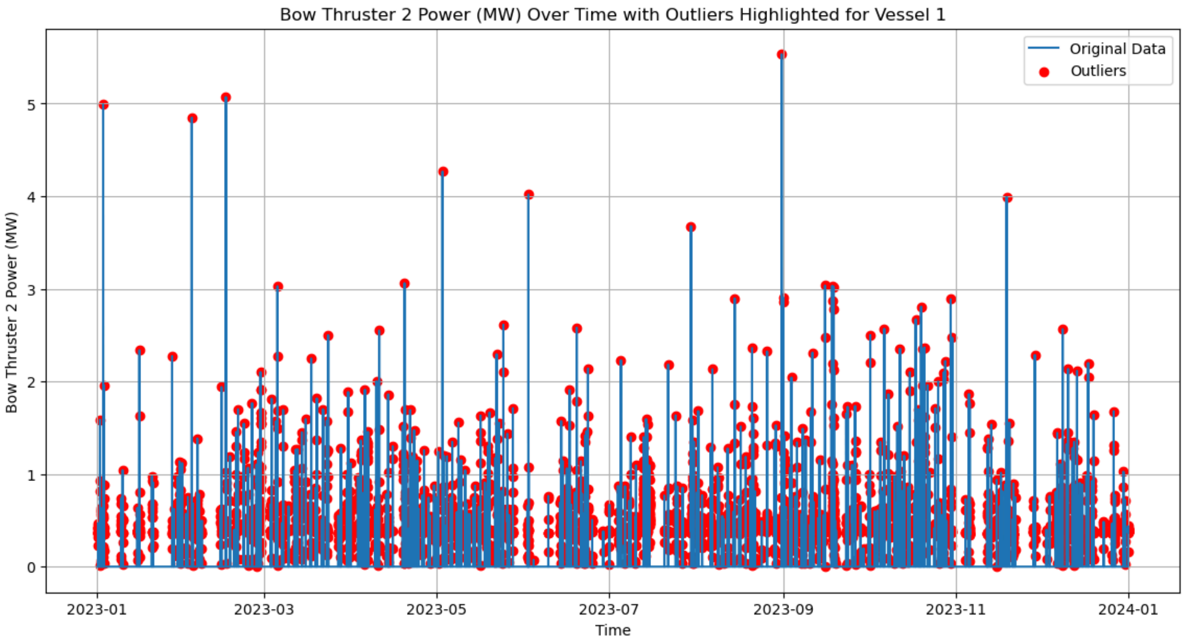




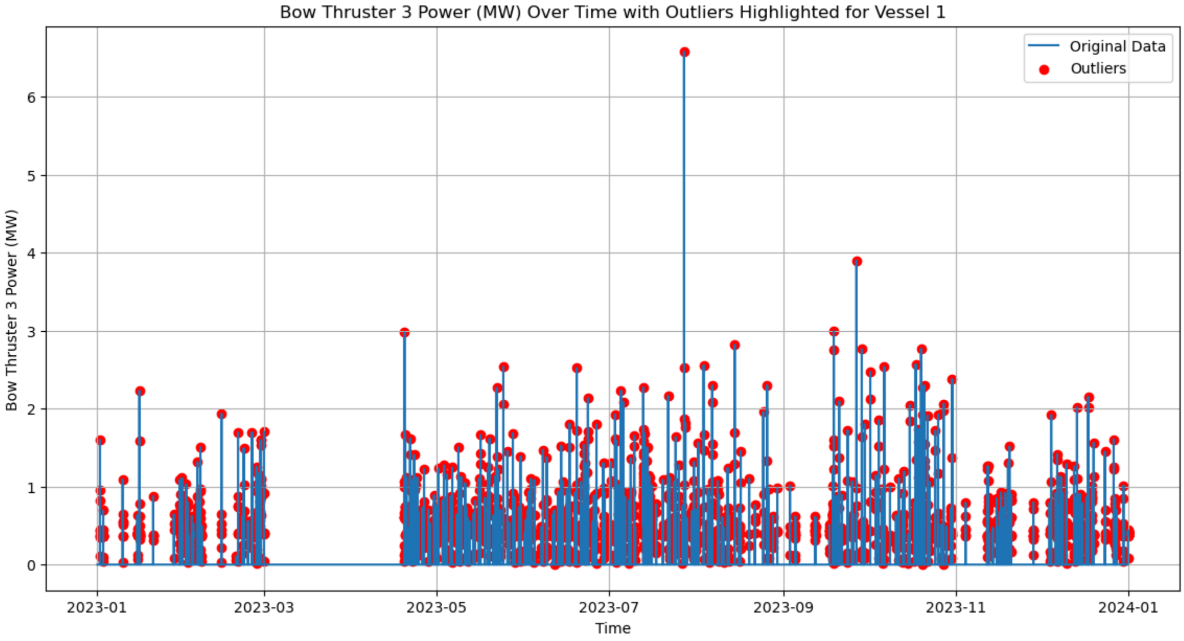
Each outlier check

In [27]:

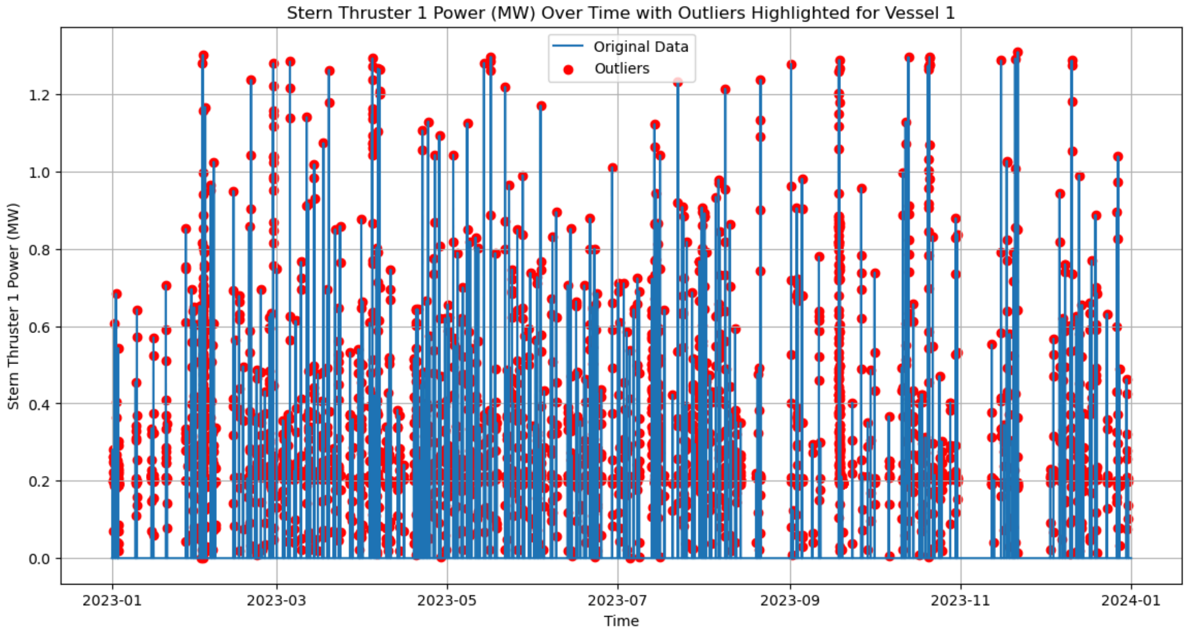
```
import numpy as np
import matplotlib.pyplot as plt

# Ensure 'Start Time' is in datetime format and set as index
df['Start Time'] = pd.to_datetime(df['Start Time'])
vessel_1_data = df[df['Vessel Name'] == 'Vessel 1'].copy()
vessel_2_data = df[df['Vessel Name'] == 'Vessel 2'].copy()
vessel_1_data.set_index('Start Time', inplace=True)
vessel_2_data.set_index('Start Time', inplace=True)

# Function to detect outliers using IQR method
def detect_outliers(df, column):
    Q1 = df[column].quantile(0.25)
    Q3 = df[column].quantile(0.75)
    IQR = Q3 - Q1
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    outliers = df[(df[column] < lower_bound) | (df[column] > upper_bound)]
    return outliers

# Function to plot original data with outliers highlighted
def plot_with_outliers(df, column, outliers, vessel_name):
    plt.figure(figsize=(14, 7))
    plt.plot(df.index, df[column], label='Original Data')
    plt.scatter(outliers.index, outliers[column], color='red', label='Outliers')
    plt.xlabel('Time')
    plt.ylabel(column)
    plt.title(f'{column} Over Time with Outliers Highlighted for {vessel_name}')
    plt.legend()
    plt.grid(True)
    plt.show()

# Function to calculate and print statistical summaries
def print_summaries(df, column, outliers):
    data_without_outliers = df[~df.index.isin(outliers.index)]
    summary_with_outliers = df[column].describe()
    summary_without_outliers = data_without_outliers[column].describe()

    print(f"Summary with Outliers for {column}:₩n", summary_with_outliers)
    print(f"₩nSummary without Outliers for {column}:₩n", summary_without_outliers)

# Detect outliers, plot, and print summaries for each relevant column in vessel_1_da
columns_to_check = ['Bow Thruster 2 Power (MW)', 'Bow Thruster 3 Power (MW)',
    'Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)']

for column in columns_to_check:
    outliers = detect_outliers(vessel_1_data, column)
    plot_with_outliers(vessel_1_data, column, outliers, 'Vessel 1')
    print_summaries(vessel_1_data, column, outliers)
```



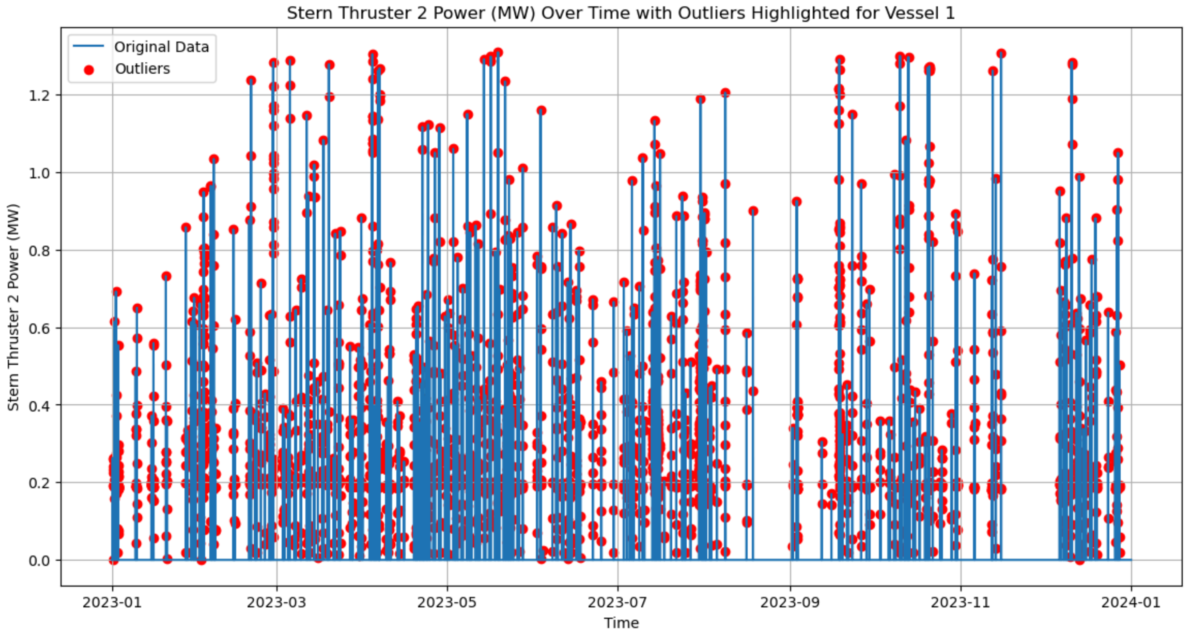
```
Summary with Outliers for Bow Thruster 2 Power (MW):
 count    105119.000000
mean          0.018148
std           0.126050
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           5.533500
Name: Bow Thruster 2 Power (MW), dtype: float64

Summary without Outliers for Bow Thruster 2 Power (MW):
 count    101637.0
mean          0.0
std           0.0
min           0.0
25%           0.0
50%           0.0
75%           0.0
max           0.0
Name: Bow Thruster 2 Power (MW), dtype: float64
```



```
Summary with Outliers for Bow Thruster 3 Power (MW):
 count    105119.000000
mean          0.014163
std           0.106515
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           6.574400
Name: Bow Thruster 3 Power (MW), dtype: float64

Summary without Outliers for Bow Thruster 3 Power (MW):
 count    102326.0
mean          0.0
std           0.0
min           0.0
25%           0.0
50%           0.0
75%           0.0
max           0.0
Name: Bow Thruster 3 Power (MW), dtype: float64
```



```
Summary with Outliers for Stern Thruster 1 Power (MW):
 count    105119.000000
mean          0.009957
std           0.070861
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           1.308500
Name: Stern Thruster 1 Power (MW), dtype: float64

Summary without Outliers for Stern Thruster 1 Power (MW):
 count    101994.0
mean          0.0
std           0.0
min           0.0
25%           0.0
50%           0.0
75%           0.0
max           0.0
Name: Stern Thruster 1 Power (MW), dtype: float64
```



```
Summary with Outliers for Stern Thruster 2 Power (MW):
 count    105119.000000
mean          0.008318
std           0.065344
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max           1.308600
Name: Stern Thruster 2 Power (MW), dtype: float64

Summary without Outliers for Stern Thruster 2 Power (MW):
 count    102466.0
mean          0.0
std           0.0
min           0.0
25%           0.0
50%           0.0
75%           0.0
max           0.0
Name: Stern Thruster 2 Power (MW), dtype: float64
```

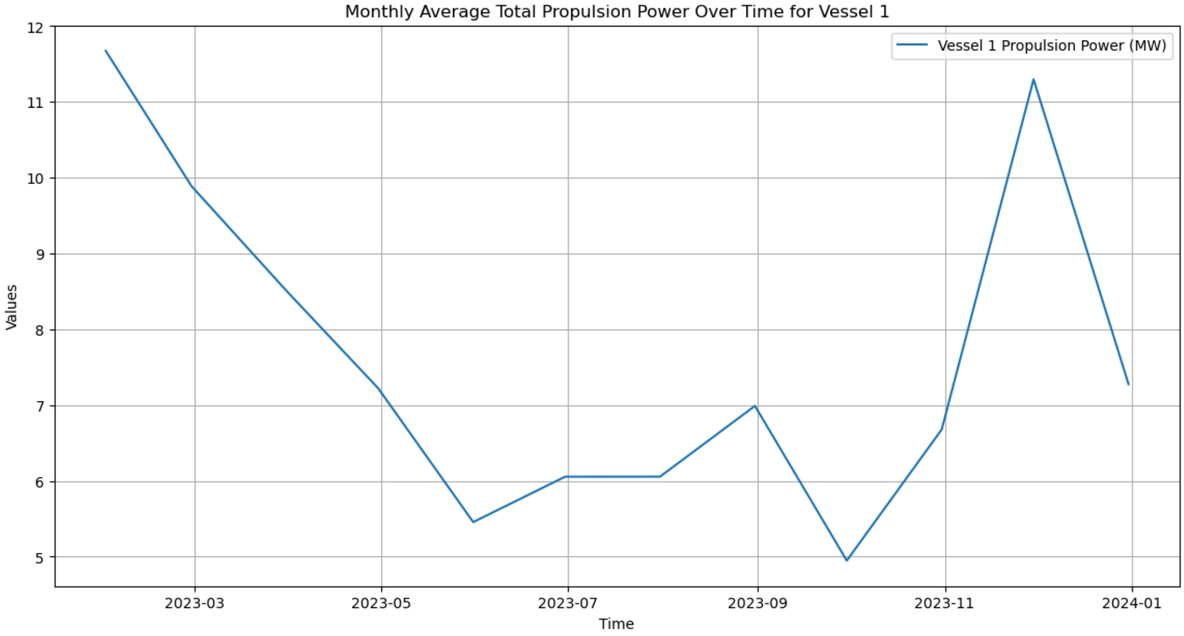
Analysing with outliers becasue outliers consistently shows a distribution

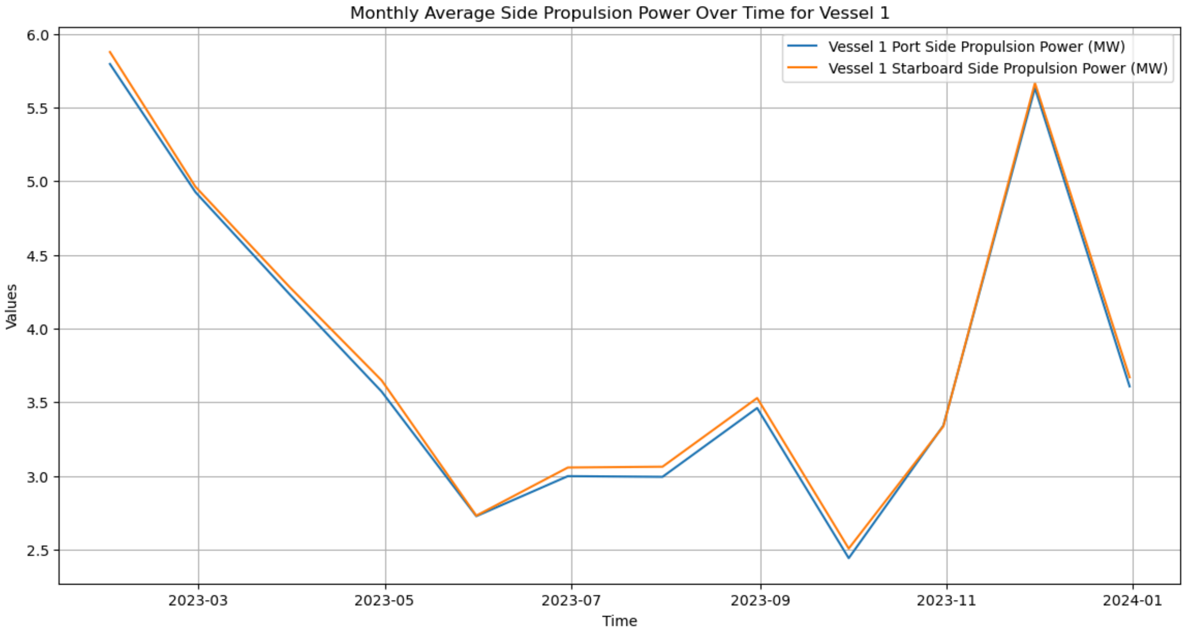
Data Analysis

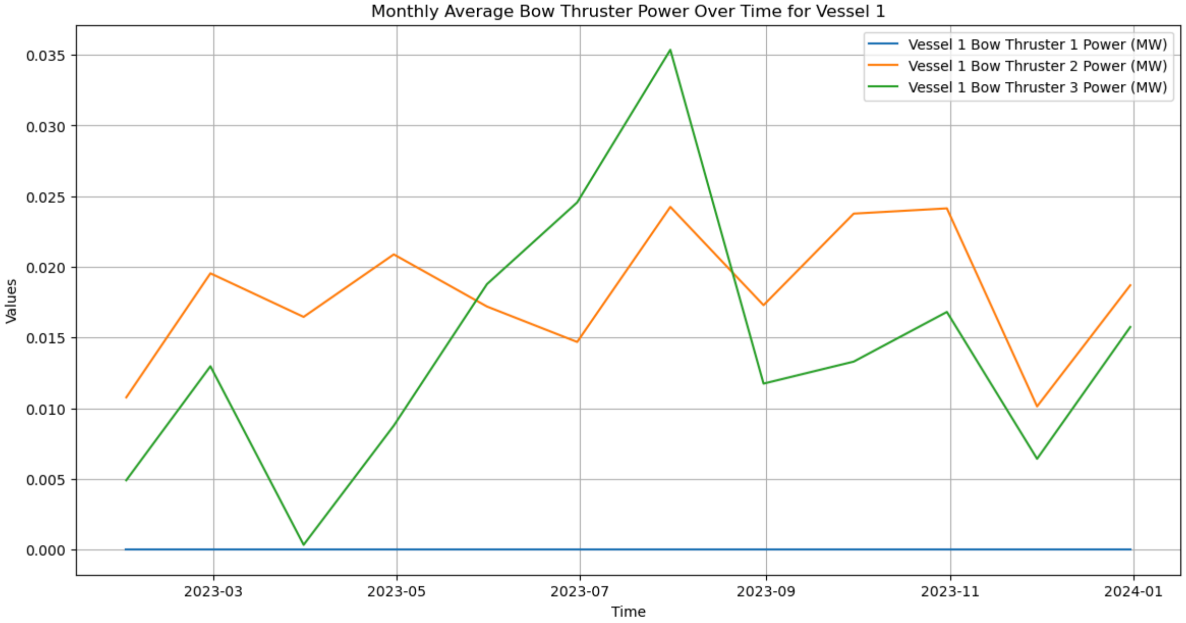
In [28]:

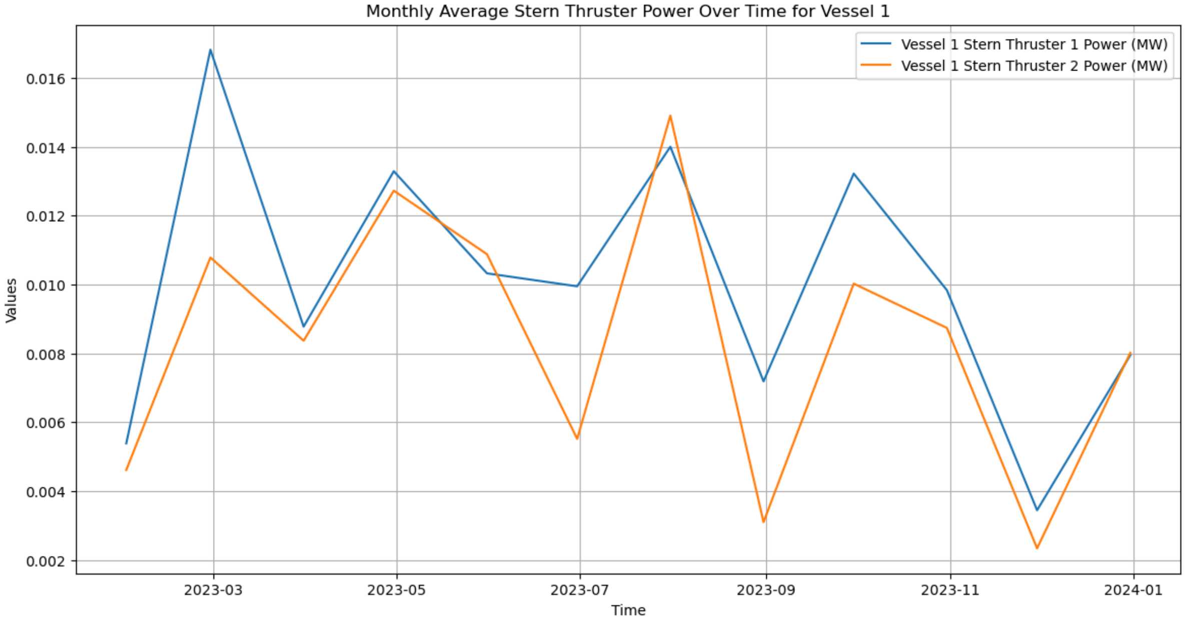
```
# Define categories
total_propulsion = ['Propulsion Power (MW)']
side_propulsion = ['Port Side Propulsion Power (MW)', 'Starboard Side Propulsion Pow
bow_thrusters = ['Bow Thruster 1 Power (MW)', 'Bow Thruster 2 Power (MW)', 'Bow Thru
stern_thrusters = ['Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)']
speed_metrics = ['Speed Over Ground (knots)', 'Speed Through Water (knots)']

# Combine all columns into a single list
propulsion_columns = total_propulsion + side_propulsion + bow_thrusters + stern_thru

# Ensure all columns are numeric
df[propulsion_columns] = df[propulsion_columns].apply(pd.to_numeric, errors='coerce'

# Assuming 'Vessel Name' is the column indicating the vessel
vessel_1_data = df[df['Vessel Name'] == 'Vessel 1'].copy()
vessel_2_data = df[df['Vessel Name'] == 'Vessel 2'].copy()

# Calculate monthly averages for Vessel 1
vessel_1_data.set_index('Start Time', inplace=True)
vessel_1_monthly_avg = vessel_1_data[propulsion_columns].resample('M').mean()

# Calculate monthly averages for Vessel 2
vessel_2_data.set_index('Start Time', inplace=True)
vessel_2_monthly_avg = vessel_2_data[propulsion_columns].resample('M').mean()

# Function to plot monthly averages for Vessel 1
def plot_monthly_averages_vessel_1(columns, title):
    plt.figure(figsize=(14, 7))
    for column in columns:
        plt.plot(vessel_1_monthly_avg.index, vessel_1_monthly_avg[column], label=f'V
    plt.xlabel('Time')
    plt.ylabel('Values')
    plt.title(f'Monthly Average {title} Over Time for Vessel 1')
    plt.legend()
    plt.grid(True)
    plt.show()

# Function to plot monthly averages for Vessel 2
def plot_monthly_averages_vessel_2(columns, title):
    plt.figure(figsize=(14, 7))
    for column in columns:
        plt.plot(vessel_2_monthly_avg.index, vessel_2_monthly_avg[column], label=f'V
    plt.xlabel('Time')
    plt.ylabel('Values')
    plt.title(f'Monthly Average {title} Over Time for Vessel 2')
    plt.legend()
    plt.grid(True)
    plt.show()

# Plotting for each category for Vessel 1
plot_monthly_averages_vessel_1(total_propulsion, 'Total Propulsion Power')
plot_monthly_averages_vessel_1(side_propulsion, 'Side Propulsion Power')
plot_monthly_averages_vessel_1(bow_thrusters, 'Bow Thruster Power')
plot_monthly_averages_vessel_1(stern_thrusters, 'Stern Thruster Power')
plot_monthly_averages_vessel_1(speed_metrics, 'Speed Metrics')

# Plotting for each category for Vessel 2
plot_monthly_averages_vessel_2(total_propulsion, 'Total Propulsion Power')
plot_monthly_averages_vessel_2(side_propulsion, 'Side Propulsion Power')
plot_monthly_averages_vessel_2(bow_thrusters, 'Bow Thruster Power')
plot_monthly_averages_vessel_2(stern_thrusters, 'Stern Thruster Power')
plot_monthly_averages_vessel_2(speed_metrics, 'Speed Metrics')
```

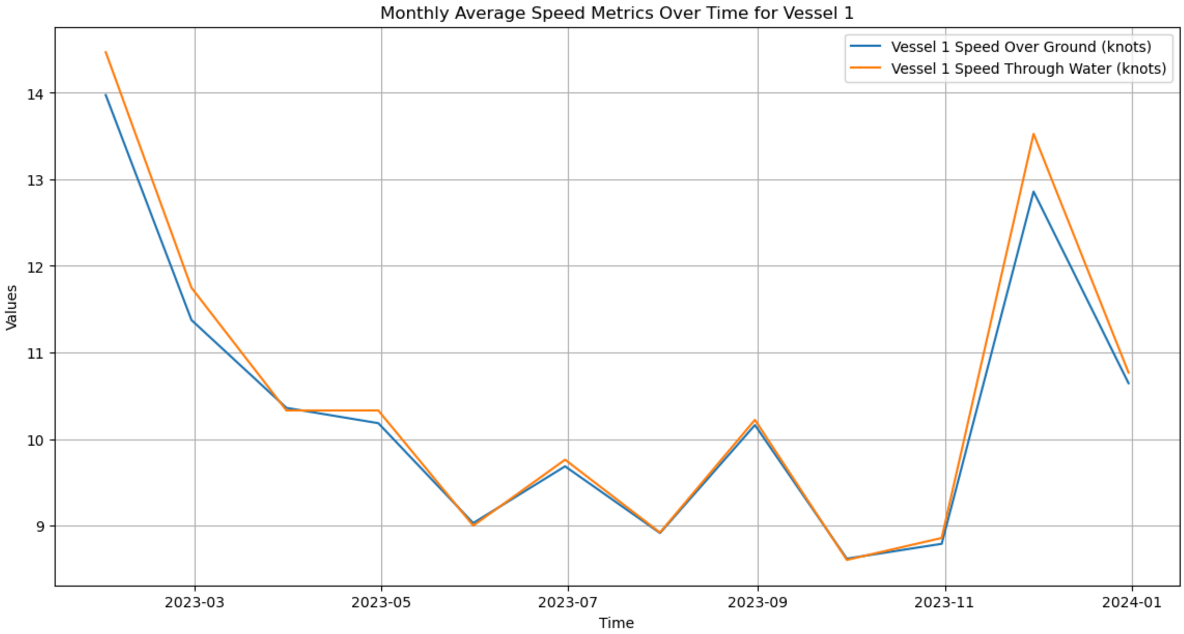
```
C:₩Users₩seoin₩AppData₩Local₩Temp₩ipykernel_16872₩142334750.py:20: FutureWarning:
'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  vessel_1_monthly_avg = vessel_1_data[propulsion_columns].resample('M').mean()
C:₩Users₩seoin₩AppData₩Local₩Temp₩ipykernel_16872₩142334750.py:24: FutureWarning:
'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  vessel_2_monthly_avg = vessel_2_data[propulsion_columns].resample('M').mean()
```

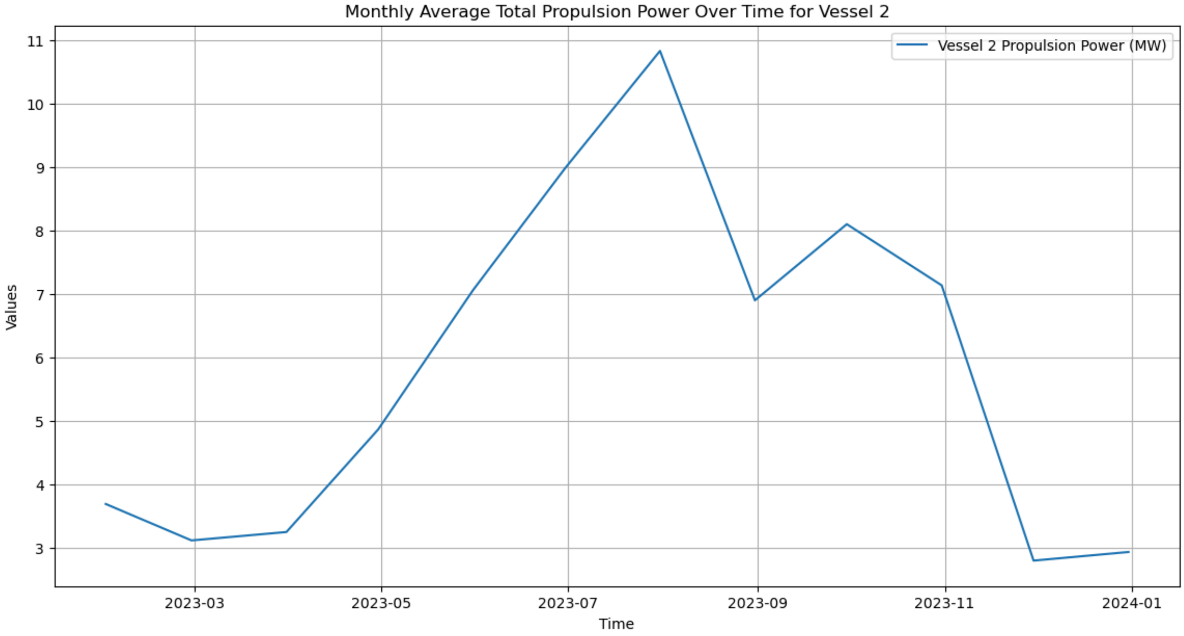


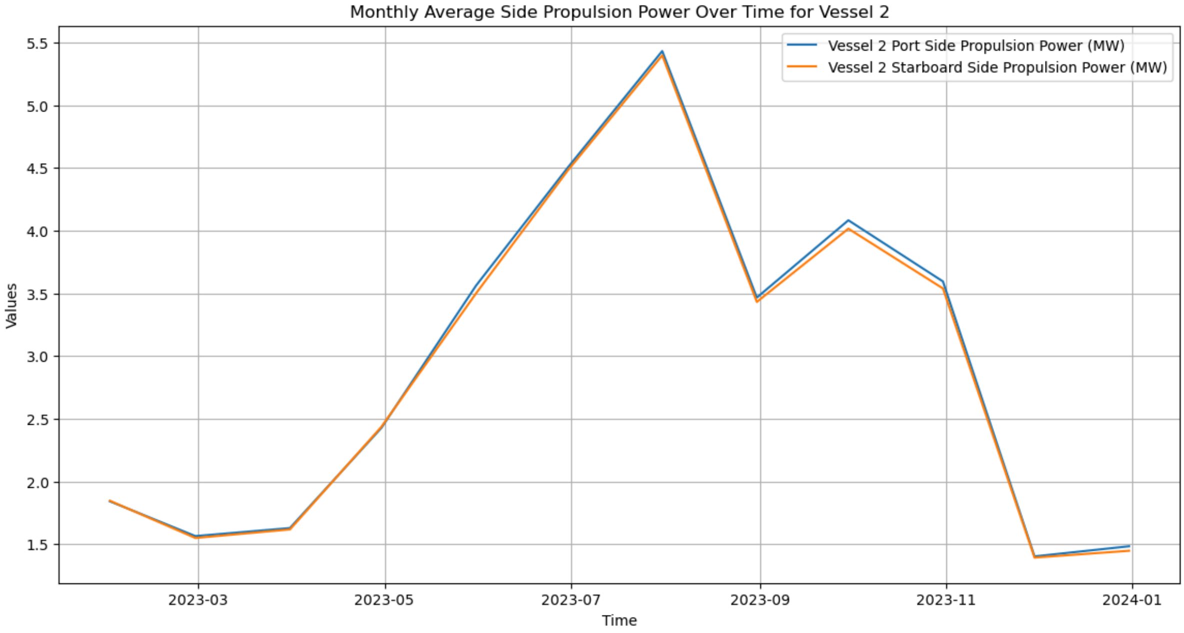

Monthly Average Side Propulsion Power Over Time for Vessel 1

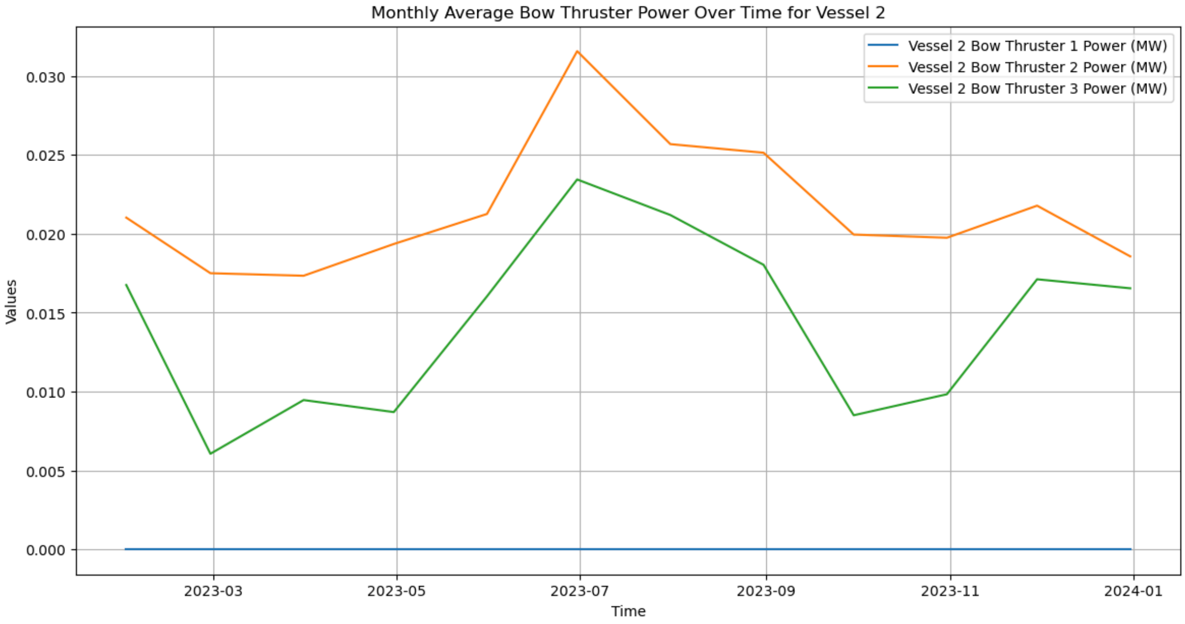

Monthly Average Bow Thruster Power Over Time for Vessel 1

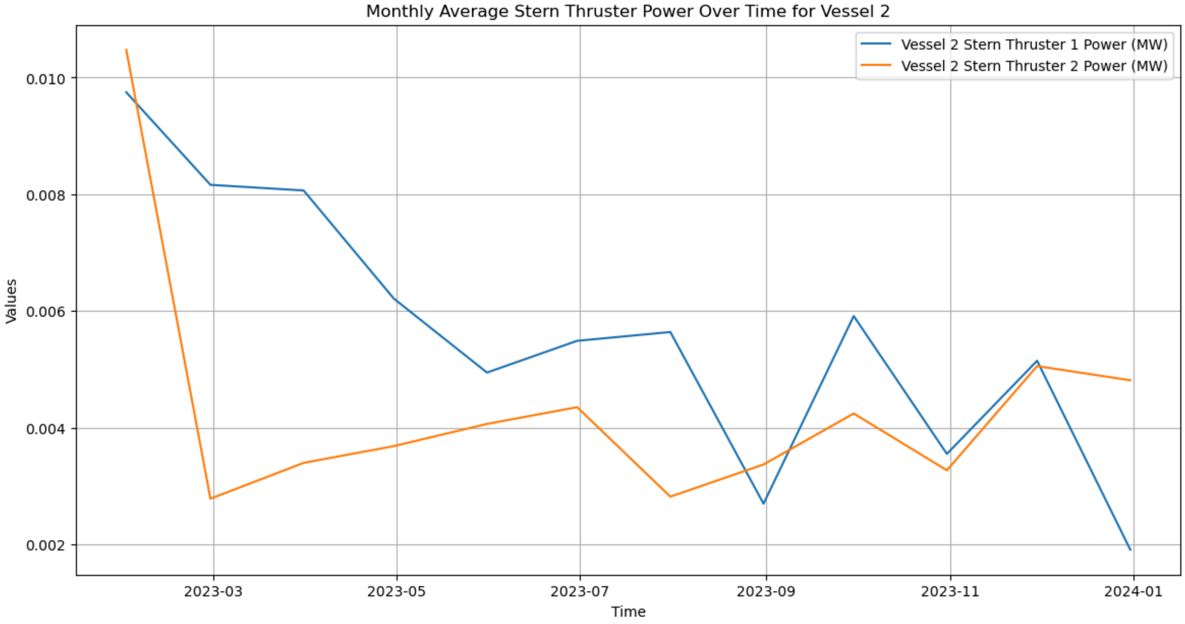

Monthly Average Stern Thruster Power Over Time for Vessel 1

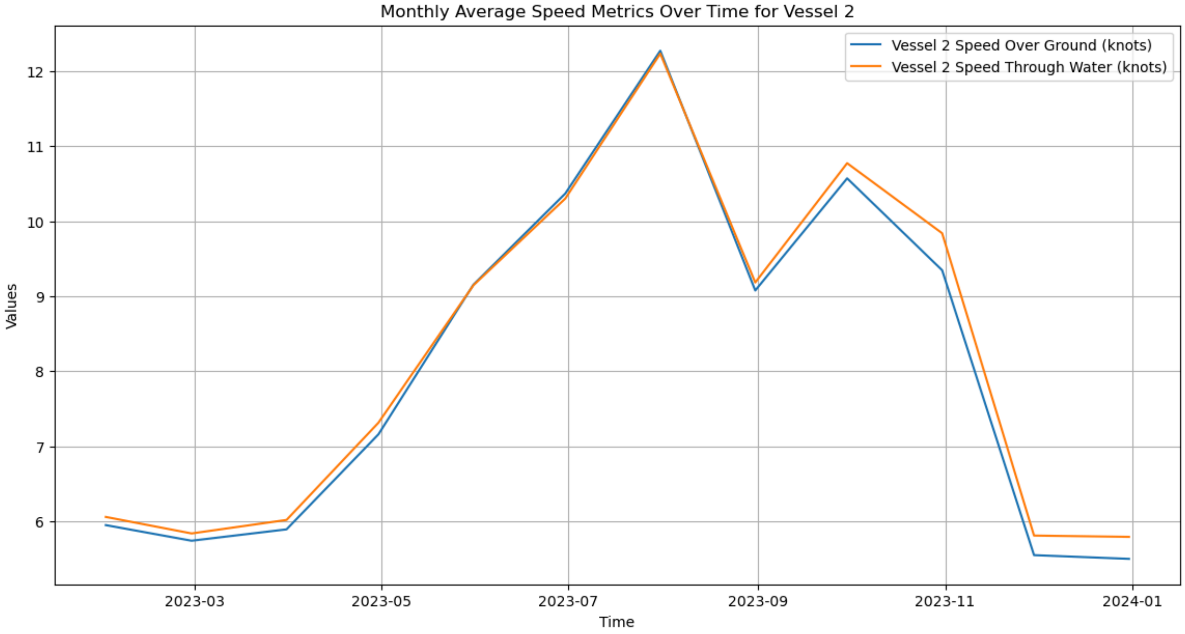

Monthly Average Speed Metrics Over Time for Vessel 1
Vessel 1 Speed Over Ground (knots)
Vessel 1 Speed Through Water (knots)
Values
14
13
12
11
10
9
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time


Monthly Average Total Propulsion Power Over Time for Vessel 2
Vessel 2 Propulsion Power (MW)
Values
11
10
9
8
7
6
5
4
3
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time


Monthly Average Side Propulsion Power Over Time for Vessel 2
Vessel 2 Port Side Propulsion Power (MW)
Vessel 2 Starboard Side Propulsion Power (MW)
Values
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
2023-03
2023-05
2023-07
2023-09
2023-11
2024-01
Time

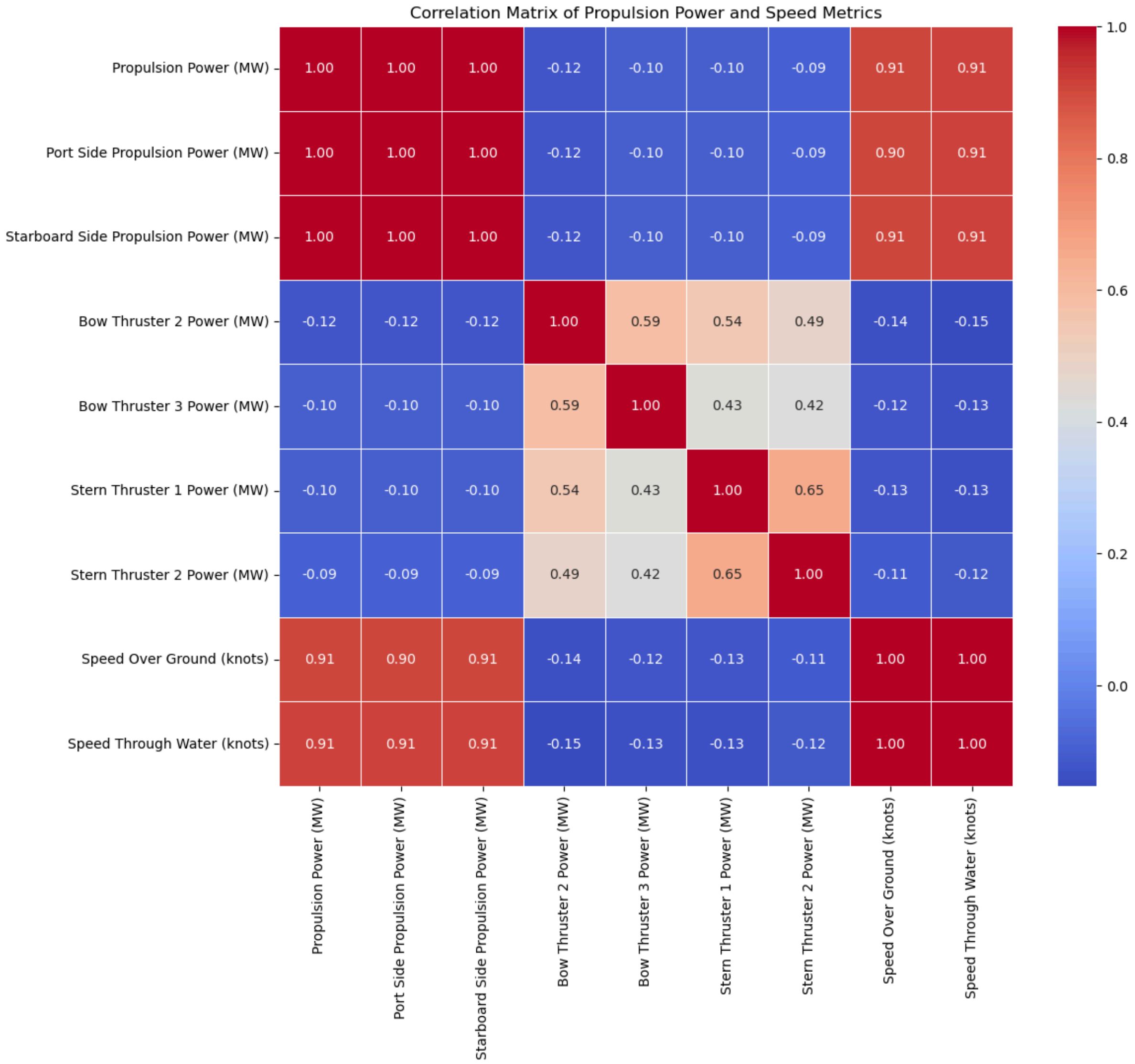






In [29]:

```
# Define the columns for correlation analysis
propulsion_columns = [
    'Propulsion Power (MW)', 'Port Side Propulsion Power (MW)',
    'Starboard Side Propulsion Power (MW)',
    'Bow Thruster 2 Power (MW)', 'Bow Thruster 3 Power (MW)',
    'Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)',
    'Speed Over Ground (knots)', 'Speed Through Water (knots)'
]

# Calculate the correlation matrix
correlation_matrix = df[propulsion_columns].corr()

# Plot the heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f', linewidths=
plt.title('Correlation Matrix of Propulsion Power and Speed Metrics')
plt.show()
```



Correlation Matrix of Propulsion Power and Speed Metrics

	Propulsion Power (MW)	Port Side Propulsion Power (MW)	Starboard Side Propulsion Power (MW)	Bow Thruster 2 Power (MW)	Bow Thruster 3 Power (MW)	Stern Thruster 1 Power (MW)	Stern Thruster 2 Power (MW)	Speed Over Ground (knots)	Speed Through Water (knots)
Propulsion Power (MW)	1.00	1.00	1.00	-0.12	-0.10	-0.10	-0.09	0.91	0.91
Port Side Propulsion Power (MW)	1.00	1.00	1.00	-0.12	-0.10	-0.10	-0.09	0.90	0.91
Starboard Side Propulsion Power (MW)	1.00	1.00	1.00	-0.12	-0.10	-0.10	-0.09	0.91	0.91
Bow Thruster 2 Power (MW)	-0.12	-0.12	-0.12	1.00	0.59	0.54	0.49	-0.14	-0.15
Bow Thruster 3 Power (MW)	-0.10	-0.10	-0.10	0.59	1.00	0.43	0.42	-0.12	-0.13
Stern Thruster 1 Power (MW)	-0.10	-0.10	-0.10	0.54	0.43	1.00	0.65	-0.13	-0.13
Stern Thruster 2 Power (MW)	-0.09	-0.09	-0.09	0.49	0.42	0.65	1.00	-0.11	-0.12
Speed Over Ground (knots)	0.91	0.90	0.91	-0.14	-0.12	-0.13	-0.11	1.00	1.00
Speed Through Water (knots)	0.91	0.91	0.91	-0.15	-0.13	-0.13	-0.12	1.00	1.00

In [32]:

```
# Define categories for propulsion columns
total_propulsion = ['Propulsion Power (MW)']
side_propulsion = ['Port Side Propulsion Power (MW)', 'Starboard Side Propulsion Pow
bow_thrusters = ['Bow Thruster 1 Power (MW)', 'Bow Thruster 2 Power (MW)', 'Bow Thru
stern_thrusters = ['Stern Thruster 1 Power (MW)', 'Stern Thruster 2 Power (MW)']
speed_metrics = ['Speed Over Ground (knots)', 'Speed Through Water (knots)']

# Combine all columns into a single list for propulsion analysis
propulsion_columns = total_propulsion + side_propulsion + bow_thrusters + stern_thru

# Assuming 'Vessel Name' is the column indicating the vessel
vessel_1_data = df[df['Vessel Name'] == 'Vessel 1'].copy()
vessel_2_data = df[df['Vessel Name'] == 'Vessel 2'].copy()
```

```
# Convert index to datetime if it's not already
vessel_1_data.index = pd.to_datetime(vessel_1_data.index)
vessel_2_data.index = pd.to_datetime(vessel_2_data.index)

# Calculate monthly averages for Vessel 1
vessel_1_monthly_avg = vessel_1_data[propulsion_columns].resample('M').mean()

# Calculate monthly averages for Vessel 2
vessel_2_monthly_avg = vessel_2_data[propulsion_columns].resample('M').mean()

# Add 'Vessel' column to identify each vessel
vessel_1_monthly_avg['Vessel'] = 'Vessel 1'
vessel_2_monthly_avg['Vessel'] = 'Vessel 2'

# Combine all results
combined_results = pd.concat([vessel_1_monthly_avg, vessel_2_monthly_avg])

# Store combined results into SQLite database
conn = sqlite3.connect('your_database.db')
combined_results.to_sql('monthly_propulsion_data', conn, if_exists='replace', index=

# Verify data is stored correctly
query = "SELECT * FROM monthly_propulsion_data"
df_sql = pd.read_sql_query(query, conn)
print(df_sql.head())

# Close the connection
conn.close()
```

```
                 index  Propulsion Power (MW)  ₩
0  1970-01-31 00:00:00               7.655666
1  1970-01-31 00:00:00               5.820996

   Port Side Propulsion Power (MW)  Starboard Side Propulsion Power (MW)  ₩
0                         3.802888                              3.852779
1                         2.924577                              2.896418

   Bow Thruster 1 Power (MW)  Bow Thruster 2 Power (MW)  ₩
0                        0.0                   0.018148
1                        0.0                   0.021591

   Bow Thruster 3 Power (MW)  Stern Thruster 1 Power (MW)  ₩
0                   0.014163                     0.009957
1                   0.014371                     0.005601

   Stern Thruster 2 Power (MW)    Vessel
0                     0.008318  Vessel 1
1                     0.004372  Vessel 2
```

```
C:₩Users₩seoin₩AppData₩Local₩Temp₩ipykernel_16872₩3192882156.py:20: FutureWarning:
'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  vessel_1_monthly_avg = vessel_1_data[propulsion_columns].resample('M').mean()
C:₩Users₩seoin₩AppData₩Local₩Temp₩ipykernel_16872₩3192882156.py:23: FutureWarning:
'M' is deprecated and will be removed in a future version, please use 'ME' instead.
  vessel_2_monthly_avg = vessel_2_data[propulsion_columns].resample('M').mean()
```

Performance Analysis Report: Propulsion Power and Speed Metrics

Introduction This report presents an analysis of propulsion power and speed metrics for two vessels over a defined period. The aim is to understand the performance trends, correlations among various power and speed-related parameters, and insights for operational efficiency.

Data Preparation

Data was filtered for two specific vessels. Missing values were handled through interpolation.

Weekly averages were computed for each metric.

The dataset includes the following columns:

- Total Propulsion Power: Propulsion Power (MW)
- Side Propulsion Power: Port Side Propulsion Power (MW) Starboard Side Propulsion Power (MW)
- Thruster Power: Bow Thruster 1 Power (MW) Bow Thruster 2 Power (MW) Bow Thruster 3 Power (MW) Stern Thruster 1 Power (MW) Stern Thruster 2 Power (MW)
- Speed Metrics: Speed Over Ground (knots) Speed Through Water (knots)

Insights for Performance Analysis:

Monitoring Propulsion Power and Speed:

Given the strong correlations, monitoring the total propulsion power will give a good indication of the vessel's speed. This is crucial for assessing the efficiency and performance of the vessel.

Thruster Operations:

Since the thrusters show weak correlations with propulsion power and speed, their use might be more operation-specific, such as docking, undocking, and maneuvering in constrained spaces. This could be further investigated by correlating thruster usage with specific operational events or locations.

Propulsion Power Performance Trend Analysis

This section provides a detailed analysis of the propulsion power performance trends for Vessel 1 and Vessel 2. The analysis covers the following categories:

- Total Propulsion Power

Vessel 1: The total propulsion power shows significant fluctuations over the year, with peaks observed in January, March, and towards the end of the year. Vessel 2: The total propulsion power also fluctuates but shows a more pronounced peak during the summer months (June to August) and a significant peak in November.

- Side Propulsion Power (Port Side and Starboard Side)

Vessel 1: Both port side and starboard side propulsion powers follow a closely synchronized pattern, with peaks and troughs aligned. Vessel 2: Similar synchronization is observed, with increased usage around mid-year and a sharp peak in November.

- Bow Thrusters Power (Bow Thruster 1, 2, 3)

Vessel 1: Bow thruster power usage shows frequent peaks, particularly around March and mid-year. Vessel 2: Bow thruster power usage shows more pronounced fluctuations with higher peaks mid-year and towards the end of the year.

- Stern Thrusters Power (Stern Thruster 1, 2)

Vessel 1: Stern thruster power usage shows noticeable peaks, especially in the first and third quarters, with synchronized usage between the two thrusters. Vessel 2: Similar synchronized usage patterns, with notable decreases in usage towards the end of the year.

- Speed Metrics (Speed Over Ground and Speed Through Water)

Vessel 1: Speed over ground and speed through water follow closely similar trends, with peaks at the beginning and mid-year. Vessel 2: Similar patterns, with synchronized speeds and a sharp peak in November.

Summary

- Seasonal Influence: Both vessels exhibit seasonal trends in power usage and speeds, with higher values generally observed mid-year and specific peaks around November and December.
- Operational Patterns: Synchronized usage of propulsion systems (port, starboard, bow, and stern thrusters) suggests coordinated maneuvers or operational activities.
- Efficiency: Vessel 2 shows a more pronounced peak in total propulsion power mid-year, whereas Vessel 1 shows more fluctuations throughout the year. This could indicate different operational strategies or conditions faced by the vessels.

Correlation Analysis

Correlation with Speed Metrics:

There is a very strong positive correlation between Propulsion Power (MW), Port Side Propulsion Power (MW), Starboard Side Propulsion Power (MW), and both speed metrics (Speed Over Ground (knots) and Speed Through Water (knots)), with coefficients around 0.90 to 0.91. This indicates that as the propulsion power increases, the speed of the vessel, both over ground and through water, also increases significantly.

Weak Correlation with Thrusters:

Bow Thruster 1 Power (MW) has missing values (denoted by -) indicating that there might be no data available for this metric, or it could be a constant value across the dataset.

Bow Thruster 2 Power (MW), Bow Thruster 3 Power (MW), Stern Thruster 1 Power (MW), and Stern Thruster 2 Power (MW) show weak negative correlations with the propulsion power metrics and speed metrics (values ranging from -0.09 to -0.15). This suggests that the thrusters' power does not significantly increase with the propulsion power or speed, which makes sense as thrusters are typically used for maneuvering rather than continuous propulsion.

Conclusion

The analysis of propulsion power performance trends reveals that both Vessel 1 and Vessel 2 exhibit distinct seasonal and operational patterns in their power usage and speeds. Understanding these trends can aid in optimizing operational efficiency, planning maintenance, and developing strategies to manage power usage effectively throughout the year.

```
In [ ]:
```