In [1]:

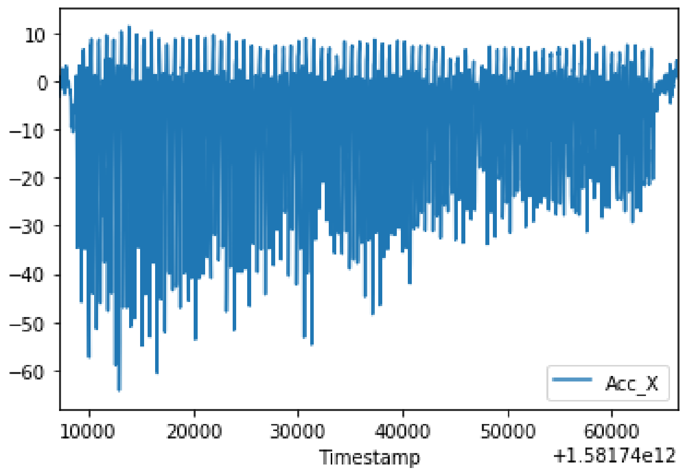
```
# Importing the required libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from datetime import datetime
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import classification_report
```

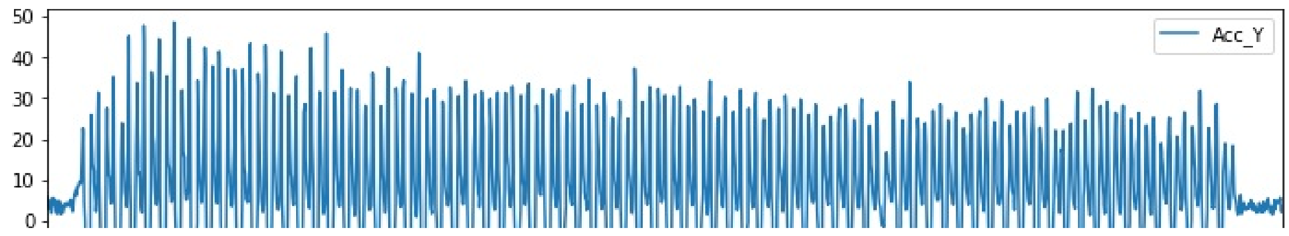
In [2]:

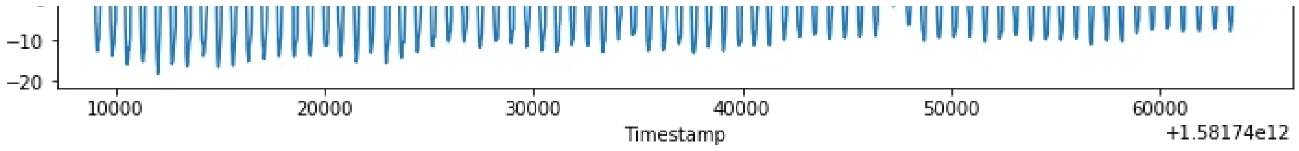
```
# Reading the CSV files
acc_jog = pd.read_csv("Acc_JOGGING.csv")
acc_run = pd.read_csv("Acc_RUNNING.csv")
acc_sit = pd.read_csv("Acc_SITTING.csv")
acc_sup = pd.read_csv("Acc_STAIRCLIMB.csv")
acc_sdn = pd.read_csv("Acc_STAIRDOWN.csv")
acc_wlk = pd.read_csv("Acc_WALKING.csv")

gyr_jog = pd.read_csv("Gyr_JOGGING.csv")
gyr_run = pd.read_csv("Gyr_RUNNING.csv")
gyr_sit = pd.read_csv("Gyr_SITTING.csv")
gyr_sup = pd.read_csv("Gyr_STAIRCLIMB.csv")
gyr_sdn = pd.read_csv("Gyr_STAIRDOWN.csv")
gyr_wlk = pd.read_csv("Gyr_WALKING.csv")
```

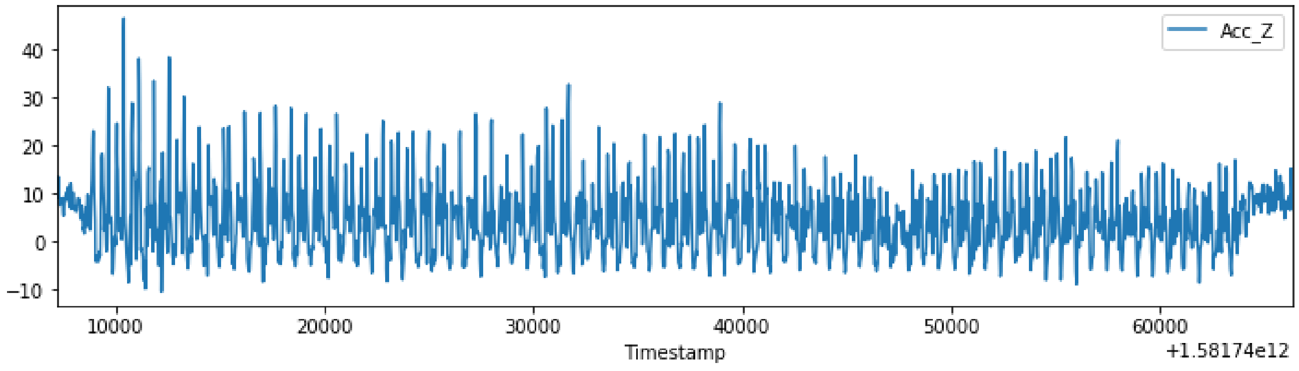
In [3]:

```
# Plotting the time series data
acc_jog.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_jog.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_jog.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```



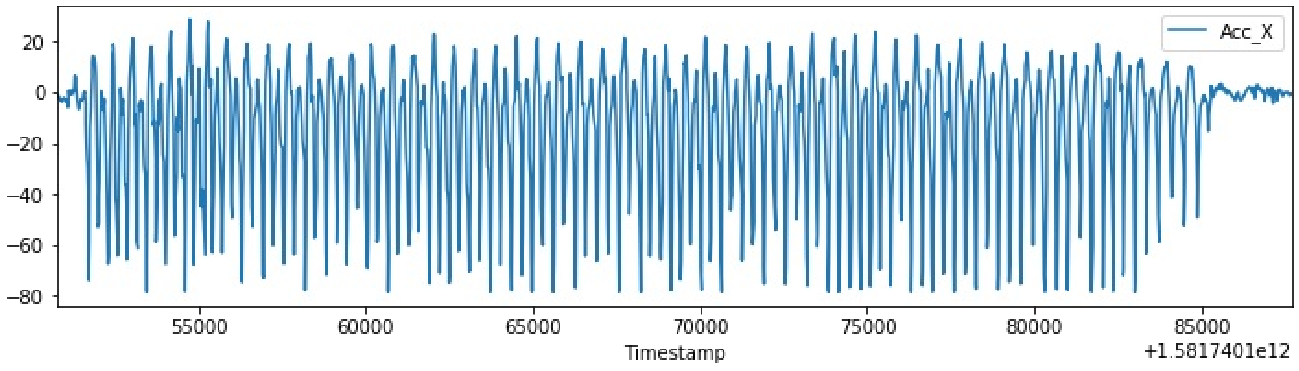


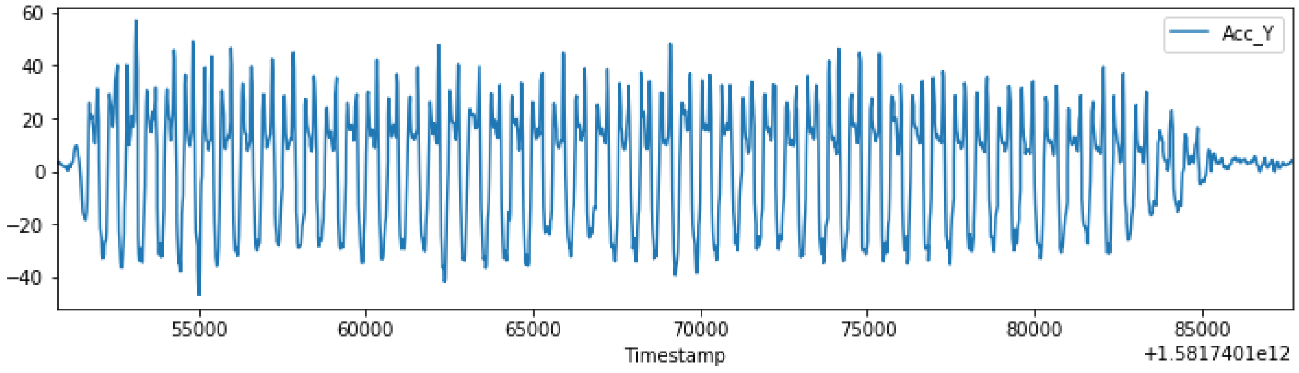


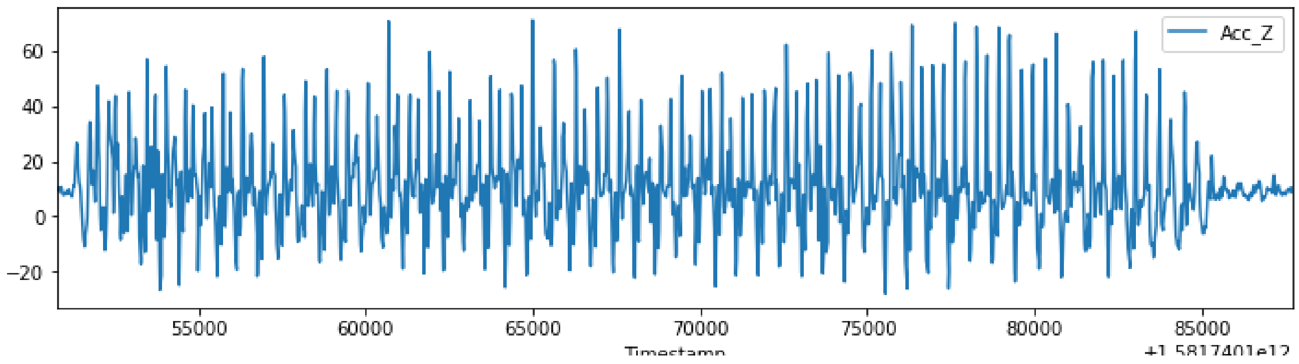


In [4]:

```
# Plotting the time series data
acc_run.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_run.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_run.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

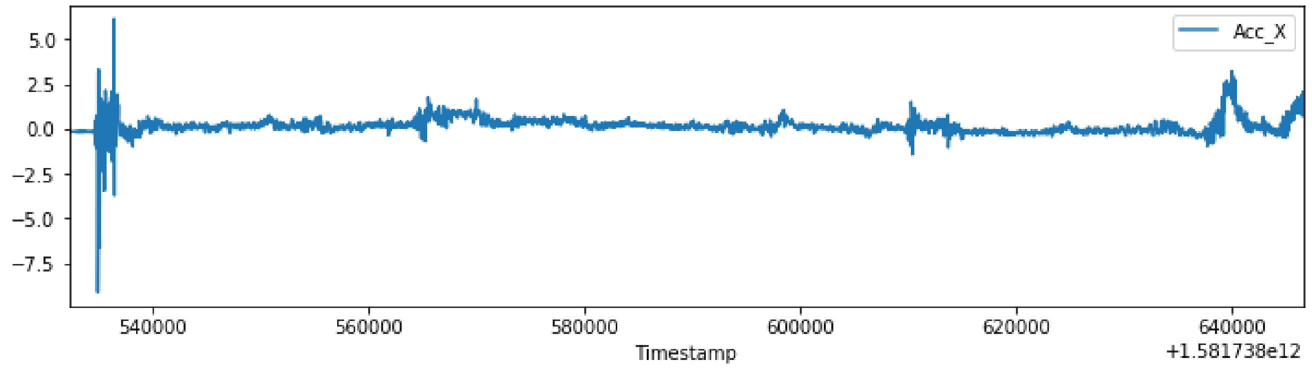


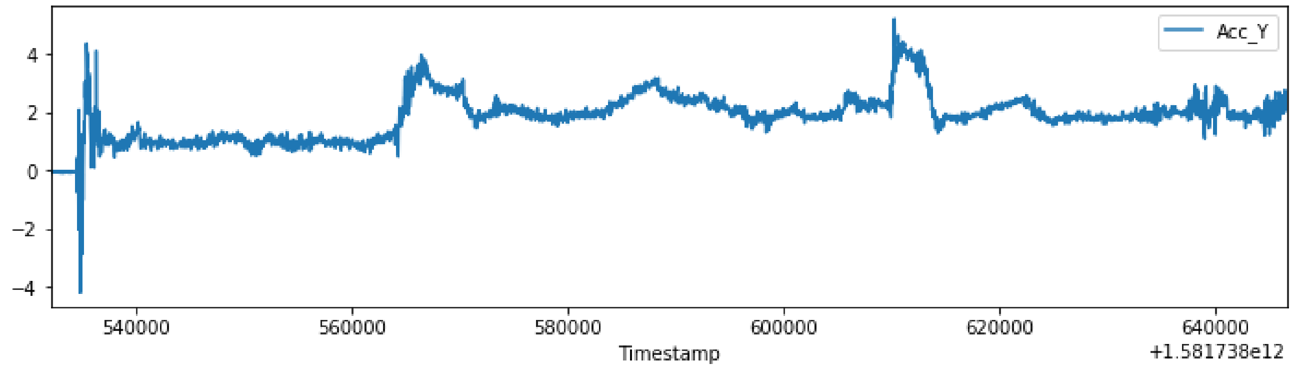


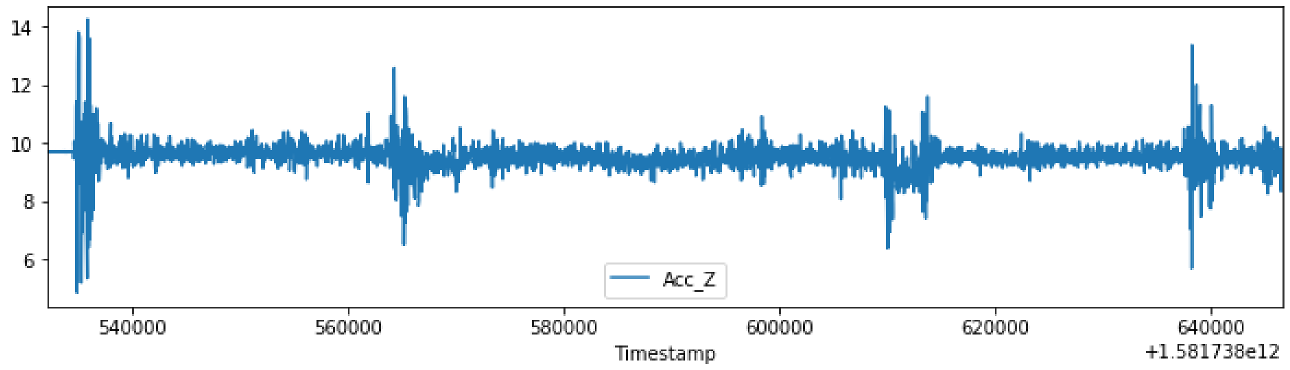


In [5]:

```python
# Plotting the time series data
acc_sit.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sit.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sit.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

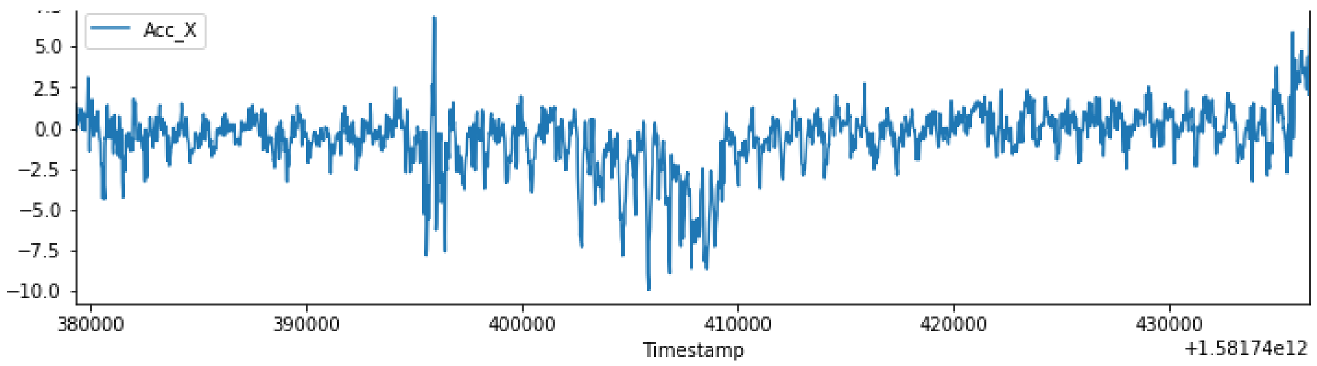


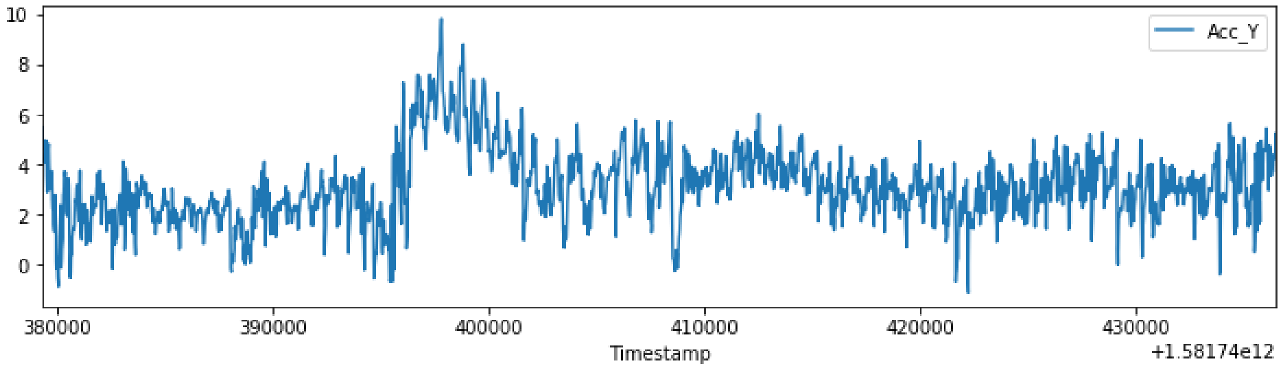


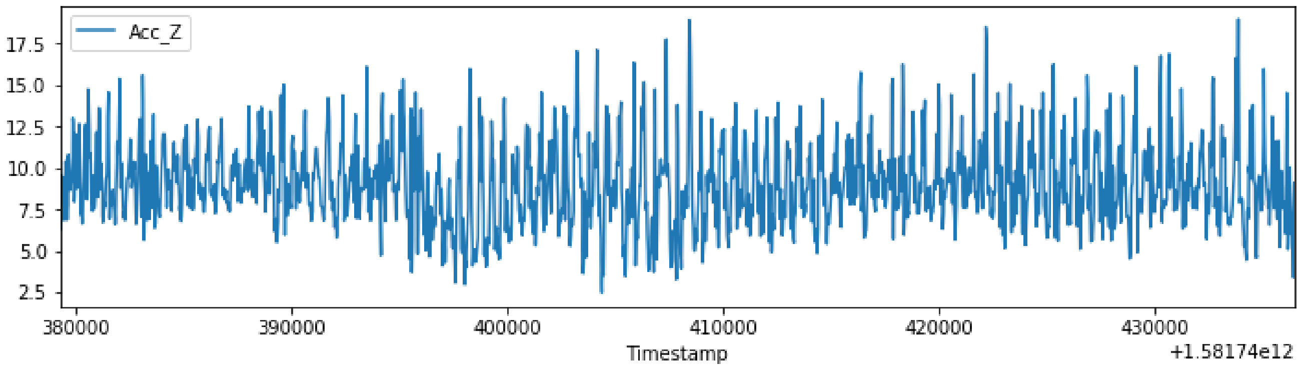


In [6]:

```python
# Plotting the time series data
acc_sup.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sup.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sup.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

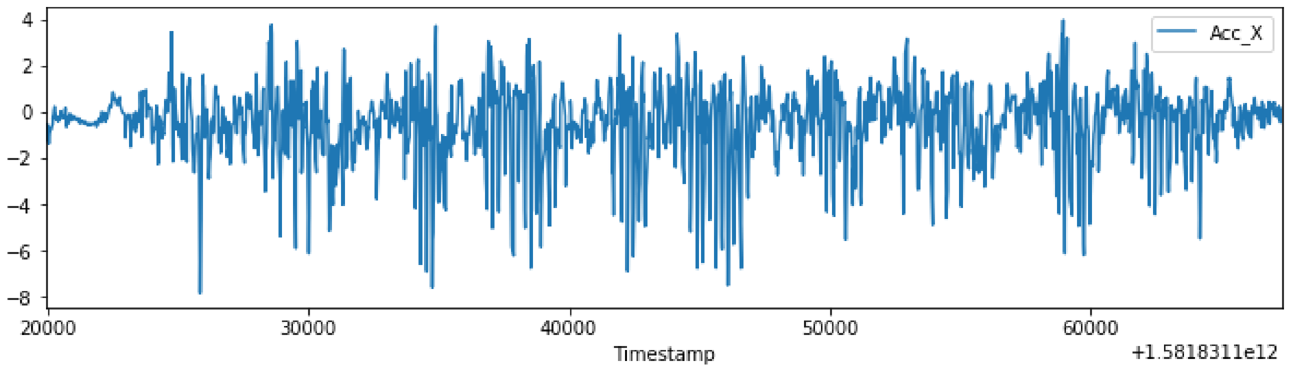


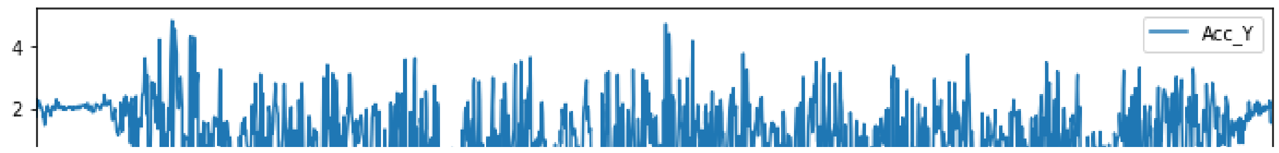


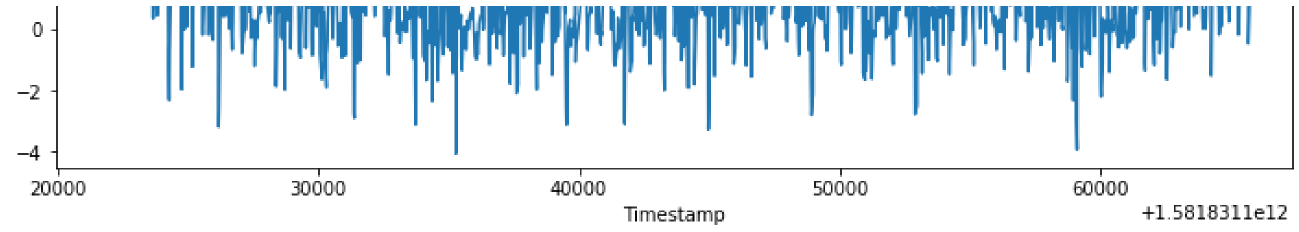


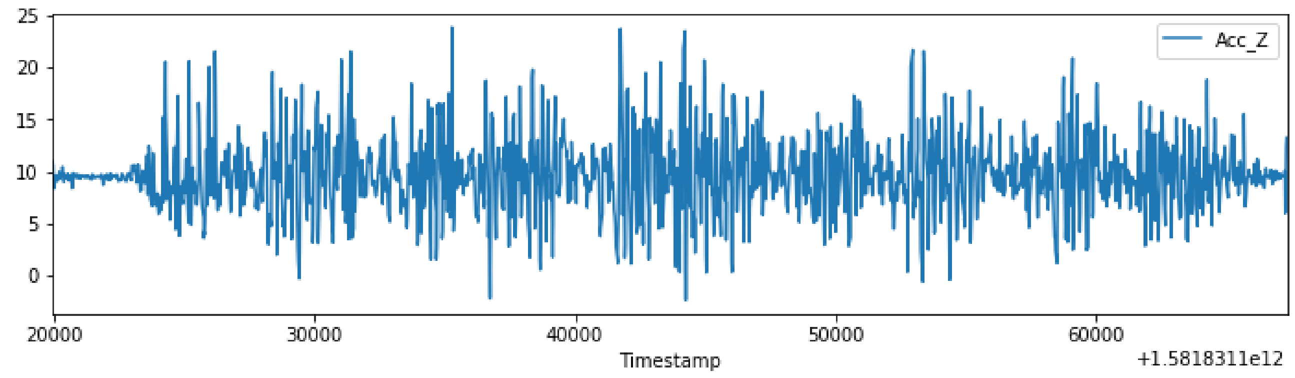
In [7]:

```
# Plotting the time series data
acc_sdn.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sdn.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_sdn.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```



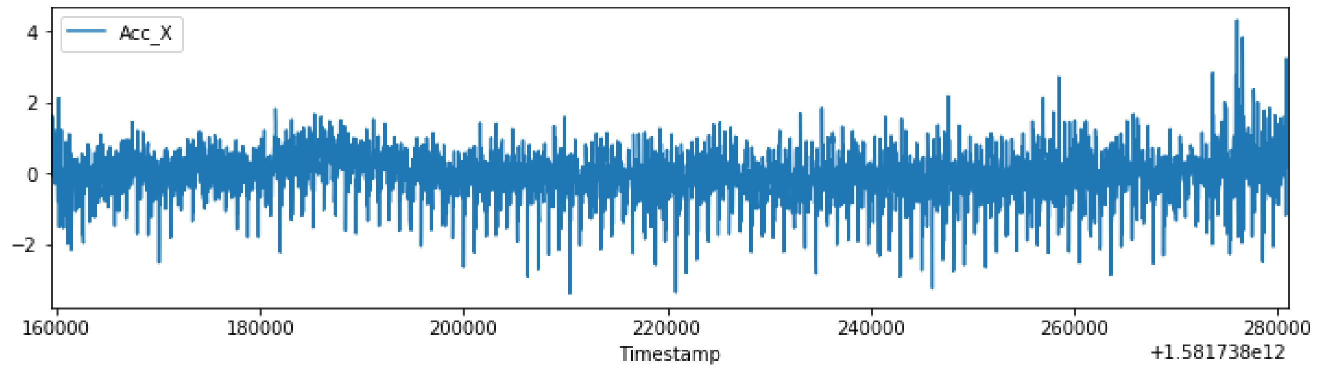


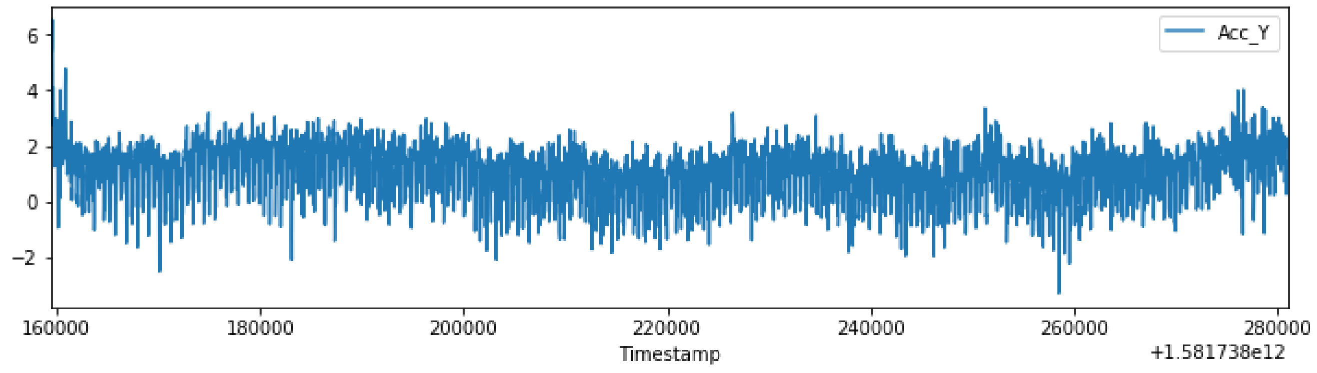


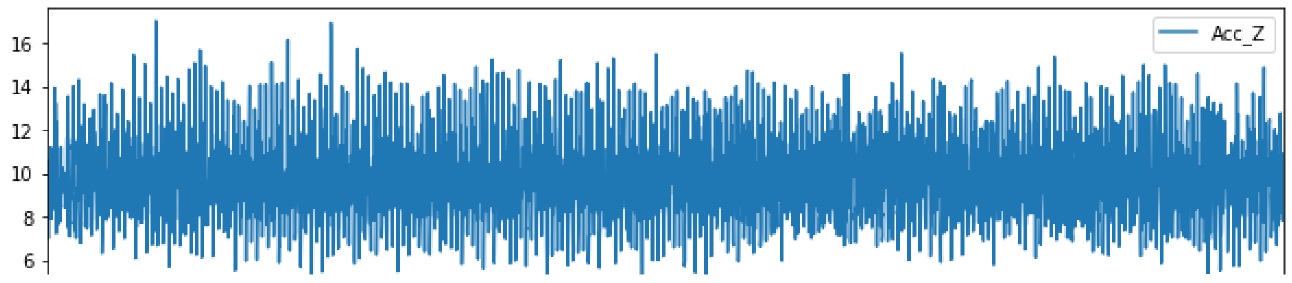


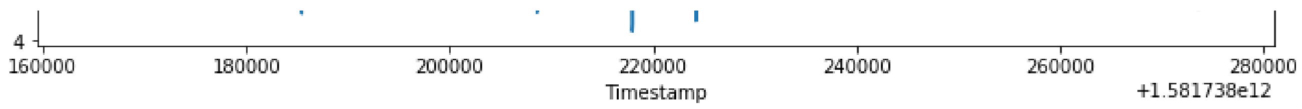
In [8]:

```
# Plotting the time series data
acc_wlk.plot(x='Timestamp',y='Acc_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_wlk.plot(x='Timestamp',y='Acc_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

acc_wlk.plot(x='Timestamp',y='Acc_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```



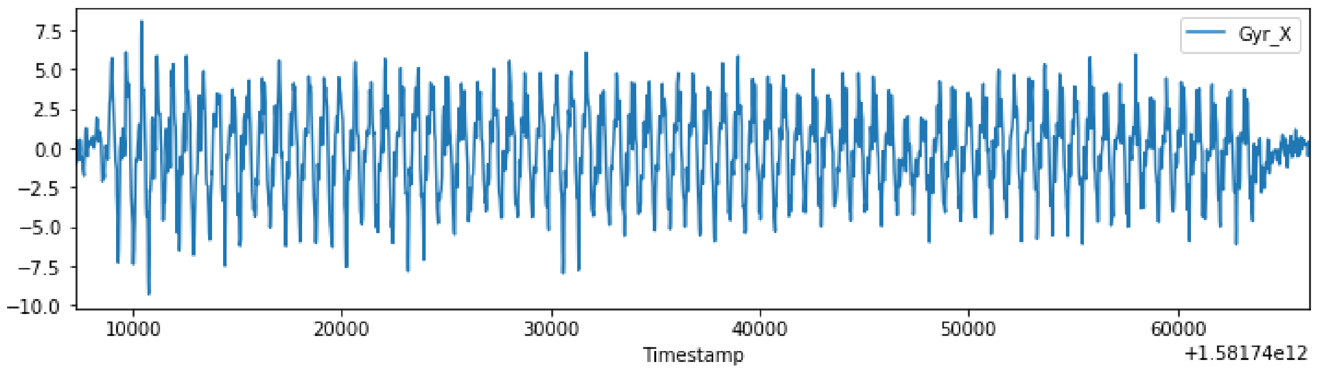


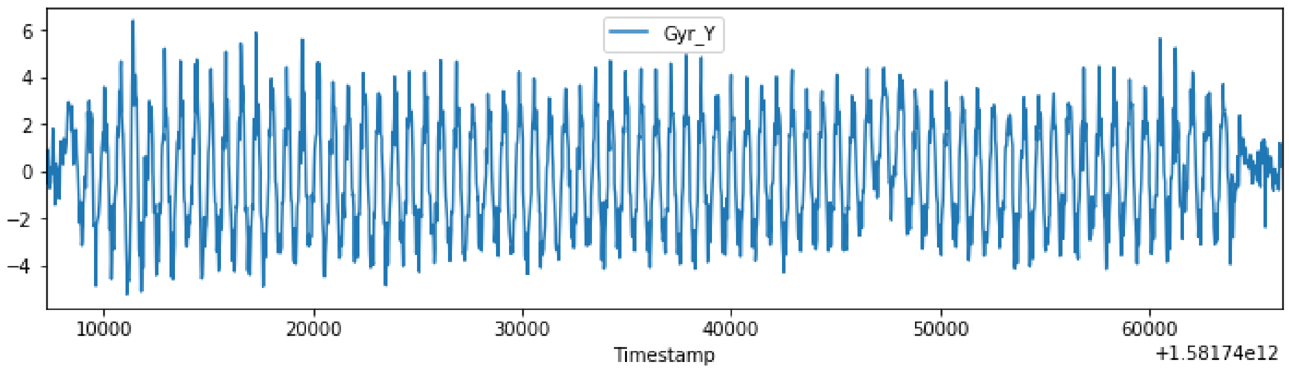


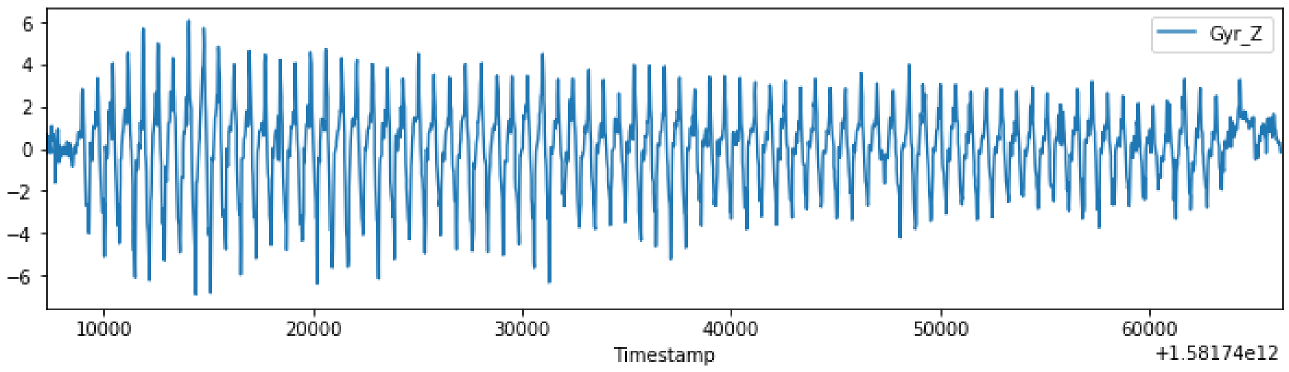


In [9]:

```
# Plotting the time series data
gyr_jog.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_jog.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_jog.plot(x='Timestamp',y='Gyr_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```



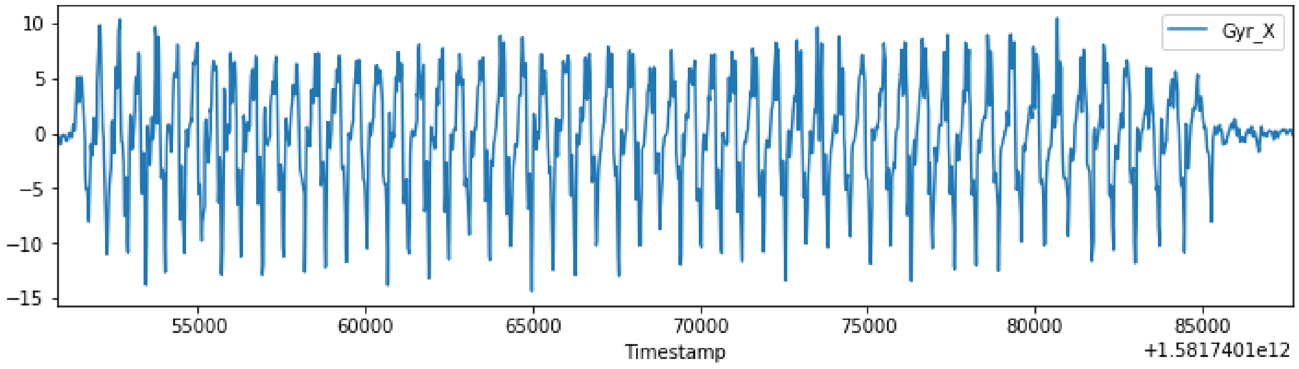


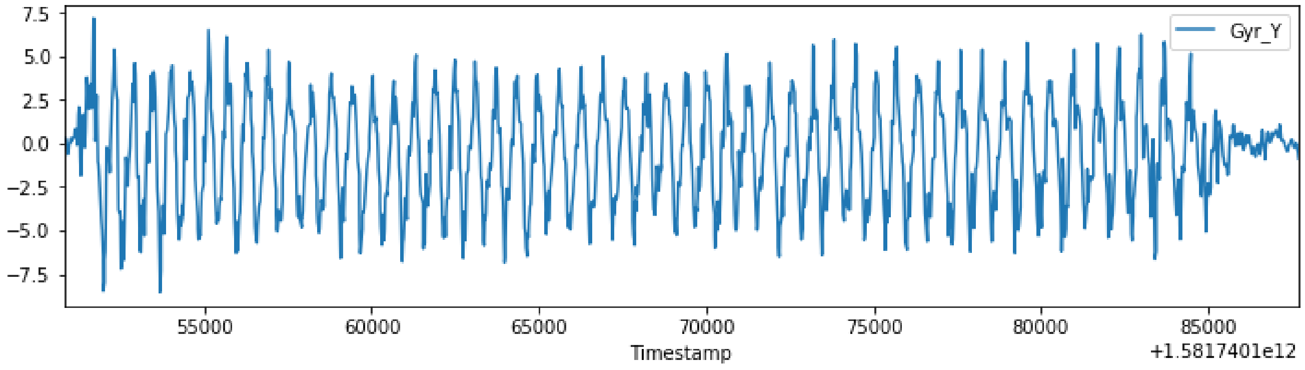


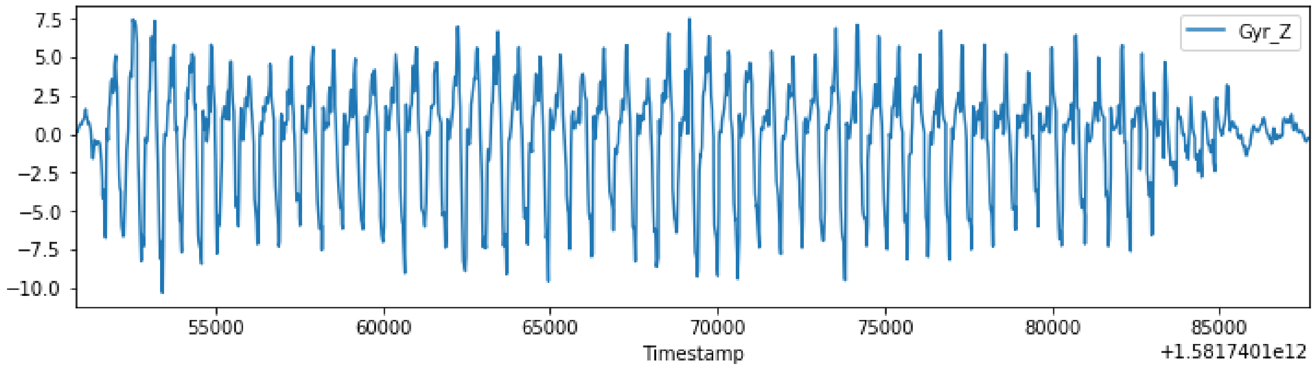
In [10]:

```
# Plotting the time series data
gyr_run.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_run.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_run.plot(x='Timestamp',y='Gyr_Z')
```

```
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

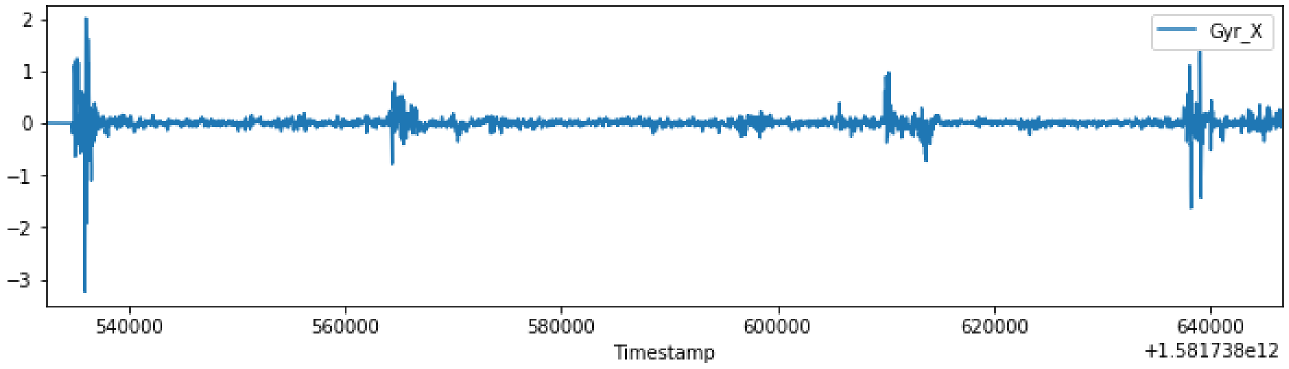


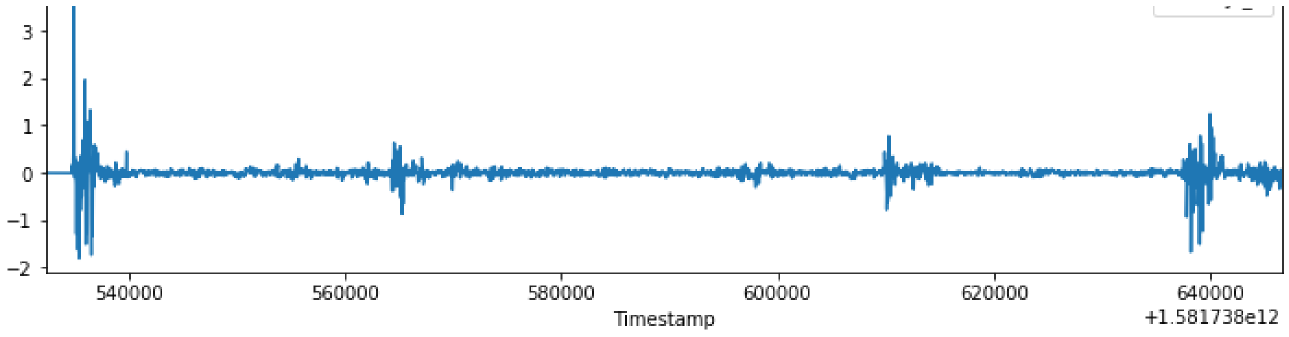


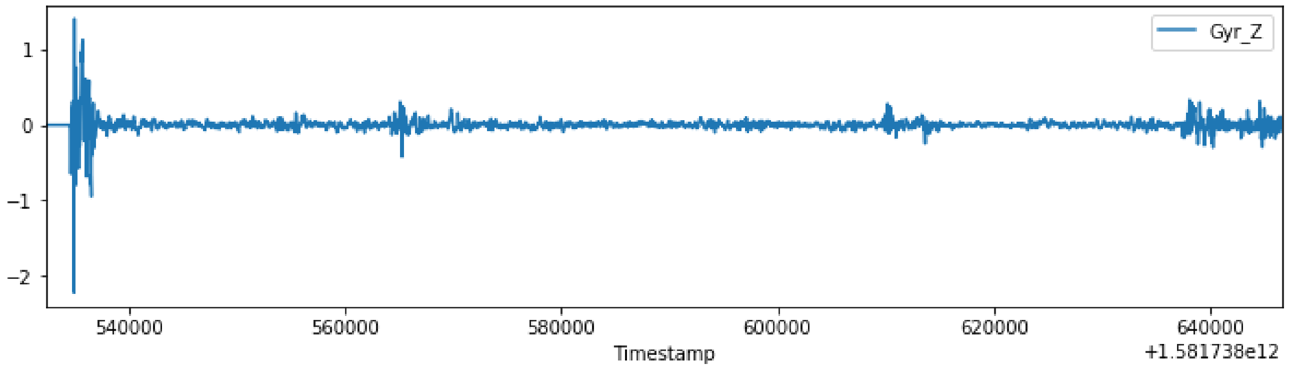


In [11]:

```
# Plotting the time series data
gyr_sit.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sit.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sit.plot(x='Timestamp',y='Gyr_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

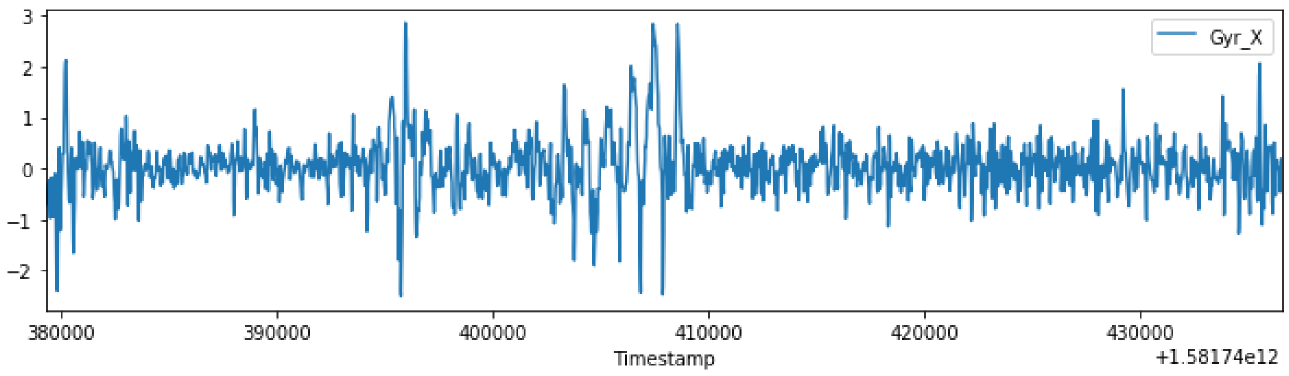


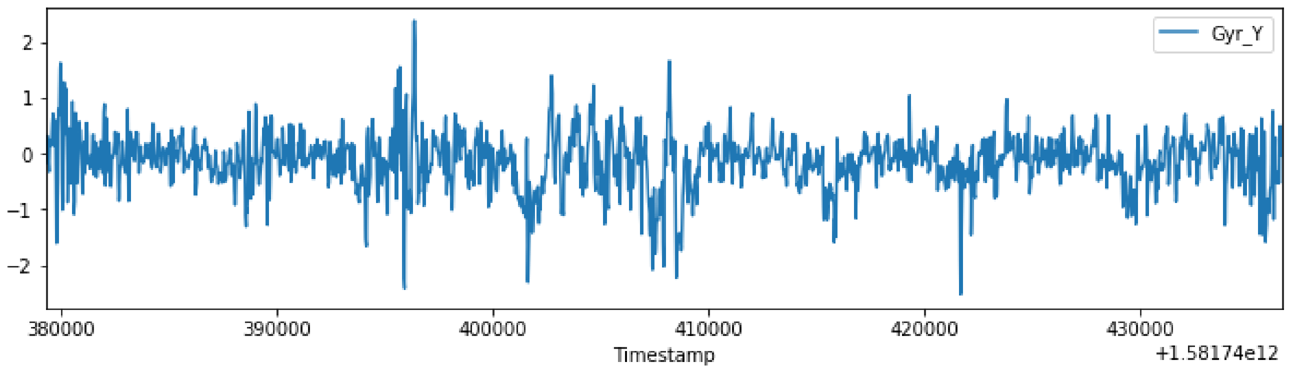


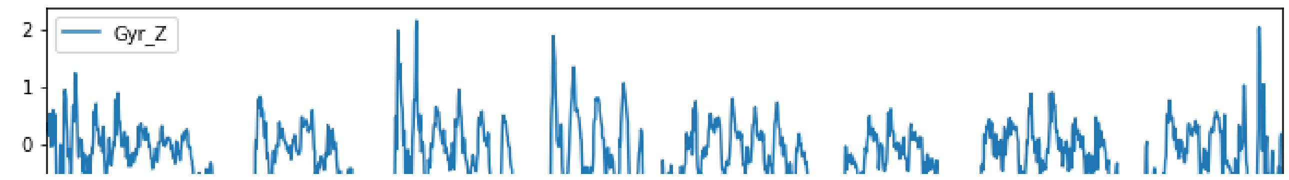


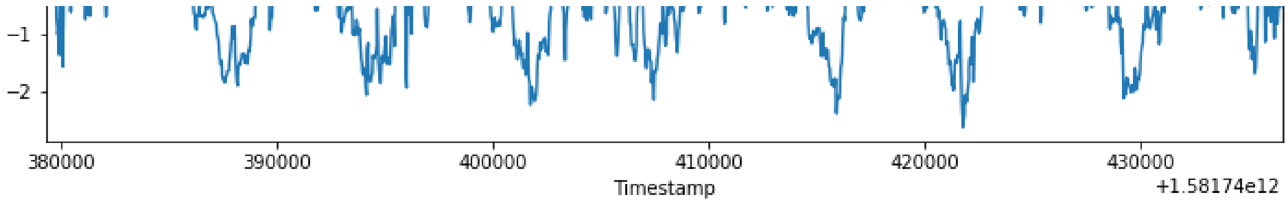
In [12]:

```
# Plotting the time series data
gyr_sup.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sup.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sup.plot(x='Timestamp',y='Gyr_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```



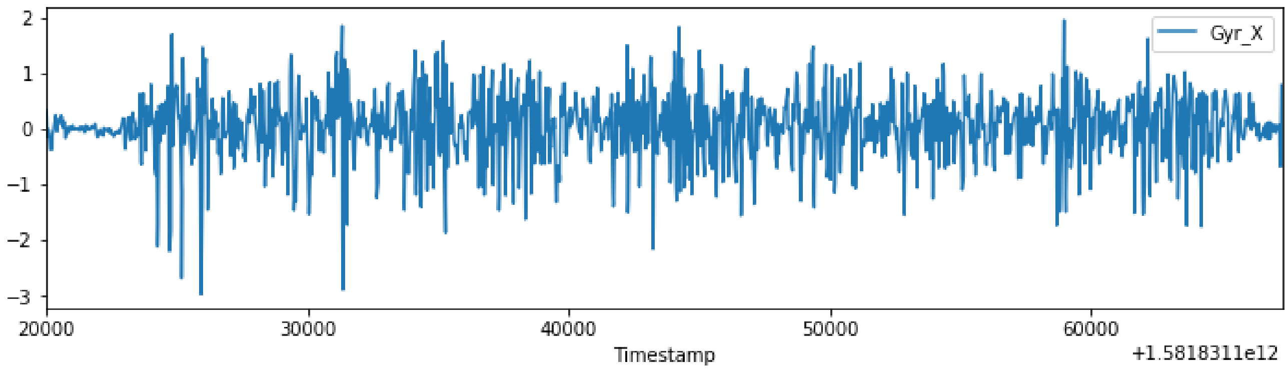


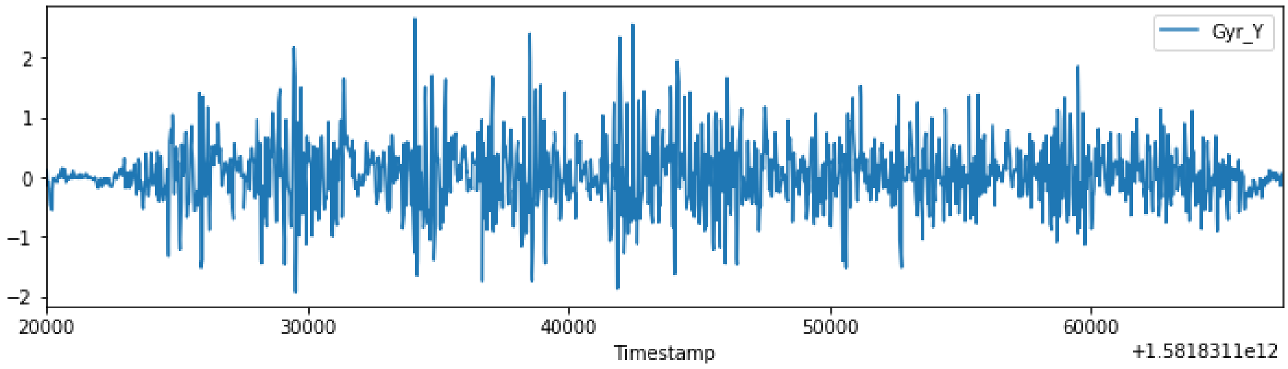


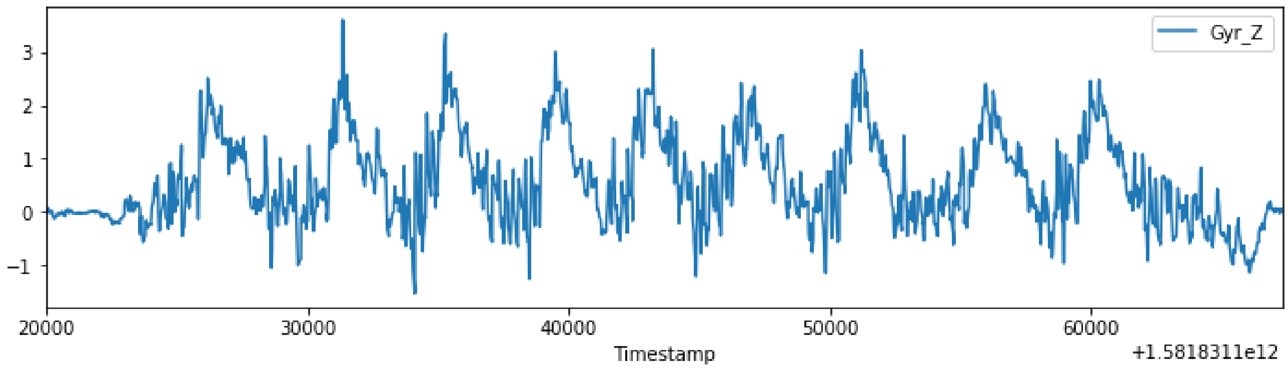


In [13]:

```python
# Plotting the time series data
gyr_sdn.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sdn.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_sdn.plot(x='Timestamp',y='Gyr_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```

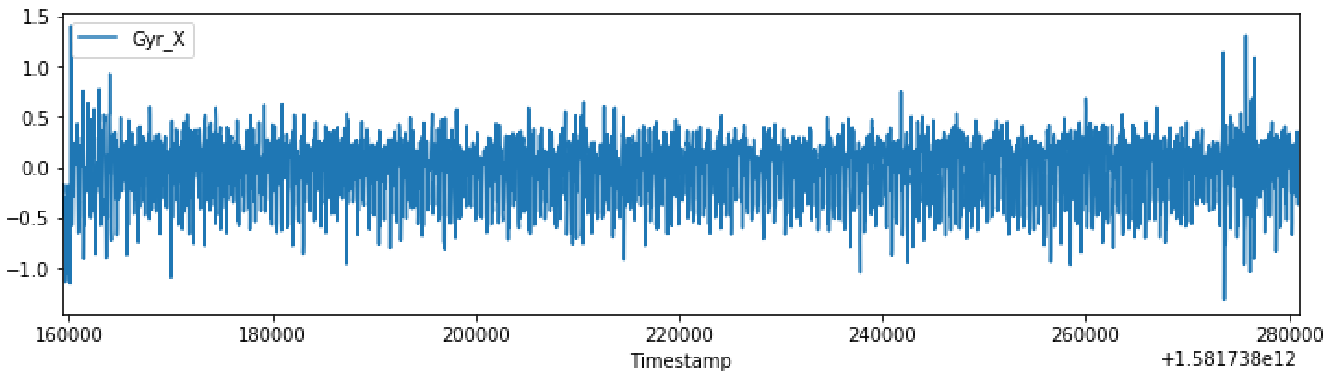


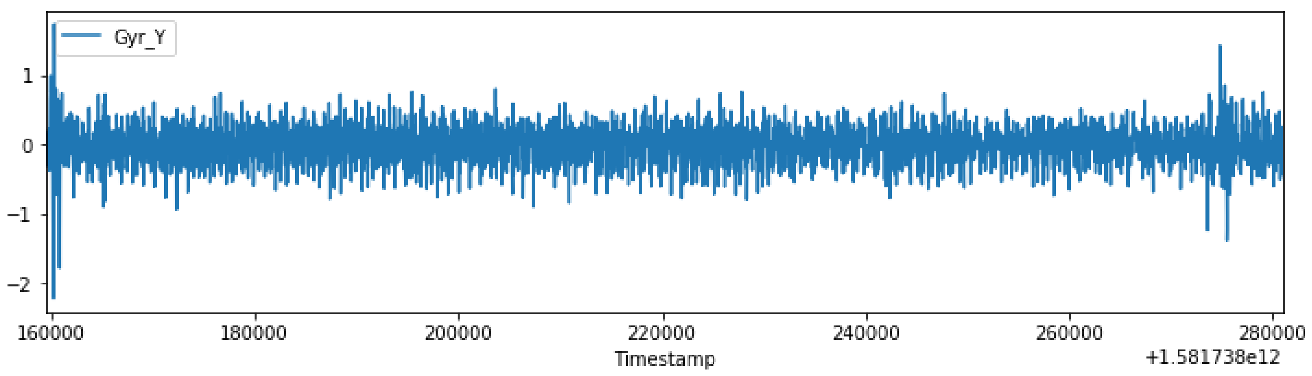


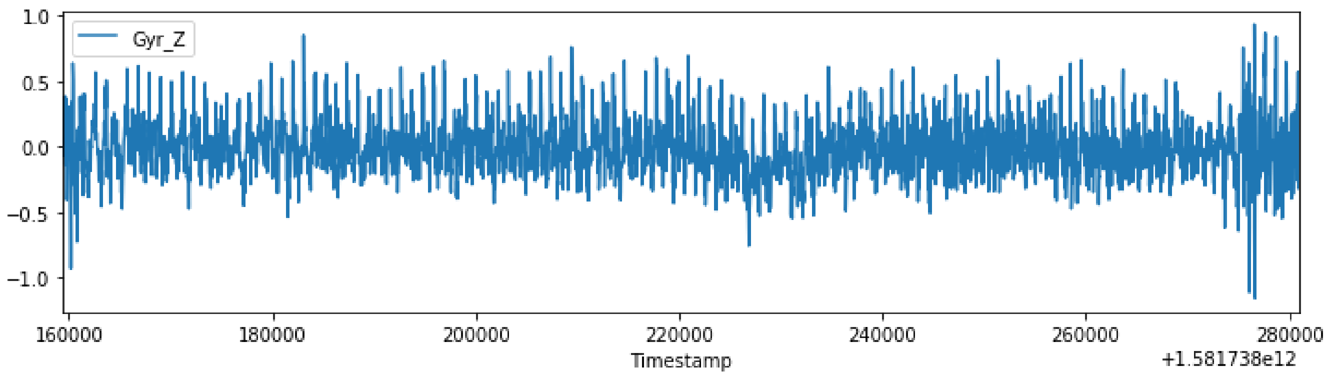


In [14]:

```python
# Plotting the time series data
gyr_wlk.plot(x='Timestamp',y='Gyr_X')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_wlk.plot(x='Timestamp',y='Gyr_Y')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()

gyr_wlk.plot(x='Timestamp',y='Gyr_Z')
plt.rcParams["figure.figsize"]=(12,3)
plt.show()
```







In [15]:

```
# Removing certain samples from start and end of sensor readings, to avoid the error asso
ciated

# Jogging
acc_jog[(acc_jog.Acc_Y > 9.98)].head(1)
acc_jog = acc_jog.iloc[81:]
gyr_jog = gyr_jog.iloc[81:]

acc_jog[(acc_jog.Acc_Y > 9.98)].tail(1)
acc_jog = acc_jog.iloc[:-111]
gyr_jog = gyr_jog.iloc[:-111]

acc_jog.count()
```

Out[15]:

```
Timestamp    2789
Acc_X        2789
Acc_Y        2789
Acc_Z        2789
Activity     2789
dtype: int64
```

In [16]:

```
# Running
acc_run[(acc_run.Acc_X < -20)].head(1)
acc_run = acc_run.iloc[44:]
gyr_run = gyr_run.iloc[44:]

acc_run[(acc_run.Acc_X < -20)].tail(1)
acc_run = acc_run.iloc[:-120]
gyr_run = gyr_run.iloc[:-120]

acc_run.count()
```

Out[16]:

```
Timestamp    1697
Acc_X        1697
Acc_Y        1697
Acc_Z        1697
Activity     1697
dtype: int64
```

In [17]:

```
# Sitting
acc_sit[(acc_sit.Acc_X > 5)].head(1)
acc_sit = acc_sit.iloc[240:]
gyr_sit = gyr_sit.iloc[240:]

acc_sit[(acc_sit.Acc_X > 2.5)].tail(1)
acc_sit = acc_sit.iloc[:-134]
gyr_sit = gyr_sit.iloc[:-134]

acc_sit.count()
```

Out[17]:

```
Timestamp    5400
Acc_X        5400
Acc_Y        5400
Acc_Z        5400
Activity     5400
dtype: int64
```

In [18]:

```
# Stair down
acc_sdn[(acc_sdn.Acc_X > 1)].head(1)
acc_sdn = acc_sdn.iloc[247:]
gyr_sdn = gyr_sdn.iloc[247:]

acc_sdn[(acc_sdn.Acc_X > 1)].tail(1)
acc_sdn = acc_sdn.iloc[:-150]
gyr_sdn = gyr_sdn.iloc[:-150]

acc_sdn.count()
```

Out[18]:

```
Timestamp    2000
Acc_X        2000
Acc_Y        2000
Acc_Z        2000
Activity     2000
dtype: int64
```

In [19]:

```
# Walking
acc_wlk = acc_wlk.iloc[200:]
gyr_wlk = gyr_wlk.iloc[200:]

acc_wlk = acc_wlk.iloc[:-327]
gyr_wlk = gyr_wlk.iloc[:-327]
```

```
acc_wlk.count()
```

Out[19]:

```
Timestamp    5600
Acc_X        5600
Acc_Y        5600
Acc_Z        5600
Activity     5600
dtype: int64
```

In [20]:

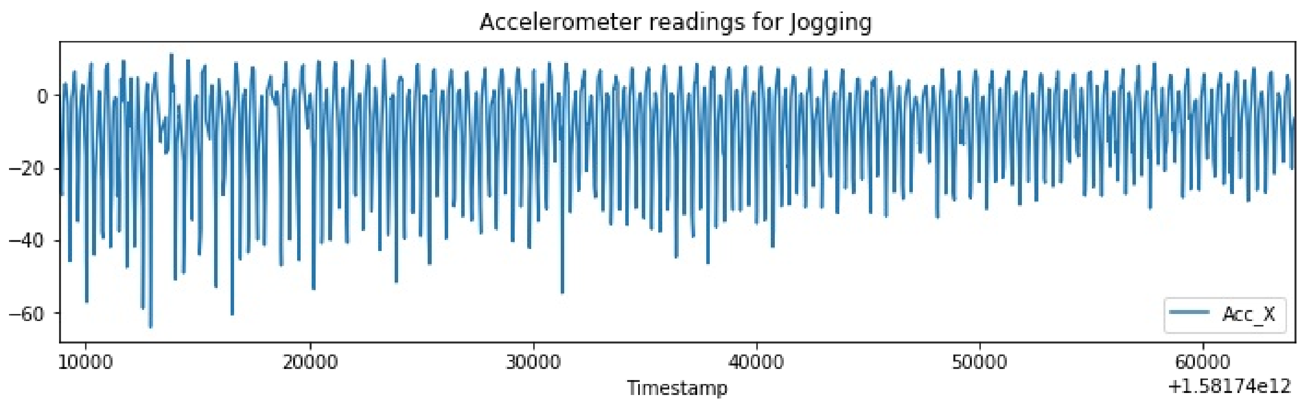
```
# Merging the two frames consisting of Accelerometer and Gyroscope readings
sen_jog = pd.merge(acc_jog,gyr_jog, on=['Timestamp','Activity'])
sen_run = pd.merge(acc_run,gyr_run, on=['Timestamp','Activity'])
sen_wlk = pd.merge(acc_wlk,gyr_wlk, on=['Timestamp','Activity'])
sen_sup = pd.merge(acc_sup,gyr_sup, on=['Timestamp','Activity'])
sen_sdn = pd.merge(acc_sdn,gyr_sdn, on=['Timestamp','Activity'])
sen_sit = pd.merge(acc_sit,gyr_sit, on=['Timestamp','Activity'])
```

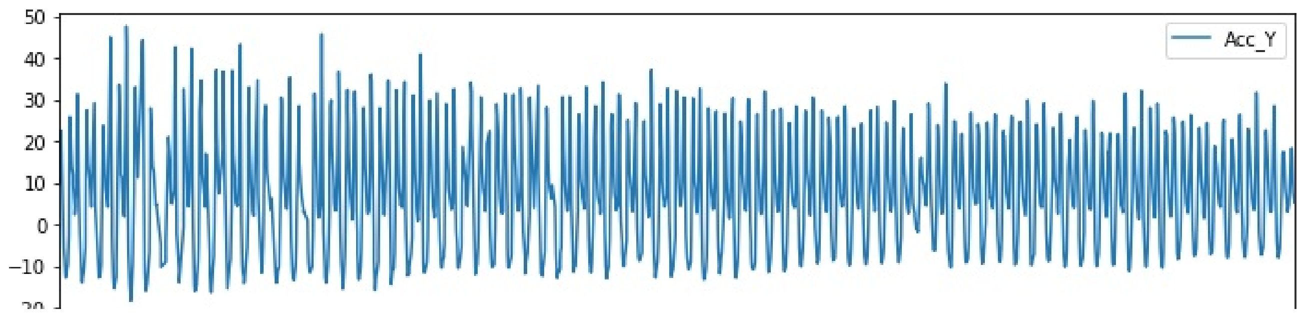
In [21]:

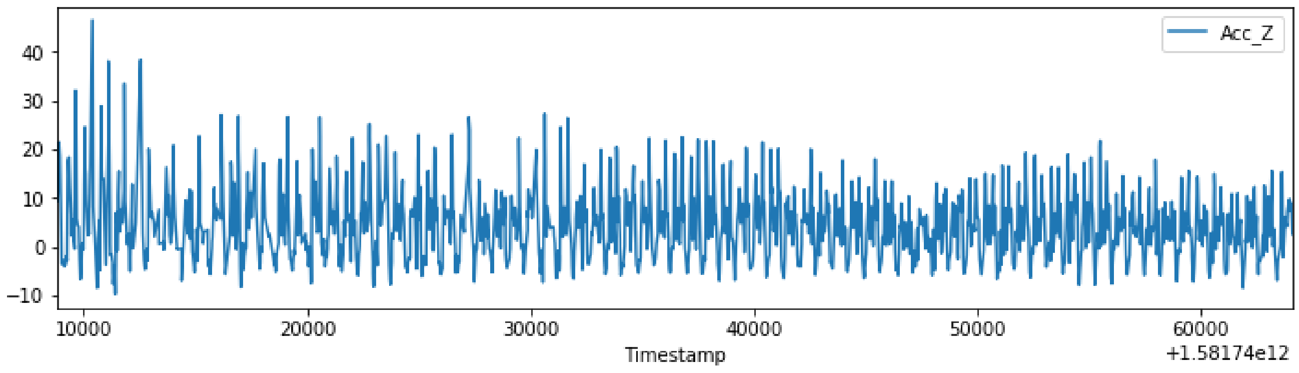
```
# Before forming the features, declaring an empty data frame to append features
features = ['meanAccX','meanAccY','meanAccZ','meanGyrX','meanGyrY','meanGyrZ',
             'maxAccX', 'maxAccY', 'maxAccZ', 'maxGyrX', 'maxGyrY', 'maxGyrZ',
             'minAccX', 'minAccY', 'minAccZ', 'minGyrX', 'minGyrY', 'minGyrZ',
             'sdAccX' , 'sdAccY' , 'sdAccZ' , 'sdGyrX' , 'sdGyrY',  'sdGyrZ',
             'madAccX', 'madAccY', 'maxAccZ', 'madGyrX', 'madGyrY', 'maxGyrZ',
             'Activity']
dataset = pd.DataFrame(columns=features)
```

In [22]:

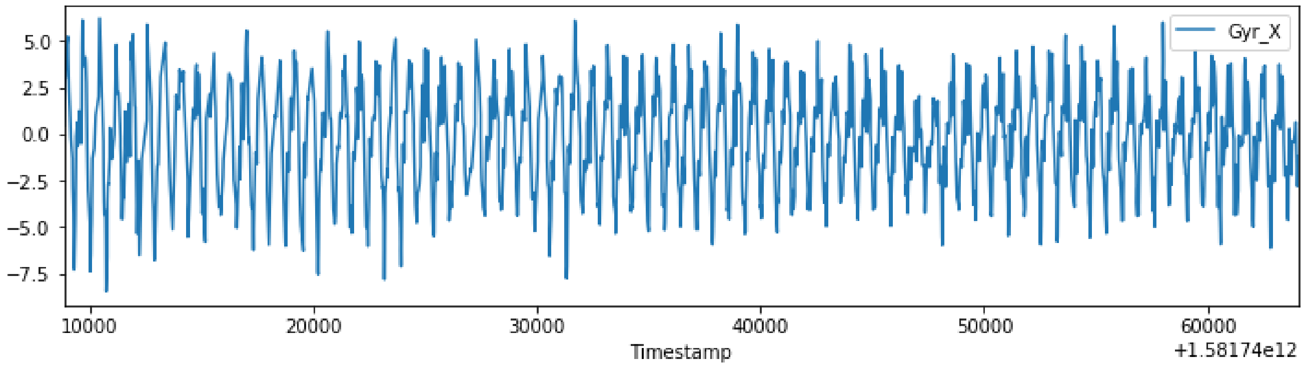
```
sen_jog.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Jogging')
plt.show()
sen_jog.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_jog.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_jog.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Jogging')
plt.show()
sen_jog.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_jog.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```

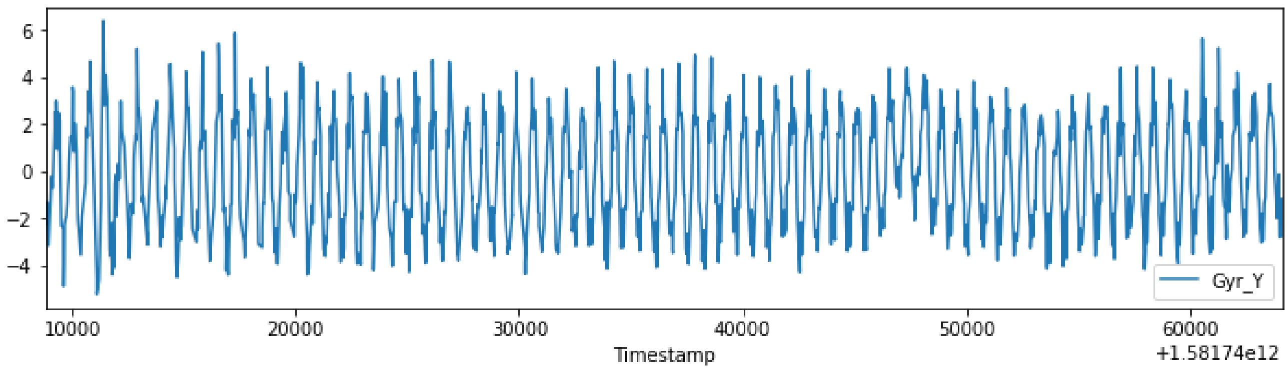


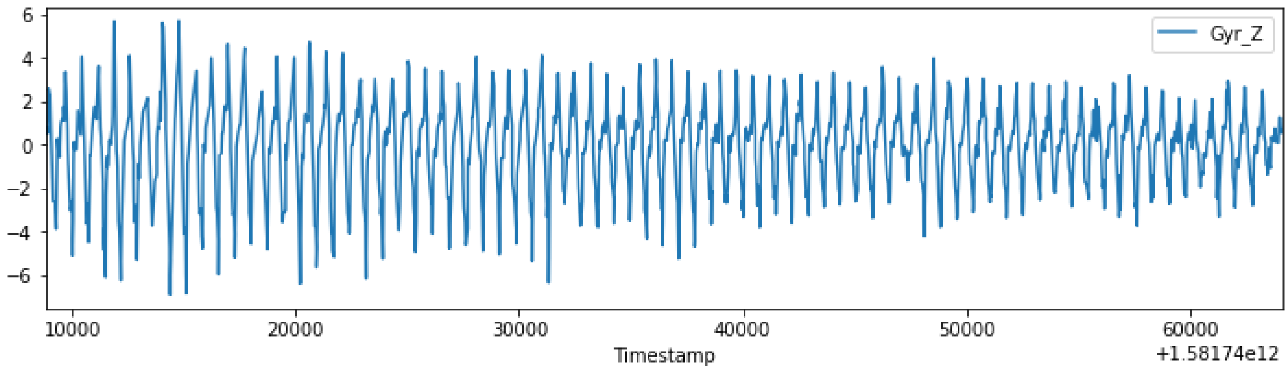




Gyroscope readings for Jogging





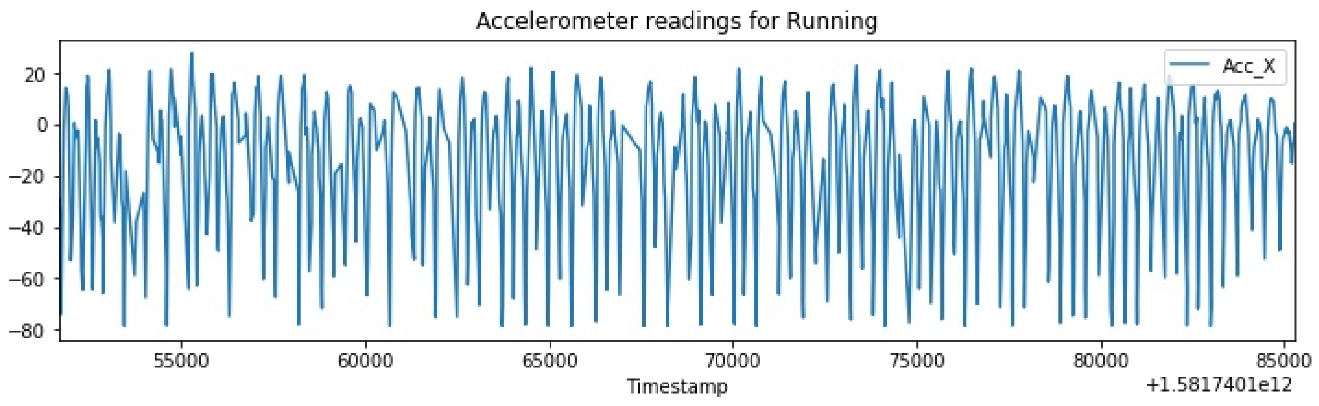


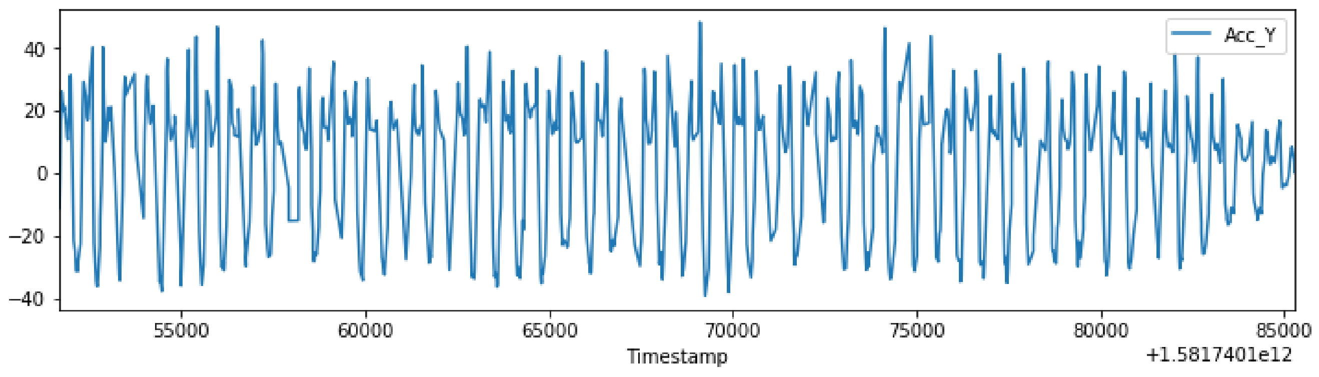
In [23]:

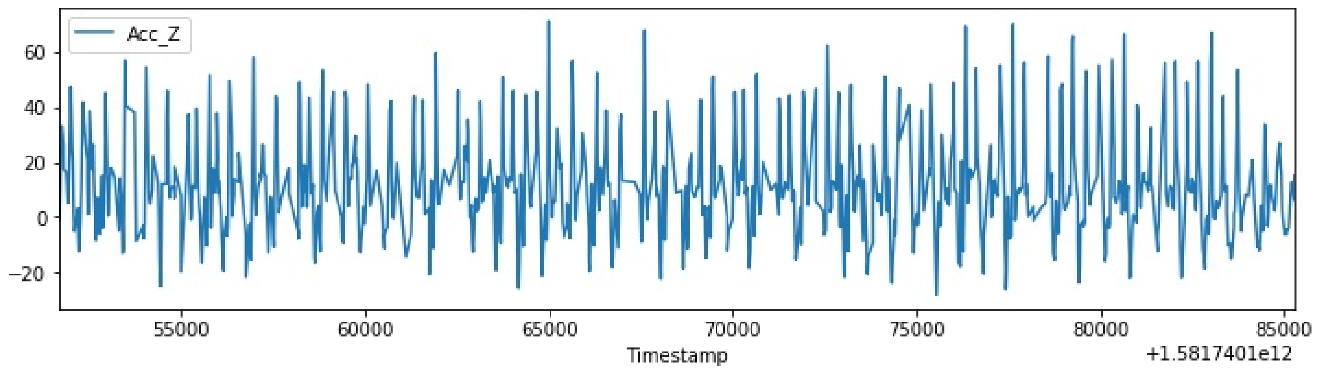
```
# For jogging, it is observed that, a cycle of activity repeats at around 25 samples
# Window of 25 is cut and parameters are computed.
# The count of records is 2314 currently, to make it a multiple of window size, ignoring
the last 14 records.
sen_jog = sen_jog.iloc[:-14]
sen_jog.count()

for i in range(0,sen_jog.Activity.count()-1,25):
    temp = sen_jog.iloc[i:i+25]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
        'meanGyrX':temp.Gyr_X.mean(),
        'meanGyrY':temp.Gyr_Y.mean(),
        'meanGyrZ':temp.Gyr_Z.mean(),
        'maxAccX' :temp.Acc_X.max(),
        'maxAccY' :temp.Acc_Y.max(),
        'maxAccZ' :temp.Acc_Z.max(),
        'maxGyrX' :temp.Gyr_X.max(),
        'maxGyrY' :temp.Gyr_Y.max(),
        'maxGyrZ' :temp.Gyr_Z.max(),
        'minAccX' :temp.Acc_X.min(),
        'minAccY' :temp.Acc_Y.min(),
        'minAccZ' :temp.Acc_Z.min(),
        'minGyrX' :temp.Gyr_X.min(),
        'minGyrY' :temp.Gyr_Y.min(),
        'minGyrZ' :temp.Gyr_Z.min(),
        'sdAccX'  :temp.Acc_X.std(),
        'sdAccY'  :temp.Acc_Y.std(),
        'sdAccZ'  :temp.Acc_Z.std(),
        'sdGyrX'  :temp.Gyr_X.std(),
        'sdGyrY'  :temp.Gyr_Y.std(),
        'sdGyrZ'  :temp.Gyr_Z.std(),
        'madAccX' :temp.Acc_X.mad(),
        'madAccY' :temp.Acc_Y.mad(),
        'maxAccZ' :temp.Acc_Z.mad(),
        'madGyrX' :temp.Gyr_X.mad(),
        'madGyrY' :temp.Gyr_Y.mad(),
        'maxGyrZ' :temp.Gyr_Z.mad(),
        'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

In [24]:

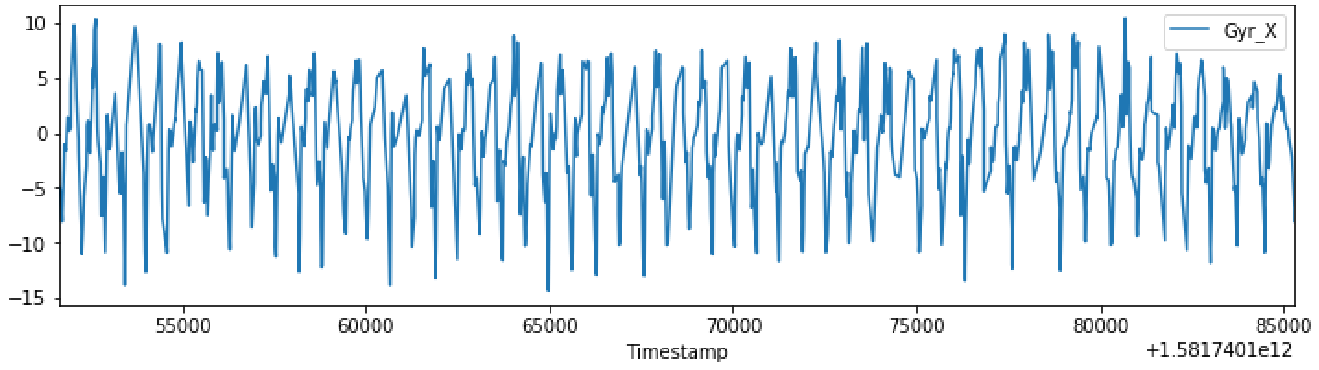
```
sen_run.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Running')
plt.show()
sen_run.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_run.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_run.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Running')
plt.show()
sen_run.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_run.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```

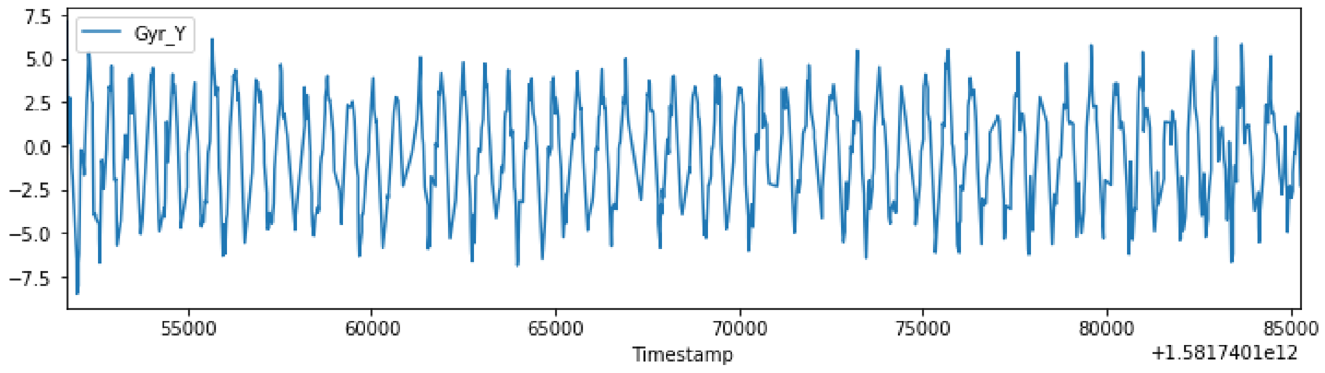


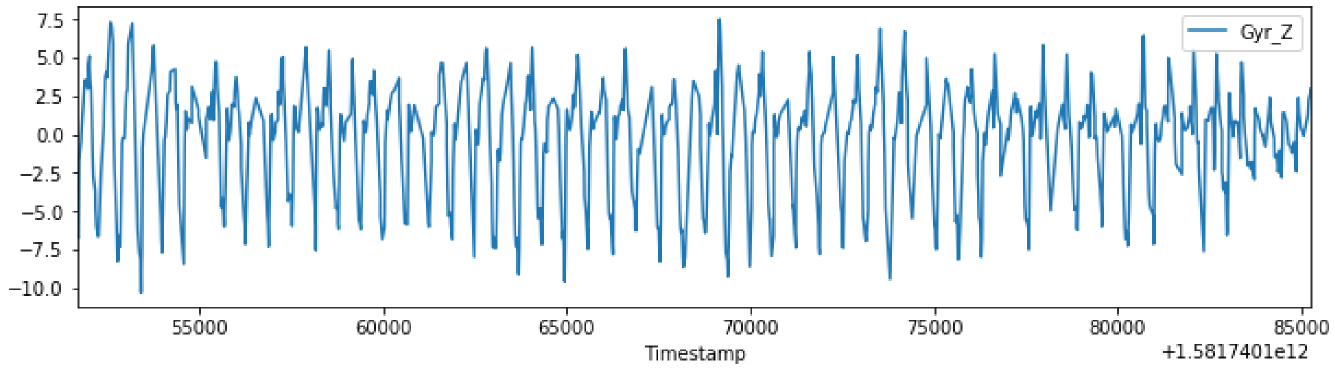




Gyroscope readings for Running





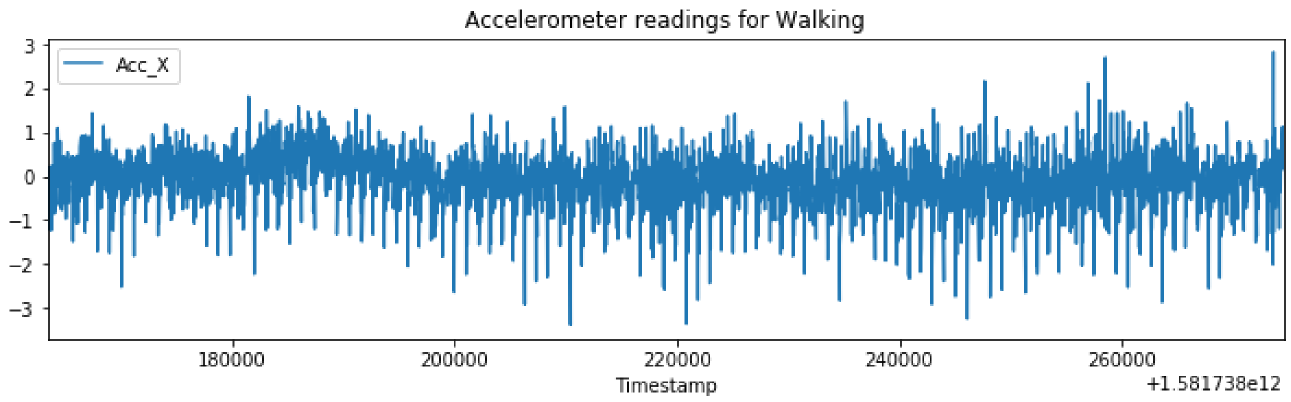


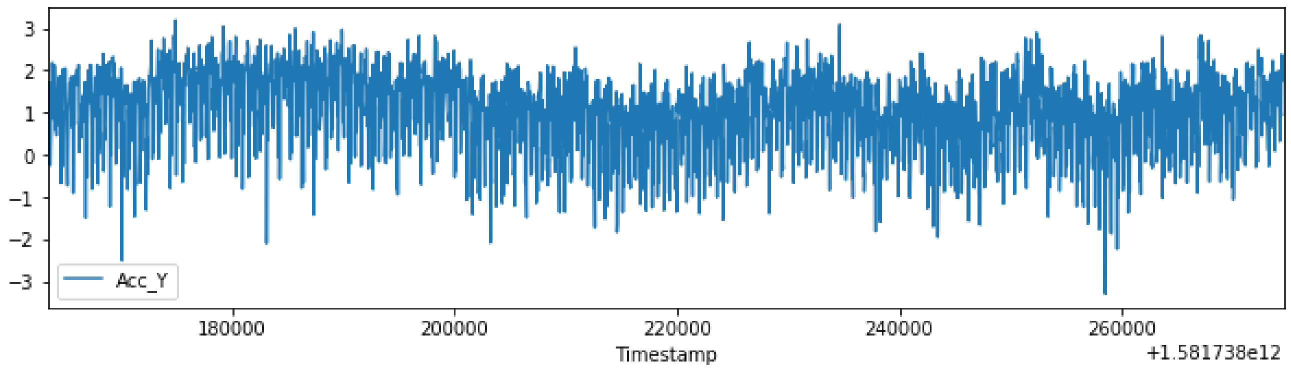
In [25]:

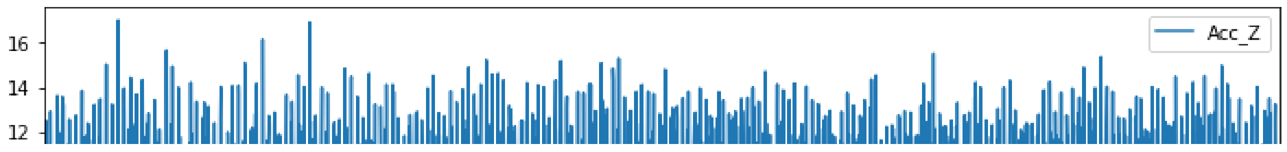
```
# For running, it is observed that, a cycle of activity repeats at around 24 samples
# Window of 24 is cut and parameters are computed.
sen_run = sen_run.iloc[:-11]
sen_run.count()

for i in range(0,sen_run.Activity.count()-1,24):
    temp = sen_run.iloc[i:i+24]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
         'meanGyrX':temp.Gyr_X.mean(),
         'meanGyrY':temp.Gyr_Y.mean(),
         'meanGyrZ':temp.Gyr_Z.mean(),
         'maxAccX' :temp.Acc_X.max(),
         'maxAccY' :temp.Acc_Y.max(),
         'maxAccZ' :temp.Acc_Z.max(),
         'maxGyrX' :temp.Gyr_X.max(),
         'maxGyrY' :temp.Gyr_Y.max(),
         'maxGyrZ' :temp.Gyr_Z.max(),
         'minAccX' :temp.Acc_X.min(),
         'minAccY' :temp.Acc_Y.min(),
         'minAccZ' :temp.Acc_Z.min(),
         'minGyrX' :temp.Gyr_X.min(),
         'minGyrY' :temp.Gyr_Y.min(),
         'minGyrZ' :temp.Gyr_Z.min(),
         'sdAccX'  :temp.Acc_X.std(),
         'sdAccY'  :temp.Acc_Y.std(),
         'sdAccZ'  :temp.Acc_Z.std(),
         'sdGyrX'  :temp.Gyr_X.std(),
         'sdGyrY'  :temp.Gyr_Y.std(),
         'sdGyrZ'  :temp.Gyr_Z.std(),
         'madAccX' :temp.Acc_X.mad(),
         'madAccY' :temp.Acc_Y.mad(),
         'maxAccZ' :temp.Acc_Z.mad(),
         'madGyrX' :temp.Gyr_X.mad(),
         'madGyrY' :temp.Gyr_Y.mad(),
         'maxGyrZ' :temp.Gyr_Z.mad(),
         'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

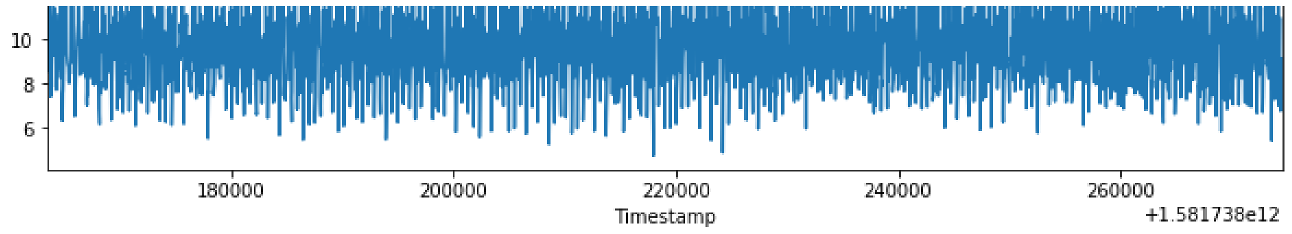
In [26]:

```
sen_wlk.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Walking')
plt.show()
sen_wlk.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_wlk.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_wlk.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Walking')
plt.show()
sen_wlk.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_wlk.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```

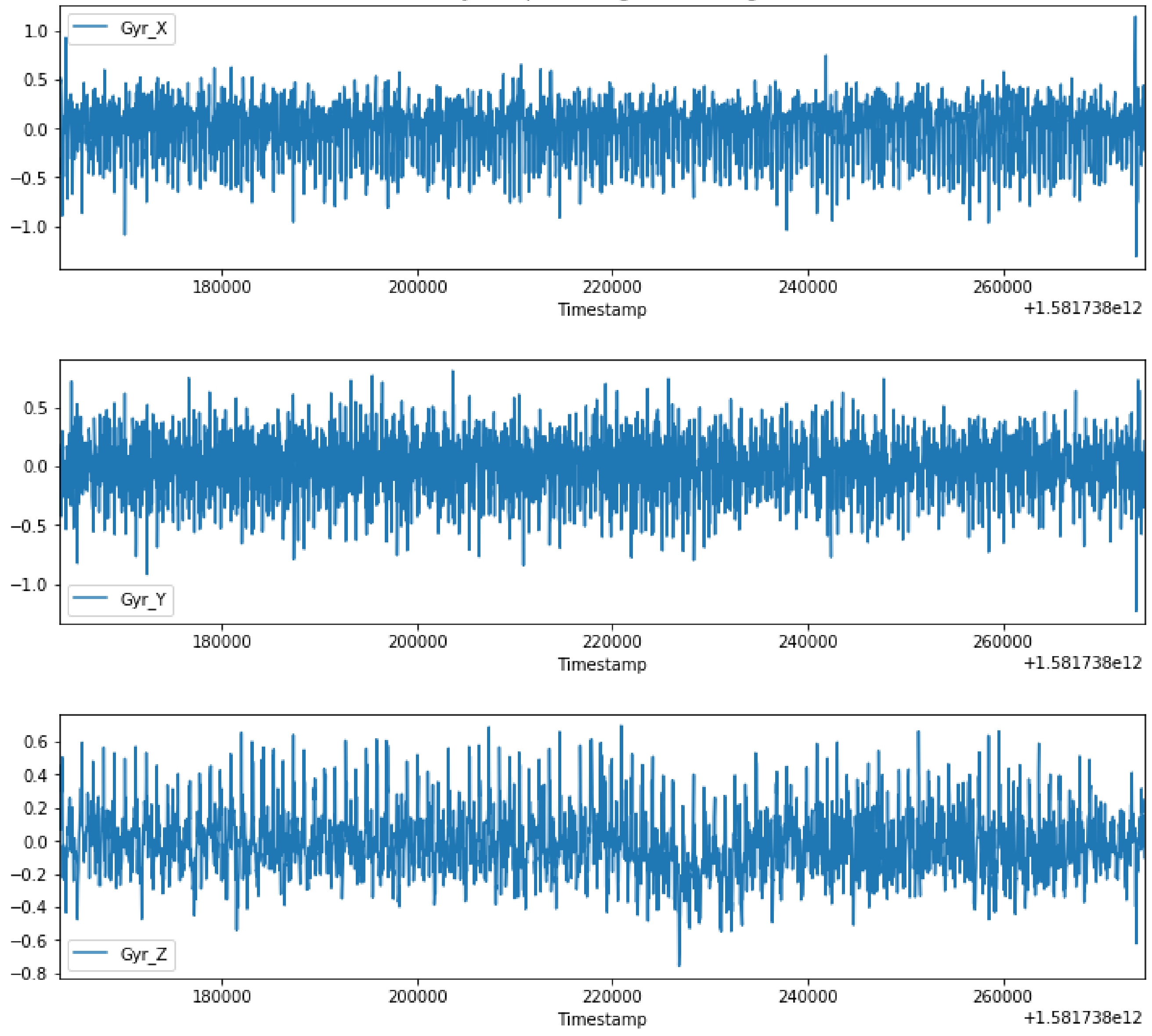








Gyroscope readings for Walking

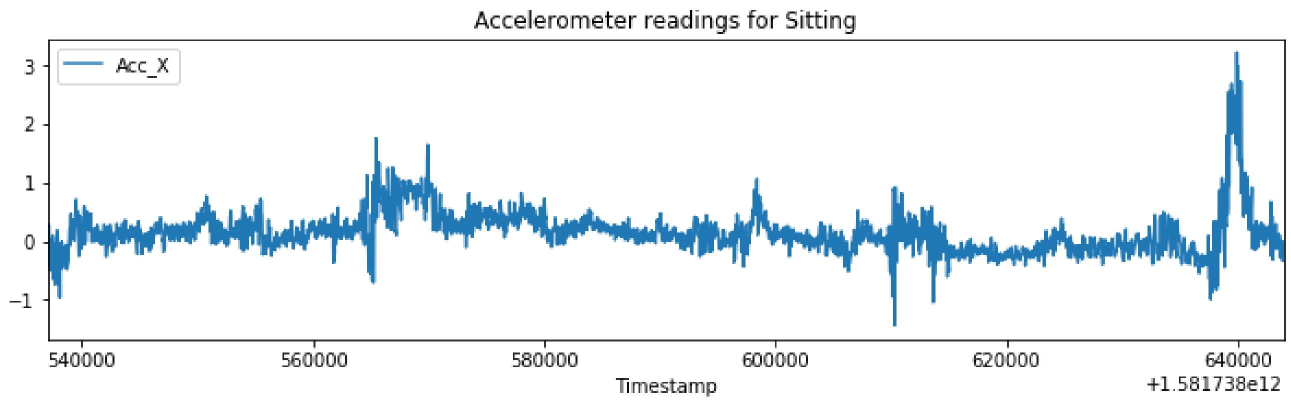


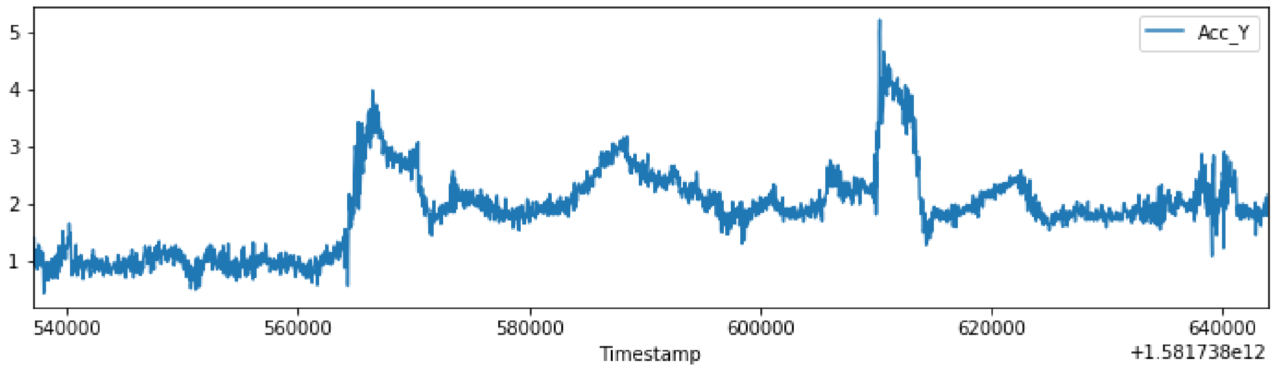
In [27]:

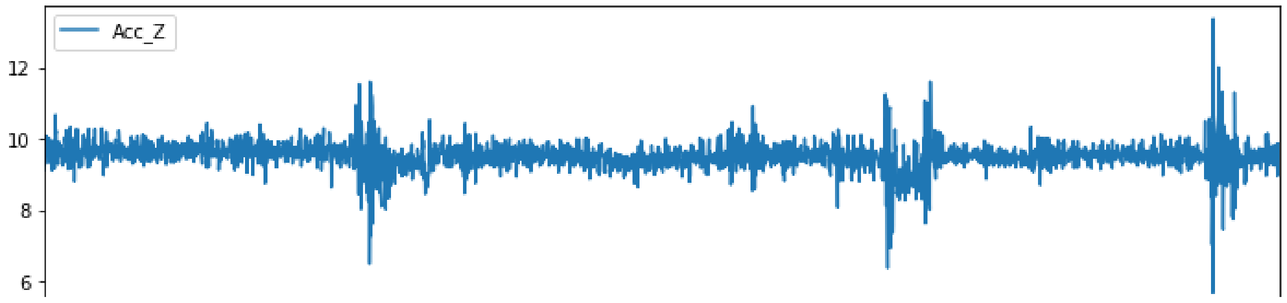
```
# For walking, it is observed that, a cycle of activity repeats at around 37 samples
# Window of 37 is cut and parameters are computed.
sen_wlk = sen_wlk.iloc[:-35]
sen_wlk.count()

for i in range(0,sen_wlk.Activity.count()-1,37):
    temp = sen_wlk.iloc[i:i+37]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
         'meanGyrX':temp.Gyr_X.mean(),
         'meanGyrY':temp.Gyr_Y.mean(),
         'meanGyrZ':temp.Gyr_Z.mean(),
         'maxAccX' :temp.Acc_X.max(),
         'maxAccY' :temp.Acc_Y.max(),
         'maxAccZ' :temp.Acc_Z.max(),
         'maxGyrX' :temp.Gyr_X.max(),
         'maxGyrY' :temp.Gyr_Y.max(),
         'maxGyrZ' :temp.Gyr_Z.max(),
         'minAccX' :temp.Acc_X.min(),
         'minAccY' :temp.Acc_Y.min(),
         'minAccZ' :temp.Acc_Z.min(),
         'minGyrX' :temp.Gyr_X.min(),
         'minGyrY' :temp.Gyr_Y.min(),
         'minGyrZ' :temp.Gyr_Z.min(),
         'sdAccX'  :temp.Acc_X.std(),
         'sdAccY'  :temp.Acc_Y.std(),
         'sdAccZ'  :temp.Acc_Z.std(),
         'sdGyrX'  :temp.Gyr_X.std(),
         'sdGyrY'  :temp.Gyr_Y.std(),
         'sdGyrZ'  :temp.Gyr_Z.std(),
         'madAccX' :temp.Acc_X.mad(),
         'madAccY' :temp.Acc_Y.mad(),
         'maxAccZ' :temp.Acc_Z.mad(),
         'madGyrX' :temp.Gyr_X.mad(),
         'madGyrY' :temp.Gyr_Y.mad(),
         'maxGyrZ' :temp.Gyr_Z.mad(),
         'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

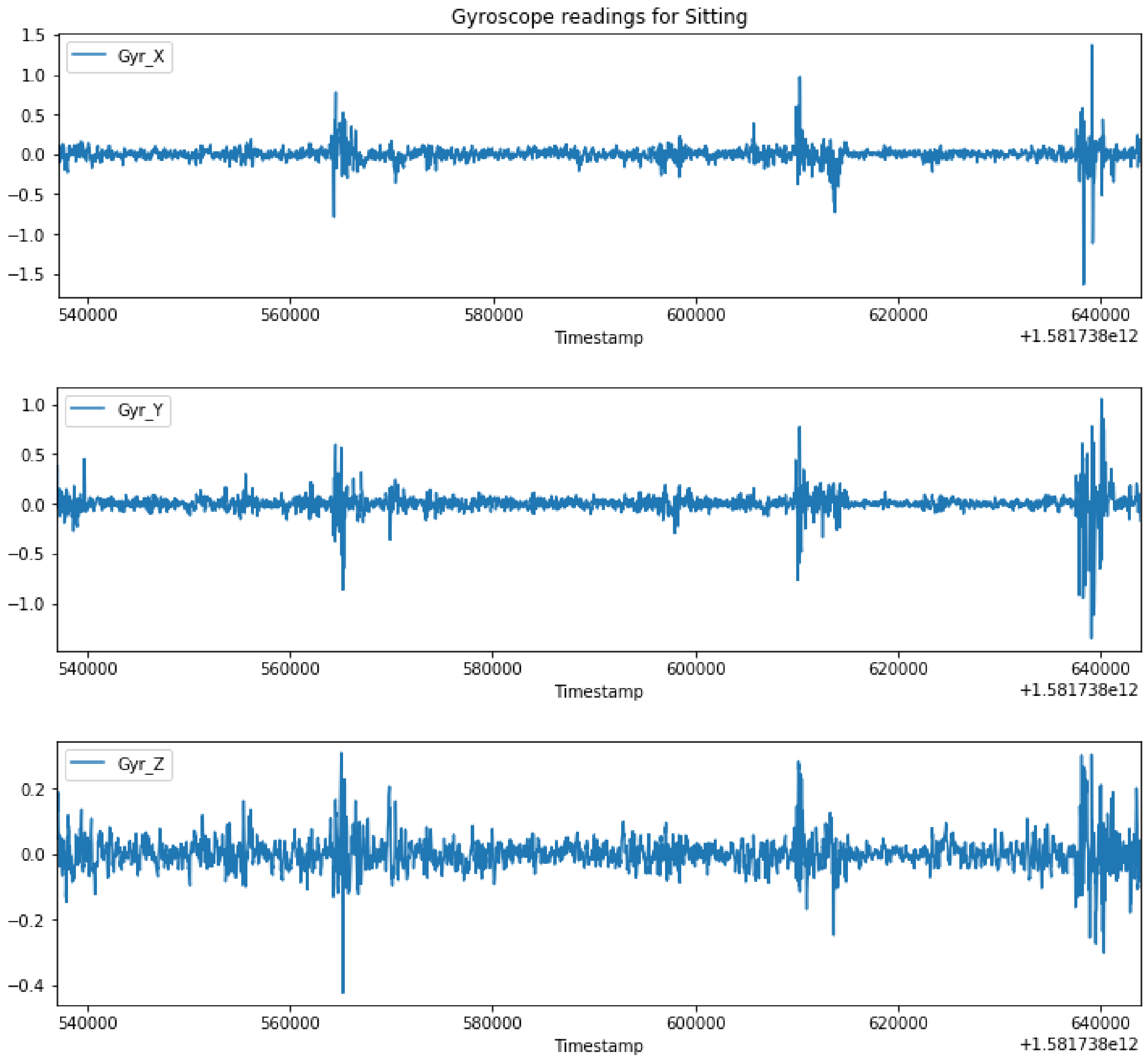
In [28]:

```
sen_sit.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Sitting')
plt.show()
sen_sit.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_sit.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_sit.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Sitting')
plt.show()
sen_sit.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_sit.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```







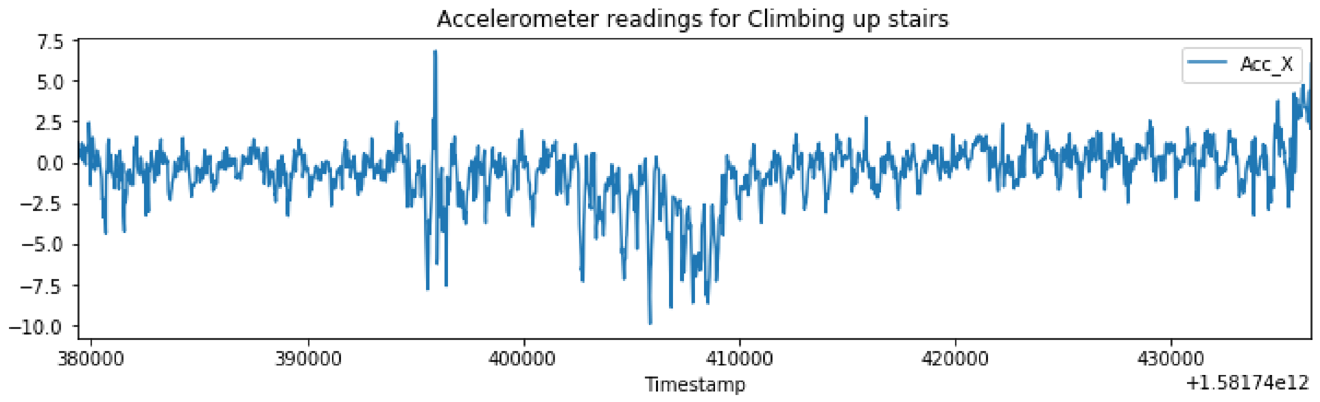


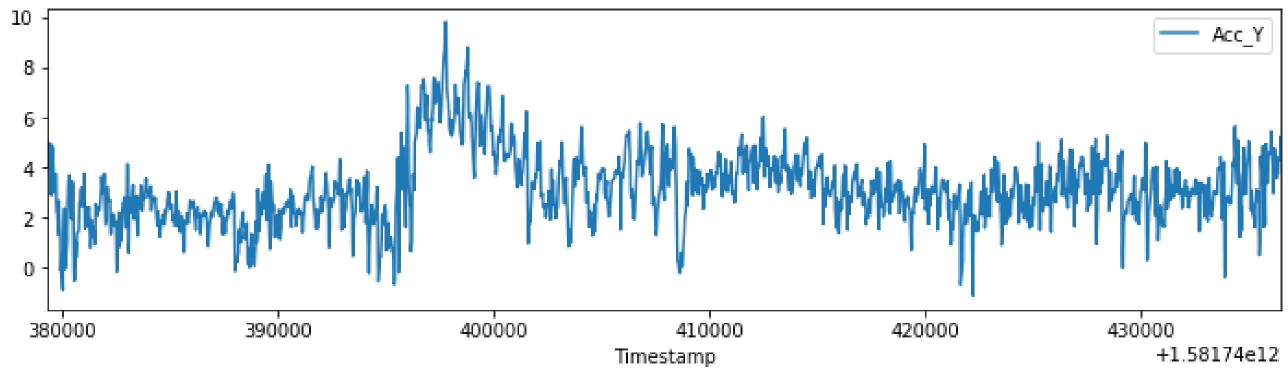
In [29]:

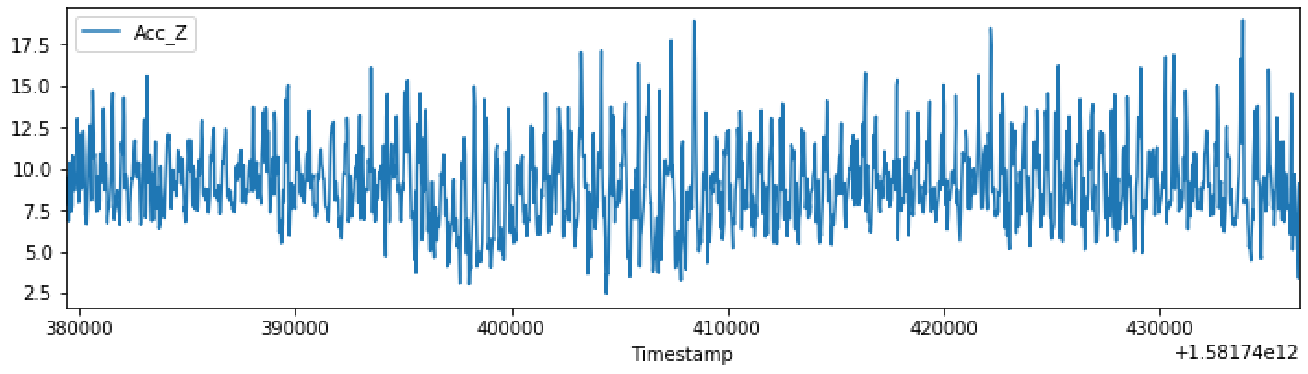
```
# For sitting, it is observed that, a cycle of activity repeats at around 30 samples
# Window of 30 is cut and parameters are computed.
sen_sit = sen_sit.iloc[:-26]
sen_sit.count()

for i in range(0,sen_sit.Activity.count()-1,30):
    temp = sen_sit.iloc[i:i+30]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
         'meanGyrX':temp.Gyr_X.mean(),
         'meanGyrY':temp.Gyr_Y.mean(),
         'meanGyrZ':temp.Gyr_Z.mean(),
         'maxAccX' :temp.Acc_X.max(),
         'maxAccY' :temp.Acc_Y.max(),
         'maxAccZ' :temp.Acc_Z.max(),
         'maxGyrX' :temp.Gyr_X.max(),
         'maxGyrY' :temp.Gyr_Y.max(),
         'maxGyrZ' :temp.Gyr_Z.max(),
         'minAccX' :temp.Acc_X.min(),
         'minAccY' :temp.Acc_Y.min(),
         'minAccZ' :temp.Acc_Z.min(),
         'minGyrX' :temp.Gyr_X.min(),
         'minGyrY' :temp.Gyr_Y.min(),
         'minGyrZ' :temp.Gyr_Z.min(),
         'sdAccX'  :temp.Acc_X.std(),
         'sdAccY'  :temp.Acc_Y.std(),
         'sdAccZ'  :temp.Acc_Z.std(),
         'sdGyrX'  :temp.Gyr_X.std(),
         'sdGyrY'  :temp.Gyr_Y.std(),
         'sdGyrZ'  :temp.Gyr_Z.std(),
         'madAccX' :temp.Acc_X.mad(),
         'madAccY' :temp.Acc_Y.mad(),
         'maxAccZ' :temp.Acc_Z.mad(),
         'madGyrX' :temp.Gyr_X.mad(),
         'madGyrY' :temp.Gyr_Y.mad(),
         'maxGyrZ' :temp.Gyr_Z.mad(),
         'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

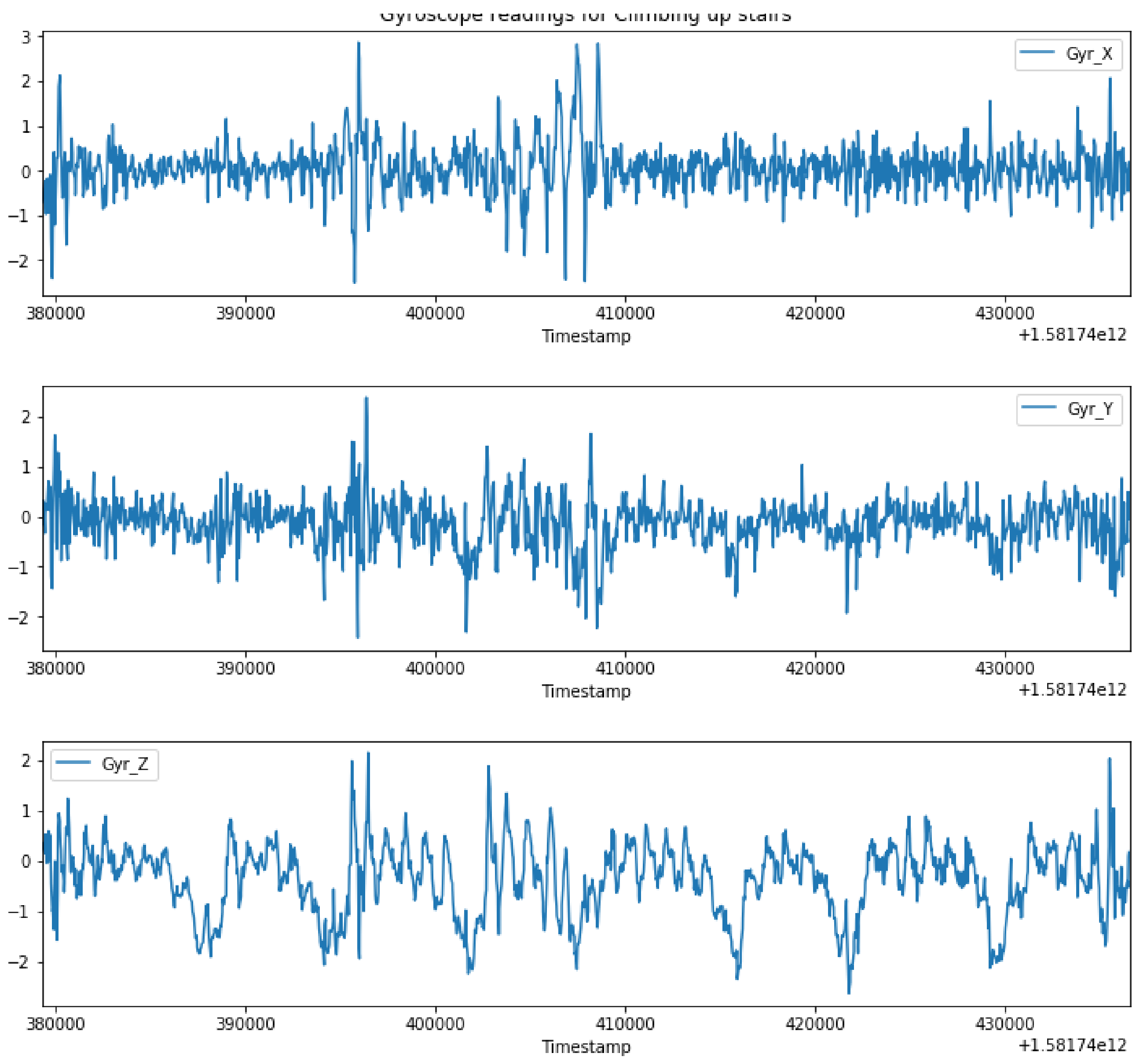
In [30]:

```
sen_sup.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Climbing up stair
s')
plt.show()
sen_sup.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_sup.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_sup.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Climbing up stairs')
plt.show()
sen_sup.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_sup.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```







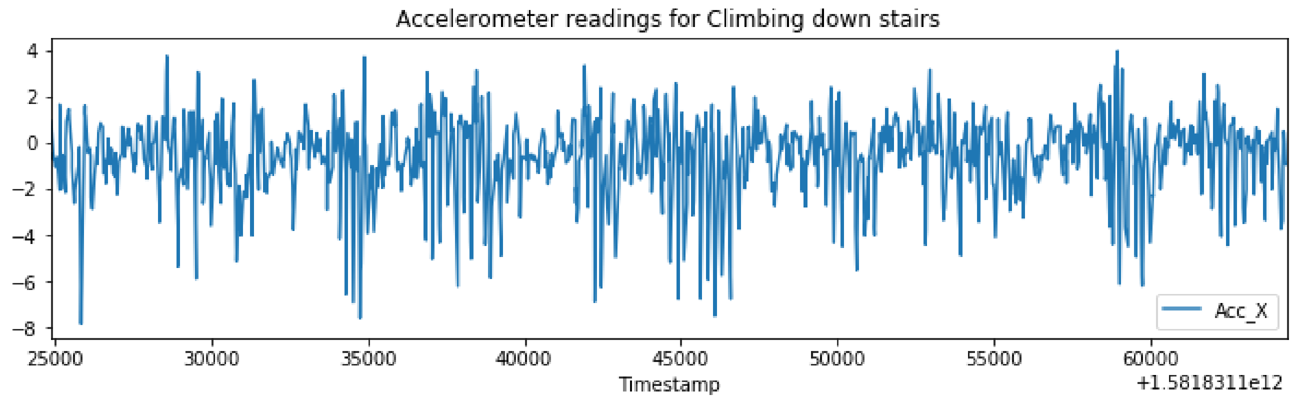


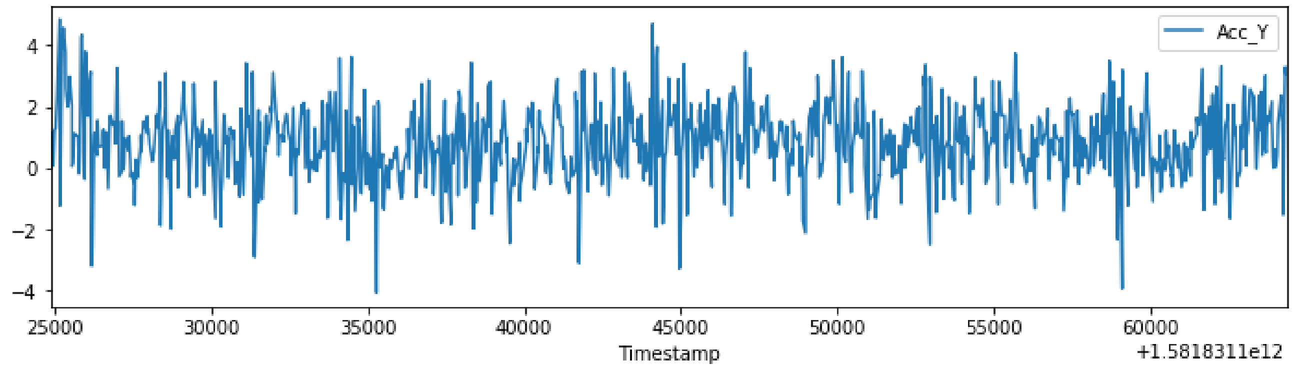
In [31]:

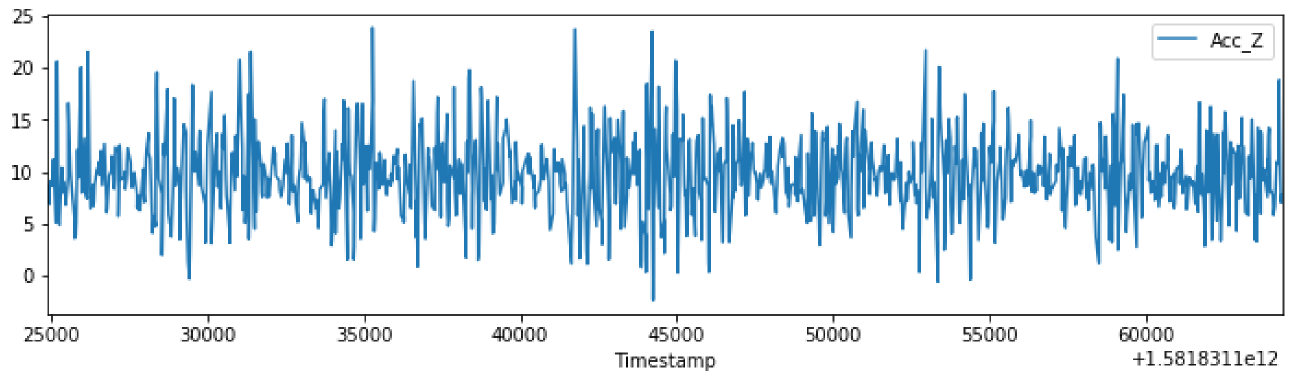
```
# For climbing up stairs, it is observed that, a cycle of activity repeats at around 45 s
amples
# Window of 45 is cut and parameters are computed.
sen_sup = sen_sup.iloc[:-24]
sen_sup.count()

for i in range(0,sen_sup.Activity.count()-1,45):
    temp = sen_sup.iloc[i:i+45]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
         'meanGyrX':temp.Gyr_X.mean(),
         'meanGyrY':temp.Gyr_Y.mean(),
         'meanGyrZ':temp.Gyr_Z.mean(),
         'maxAccX' :temp.Acc_X.max(),
         'maxAccY' :temp.Acc_Y.max(),
         'maxAccZ' :temp.Acc_Z.max(),
         'maxGyrX' :temp.Gyr_X.max(),
         'maxGyrY' :temp.Gyr_Y.max(),
         'maxGyrZ' :temp.Gyr_Z.max(),
         'minAccX' :temp.Acc_X.min(),
         'minAccY' :temp.Acc_Y.min(),
         'minAccZ' :temp.Acc_Z.min(),
         'minGyrX' :temp.Gyr_X.min(),
         'minGyrY' :temp.Gyr_Y.min(),
         'minGyrZ' :temp.Gyr_Z.min(),
         'sdAccX'  :temp.Acc_X.std(),
         'sdAccY'  :temp.Acc_Y.std(),
         'sdAccZ'  :temp.Acc_Z.std(),
         'sdGyrX'  :temp.Gyr_X.std(),
         'sdGyrY'  :temp.Gyr_Y.std(),
         'sdGyrZ'  :temp.Gyr_Z.std(),
         'madAccX' :temp.Acc_X.mad(),
         'madAccY' :temp.Acc_Y.mad(),
         'maxAccZ' :temp.Acc_Z.mad(),
         'madGyrX' :temp.Gyr_X.mad(),
         'madGyrY' :temp.Gyr_Y.mad(),
         'maxGyrZ' :temp.Gyr_Z.mad(),
         'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

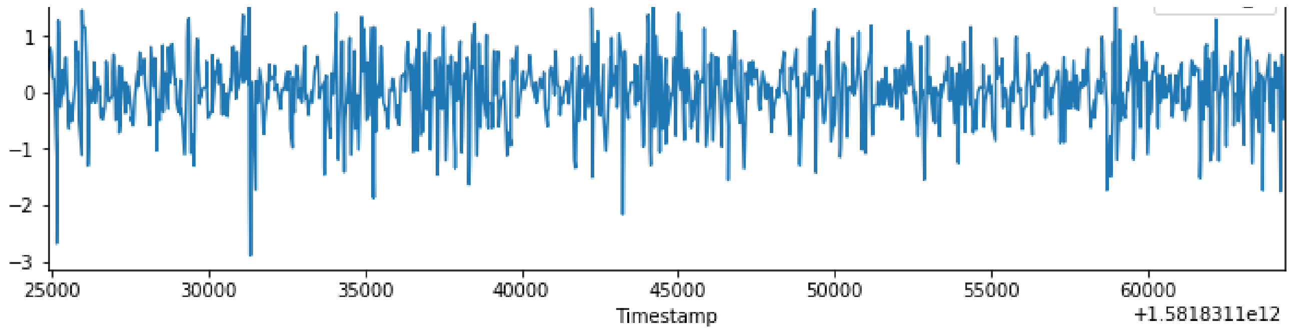
In [32]:

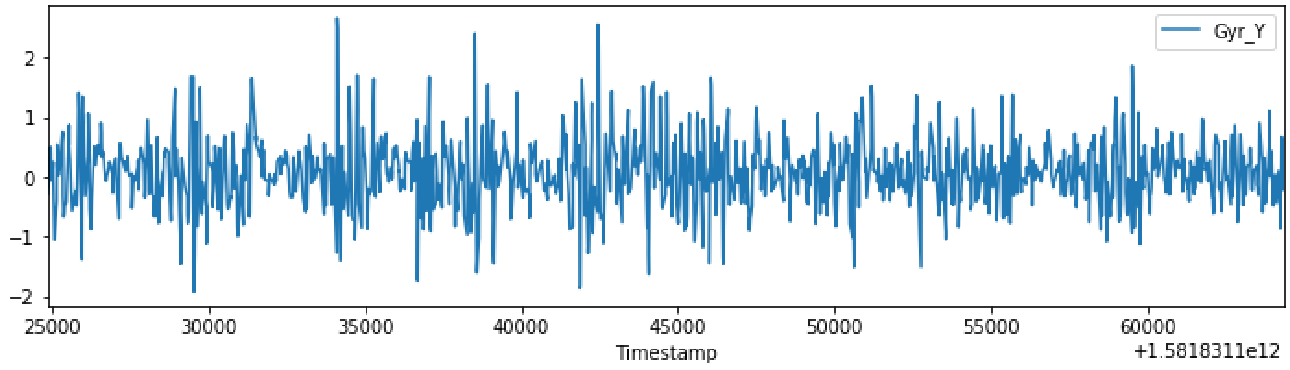
```
sen_sdn.plot(x='Timestamp',y='Acc_X', title='Accelerometer readings for Climbing down sta
irs')
plt.show()
sen_sdn.plot(x='Timestamp',y='Acc_Y')
plt.show()
sen_sdn.plot(x='Timestamp',y='Acc_Z')
plt.show()
sen_sdn.plot(x='Timestamp',y='Gyr_X', title='Gyroscope readings for Climbing down stairs'
)
plt.show()
sen_sdn.plot(x='Timestamp',y='Gyr_Y')
plt.show()
sen_sdn.plot(x='Timestamp',y='Gyr_Z')
plt.show()
```

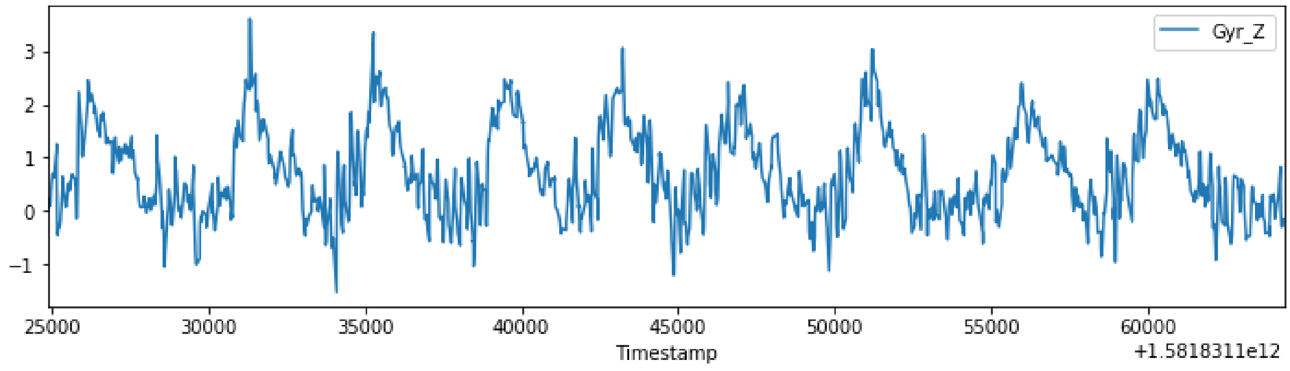










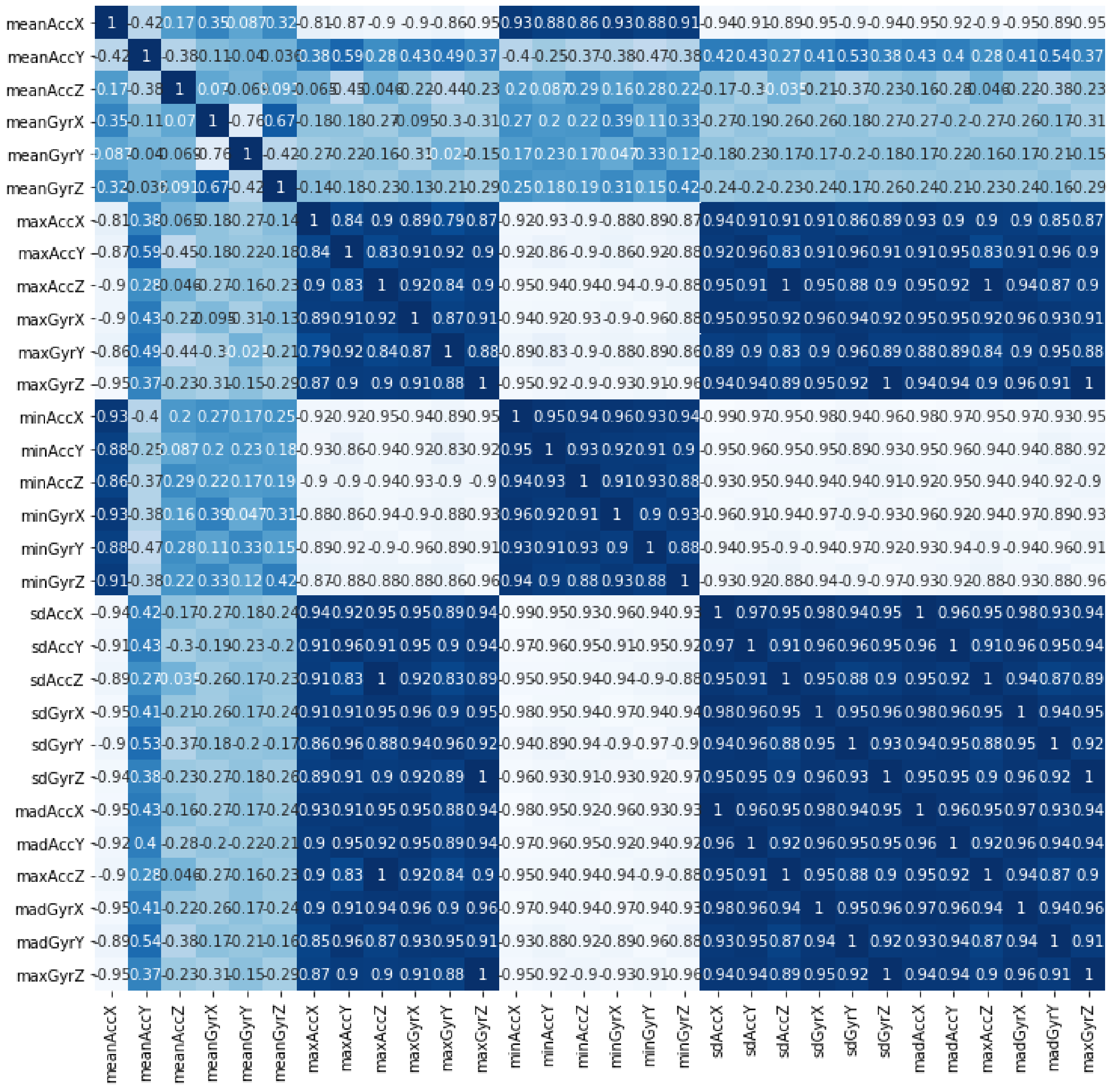


In [33]:

```
# For climbing down stairs, it is observed that, a cycle of activity repeats at around 45
samples
# Window of 45 is cut and parameters are computed.
sen_sdn = sen_sdn.iloc[:-13]
sen_sdn.count()

for i in range(0,sen_sdn.Activity.count()-1,45):
    temp = sen_sdn.iloc[i:i+45]
    f = {'meanAccX':temp.Acc_X.mean(),
         'meanAccY':temp.Acc_Y.mean(),
         'meanAccZ':temp.Acc_Z.mean(),
         'meanGyrX':temp.Gyr_X.mean(),
         'meanGyrY':temp.Gyr_Y.mean(),
         'meanGyrZ':temp.Gyr_Z.mean(),
         'maxAccX' :temp.Acc_X.max(),
         'maxAccY' :temp.Acc_Y.max(),
         'maxAccZ' :temp.Acc_Z.max(),
         'maxGyrX' :temp.Gyr_X.max(),
         'maxGyrY' :temp.Gyr_Y.max(),
         'maxGyrZ' :temp.Gyr_Z.max(),
         'minAccX' :temp.Acc_X.min(),
         'minAccY' :temp.Acc_Y.min(),
         'minAccZ' :temp.Acc_Z.min(),
         'minGyrX' :temp.Gyr_X.min(),
         'minGyrY' :temp.Gyr_Y.min(),
         'minGyrZ' :temp.Gyr_Z.min(),
         'sdAccX'  :temp.Acc_X.std(),
         'sdAccY'  :temp.Acc_Y.std(),
         'sdAccZ'  :temp.Acc_Z.std(),
         'sdGyrX'  :temp.Gyr_X.std(),
         'sdGyrY'  :temp.Gyr_Y.std(),
         'sdGyrZ'   :temp.Gyr_Z.std(),
         'madAccX' :temp.Acc_X.mad(),
         'madAccY' :temp.Acc_Y.mad(),
         'maxAccZ' :temp.Acc_Z.mad(),
         'madGyrX' :temp.Gyr_X.mad(),
         'madGyrY' :temp.Gyr_Y.mad(),
         'maxGyrZ' :temp.Gyr_Z.mad(),
         'Activity':temp.Activity[i]}
    dataset = dataset.append(f,ignore_index=True)
```

In [34]:

```
plt.figure(figsize=(12,12))
ax = sns.heatmap(
    dataset.corr(),
    annot=True,
    cbar=False,
    cmap='Blues'
)
```



In [35]:

```
X = dataset.loc[:, dataset.columns != 'Activity']
Y = dataset.loc[:, dataset.columns == 'Activity']
```

In [36]:

```
# Split the data in 70:30 ratio for train and test
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.3, random_state=42
)
```

In [37]:

```
clf = LogisticRegression(random_state=0, multi_class='multinomial', solver='newton-cg')
model = clf.fit(x_train, y_train.Activity)
y_predicted = model.predict(x_test)
```

In [38]:

```
model.score(x_test,y_test)
```

Out[38]:

```
0.9828571428571429
```

In [39]:

```
print(confusion_matrix(y_test, y_predicted))
```

```
[[34  1  0  0  0  0]
 [ 0 13  0  0  0  0]
 [ 0  0 49  1  0  0]
 [ 0  0  0 14  0  0]
 [ 0  0  0  0 19  1]
 [ 0  0  0  0  0 43]]
```

In [40]:

```
print(classification_report(y_test,
                            y_predicted,
                            target_names=['JOGGING','RUNNING','SITTING','STAIRCLIMB','ST
AIRDOWN','WALKING']))
```

```
              precision    recall  f1-score   support

     JOGGING       1.00      0.97      0.99        35
     RUNNING       0.93      1.00      0.96        13
     SITTING       1.00      0.98      0.99        50
  STAIRCLIMB       0.93      1.00      0.97        14
   STAIRDOWN       1.00      0.95      0.97        20
     WALKING       0.98      1.00      0.99        43

    accuracy                           0.98       175
   macro avg       0.97      0.98      0.98       175
weighted avg       0.98      0.98      0.98       175
```