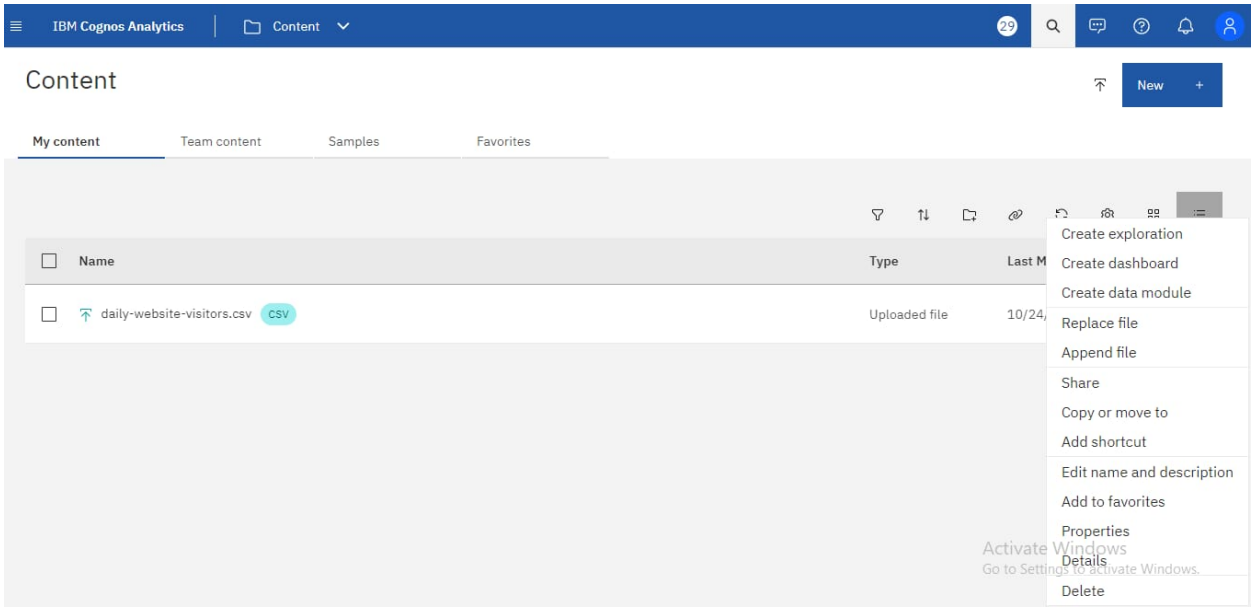
WEBSITE TRAFFIC ANALYSIS

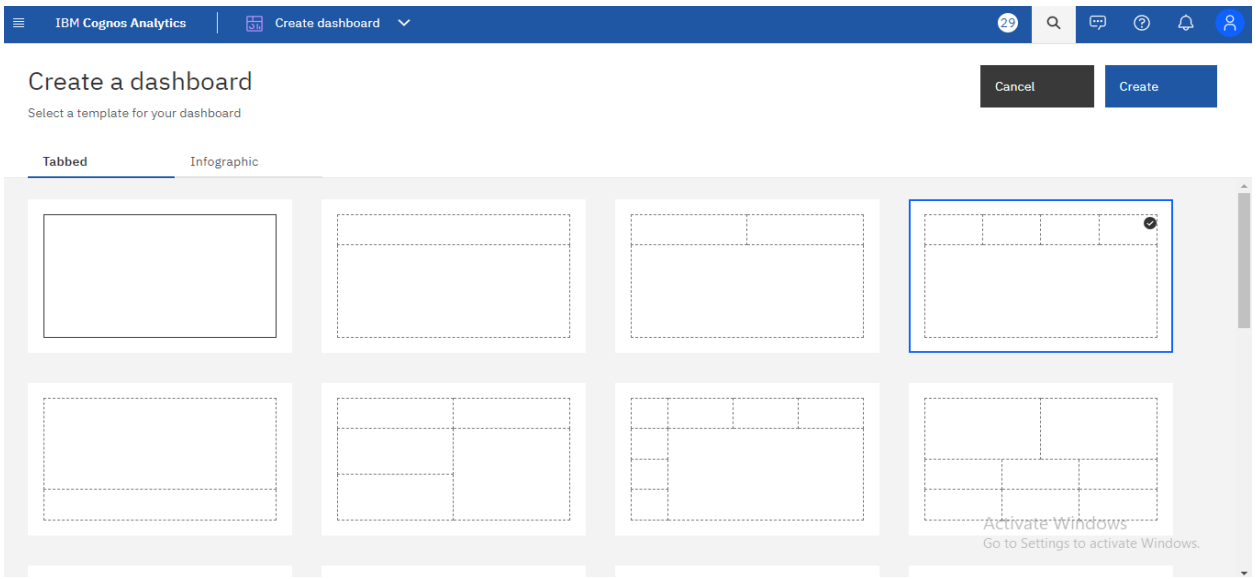
ADVANCE VISUALIZATION USING IBM COGNOS

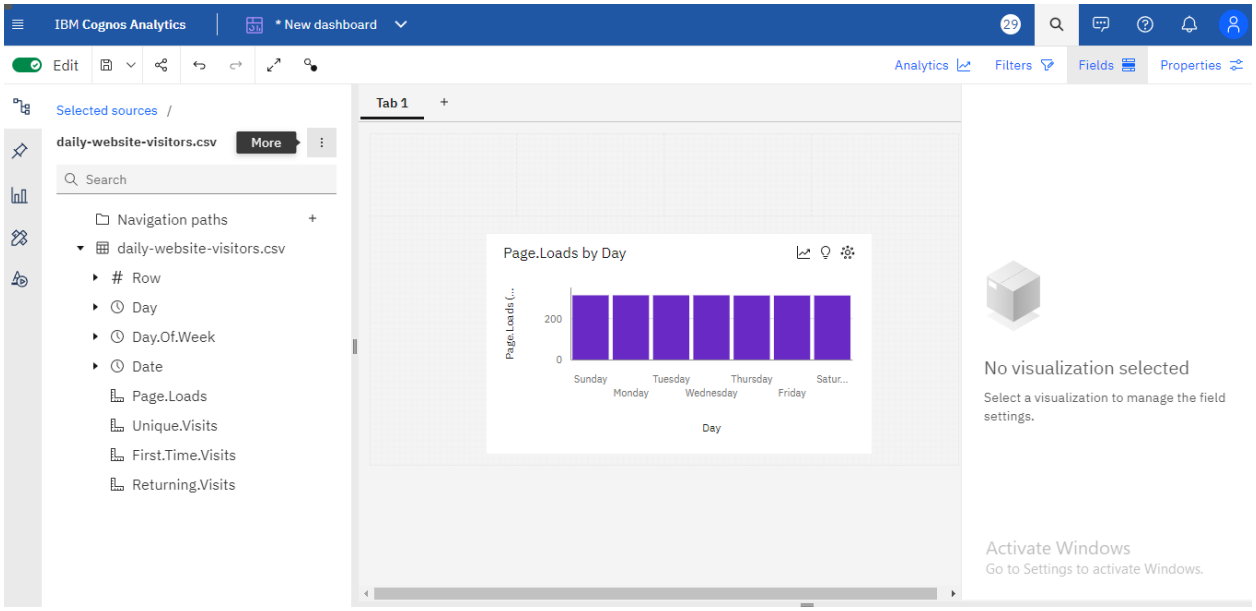
PHASE 4: DEVELOPMENTPART 2

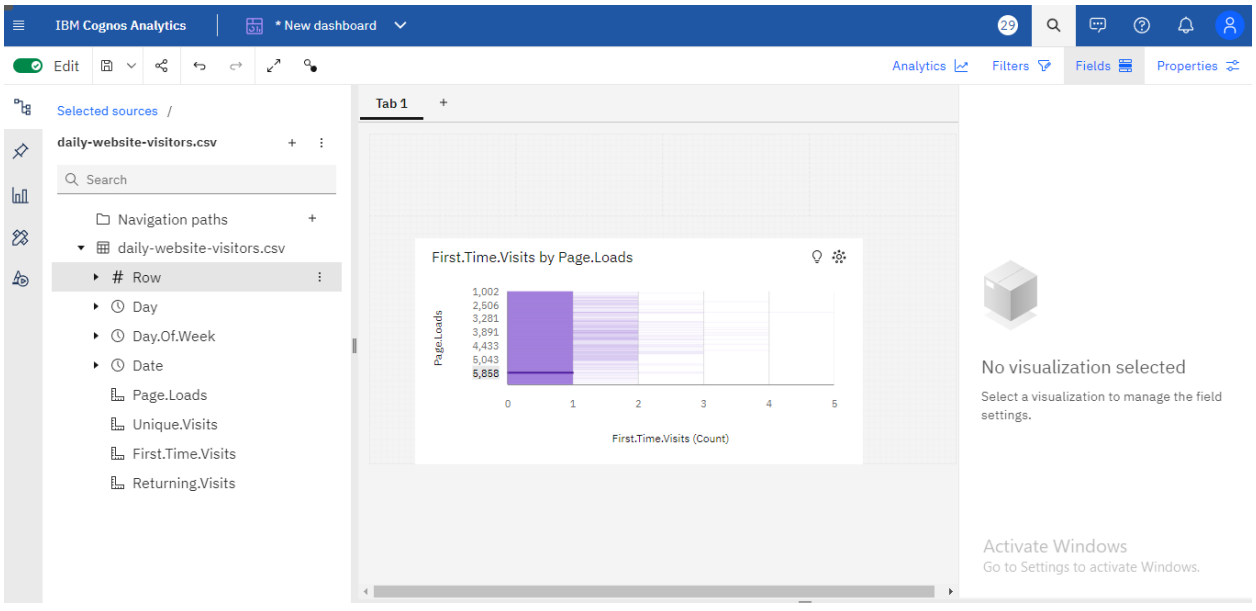
STEP1: Creating Dashboard

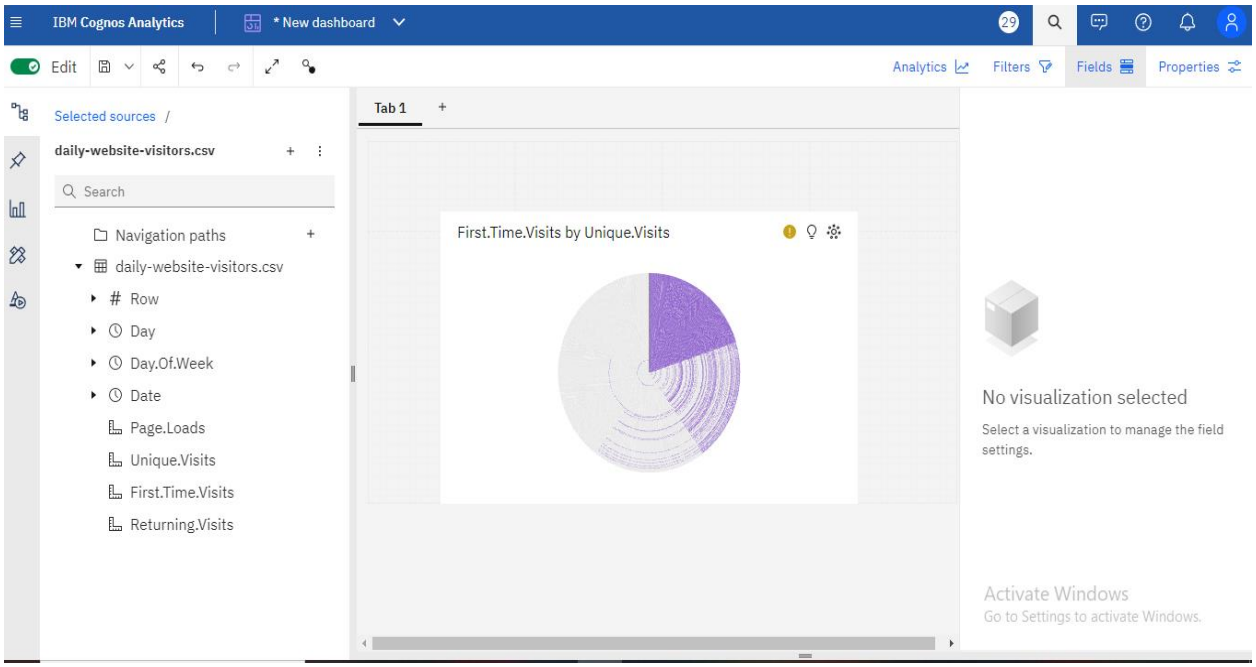
Create the dashboard with dataset.

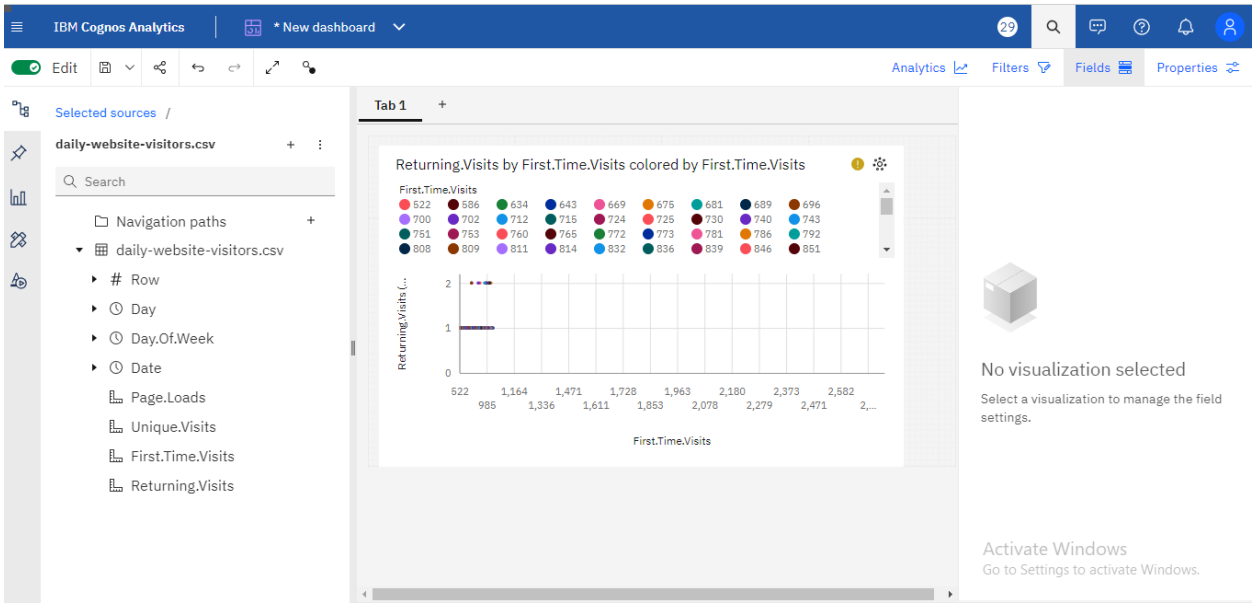





IBM Cognos Analytics
* New dashboard
29
Edit
Analytics
Filters
Fields
Properties
Selected sources /
daily-website-visitors.csv
More
Search
Navigation paths
daily-website-visitors.csv
Row
Day
Day.Of.Week
Date
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Tab 1
Page.Loads by Day
Page.Loads (...
200
0
Sunday
Monday
Tuesday
Wednesday
Thursday
Friday
Satur...
Day
No visualization selected
Select a visualization to manage the field settings.
Activate Windows
Go to Settings to activate Windows.


IBM Cognos Analytics
* New dashboard
29
Edit
Analytics
Filters
Fields
Properties
Selected sources /
daily-website-visitors.csv
Search
Navigation paths
daily-website-visitors.csv
Row
Day
Day.Of.Week
Date
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Tab 1
First.Time.Visits by Page.Loads
Page.Loads
1,002
2,506
3,281
3,891
4,433
5,043
5,858
0
1
2
3
4
5
First.Time.Visits (Count)
No visualization selected
Select a visualization to manage the field settings.
Activate Windows
Go to Settings to activate Windows.


IBM Cognos Analytics
* New dashboard
29
Edit
Analytics
Filters
Fields
Properties
Selected sources /
daily-website-visitors.csv
Search
Navigation paths
daily-website-visitors.csv
Row
Day
Day.Of.Week
Date
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Tab 1
First.Time.Visits by Unique.Visits
No visualization selected
Select a visualization to manage the field settings.
Activate Windows
Go to Settings to activate Windows.


IBM Cognos Analytics
* New dashboard
29
Edit
Analytics
Filters
Fields
Properties
Selected sources /
daily-website-visitors.csv
Search
Navigation paths
daily-website-visitors.csv
Row
Day
Day.Of.Week
Date
Page.Loads
Unique.Visits
First.Time.Visits
Returning.Visits
Tab 1
Returning.Visits by First.Time.Visits colored by First.Time.Visits
First.Time.Visits
522 586 634 643 669 675 681 689 696
700 702 712 715 724 725 730 740 743
751 753 760 765 772 773 781 786 792
808 809 811 814 832 836 839 846 851
Returning.Visits (...
2
1
0
522 1,164 1,471 1,728 1,963 2,180 2,373 2,582
985 1,336 1,611 1,853 2,078 2,279 2,471 2,...
First.Time.Visits
No visualization selected
Select a visualization to manage the field settings.
Activate Windows
Go to Settings to activate Windows.

STEP 2: Using Google colab

In [1]:

```
import pandas as pd

FILE_LOCATION = '/kaggle/input/daily-website-visitors/daily-website-visitor
s.csv'

whole_dataset = pd.read_csv(FILE_LOCATION,
                            index_col='Date',
                            thousands=',')
whole_dataset.index = pd.to_datetime(whole_dataset.index)
whole_dataset
```

Out[1]:

	Row	Day	Day.Of.Week	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits
Date							
2014-09-14	1	Sunday	1	2146	1582	1430	152
2014-09-15	2	Monday	2	3621	2528	2297	231
2014-09-16	3	Tuesday	3	3698	2630	2352	278
2014-09-17	4	Wednesday	4	3667	2614	2327	287
2014-09-18	5	Thursday	5	3316	2366	2130	236
...	...	...	...	...	...	...	...
2020-08-15	2163	Saturday	7	2221	1696	1373	323
2020-08-16	2164	Sunday	1	2724	2037	1686	351
2020-08-17	2165	Monday	2	3456	2638	2181	457
2020-08-18	2166	Tuesday	3	3581	2683	2184	499
2020-08-19	2167	Wednesday	4	2064	1564	1297	267

Activ

In [3]:

```
whole_dataset.describe()
```

Out[3]:

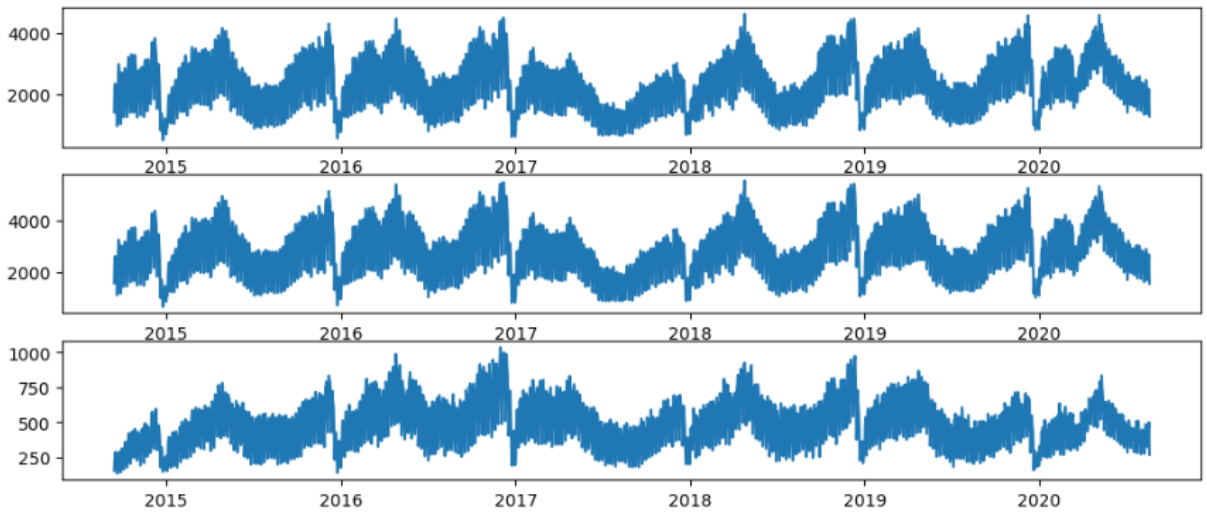
	Row	Day.Of.Week	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits
count	2167.000000	2167.000000	2167.000000	2167.000000	2167.000000	2167.000000
mean	1084.000000	3.997231	4116.989386	2943.646516	2431.824181	511.822335
std	625.703338	2.000229	1350.977843	977.886472	828.704688	168.736370
min	1.000000	1.000000	1002.000000	667.000000	522.000000	133.000000
25%	542.500000	2.000000	3114.500000	2226.000000	1830.000000	388.500000
50%	1084.000000	4.000000	4106.000000	2914.000000	2400.000000	509.000000
75%	1625.500000	6.000000	5020.500000	3667.500000	3038.000000	626.500000
max	2167.000000	7.000000	7984.000000	5541.000000	4616.000000	1036.000000

In [4]:

```
import matplotlib.pyplot as plt

fig, axs = plt.subplots(3, figsize=(12, 5))

axs[0].plot(whole_dataset['First.Time.Visits'])
axs[1].plot(whole_dataset['Unique.Visits'])
axs[2].plot(whole_dataset['Returning.Visits'])
plt.show()
```



In [5]:

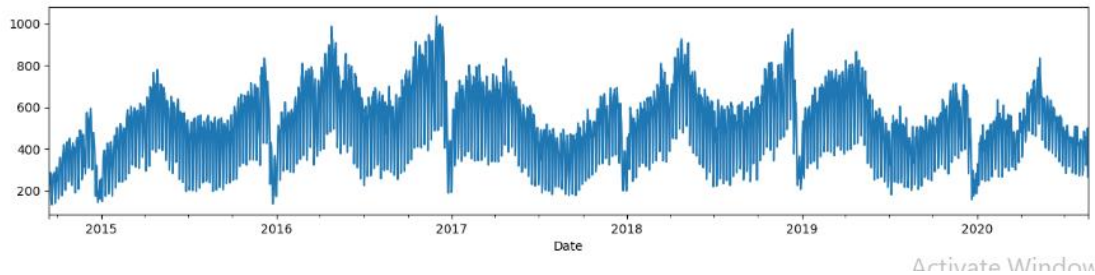
```
target_column = whole_dataset['Returning.Visits']
target_column
```

Out[5]:

```
Date
2014-09-14    152
2014-09-15    231
2014-09-16    278
2014-09-17    287
2014-09-18    236
              ...
2020-08-15    323
2020-08-16    351
2020-08-17    457
2020-08-18    499
2020-08-19    267
Name: Returning.Visits, Length: 2167, dtype: int64
```

In [6]:

```
target_column.plot(figsize=(15, 3))
plt.show()
```



In [7]:
```
len(target_column)
```

Out[7]:
```
2167
```

In [8]:
```
TEST_DATA_PERCENTAGE = 0.1

TEST_DATA_BOUNDARY_INDEX = int((1 - TEST_DATA_PERCENTAGE) * len(target_column))
print(f"Train data:\tReturning Visits [:{TEST_DATA_BOUNDARY_INDEX}] ({TEST_DATA_BOUNDARY_INDEX + 1})")
print(f"Test data:\tReturning Visits [{TEST_DATA_BOUNDARY_INDEX}:] ({len(target_column) - TEST_DATA_BOUNDARY_INDEX})")
print(f"\nLast target on train data: {target_column[TEST_DATA_BOUNDARY_INDEX]}")
```

```
Train data:     Returning Visits [:1950] (1951)
Test data:      Returning Visits [1950:] (217)

Last target on train data: 441
```

In [11]:
```
target_column[TEST_DATA_BOUNDARY_INDEX-10:TEST_DATA_BOUNDARY_INDEX+10].values, (list(train_dataset)[-1][0][-1].numpy(), list(train_dataset)[-1][1][-1].numpy())
```

Out[11]:
```
(array([429, 423, 442, 464, 372, 253, 277, 515, 434, 394, 441, 413, 246,
        314, 443, 484, 473, 490, 353, 249]),
 (array([277, 515, 434]), 394))
```

In [12]:
```
test_dataset = timeseries_dataset_from_array(target_column[TEST_DATA_BOUNDARY_INDEX - WINDOW_SIZE:],
                                             target_column[TEST_DATA_BOUNDARY_INDEX:],
                                             sequence_length=WINDOW_SIZE
                                            )
len(test_dataset), len(list(test_dataset.unbatch()))
```

Out[12]:
```
(2, 217)
```

In [13]:
```
target_column[TEST_DATA_BOUNDARY_INDEX-10:TEST_DATA_BOUNDARY_INDEX+10].values, list(test_dataset)[0]
[0][0].numpy(), list(test_dataset)[0][1][0].numpy()
```

Out[13]:
```
(array([429, 423, 442, 464, 372, 253, 277, 515, 434, 394, 441, 413, 246,
        314, 443, 484, 473, 490, 353, 249]),
 array([515, 434, 394]),
 441)
```

In [14]:
```
# First point in test dataset
list(test_dataset)[0][0][0].numpy(), list(test_dataset)[0][1][0].numpy()
```

Out[14]:
```
(array([515, 434, 394]), 441)
```

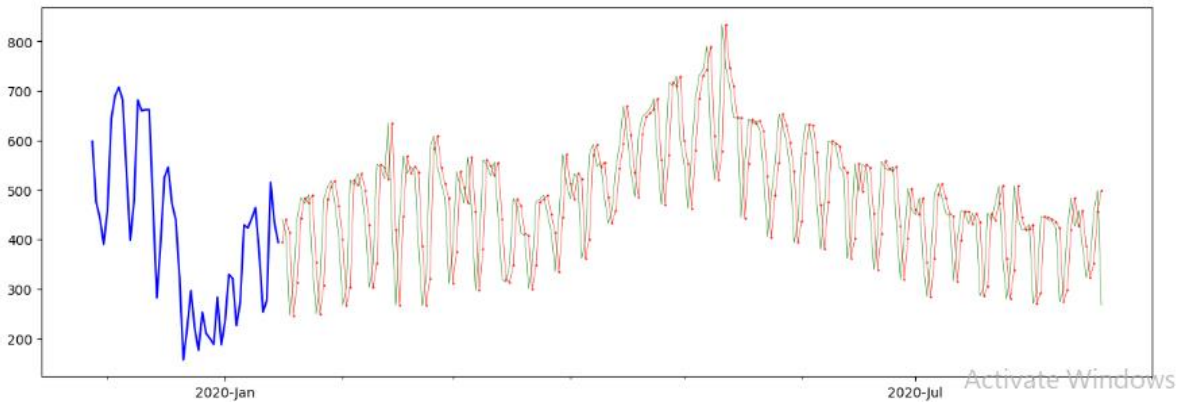
In [15]:
```
# Last point in test dataset
list(test_dataset)[-1][0][-1].numpy(), list(test_dataset)[-1][1][-1].numpy()
```

Out[15]:

```
(array([351, 457, 499]), 267)
```

In [20]:

```
plot_time_series(baseline_predictions.ravel(), start_index=1900)
```



In [21]:

```
y_true = target_column[TEST_DATA_BOUNDARY_INDEX : ]

len(y_true), y_true
```

Out[21]:

```
(217,
 Date
 2020-01-16    441
 2020-01-17    413
 2020-01-18    246
 2020-01-19    314
 2020-01-20    443
               ...
 2020-08-15    323
 2020-08-16    351
 2020-08-17    457
 2020-08-18    499
 2020-08-19    267
 Name: Returning.Visits, Length: 217, dtype: int64)
```

In [22]:

```
from sklearn.metrics import mean_absolute_error, mean_squared_error, mean_absolute_percentage_error

def evaluate_predictions(y_true, y_preds):
    mae = mean_absolute_error(y_true, y_preds)
    mse = mean_squared_error(y_true, y_preds)
    rmse = np.sqrt(mse)
    mape = mean_absolute_percentage_error(y_true, y_preds)

    return {
        'mae': mae,
        'mse': mse,
        "rmse": rmse,
        "mape": mape
    }

evaluate_predictions(y_true, baseline_predictions)
```

Out[22]:

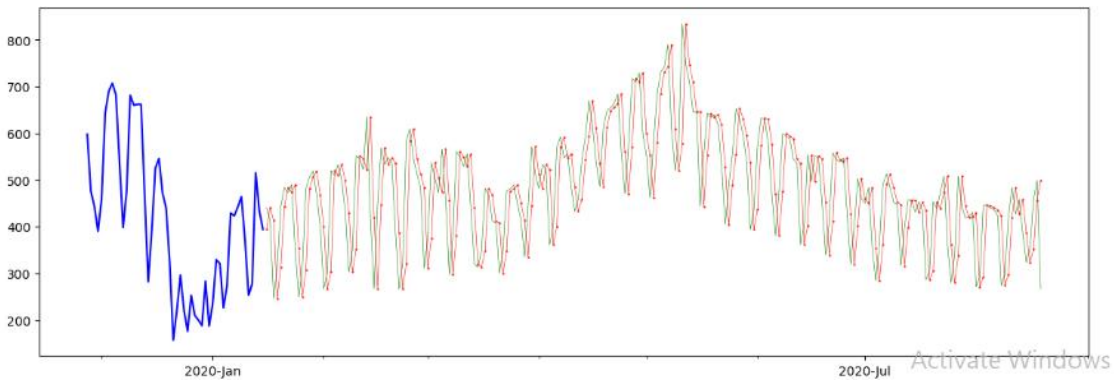
```
{'mae': 72.19815668202764,
 'mse': 8508.622119815669,
 'rmse': 92.24219273096054,
 'mape': 0.16713927858326993}
```

In [24]:

```
evaluate_model(baseline_model)
```

Out[24]:

```
{'mae': 72.19815668202764,
 'mse': 8508.622119815669,
 'rmse': 92.24219273096054,
 'mape': 0.16713927858326993}
```



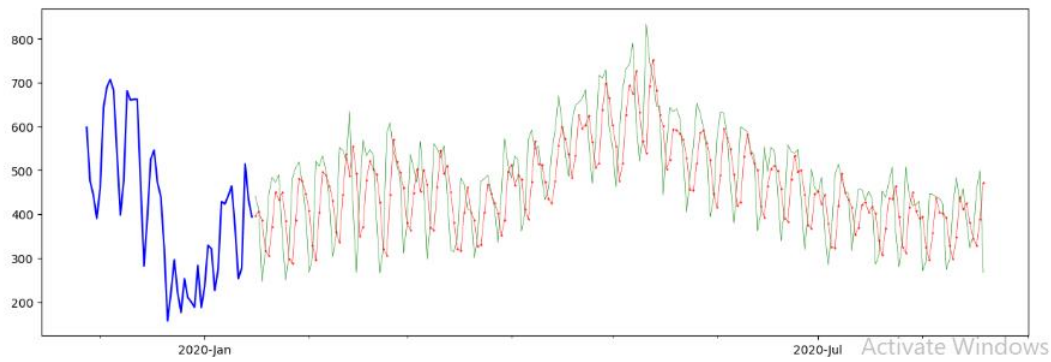
In [25]:
```
MODEL_METRICS
```

Out[25]:

	mae	mse	rmse	mape
model_0	72.198157	8508.62212	92.242193	0.167139

In [28]:
```
evaluate_model(model_1)
```

Out[28]:
```
{'mae': 75.97736101985527,
 'mse': 9384.529501460089,
 'rmse': 96.8737812901927,
 'mape': 0.16859899052082924}
```



In [29]:
```
MODEL_METRICS
```

Out[29]:

	mae	mse	rmse	mape
model_0	72.198157	8508.622120	92.242193	0.167139
model_1	75.977361	9384.529501	96.873781	0.168599

In [30]:

```
unbatched_train_dataset = whole_dataset[:TEST_DATA_BOUNDARY_INDEX + 1].copy()
unbatched_train_dataset
```

Out[30]:

	Row	Day	Day.Of.Week	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits
Date							
2014-09-14	1	Sunday	1	2146	1582	1430	152
2014-09-15	2	Monday	2	3621	2528	2297	231
2014-09-16	3	Tuesday	3	3698	2630	2352	278
2014-09-17	4	Wednesday	4	3667	2614	2327	287
2014-09-18	5	Thursday	5	3316	2366	2130	236
...	...	...	...	...	...	...	...
2020-01-12	1947	Sunday	1	2762	2238	1961	277
2020-01-13	1948	Monday	2	4298	3242	2727	515
2020-01-14	1949	Tuesday	3	3838	2884	2450	434
2020-01-15	1950	Wednesday	4	3754	2864	2470	394
2020-01-16	1951	Thursday	5	3817	2951	2510	441

In [31]:

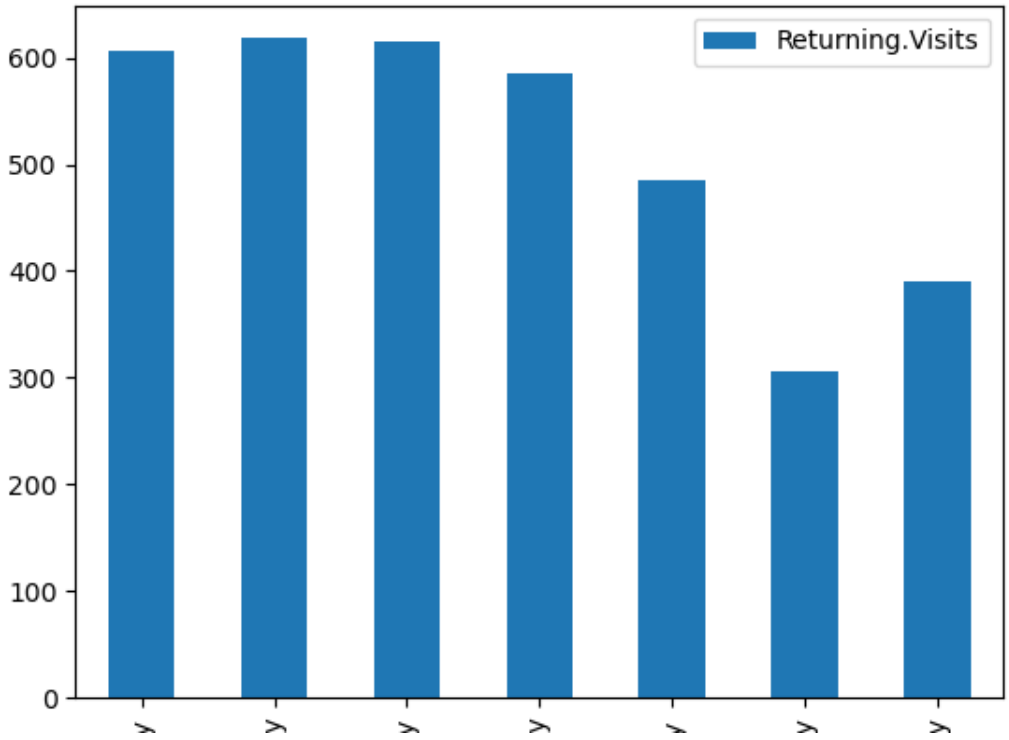
```
dataset_by_day = unbatched_train_dataset.groupby(by=['Day'])
dataset_by_day['Returning.Visits'].mean()
```

Out[31]:

```
Day
Friday       484.697842
Monday       606.512545
Saturday     306.071942
Sunday       390.573477
Thursday     584.627240
Tuesday      617.888889
Wednesday    614.369176
Name: Returning.Visits, dtype: float64
```

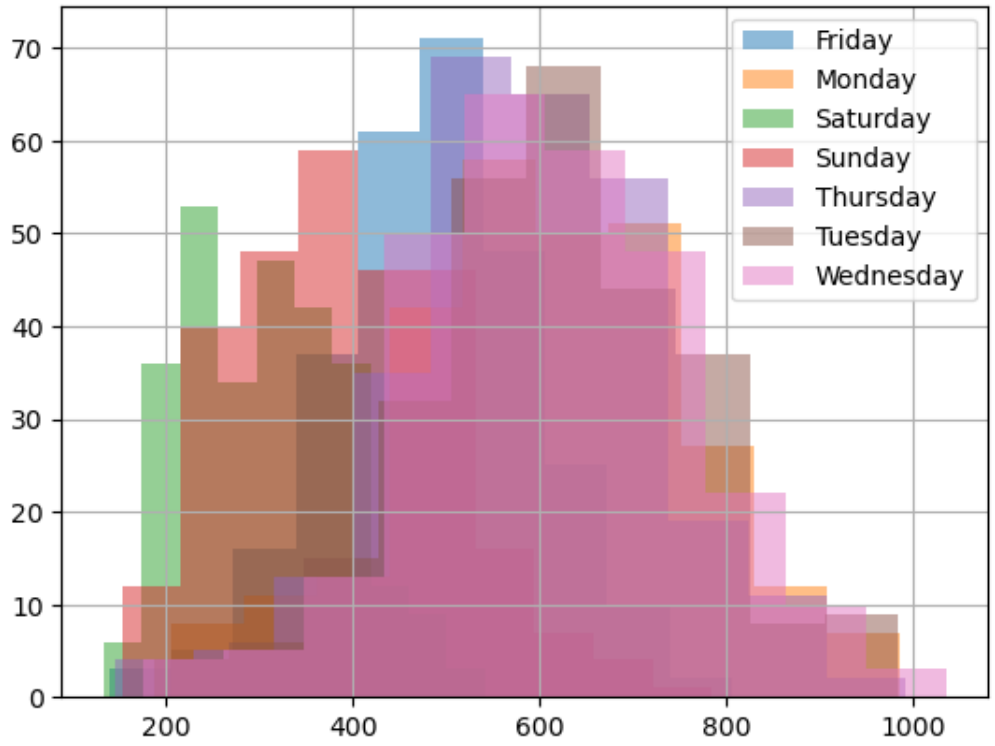
In [32]:

```
DAYS_OF_WEEK = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
pd.DataFrame(dataset_by_day['Returning.Visits'].mean()).loc[DAYS_OF_WEEK].plot(kind='bar')
```



In [33]:

```
dataset_by_day['Returning.Visits'].hist(legend=True, alpha=0.5)
plt.show()
```



In [34]:

```
import calendar

train_dataset_with_months = unbatched_train_dataset.copy()
train_dataset_with_months['Month.Name'] = pd.Series(train_dataset_with_months.index,
                                                    index=train_dataset_with_months.index)\
                                          .apply(lambda x: calendar.month_name[x.month])

train_dataset_with_months
```

Out[34]:

	Row	Day	Day.Of.Week	Page.Loads	Unique.Visits	First.Time.Visits	Returning.Visits	Month.Name
Date								
2014-09-14	1	Sunday	1	2146	1582	1430	152	September
2014-09-15	2	Monday	2	3621	2528	2297	231	September
2014-09-16	3	Tuesday	3	3698	2630	2352	278	September
2014-09-17	4	Wednesday	4	3667	2614	2327	287	September
2014-09-18	5	Thursday	5	3316	2366	2130	236	September
...	...	...	...	...	...	...	...	...
2020-01-12	1947	Sunday	1	2762	2238	1961	277	January
2020-01-13	1948	Monday	2	4298	3242	2727	515	January
2020-01-14	1949	Tuesday	3	3838	2884	2450	434	January
2020-01-15	1950	Wednesday	4	3754	2864	2470	394	January
2020-01-16	1951	Thursday	5	3817	2951	2510	441	January

1951 rows × 8 columns