WEBSITE TRAFFIC ANALYSIS

PHASE 4: **Development Part 2**

TOPIC:

Building the website traffic analysis using IBM Cognos for visualization define the objectives of the analysis and load website traffic data from the source shared.

* Using IBM Cognos to create interactive dashboards and reports that display insights such as popular pages, traffic sources, and user engagement metrics.
* Using Python libraries like Pandas and Matplotlib to perform more complex analyses on the data, such as time series analysis, user segmentation, or machine learning-based predictions.

Overview of Website Traffic

**Daily Visitors**

The number of daily visitors is a key metric for website traffic analysis. It can help identify trends, such as peak traffic times, and inform decisions on website design and content strategy.

### Traffic Sources

Understanding the sources of website traffic, such as organic search, social media, and referral links, can help optimize marketing efforts and improve overall website performance.

### Geographic Location of Visitors

Knowing the geographic location of website visitors can help tailor content and marketing efforts to specific regions or countries. It can also inform decisions on website translation and localization.

**DATA SET**

In [1]:

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

In [2]:

my\_data = pd.read\_csv("/kaggle/input/daily-website-visitors/daily-website-visitors.csv", delimiter=",")

In [3]:

linkcode

my\_data

Out[3]:

| Row | Day | Day.Of.Week | Date | Page.Loads | Unique.Visits | First.Time.Visits | Returning.Visits |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | Sunday | 1 | 9/14/2014 | 2,146 | 1,582 | 1,430 | 152 |
| 1 | 2 | Monday | 2 | 9/15/2014 | 3,621 | 2,528 | 2,297 | 231 |
| 2 | 3 | Tuesday | 3 | 9/16/2014 | 3,698 | 2,630 | 2,352 | 278 |
| 3 | 4 | Wednesday | 4 | 9/17/2014 | 3,667 | 2,614 | 2,327 | 287 |
| 4 | 5 | Thursday | 5 | 9/18/2014 | 3,316 | 2,366 | 2,130 | 236 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 2162 | 2163 | Saturday | 7 | 8/15/2020 | 2,221 | 1,696 | 1,373 | 323 |
| 2163 | 2164 | Sunday | 1 | 8/16/2020 | 2,724 | 2,037 | 1,686 | 351 |
| 2164 | 2165 | Monday | 2 | 8/17/2020 | 3,456 | 2,638 | 2,181 | 457 |
| 2165 | 2166 | Tuesday | 3 | 8/18/2020 | 3,581 | 2,683 | 2,184 | 499 |
| 2166 | 2167 | Wednesday | 4 | 8/19/2020 | 2,064 | 1,564 | 1,297 | 267 |

2167 rows × 8 columns

In [4]:

my\_data["Page.Loads"].map(lambda x: float(x.replace(",", "")))

Out[4]:

0 2146.0

1 3621.0

2 3698.0

3 3667.0

4 3316.0

...

2162 2221.0

2163 2724.0

2164 3456.0

2165 3581.0

2166 2064.0

Name: Page.Loads, Length: 2167, dtype: float64

In [5]:

X = (my\_data["Unique.Visits"].map(lambda x: float(x.replace(",", ""))))

*# X = my\_data["Unique.Visits"]*

In [6]:

X1 = pd.to\_datetime(my\_data["Date"])

X2 = pd.concat([X1, X], axis = 1)

X2

Out[6]:

|  | Date | Unique.Visits |
| --- | --- | --- |
| 0 | 2014-09-14 | 1582.0 |
| 1 | 2014-09-15 | 2528.0 |
| 2 | 2014-09-16 | 2630.0 |
| 3 | 2014-09-17 | 2614.0 |
| 4 | 2014-09-18 | 2366.0 |
| ... | ... | ... |
| 2162 | 2020-08-15 | 1696.0 |
| 2163 | 2020-08-16 | 2037.0 |
| 2164 | 2020-08-17 | 2638.0 |
| 2165 | 2020-08-18 | 2683.0 |
| 2166 | 2020-08-19 | 1564.0 |

2167 rows × 2 columns

In [7]:

X3 = pd.read\_csv("/kaggle/input/daily-website-visitors/daily-website-visitors.csv", index\_col = "Date", parse\_dates = True)

X3 = X3["Unique.Visits"].map(lambda x: float(x.replace(",", "")))

X3

Out[7]:

Date

2014-09-14 1582.0

2014-09-15 2528.0

2014-09-16 2630.0

2014-09-17 2614.0

2014-09-18 2366.0

...

2020-08-15 1696.0

2020-08-16 2037.0

2020-08-17 2638.0

2020-08-18 2683.0

2020-08-19 1564.0

Name: Unique.Visits, Length: 2167, dtype: float64

In [8]:

X2.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 2167 entries, 0 to 2166

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 Date 2167 non-null datetime64[ns]

1 Unique.Visits 2167 non-null float64

dtypes: datetime64[ns](1), float64(1)

memory usage: 34.0 KB

In [ ]:

# **Visualize the time series**

In [9]:

*# X = my\_data[["Row","Page.Loads"]].values*

In [10]:

*# X = X[0:50]*

In [11]:

X

Out[11]:

0 1582.0

1 2528.0

2 2630.0

3 2614.0

4 2366.0

...

2162 1696.0

2163 2037.0

2164 2638.0

2165 2683.0

2166 1564.0

Name: Unique.Visits, Length: 2167, dtype: float64

In [12]:

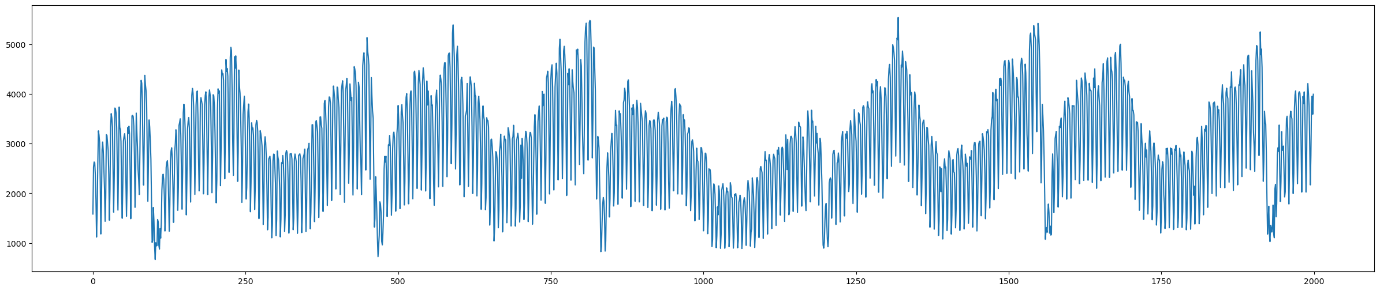
x = range(0,2000)

y = X[0:2000]

plt.figure(figsize=(30,6))

plt.plot(x,y)

plt.show()



In [13]:

import statsmodels.tsa.seasonal as statseason

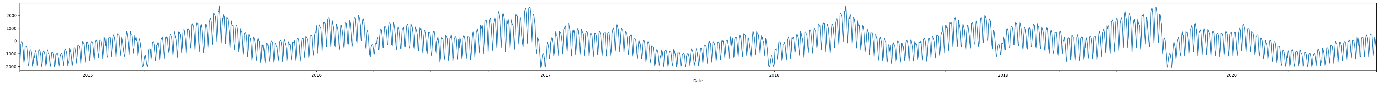
In [14]:

plt.figure(figsize=(60,3))

season = statseason.seasonal\_decompose(X3, model = "additive", period = 1000)

season.seasonal.plot()

plt.show()



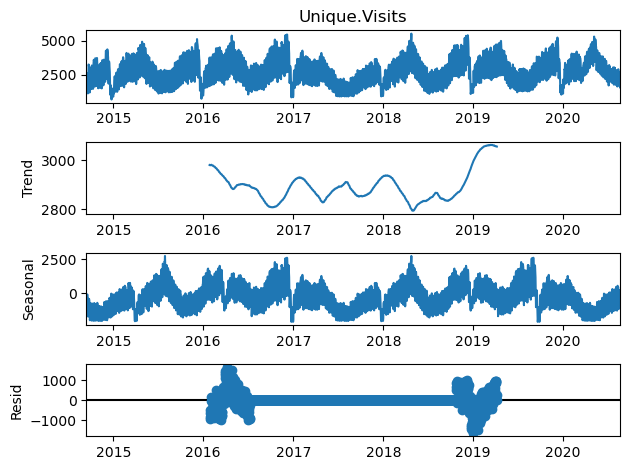
In [15]:

linkcode

season.plot()

plt.show()

plt.show()



# **Stationarity Testing**

Augmented Dickey–Fuller test

In [16]:

import statsmodels.tsa.stattools as stattools

In [17]:

adf\_test = stattools.adfuller(X)

*# Print the results*

print(adf\_test)

(-4.475968574445399, 0.00021726409300080682, 26, 2140, {'1%': -3.4334094211542983, '5%': -2.8628915360971003, '10%': -2.5674894918770197}, 29336.11247026125)

In [18]:

*# ADF Test*

result = stattools.adfuller(X, autolag='AIC')

*#Extracting the values from the results:*

print('ADF Statistic: **%f**' % result[0])

print('p-value: **%f**' % result[1])

print('Critical Values:')

for key, value **in** result[4].items():

print('**\t%s**: **%.3f**' % (key, value))

if result[0] < result[4]["5%"]:

print ("Reject H0 - Time Series is Stationary")

else:

print ("Failed to Reject Ho - Time Series is Non-Stationary")

ADF Statistic: -4.475969

p-value: 0.000217

Critical Values:

1%: -3.433

5%: -2.863

10%: -2.567

Reject H0 - Time Series is Stationary

In [ ]:

# **Seasonality Testing**

In [19]:

X

Out[19]:

0 1582.0

1 2528.0

2 2630.0

3 2614.0

4 2366.0

...

2162 1696.0

2163 2037.0

2164 2638.0

2165 2683.0

2166 1564.0

Name: Unique.Visits, Length: 2167, dtype: float64

BOUNCE RATE

### Causes of High Bounce Rate

Slow loading timesPoor website design and navigationIrrelevant or low-quality contentTechnical errors or broken links

### Reducing Bounce Rate

To reduce the bounce rate, it is important to improve website design and navigation, optimize page loading times, and provide high-quality, relevant content. Conducting A/B testing and analyzing user behavior can also help identify areas for improvement.

### Factors Affecting Conversion Rate

There are several factors that can impact a website's conversion rate, including:

* Website design and user experience
* Quality and relevance of website content
* Pricing and product offerings

**Conclusion:**

In conclusion, website traffic analysis using machine learning is a dynamic and evolving field with numerous applications. It empowers website owners, marketers, and IT teams to make data-driven decisions that enhance user experience, increase conversion rates, and ultimately drive the success of online platforms. As the volume of data continues to grow, machine learning will play an increasingly vital role in harnessing its full potential for website optimization.