WEBSITE TRAFFIC ANALYSIS

PHASE5: DOCUMENTATION

COLLEGE CODE & NAME: 5119 / Priyadarshini Engineering College

INTRODUCTION:

The popularity of the World-Wide Web(also called WWW, or the Web) has increased dramatically in the past few years. Data on the web is rapidly increasing day by day. Web is an open medium .Today, WWW traffic is one of the dominating components of Internet traffic. There are many reasons behind this explosive growth in Web traffic. These reasons include: the ease of use of the Web, the availability of graphical user interfaces for navigating the Web, the availability of editors and support tools for creating and publishing web documents, the machine-independent nature of the languages and protocols used for constructing and exchanging Web documents and a continuing exponential increase in the number of Internet hosts and users. The Web Server data is actually the user logs that are generated on the Web Server. These logs enable the analyst to keep a track of and analyze the user‘s behaviours who visit the website. The process of analyzing server performance begins with the collection of log files spanning some analysis time period. To understand traffic trends there is need to collect logs over a certain period. The phenomenal growth in Web traffic has sparked much research activity on improving the World-Wide Web. Much of this recent research activity has been aimed at improving Web performance. The key performance factors to consider are how to reduce the volume of network traffic produced by Web clients and servers, and how to improve the response time for WWW users. The website traffic analysis is a comprehensive analysis of visitor data to gain valuable insights into website performance and user behaviour.

ABSTRACT:

In this project on website traffic analysis, we present a comprehensive analysis of visitor data to gain valuable insights into website performance and user behaviour. By examining visitor patterns, trends, and engagement metrics, we aim to optimize the user experience and enhance website performance. Through data-driven decision-making, we can identify opportunities for improvement, increase traffic, and ultimately achieve business objectives. This analysis allows us to understand user behaviour, preferences, and interests, enabling us to tailor our website content and design to better meet their needs.

PROBLEM STATEMENT:

To understand the patterns, trends, and behaviour of visitors to a website in order to make data-driven decisions for improving user experience and increasing website performance.

In an increasingly digital world, understanding and optimizing website traffic is crucial for businesses and organizations to succeed online. The system should be capable of collecting, processing, and visualizing data from various sources, such as web server logs, analytics, and user interactions.

This involves analyzing website traffic data to gain insights into user behaviour, popular pages, and traffic sources. The goal is to help website owners enhance the user experience by understanding how visitors interact with the site. It encompasses defining the analysis objectives, collecting website traffic data, using IBM Cognos for data visualization, and integrating Python code for advanced analysis.

DESIGN THINKING:

1. ANALYSIS OBJECTIVES:

The primary objective of this project is to extract meaningful insights from the website traffic dataset. Key insights we aim to derive include:

- ✓ **Identifying Popular Pages:**
  - Determine which webpages on the website are the most visited or have the highest page views.
- ✓ **Traffic Trends:**
  - Analyze daily, weekly, and monthly traffic trends to understand patterns and seasonality.
- ✓ **User Engagement Metrics:**
  - Measure user engagement using metrics such as bounce rate, session duration, and the number of unique visitors.

2. DATA COLLECTION:

1) Data Source:

The data for this analysis will be sourced from the Kaggle dataset [Link], which provides daily website traffic statistics.

About Dataset:

Context:

The dataset contains 5 years of daily time series data for several measures of traffic on a statistical forecasting teaching notes website whose alias is statforecasting.com. The variables have complex seasonality that is keyed to the day of the week and to the academic calendar. The patterns you you see here are similar in principle to what you would see in other daily data with day-of-week and time-of-year effects. Some good exercises are to develop a 1-day-ahead forecasting model, a 7-day ahead forecasting model, and an entire-next-week forecasting model (i.e., next 7 days) for unique visitors.

Content:

The variables are daily counts of page loads, unique visitors, first-time visitors, and returning visitors to an academic teaching notes website. There are 2167 rows of data spanning the date range from September 14, 2014, to August 19, 2020. A visit is defined as a stream of hits on one or more pages on the site on a given day by the same user, as identified by IP address. Multiple individuals with a shared IP address (e.g., in a computer lab) are considered as a single user, so real users may be undercounted to some extent. A visit is classified as "unique" if a hit from the same IP address has not come within the last 6 hours. Returning visitors are identified by cookies if those are accepted. All others are classified as first-time visitors, so the count of unique visitors is the sum of the counts of returning and first-time visitors by definition. The data was collected through a traffic monitoring service known as StatCounter.

2) Data Collection Method:

Data collection will involve the following steps:

i. **Data Download:**
   Download the dataset from Kaggle or fetch it programmatically using Kaggle's API.

ii. **Data Cleaning:**
   Perform data cleaning to address missing values, outliers, or any inconsistencies in the dataset.

3. VISUALIZATION:

IBM Cognos Dashboard and Reports:

To visualize the insights, we will use IBM Cognos, which offers powerful visualization capabilities. The visualization plan includes:

1) Data Import:

Import the cleaned dataset into IBM Cognos.

2) Dashboard Design:

Create interactive dashboards containing graphs, charts, and KPIs to visualize popular pages, traffic trends, and user engagement metrics.

3) Report Generation:

Generate comprehensive reports summarizing the key insights and trends for stakeholders.

4) Automated Updates:

Implement automated updates to ensure that reports and dashboards are continually refreshed with new data.

4. PYTHON INTEGRATION:

Machine Learning Models:

Incorporating Python-based machine learning models will enhance the analysis and potentially predict future traffic trends or user behaviour patterns. Possible integration points include:

1) Time Series Forecasting:

Utilize time series forecasting models to predict future website traffic based on historical data.

2) User Segmentation:

Implement clustering algorithms like K-Means to segment users based on their behaviour, allowing for targeted marketing efforts.

3) Anomaly Detection:

Employ anomaly detection algorithms to identify unusual traffic patterns or detect potential security breaches.

4) Content Recommendation:

Develop recommendation systems to suggest content to users based on their past behaviour and preferences.

OBJECTIVES:

1. Data Collection:

Implement a data collection mechanism to gather information on user interactions, page views, session duration, and other relevant metrics. This should support multiple data sources and be scalable to handle large amounts of data.

2. Data Processing:

Develop data processing algorithms to clean, transform, and aggregate raw data into meaningful statistics. This includes identifying unique users, filtering out bots, and calculating performance metrics.

3. User Behaviour Analysis:

Create analytical models to understand user behaviour patterns, including common navigation paths, bounce rates, and conversion funnels. Detect anomalies and trends in user interactions.

4. Visualization:

Build a user-friendly dashboard or reporting tool to visualize website traffic data. Provide interactive charts, graphs, and reports that enable stakeholders to gain insights quickly and make informed decisions.

5. Integration:

Ensure compatibility with various web platforms, content management systems, and analytics tools. Offer the flexibility to integrate with third-party services and APIs.

6. Security and Privacy:

Implement robust security measures to protect sensitive user data and ensure compliance with data privacy regulations, such as GDPR and CCPA.

7. Performance Optimization:

Optimize the system for efficiency and scalability to handle high traffic websites without performance degradation.

8. Recommendations:

Develop a feature that provides recommendations for improving website performance and user engagement based on the analysis of traffic data.

9. User Training:

Provide training and documentation for users to effectively use the traffic analysis system.

APPLYING SOLUTION:

The following are the steps for applying the innovative solution:

1. **DATA COLLECTION:**

   The data for this analysis will be sourced from the Kaggle dataset [Link], which provides daily website traffic statistics.

2. **DATA CLEANING:**

   Perform data cleaning to address missing values, outliers, or any inconsistencies in the dataset.

3. **IMPORT DATA:**

   Import the cleaned dataset into IBM Cognos.

4. WORKING ON DATA USING MACHINE LEARNING MODELS:

Incorporating Python-based machine learning models will enhance the analysis and potentially predict future traffic trends or user behaviour patterns. The following are the machine learning models that are used in this project:

- **Regression Analysis:**
  Linear regression like polynomial regression are used to model and predict traffic trends over time.

- **Time Series Analysis:**
  Techniques like ARIMA (Auto Regressive Integrated Moving Average) are used in forecasting website traffic based on historical data.

- **Classification Algorithms:**
  These algorithms are used to categorize website visitors, such as decision trees, random forests, or support vector machines, to identify different user segments or traffic sources.

- **Clustering Algorithms:**
  K-Means clustering or hierarchical clustering can group website visitors or web pages based on similar behaviour, helping to understand user preferences and patterns.

- **Anomaly Detection:**
  Algorithms like Isolation Forest or One-Class SVM can detect unusual or fraudulent activities on the website, such as bot traffic or security breaches.

- **Reinforcement Learning:**
  In some cases, reinforcement learning models can optimize website elements for user engagement, such as A/B testing or content recommendation.

We can deploy these machine learning models to IBM Cognos Analytics using its built-in machine learning deployment capabilities and use it for prediction and analysis through visualizations.

5. VISUALIZING THE ANALYIS RESULT:

Creating dashboards in IBM Cognos allows us to combine various visualizations and key metrics on a single screen, offering a comprehensive overview of website traffic. Some of the Visualizations are:

- ✓ **Bar Charts:**

  Bar charts are useful for displaying metrics like the number of page views, unique visitors, or bounce rates over a specific time period. You can create bar charts to compare different time intervals or website sections.

- ✓ **Line Charts:**

  Line charts are effective for showing trends in website traffic data, such as changes in visitor numbers over time. They can help identify seasonal patterns and long-term trends.

- ✓ **Pie Charts:**

  Pie charts can be used to represent the distribution of traffic sources, showing the percentage of traffic coming from direct visits, search engines, social media, etc.

- ✓ **Area Charts:**

  Area charts are similar to line charts but can be used to display the cumulative effect of website traffic data over time. They are good for visualizing total page views or unique visitors.

- ✓ **Heat Maps:**

  Heat maps can provide insights into user engagement by showing which parts of a webpage receive the most clicks or interaction. This helps in optimizing the website's layout.

6. PREDICTING FUTURE TRAFFIC TRENDS:

Once we have deployed the machine learning models, we can use it to predict future traffic trends or user behaviour patterns. For example, we can use our model to predict:

" How much traffic we can expect to receive on the website on a given day or week? "
" Which pages of the website are most likely to be visited by a particular user segment? "
" Which products or services are most likely to be purchased by a particular user segment? "

7. KEY FINDINGS:

These are the key metrics to track for Website Traffic Improvement:

- **Visits:**
  An estimate of total visits to the website over the chosen month

- **Unique Visitors:**
  An estimate of total unique visitors to the website over the chosen month

- **Purchase Conversion:**
  Estimated percentage of sessions that ended with a checkout

- **Pages/Visit:**
  An estimate of how many pages (on average) a person visits in one session on the website

- **Average Visit Duration:**
  An average estimate of the amount of time spent on the site during each visit

- **Bounce Rate:**
  An estimate of the website's average bounce rate, or percentage of visitors that leave the website after viewing just one page

PERFORMING DATA PROCESSING AND CLEANING USING PYTHON:

WEBSITE TRAFFIC ANALYSIS

Importing libraries

In [5]:
```
import numpy as np
import pandas as pd
import warnings
warnings.filterwarnings('ignore')
```

Loading the dataset "Daily website visitors" which is downloaded from https://www.kaggle.com/bobnau/daily-website-visitors that contains 5 years of daily time series data for several measures of traffic on the website http://statforecasting.com/. So we are analyzing the traffic of this website in our project.

In [8]:
```
#import dataset
df=pd.read_csv(r'C:\Users\ELCOT\Downloads\daily-website-visitors.csv')
df.head()
```

Out[8]:

	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

Renaming the columns, removing the commas from the columns and converting their data types.

In [9]:
```
#data transformation
df.rename(columns = {'Day.Of.Week':'day_of_week'
                     ,'Page.Loads':'page_loads'
                     ,'Unique.Visits':'unique_visits'
                     ,'First.Time.Visits':'first_visits'
                     ,'Returning.Visits':'returning_visits'}, inplace = True)

df=df.replace(',','',regex=True)
df['page_loads']=df['page_loads'].astype(int)
df['unique_visits']=df['unique_visits'].astype(int)
df['first_visits']=df['first_visits'].astype(int)
df['returning_visits']=df['returning_visits'].astype(int)
```

In [10]:
```
df
```

Out[10]:

	Row	Day	day_of_week	Date	page_loads	unique_visits	first_visits	returning_visits
0	1	Sunday	1	9/14/2014	2146	1582	1430	152
1	2	Monday	2	9/15/2014	3621	2528	2297	231
2	3	Tuesday	3	9/16/2014	3698	2630	2352	278
3	4	Wednesday	4	9/17/2014	3667	2614	2327	287
4	5	Thursday	5	9/18/2014	3316	2366	2130	236
...	...	...	...	...	...	...	...	...
2162	2163	Saturday	7	8/15/2020	2221	1696	1373	323
2163	2164	Sunday	1	8/16/2020	2724	2037	1686	351
2164	2165	Monday	2	8/17/2020	3456	2638	2181	457
2165	2166	Tuesday	3	8/18/2020	3581	2683	2184	499
2166	2167	Wednesday	4	8/19/2020	2064	1564	1297	267

2167 rows × 8 columns

Performing data cleaning by checking for the null values and duplicate values in the dataset if present

In [11]:
```
#checking null values
df.isna().sum()
```

Out[11]:
```
Row             0
Day             0
day_of_week     0
Date            0
page_loads      0
unique_visits   0
first_visits    0
returning_visits    0
dtype: int64
```

In [12]:
```
#checking duplicate values
df.duplicated().sum()
```

Out[12]:
```
0
```

Thus the data is processed as well as cleaned and its accuracy and consistency is ensured.

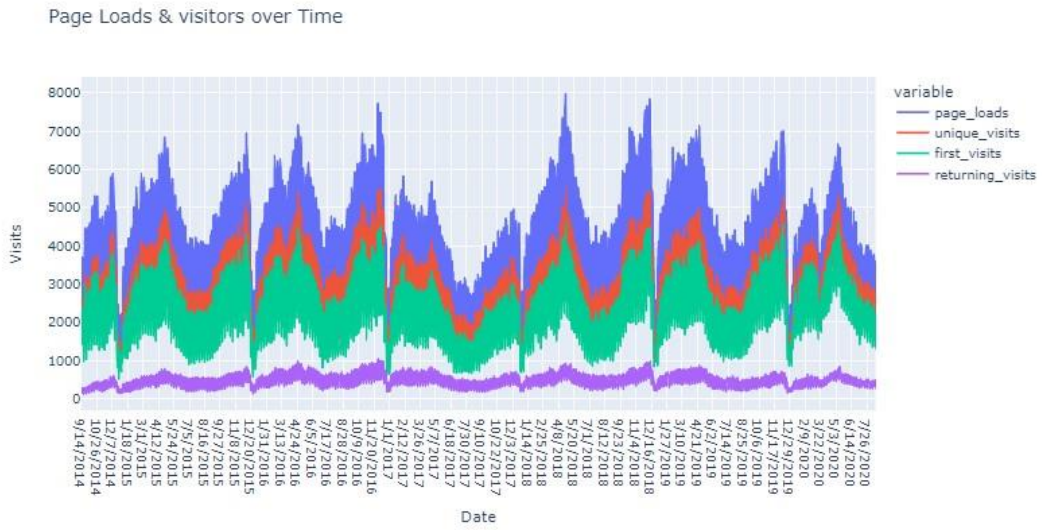
ANALYZING AND VISUALIZING USING PYTHON:

Now that the dataset is cleaned and processed we are now analyzing the data into visualizations to get insights.

VISUALIZING DATA:

Line plot Generating line plot for visualizing the trend of page loads and visits over time series, it seems that page loads and visits have a constant fluctuation, means they have trend over time and are correlated to each other.

In [7]:
```
import plotly.express as px
px.line(df,x='Date',y=['page_loads' ,'unique_visits' ,'first_visits' ,'returning_visits'],
       labels={'value':'Visits'}
       ,title='Page Loads & visitors over Time')
```



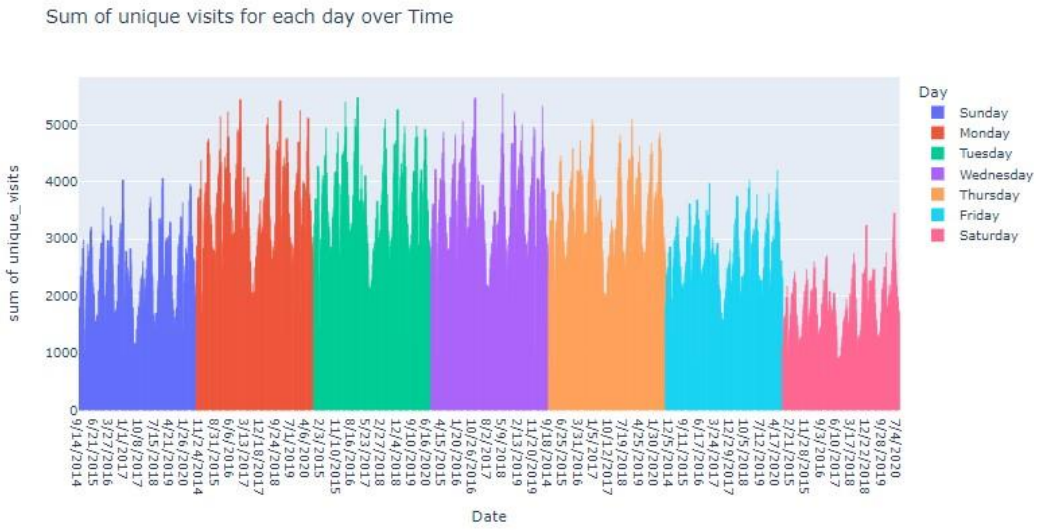
Analyzing unique visits(sum of unique visits) for each day of the week to understand how much traffic the website gets based on week days and which day has the most traffic.

In [8]:
```
day_imp=df.groupby(['Day'])['unique_visits'].agg(['sum']).sort_values(by='sum',ascending=False)
px.bar(day_imp,labels={'value':'sum of unique visits'},title='Sum of Unique visits for each day')
```

The above bar plot shows that monday, tuesday, wednesday and thursday are the days in a week when extensive amount of traffic come to this website.

Time Series Analysis The time series is used to understand which days get the most traffic and on what time intervals. Time intervals are grouped according to their relation with unique visits and days.

In [9]:
```
px.histogram(df,x='Date',y='unique_visits',color='Day',title='Sum of unique visits for each day over Time')
```



Now it is clear that in which days, months and years did the website get the most traffic.

Bar Graph Get the sum of page_loads, unique_visits, first_visits and returning_visits related to each of their days and group them together in order to create a bar chart.

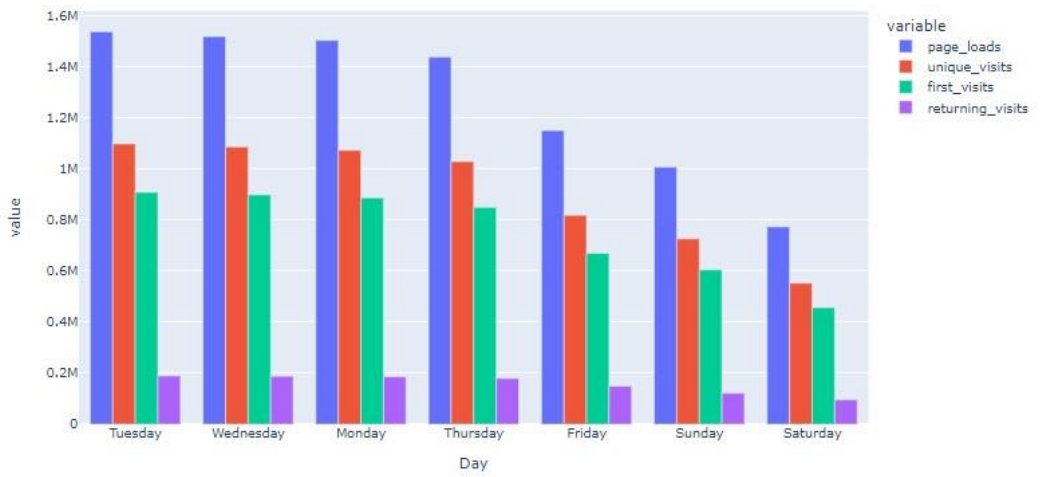
In [10]:
```
sums=df.groupby(['Day'])[['page_loads' ,'unique_visits' ,'first_visits' ,'returning_visits']].sum().sort_values(
    by='unique_visits',ascending=False)
sums
```

Out[10]:

Day	page_loads	unique_visits	first_visits	returning_visits
Tuesday	1536154	1097181	907752	189429
Wednesday	1517114	1085624	897602	188022
Monday	1502161	1072112	886036	186076
Thursday	1437269	1028214	848921	179293
Friday	1149437	817852	668805	149047
Sunday	1006564	725794	604198	121596
Saturday	772817	552105	456449	95656

Now a bar chart is created using the above calculated sum of data and it shows the sum of page_loads, unique_visits, first_visits, returning_visits for each day.

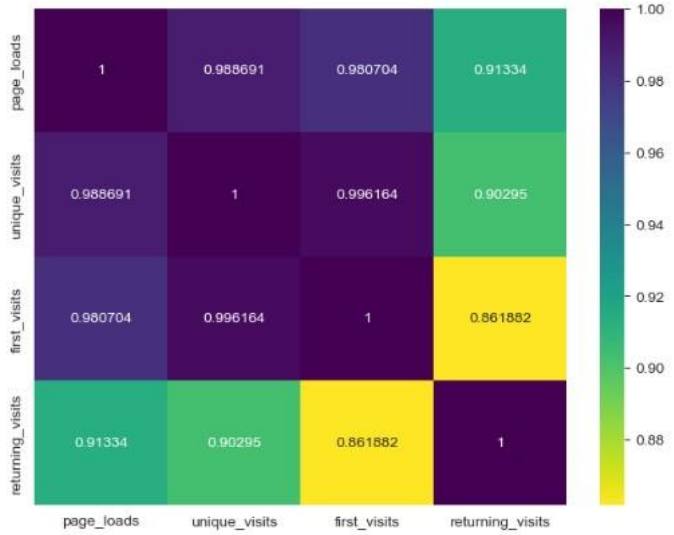
In [11]:
```
px.bar(sums,barmode='group')
```



Day

In [12]:
```
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
sns.set_style("whitegrid")
fig, ax = plt.subplots()
fig.set_size_inches(8, 6)
sns.heatmap(df[['page_loads' ,'unique_visits' ,'first_visits' ,'returning_visits']].corr(),
            annot=True,
            cmap='viridis_r',
            fmt='g')
```

Out[12]: <Axes: >



The above heat map shows the paired correlation of page_loads unique_visits first_visits returning_visits columns with annotated values we know that first visits and unique visits are correlated by 0.99 which is a great correlation and page loads have a good correlation with our target variable as well.

```
The above heat map shows the paired correlation of page_loads unique_visits first_visits returning_visits columns with
annotated values we know that first visits and unique visits are correlated by 0.99 which is a great correlation and page loads
have a good correlation with our target variable as well.
```

```
FEATURE ENGINEERING
    Tuesday, wednesday, thursday and monday are the days when our website received the most traffic so we will create a feature
days_f of them 1 value will define their existence and 0 will define the rest of the days.
```

In [13]:
```
pred_df=df[['page_loads' ,'unique_visits' ,'first_visits' ,'returning_visits','Day']]
pred_df['days_f']=np.where((df['Day']=='Tuesday') |
                           (df['Day']=='Wednesday') |
                           (df['Day']=='Thursday') |
                           (df['Day']=='Monday'),1,0)

pred_df
```

Out[13]:

	page_loads	unique_visits	first_visits	returning_visits	Day	days_f
0	2146	1582	1430	152	Sunday	0
1	3621	2528	2297	231	Monday	1
2	3698	2630	2352	278	Tuesday	1
3	3667	2614	2327	287	Wednesday	1
4	3316	2366	2130	236	Thursday	1
...	...	...	...	...	...	...
2162	2221	1696	1373	323	Saturday	0
2163	2724	2037	1686	351	Sunday	0
2164	3456	2638	2181	457	Monday	1
2165	3581	2683	2184	499	Tuesday	1
2166	2064	1564	1297	267	Wednesday	1

2167 rows × 6 columns

Multi Linear Regression model

In [19]:
```
#import libraries
from sklearn.model_selection import train_test_split, cross_val_score, GridSearchCV
from sklearn.linear_model import LinearRegression

#separate the independent variable and dependent variable

X2=pred_df[['page_loads','first_visits' ,'returning_visits','days_f']]
y2=pred_df['unique_visits']

#split the dataset in train and test samples now

X_train, X_test, y_train, y_test = train_test_split(X2, y2, test_size=0.3,random_state=42)

#train the model with train sample

regressor2 = LinearRegression(fit_intercept=False)
regressor2.fit(X_train, y_train)
```

Out[19]:
```
LinearRegression
LinearRegression(fit_intercept=False)
```

The model has been trained on the train dataset now predicting the values using test dataset.

In [20]:
```
y_pred2 = regressor2.predict(X_test)
lr2 = pd.DataFrame({'Actual': y_test, 'Predicted': y_pred2})
lr2
```

Out[20]:

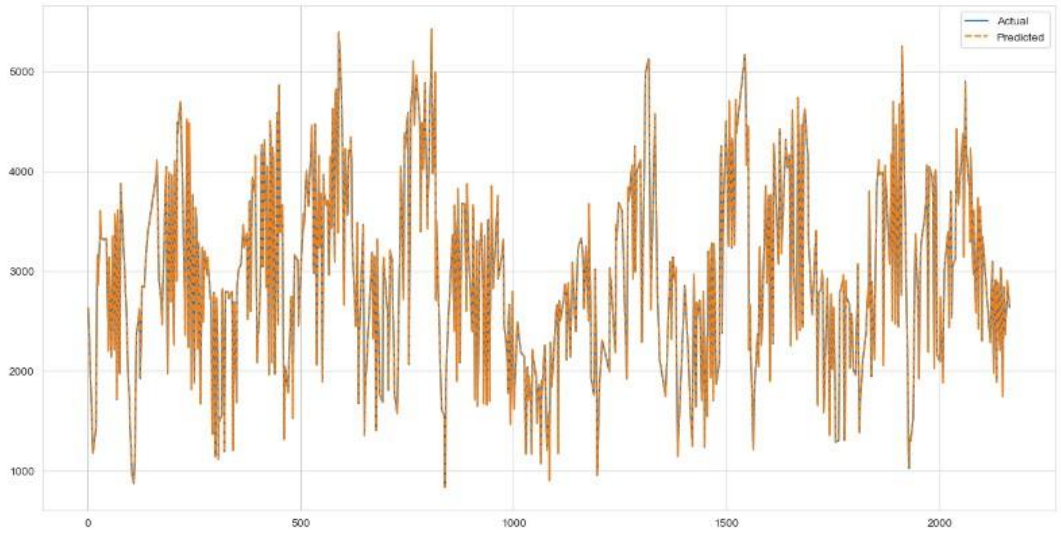
	Actual	Predicted
1486	4173	4173.0
1602	1902	1902.0
1460	2870	2870.0
1134	2142	2142.0
1513	4329	4329.0
...	...	...
439	2579	2579.0
271	2494	2494.0
244	1818	1818.0
1159	3332	3332.0
1701	2565	2565.0

651 rows × 2 columns

As we have calculated the values lets visualize the actual and predicted values

In [21]:
```
plt.figure(figsize=(16,8))
sns.lineplot(data=lr2)
```

Out[21]: <Axes: >



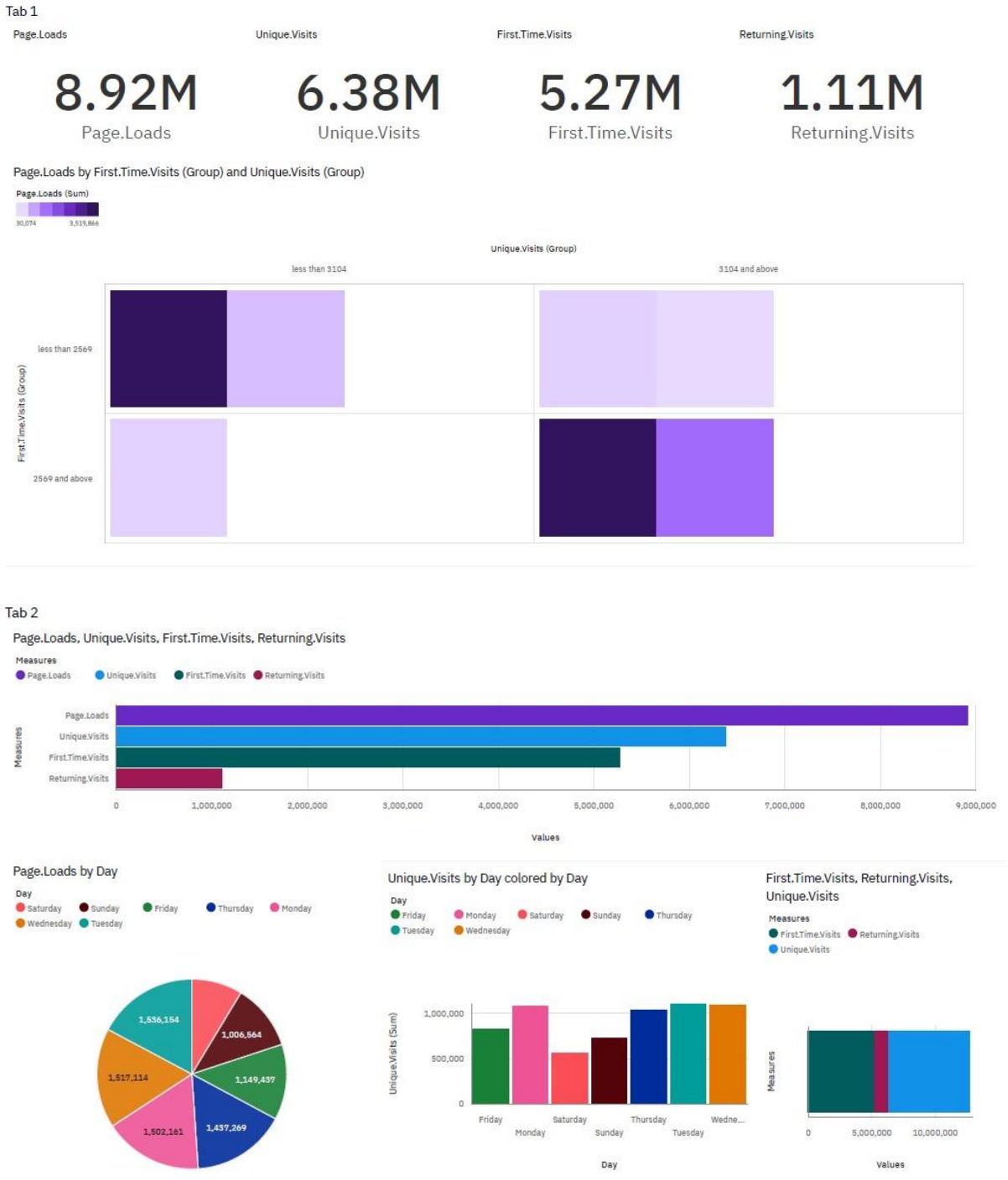
In [22]:
```
#get the accuacy score of the model.

regressor2.score(X_test,y_test)*100
```

Out[22]: 100.0

The accuracy of the above model is 100

CREATING DASHBOARD:

Using IBM Cognos Analytics we have created a dashboard for website traffic analysis as shown below:



INSIGHTS:

The insights we get from the visualizations and dashboard helps the website owners to improve the user experience of website. Some of the insights we get by implementing this project are:

- ✓ From the KPI, we observe that the Day Saturday has the lowest total Page Loads at nearly 773 thousand, followed by Sunday at over 1.0 million.
- ✓ Day Tuesday has the highest total Page Loads at 1536154.0, followed by Wednesday at 1517114.0
- ✓ The bar chart at the top represents the total values of page loads, unique visits, first time visits and returning visits.
- ✓ Then the page loads of each day of the week is visualized using pie chart.
- ✓ Then we have a bar chart representing the unique visits of each day.
- ✓ The first time visits, unique visits and returning visits is visualized using a stacked bar chart.
- ✓ The time series analysis is used to understand which days get the most traffic and on what time intervals.
- ✓ Unique visits is compared to page loads with first time visits and returning visits among page loads.

LINKS:

Website Traffic Analysis.ipynb

The above link redirects to the file that contains the codes used in the python integration part of this project.

Website Traffic Analysis Dashboard.pdf

The above link redirects to the file that contains the dashboard that is created using IBM Cognos Analytics for this Project.

SCOPE:

The scope of this project encompasses a comprehensive analysis of website traffic data from the provided dataset. Firstly, within the realm of data analysis, the project aims to identify popular pages on the website, unveiling key insights into which content attracts the most user attention. Additionally, it intends to uncover traffic trends by dissecting daily, weekly, and monthly patterns, aiding in understanding user behaviour and the impact of various factors over time. Furthermore, the project will assess user engagement metrics, including bounce rates, session durations, and unique visitor counts, to gauge the effectiveness of the website in engaging its audience.

Secondly, within the domain of data collection, the project's scope includes obtaining and cleaning the dataset. Data cleaning activities will encompass addressing missing values, outliers, and data inconsistencies to ensure the dataset's reliability for analysis. This phase establishes the foundation for deriving meaningful insights and accurate visualizations.

Lastly, the project extends into the realm of visualization and machine learning integration. It plans to utilize IBM Cognos for visualization purposes, creating interactive dashboards and insightful reports that provide a user-friendly interface for stakeholders to explore the findings. Moreover, the integration of Python-based machine learning models broadens the project's scope, enabling predictions related to future traffic trends and user behaviour patterns, potentially unlocking advanced capabilities for website optimization and content recommendation. Overall, this project's scope is comprehensive, covering data analysis, data collection, visualization, and machine learning integration, with the overarching goal of enhancing website performance and user engagement.

CONCLUSION:

In conclusion, this project presents a comprehensive approach to analyzing website traffic data sourced from the Kaggle dataset. By defining clear objectives, establishing data sources and collection methods, planning visualization through IBM Cognos, and considering the integration of Python-based machine learning models, we aim to unlock valuable insights into website performance, user engagement, and traffic trends. This holistic approach not only facilitates informed decision-making but also opens doors to potential predictive capabilities. Ultimately, this project seeks to empower businesses and website administrators with the knowledge and tools necessary to enhance their online presence, content optimization, and user experience for sustainable growth and success.