# AI MIDSEM – 1 EXAMINATION

NAME – PRATEEK BALIYAN

BRANCH – CSEAI

SECTION – C

ROLL NUMBER (UNI) – 202401100300177

ROLL NUMBER (CLASS) – 30

TOPIC NAME – TRAFFIC LIGHT CONTROL SYSTEM

# INTRODUCTION

Traffic data analysis is crucial for understanding web user behaviour, optimizing website performance, and improving user engagement. This report examines traffic data over a 20-day period, focusing on key metrics such as Page Views, Unique Visitors, and Bounce Rate.

The goal of this study is to:

Identify traffic trends over time.

Analyse the relationship between Page Views and Bounce Rate.

Provide insights for optimizing web performance.

Using Python, pandas, matplotlib, and seaborn, the dataset was processed to extract meaningful patterns and visualizations. The findings help in understanding visitor engagement and website efficiency.

# METHODOLOGY

Methodology for the Given Code

1. Data Preprocessing (AI Foundation)

Handling Missing Values:

Checking for missing values using df.isnull().sum() is a crucial preprocessing step in AI to ensure data quality.

Feature Engineering:

Extracting the hour from the timestamp (df['timestamp'].dt.hour) is a common AI technique to create meaningful features for time-series models.

2. Exploratory Data Analysis (EDA)

Statistical Analysis:

Using df.describe() helps understand the data distribution, an essential step before applying machine learning models.

Time-Series Analysis:

Grouping data by hour (df.groupby('hour').size()) is a fundamental step in predictive modeling and anomaly detection in AI.

3. Visualization (AI-Powered Insights)

Seaborn & Matplotlib Plots:

These visualizations help identify trends, outliers, and patterns, which are crucial for training AI models like time-series forecasting (e.g., ARIMA, LSTMs) or anomaly detection models (e.g., Isolation Forests, Autoencoders).

# CODE

```
import pandas as pd
import matplotlib.pyplot as plt

# Load the CSV file
file_path = "/content/traffic_data.csv"  # Update with the correct path if needed
df = pd.read_csv(file_path)

print("Dataset Information:\n")
print(df.info())

# Show first few rows
print("\nFirst 5 Rows:\n", df.head())

# Check for missing values
print("\nMissing Values:\n", df.isnull().sum())

# Basic statistics
print("\nStatistical Summary:\n", df.describe())

# Time-based traffic analysis (Assuming there's a 'timestamp' column)
if 'timestamp' in df.columns:
    df['timestamp'] = pd.to_datetime(df['timestamp'])
    df['hour'] = df['timestamp'].dt.hour
    traffic_by_hour = df.groupby('hour').size()

    plt.figure(figsize=(10, 5))
    sns.lineplot(x=traffic_by_hour.index, y=traffic_by_hour.values, marker='o')
    plt.title('Traffic Volume by Hour')
```

```
    plt.xlabel('Hour of Day')
    plt.ylabel('Traffic Count')
    plt.grid()
    plt.show()

# Convert 'Date' column to datetime format
df['Date'] = pd.to_datetime(df['Date'])

# Create the plot
plt.figure(figsize=(10, 5))
plt.plot(df['Date'], df['PageViews'], label='Page Views', marker='o', linestyle='-')
plt.plot(df['Date'], df['UniqueVisitors'], label='Unique Visitors', marker='s', linestyle='--')

# Formatting
plt.xlabel("Date")
plt.ylabel("Count")
plt.title("Website Traffic Over Time")
plt.xticks(rotation=45)
plt.legend()
plt.grid(True)

# Show the plot
plt.show()
```

# OUTPUT

```
First 5 Rows:
        Date  PageViews  UniqueVisitors  BounceRate
0 2024-01-01        828            1261   54.420000
1 2024-01-02       7065            4225   31.583887
2 2024-01-03       5861            3286   68.284703
3 2024-01-04       7163             651   60.203175
4 2024-01-05       9432             548   37.963247

Missing Values:
 Date            0
PageViews        0
UniqueVisitors   0
BounceRate       0
dtype: int64

Statistical Summary:
           PageViews  UniqueVisitors  BounceRate
count     20.000000       20.000000   20.000000
mean    5533.200000     2435.050000   49.150658
std     2595.965850     1383.480189   15.286241
min      828.000000      518.000000   28.581849
25%     3218.500000     1115.250000   37.609458
50%     6405.000000     2466.500000   49.061288
75%     7288.750000     3696.250000   60.163514
max     9432.000000     4459.000000   79.981676
```