WEATHER DATA ANALYSIS REPORT

NAME - SHREYA TYAGI

BRANCH – CSEAI

SECTION-D

COURSE-INTRODUCTION TO AI

ROLL NO-202401100300241

LIB ID-2428EEE777

INTRODUCTION

Weather analysis is a crucial task in understanding climatic patterns and predicting future weather conditions. By analyzing weather data, we can extract meaningful insights that help in various domains such as agriculture, disaster management, and urban planning.

METHODOLOGY

The methodology followed in this report includes:

1. **Data Collection:** The dataset weather_data.csv is used for analysis.
2. **Data Preprocessing:**
   - Load the dataset using Pandas.
   - Convert the Date column to datetime format.
   - Handle missing values.
3. **Exploratory Data Analysis (EDA):**
   - Display the first few rows of the dataset.
   - Generate descriptive statistics.
   - Identify missing values.
4. **Data Visualization:**
   - Temperature trends over time.
   - Rainfall distribution.
   - Wind speed distribution.
   - Correlation heatmap of weather variables.

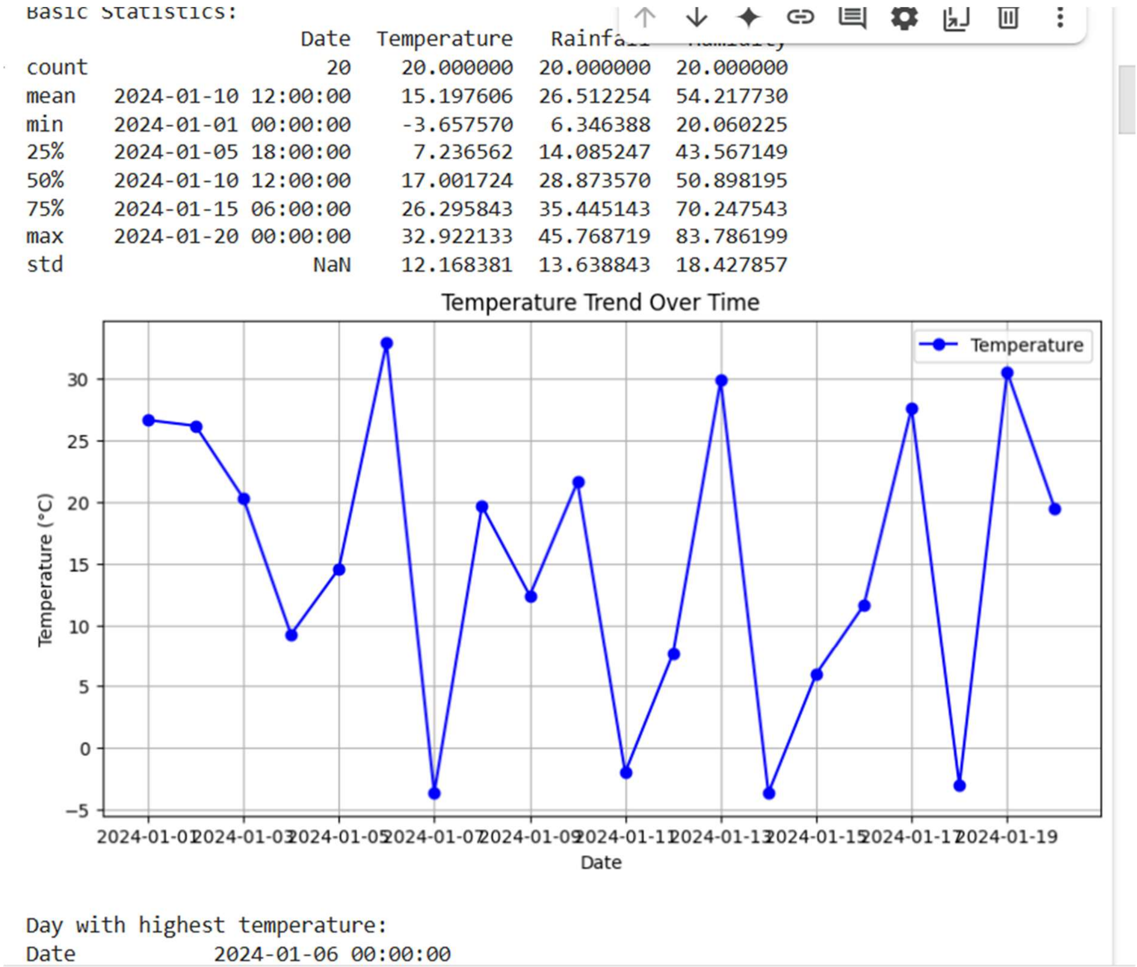
CODE IMPLEMENTATION

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

# Load the weather data CSV file
data = pd.read_csv('weather_data.csv')

# Display the first few rows of the data
print(data.head())

# Basic Data Information
print("\nData Info:")
print(data.info())

# Descriptive statistics of numerical columns
*print("\nDescriptive Statistics:")
print(data.describe())

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

# Convert 'Date' column to datetime format if it exists
if 'Date' in data.columns:
   data['Date'] = pd.to_datetime(data['Date'])
else:
   print("\nWarning: 'Date' column is missing!")
```

```
# 1. Plot Temperature Over Time
if 'Date' in data.columns and 'Temperature' in data.columns:
    plt.figure(figsize=(10, 6))
    sns.lineplot(x=data['Date'], y=data['Temperature'], label='Temperature (°C)', color='orange')
    plt.title('Temperature Over Time')
    plt.xlabel('Date')
    plt.ylabel('Temperature (°C)')
    plt.xticks(rotation=45)
    plt.legend()
    plt.grid(True)
    plt.tight_layout()
    plt.show()
else:
    print("\nSkipping Temperature Plot: Missing 'Date' or 'Temperature' column.")


# 2. Plot Rainfall Distribution
if 'Rainfall' in data.columns:
    plt.figure(figsize=(8, 5))
    sns.histplot(data['Rainfall'].dropna(), kde=True, color='blue', bins=20)
    plt.title('Rainfall Distribution')
    plt.xlabel('Rainfall (mm)')
    plt.ylabel('Frequency')
    plt.grid(True)
    plt.tight_layout()
    plt.show()
else:
    print("\nSkipping Rainfall Distribution Plot: 'Rainfall' column missing.")
```

```
# 3. Plot Wind Speed Distribution
if 'WindSpeed' in data.columns:
    plt.figure(figsize=(8, 5))
    sns.histplot(data['WindSpeed'].dropna(), kde=True, color='green', bins=20)
    plt.title('Wind Speed Distribution')
    plt.xlabel('Wind Speed (km/h)')
    plt.ylabel('Frequency')
    plt.grid(True)
    plt.tight_layout()
    plt.show()
else:
    print("\nSkipping Wind Speed Distribution Plot: 'WindSpeed' column missing.")

# 4. Correlation Heatmap
required_columns = {'Temperature', 'Rainfall', 'WindSpeed', 'Humidity'}
if required_columns.issubset(data.columns):
    plt.figure(figsize=(10, 7))
    correlation_matrix = data[list(required_columns)].corr()
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
    plt.title('Correlation Heatmap of Weather Variables')
    plt.tight_layout()
    plt.show()

    else:
        print("\nSkipping Correlation Heatmap: Missing one or more required columns.”)
```

OUTPUT



```
Basic Statistics:
                       Date  Temperature   Rainfall   Humidity
count                    20    20.000000  20.000000  20.000000
mean    2024-01-10 12:00:00    15.197606  26.512254  54.217730
min     2024-01-01 00:00:00    -3.657570   6.346388  20.060225
25%     2024-01-05 18:00:00     7.236562  14.085247  43.567149
50%     2024-01-10 12:00:00    17.001724  28.873570  50.898195
75%     2024-01-15 06:00:00    26.295843  35.445143  70.247543
max     2024-01-20 00:00:00    32.922133  45.768719  83.786199
std                     NaN    12.168381  13.638843  18.427857
```

```
Day with highest temperature:
Date            2024-01-06 00:00:00
```

REFERENCES

- Seaborn Documentation: https://seaborn.pydata.org/
- Matplotlib Documentation: https://matplotlib.org/stable/contents.html
- Pandas Documentation: https://pandas.pydata.org/
- Dataset source: Provided weather data (weather_data.csv)