```
import seaborn as sns
```

In [18]:

```
# Load the weather dataset (assuming it's in the same directory as this scrip
weather_df = pd.read_csv('weather.csv')
print(weather_df)
```

```
     MinTemp  MaxTemp  Rainfall  Evaporation  Sunshine WindGustDir  \
0        8.0     24.3       0.0          3.4       6.3          NW   
1       14.0     26.9       3.6          4.4       9.7         ENE   
2       13.7     23.4       3.6          5.8       3.3          NW   
3       13.3     15.5      39.8          7.2       9.1          NW   
4        7.6     16.1       2.8          5.6      10.6         SSE   
..       ...      ...       ...          ...       ...         ...   
361      9.0     30.7       0.0          7.6      12.1         NNW   
362      7.1     28.4       0.0         11.6      12.7           N   
363     12.5     19.9       0.0          8.4       5.3         ESE   
364     12.5     26.9       0.0          5.0       7.1          NW   
365     12.3     30.2       0.0          6.0      12.6          NW   

     WindGustSpeed WindDir9am WindDir3pm  WindSpeed9am  ...  Humidity3pm  \
0             30.0         SW         NW           6.0  ...           29   
1             39.0          E          W           4.0  ...           36   
2             85.0          N        NNE           6.0  ...           69   
3             54.0        WNW          W          30.0  ...           56   
4             50.0        SSE        ESE          20.0  ...           49   
..             ...        ...        ...           ...  ...          ...   
361           76.0        SSE         NW           7.0  ...           15   
362           48.0        NNW        NNW           2.0  ...           22   
363           43.0        ENE        ENE          11.0  ...           47   
364           46.0        SSW        WNW           6.0  ...           39   
365           78.0         NW        WNW          31.0  ...           13   

     Pressure9am  Pressure3pm  Cloud9am  Cloud3pm  Temp9am  Temp3pm  \
0         1019.7       1015.0         7         7     14.4     23.6   
1         1012.4       1008.4         5         3     17.5     25.7   
2         1009.5       1007.2         8         7     15.4     20.2   
3         1005.5       1007.0         2         7     13.5     14.1   
4         1018.3       1018.5         7         7     11.1     15.4   
..           ...          ...       ...       ...      ...      ...   
361       1016.1       1010.8         1         3     20.4     30.0   
362       1020.0       1016.9         0         1     17.2     28.2   
363       1024.0       1022.8         3         2     14.5     18.3   
364       1021.0       1016.2         6         7     15.8     25.9   
365       1009.6       1009.2         1         1     23.8     28.6   

     RainToday  RISK_MM RainTomorrow  
0           No      3.6          Yes  
1          Yes      3.6          Yes  
2          Yes     39.8          Yes  
3          Yes      2.8          Yes  
4          Yes      0.0           No  
..         ...      ...          ...  
361         No      0.0           No  
362         No      0.0           No  
363         No      0.0           No  
364         No      0.0           No  
365         No      0.0           No  
```

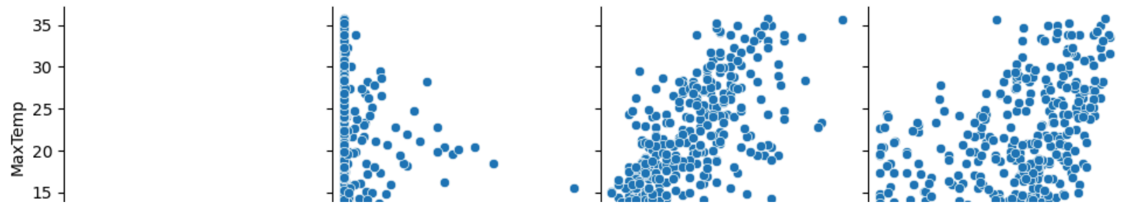
```python
print(weather_df.info())
```

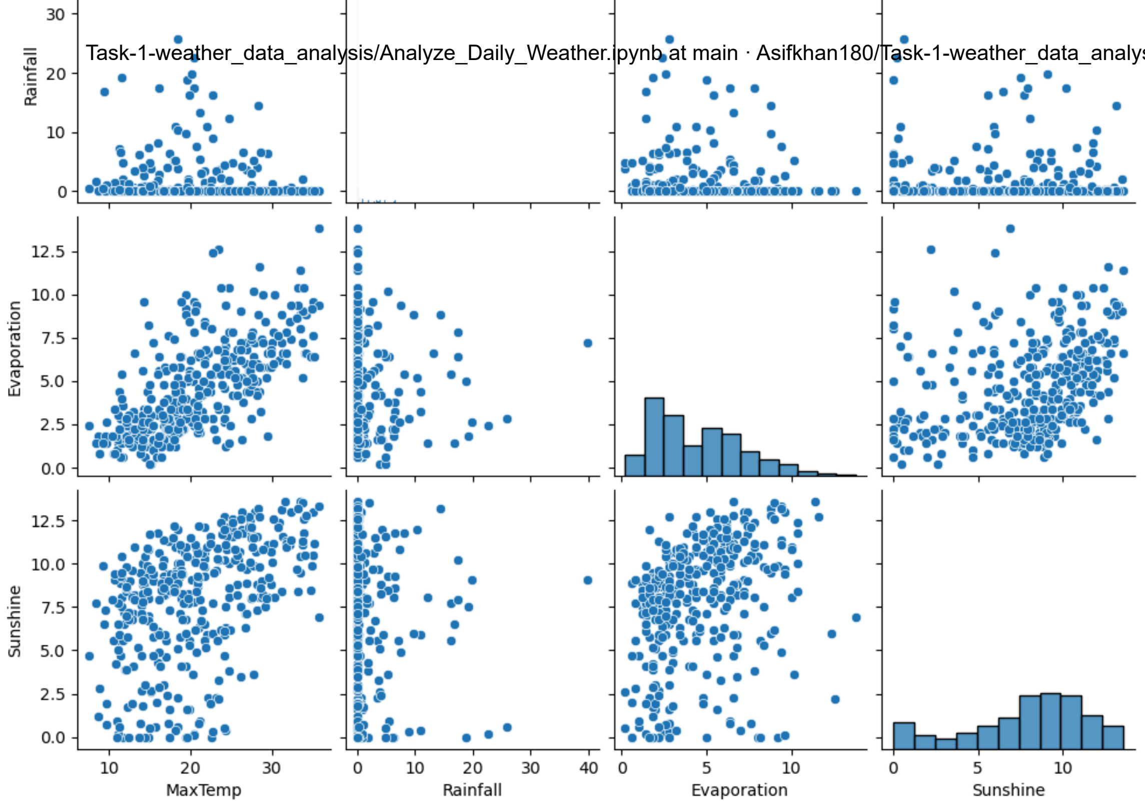
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 366 entries, 0 to 365
Data columns (total 22 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   MinTemp        366 non-null    float64
 1   MaxTemp        366 non-null    float64
 2   Rainfall       366 non-null    float64
 3   Evaporation    366 non-null    float64
 4   Sunshine       363 non-null    float64
 5   WindGustDir    363 non-null    object 
 6   WindGustSpeed  364 non-null    float64
 7   WindDir9am     335 non-null    object 
 8   WindDir3pm     365 non-null    object 
 9   WindSpeed9am   359 non-null    float64
 10  WindSpeed3pm   366 non-null    int64  
 11  Humidity9am    366 non-null    int64  
 12  Humidity3pm    366 non-null    int64  
 13  Pressure9am    366 non-null    float64
 14  Pressure3pm    366 non-null    float64
 15  Cloud9am       366 non-null    int64  
 16  Cloud3pm       366 non-null    int64  
 17  Temp9am        366 non-null    float64
 18  Temp3pm        366 non-null    float64
 19  RainToday      366 non-null    object 
 20  RISK_MM        366 non-null    float64
 21  RainTomorrow   366 non-null    object 
dtypes: float64(12), int64(5), object(5)
memory usage: 63.0+ KB
None
```

In [21]:

```python
# Create pair plots to visualize relationships between numeric variables
sns.pairplot(weather_df[['MaxTemp', 'Rainfall', 'Evaporation', 'Sunshine']])
plt.show()
```

```
R:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_
as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
R:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_
as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
R:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_
as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
R:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_
as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [22]:

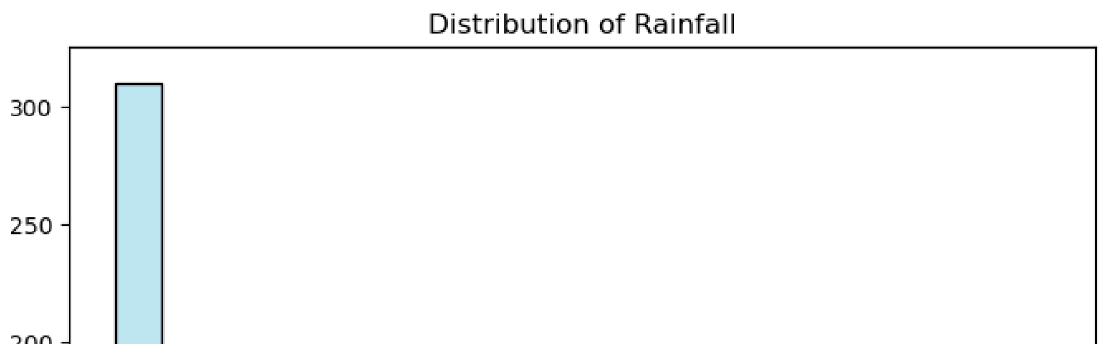
```python
# Calculate statistics for specific columns
mean_rainfall = weather_df['Rainfall'].mean()
max_temp = weather_df['MaxTemp'].max()

print(f"Mean rainfall: {mean_rainfall:.2f} mm")
print(f"Maximum temperature: {max_temp:.2f} °C")
```

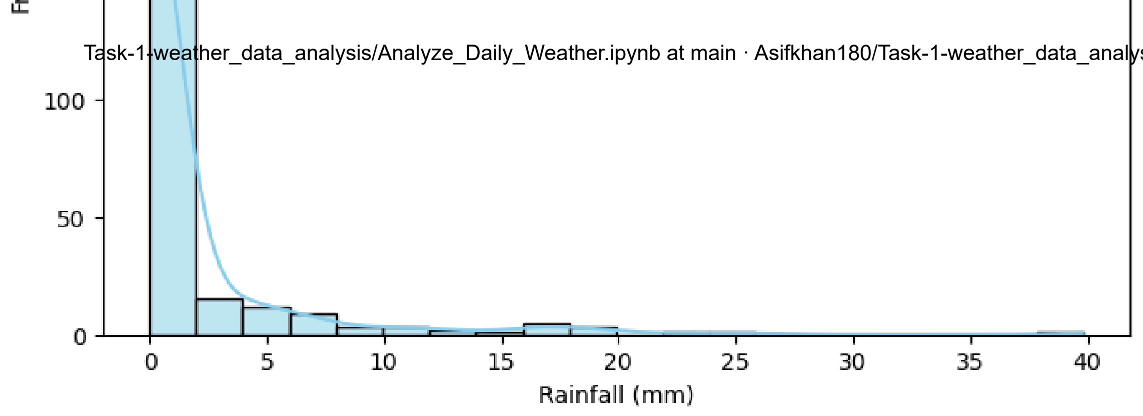
```
Mean rainfall: 1.43 mm
Maximum temperature: 35.80 °C
```

In [24]:

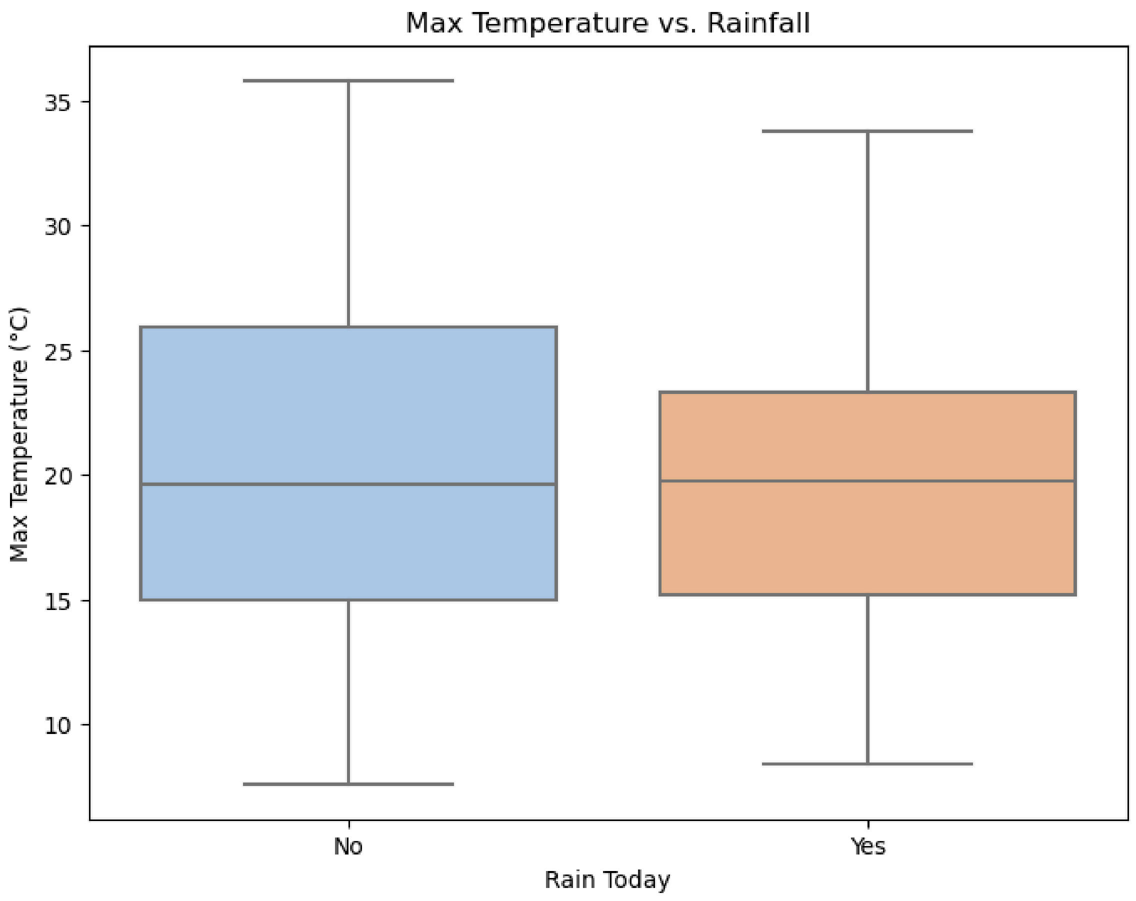
```python
# Distribution of rainfall
plt.figure(figsize=(8, 6))
sns.histplot(weather_df['Rainfall'], bins=20, kde=True, color='skyblue')
plt.xlabel('Rainfall (mm)')
plt.ylabel('Frequency')
plt.title('Distribution of Rainfall')
plt.show()
```

```
R:\anaconda\Lib\site-packages\seaborn\_oldcore.py:1119: FutureWarning: use_inf_
as_na option is deprecated and will be removed in a future version. Convert inf
values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```





In [25]:
```
# Box plot for MaxTemp and Rainfall
plt.figure(figsize=(8, 6))
sns.boxplot(x='RainToday', y='MaxTemp', data=weather_df, palette='pastel')
plt.xlabel('Rain Today')
plt.ylabel('Max Temperature (°C)')
plt.title('Max Temperature vs. Rainfall')
plt.show()
```



In [26]:
```
# Example: Rainfall prediction using Linear Regression (for demonstration pur
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split

X = weather_df[['MaxTemp']]
y = weather_df['Rainfall']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rand
rainfall_pred = model.predict(X_test)
print(f"Sample rainfall predictions: {rainfall_pred[:5]}")
```

```
Sample rainfall predictions: [1.24920459 0.95342914 0.7265329  1.37480786 0.880
4982 ]
```

In [29]:

```
#### Final Conclusion and Insights #####



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

# Read the weather dataset from 'weather.csv'
file_path = 'weather.csv'
df = pd.read_csv(file_path)

# Calculate average maximum temperature (MaxTemp)
max_temp_mean = df['MaxTemp'].mean()

# Calculate total rainfall
rainfall_sum = df['Rainfall'].sum()

# Calculate average humidity at 9 am (Humidity9am) and 3 pm (Humidity3pm)
humidity_9am_mean = df['Humidity9am'].mean()
humidity_3pm_mean = df['Humidity3pm'].mean()

# Calculate average atmospheric pressure at 9 am (Pressure9am) and 3 pm (Pres
pressure_9am_mean = df['Pressure9am'].mean()
pressure_3pm_mean = df['Pressure3pm'].mean()

# Display conclusions and insights
print("Conclusions and Insights:")
print(f"1. Average Max Temperature: {max_temp_mean:.2f} °C")
print(f"2. Total Rainfall: {rainfall_sum:.2f} mm")
print(f"3. Average Humidity at 9 am: {humidity_9am_mean:.2f}%")
print(f"4. Average Humidity at 3 pm: {humidity_3pm_mean:.2f}%")
print(f"5. Average Pressure at 9 am: {pressure_9am_mean:.2f} hPa")
print(f"6. Average Pressure at 3 pm: {pressure_3pm_mean:.2f} hPa")

# Feel free to customize and expand this analysis further!
```

```
Conclusions and Insights:
1. Average Max Temperature: 20.55 °C
2. Total Rainfall: 522.80 mm
3. Average Humidity at 9 am: 72.04%
4. Average Humidity at 3 pm: 44.52%
5. Average Pressure at 9 am: 1019.71 hPa
6. Average Pressure at 3 pm: 1016.81 hPa
```

In [ ]: