In [27]:

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

In [28]:

```
df = pd.read_csv("india_weather_crops_dataset.csv")
```

In [29]:

```
df.head()
```

Out[29]:

	Date	City	Temperature (°C)	Humidity (%)	Rainfall (mm)	Wind Speed (km/h)	Weather Condition	Recommended Crop
0	2020-04-09	Kochi	15.5	34	0.0	26.4	Sunny	Lentils
1	2024-12-17	Indore	24.4	96	292.3	15.7	Hazy	Watermelon
2	2015-11-09	Coimbatore	36.8	29	0.0	9.0	Hazy	Watermelon
3	2017-05-28	Hyderabad	29.6	92	136.3	40.4	Cloudy	Jute
4	2015-02-03	Ahmedabad	14.8	43	186.0	42.0	Sunny	Pumpkin

In [30]:

```
df.tail()
```

Out[30]:

	Date	City	Temperature (°C)	Humidity (%)	Rainfall (mm)	Wind Speed (km/h)	Weather Condition	Recommended Crop
9995	2016-01-13	Bangalore	24.8	38	0.0	5.1	Sunny	Watermelon
9996	2018-05-23	Patna	17.2	54	0.0	48.6	Cloudy	Barley
9997	2015-06-21	Delhi	17.7	34	74.6	20.2	Drizzle	Watermelon
9998	2015-07-30	Patna	20.1	38	0.0	46.6	Hazy	Bitter Gourd
9999	2016-08-12	Pune	12.7	79	0.0	20.1	Hazy	Gram

In [31]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Date               10000 non-null  object 
 1   City               10000 non-null  object 
 2   Temperature (°C)   10000 non-null  float64
 3   Humidity (%)       10000 non-null  int64  
 4   Rainfall (mm)      10000 non-null  float64
 5   Wind Speed (km/h)  10000 non-null  float64
 6   Weather Condition  10000 non-null  object 
 7   Recommended Crop   10000 non-null  object 
dtypes: float64(3), int64(1), object(4)
memory usage: 625.1+ KB
```

In [32]:

```
df.describe()
```

Out[32]:

	Temperature (°C)	Humidity (%)	Rainfall (mm)	Wind Speed (km/h)
count	10000.000000	10000.000000	10000.000000	10000.000000
mean	27.463410	59.575200	45.159530	24.940530
std	10.101007	23.190402	83.871467	14.420775
min	10.000000	20.000000	0.000000	0.000000
25%	18.800000	40.000000	0.000000	12.700000
50%	27.400000	59.000000	0.000000	24.800000
75%	36.100000	80.000000	49.925000	37.300000
max	45.000000	100.000000	300.000000	50.000000

In [34]:

```
df_city_weather = df.groupby("City").agg({
    "Temperature (°C)": "mean",
    "Rainfall (mm)": "mean",
    "Wind Speed (km/h)": "mean"
}).round(1)

print(df_city_weather)
```

```
               Temperature (°C)  Rainfall (mm)  Wind Speed (km/h)
City
Ahmedabad                  28.0           47.7               24.9
Bangalore                  27.7           40.6               25.2
Bhopal                     27.6           49.5               26.2
Chandigarh                 27.5           43.2               25.3
Chennai                    27.4           42.5               25.0
Coimbatore                 27.9           49.7               25.3
Delhi                      27.5           48.9               25.5
Hyderabad                  26.6           46.0               25.9
Indore                     27.1           44.5               25.0
Jaipur                     26.6           54.0               23.9
Kochi                      27.0           46.9               24.2
Kolkata                    28.2           47.7               24.7
Lucknow                    27.2           44.7               25.4
Mumbai                     27.6           38.6               25.4
Nagpur                     27.4           40.5               24.3
Patna                      27.7           38.0               24.3
Pune                       27.4           46.1               24.6
Visakhapatnam              28.0           44.7               23.8
```

In [37]:

```
common_weather_per_city = df.groupby("City")["Weather Condition"].agg(lambda x: x.value_counts().idxmax())
print(common_weather_per_city)
```

```
City
Ahmedabad          Rainy
Bangalore          Foggy
Bhopal            Cloudy
Chandigarh         Foggy
Chennai            Rainy
Coimbatore         Rainy
Delhi              Foggy
Hyderabad          Rainy
Indore            Cloudy
Jaipur           Drizzle
Kochi              Foggy
Kolkata            Foggy
Lucknow           Cloudy
Mumbai             Foggy
Nagpur             Sunny
Patna             Cloudy
Pune              Stormy
Visakhapatnam     Cloudy
Name: Weather Condition, dtype: object
```

In [38]:

```
best_crop_per_city = df.groupby("City")["Recommended Crop"].agg(lambda x: x.value_counts().idxmax())
print(best_crop_per_city)
```

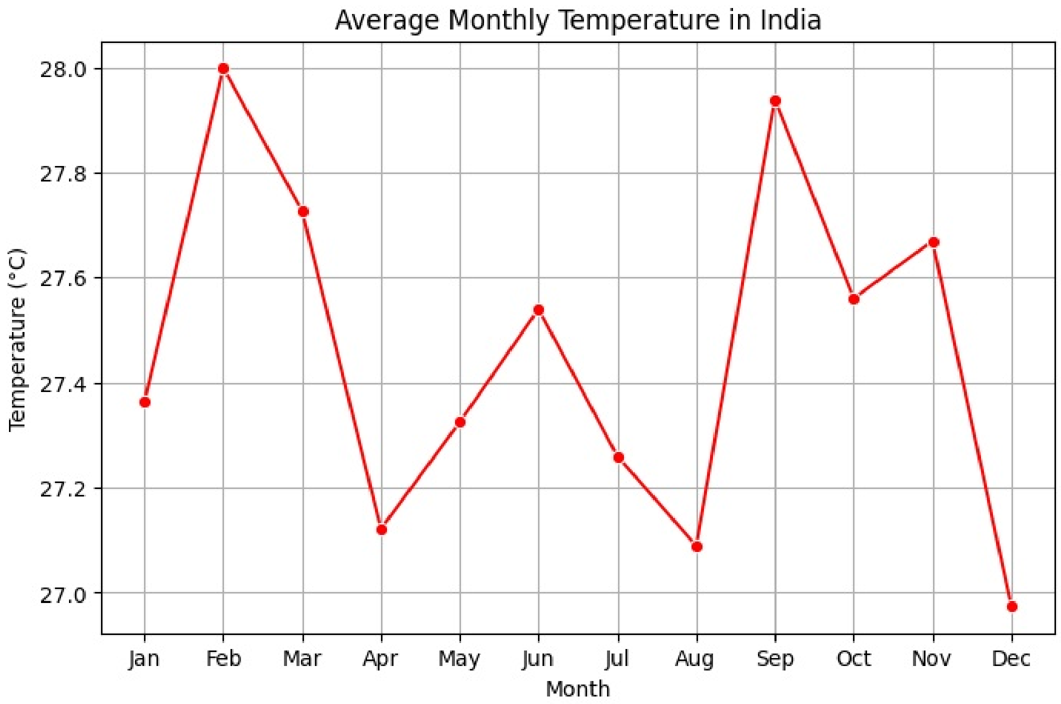
```
City
Ahmedabad           Muskmelon
Bangalore            Cucumber
Bhopal               Cucumber
Chandigarh            Pumpkin
Chennai              Cucumber
Coimbatore          Muskmelon
Delhi                Cucumber
Hyderabad        Bitter Gourd
Indore               Cucumber
Jaipur                Pumpkin
Kochi            Bitter Gourd
Kolkata            Watermelon
Lucknow          Bitter Gourd
Mumbai               Cucumber
Nagpur               Cucumber
Patna            Bitter Gourd
Pune             Bitter Gourd
Visakhapatnam    Bitter Gourd
Name: Recommended Crop, dtype: object
```

In [40]:

```
df["Date"] = pd.to_datetime(df["Date"])
df["Month"] = df["Date"].dt.month
```

In [41]:

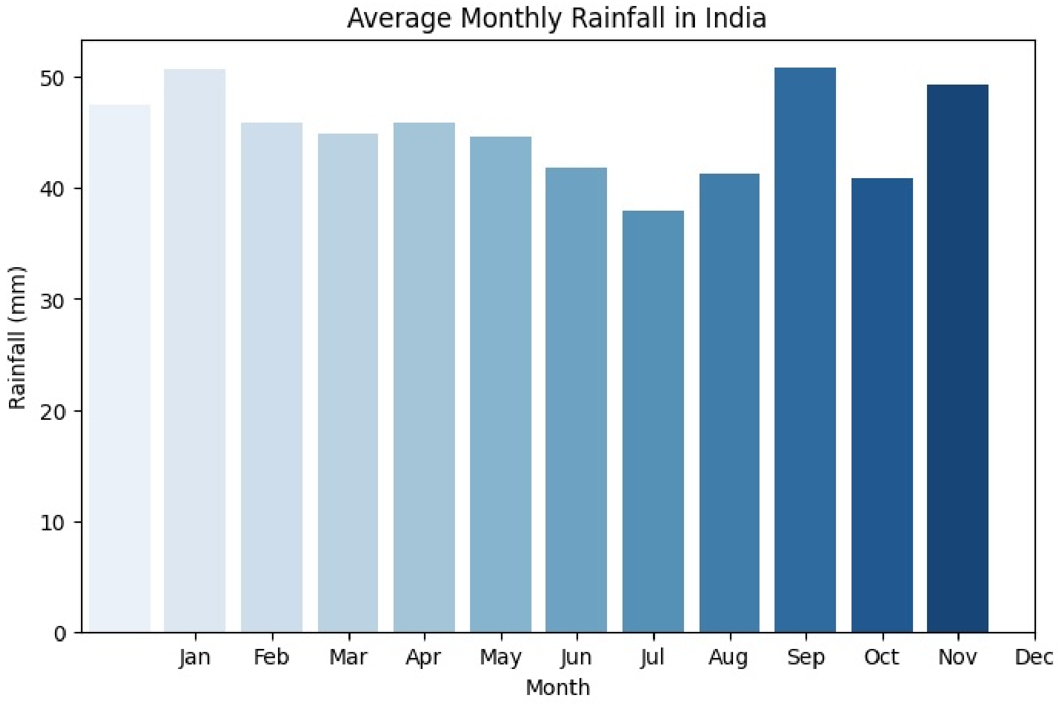
```
monthly_temp = df.groupby("Month")["Temperature (°C)"].mean()

plt.figure(figsize=(8,5))
sns.lineplot(x=monthly_temp.index, y=monthly_temp.values, marker="o", color="red")
plt.title("Average Monthly Temperature in India")
plt.xlabel("Month")
plt.ylabel("Temperature (°C)")
plt.xticks(range(1,13), ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"])
plt.grid()
plt.show()
```



In [42]:

```
monthly_rainfall = df.groupby("Month")["Rainfall (mm)"].mean()

plt.figure(figsize=(8,5))
sns.barplot(x=monthly_rainfall.index, y=monthly_rainfall.values, palette="Blues")
plt.title("Average Monthly Rainfall in India")
plt.xlabel("Month")
plt.ylabel("Rainfall (mm)")
plt.xticks(range(1,13), ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"])
plt.show()
```

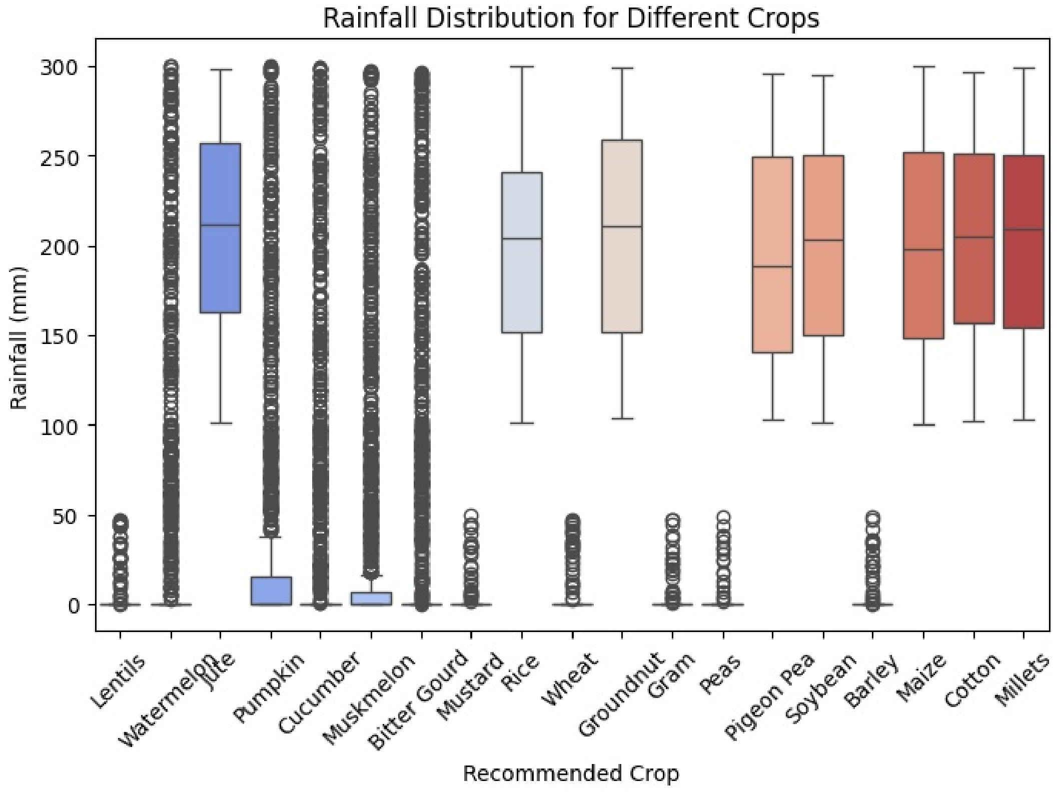
```
C:\Users\sunit\AppData\Local\Temp\ipykernel_14816\258503637.py:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=monthly_rainfall.index, y=monthly_rainfall.values, palette="Blues")
```



In [43]:

```
plt.figure(figsize=(8,5))
sns.boxplot(data=df, x="Recommended Crop", y="Rainfall (mm)", palette="coolwarm")
plt.xticks(rotation=45)
plt.title("Rainfall Distribution for Different Crops")
plt.show()
```

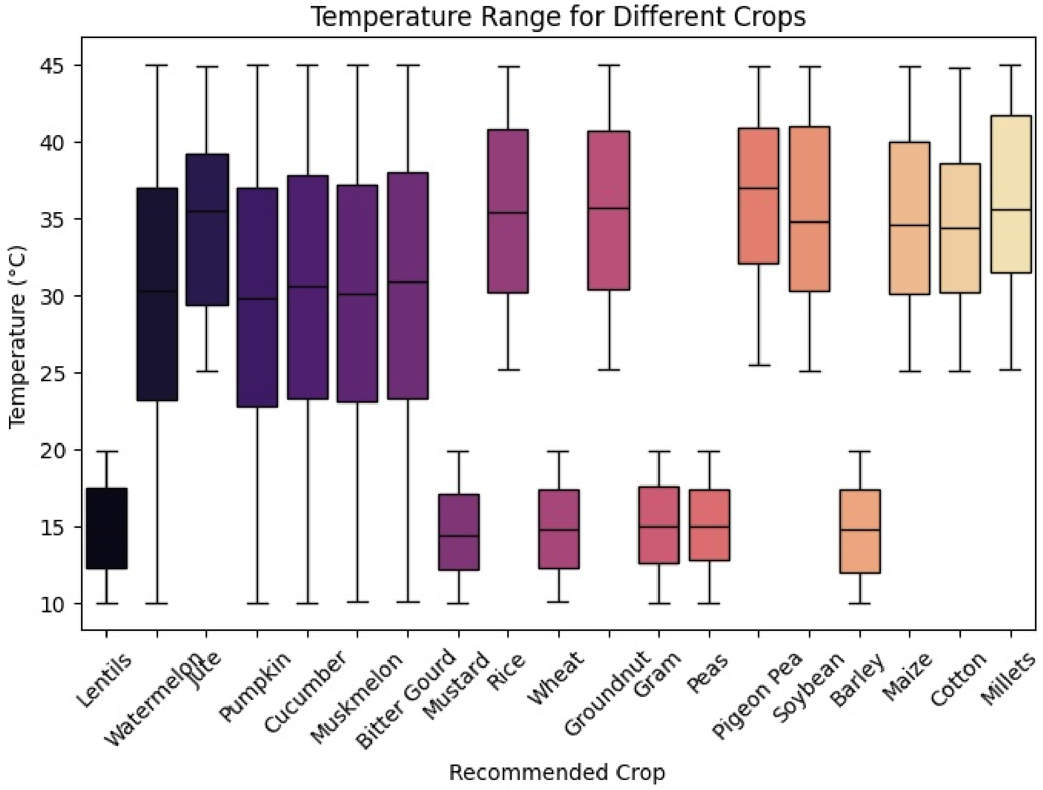
```
C:\Users\sunit\AppData\Local\Temp\ipykernel_14816\2024899777.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.boxplot(data=df, x="Recommended Crop", y="Rainfall (mm)", palette="coolwarm")
```



In [44]:

```
plt.figure(figsize=(8,5))
sns.boxplot(data=df, x="Recommended Crop", y="Temperature (°C)", palette="magma")
plt.xticks(rotation=45)
plt.title("Temperature Range for Different Crops")
plt.show()
```

```
C:\Users\sunit\AppData\Local\Temp\ipykernel_14816\1343593248.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.boxplot(data=df, x="Recommended Crop", y="Temperature (°C)", palette="magma")
```



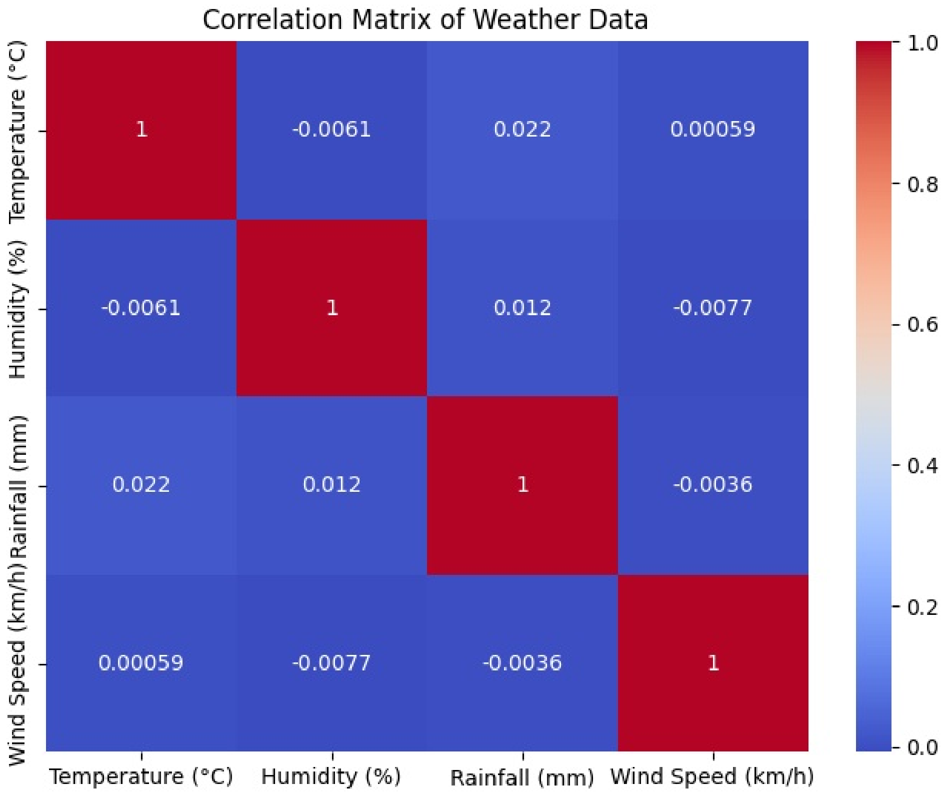
In [48]:

```
df_rainy=df[df["Weather Condition"].isin(["Rainy","Drizzle","Dtromy"])]
df_dry=df[df["Weather Condition"].isin(["Sunny","Hazy"])]
df_moderate=df[df["Weather Condition"].isin(["Cloudy","Foggy"])]

rainy_crops=df_rainy["Recommended Crop"].value_counts()
dry_crops=df_dry["Recommended Crop"].value_counts()
moderate_crops=df_moderate["Recommended Crop"].value_counts()

print("Most recommended crops in Rainy Weather:\n",rainy_crops)
print("Most recommended crops in Dry Weather:\n",dry_crops)
print("Most recommended crops in moderate Weather:\n",moderate_crops)
```

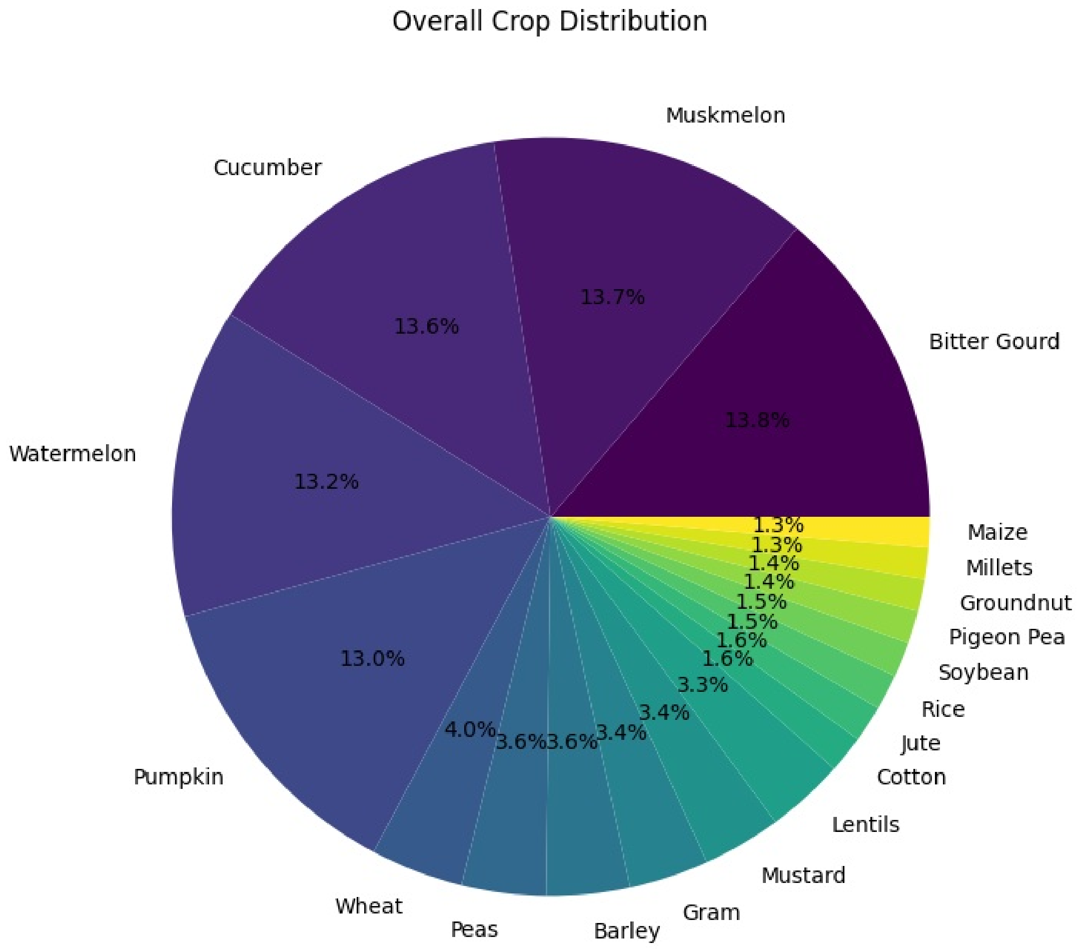
```
Most recommended crops in Rainy Weather:
 Recommended Crop
Cucumber        394
Bitter Gourd    387
Watermelon      377
Muskmelon       367
Pumpkin         365
Lentils         109
Gram            108
Wheat           108
Peas             91
Barley           88
Mustard          86
Soybean          48
Pigeon Pea       44
Rice             42
Millets          40
Jute             39
Cotton           37
Maize            35
Groundnut        30
Name: count, dtype: int64
Most recommended crops in Dry Weather:
 Recommended Crop
Bitter Gourd    399
Muskmelon       392
Cucumber        366
Pumpkin         364
Watermelon      353
Barley          112
Wheat           108
Peas            108
Lentils         103
Gram             98
Mustard          93
Cotton           50
Pigeon Pea       43
Jute             43
Millets          42
Soybean          41
Maize            40
Rice             38
Groundnut        36
Name: count, dtype: int64
Most recommended crops in moderate Weather:
 Recommended Crop
Watermelon      445
Muskmelon       405
Pumpkin         404
Bitter Gourd    401
Cucumber        391
Wheat           117
Peas            114
Barley           95
Mustard          94
Gram             94
Lentils          79
Cotton           52
Jute             51
Rice             49
Groundnut        46
Soybean          45
Maize            37
Pigeon Pea       36
Millets          32
Name: count, dtype: int64
```

In [53]:

```
plt.figure(figsize=(8,6))
sns.heatmap(df[["Temperature (°C)", "Humidity (%)", "Rainfall (mm)", "Wind Speed (km/h)"]].corr(), annot=True,
plt.title("Correlation Matrix of Weather Data")
plt.show()
```



```
In [54]: plt.figure(figsize=(8, 8))
         df["Recommended Crop"].value_counts().plot.pie(autopct="%1.1f%%", cmap="viridis")
         plt.title("Overall Crop Distribution")
         plt.ylabel("")
         plt.show()
```



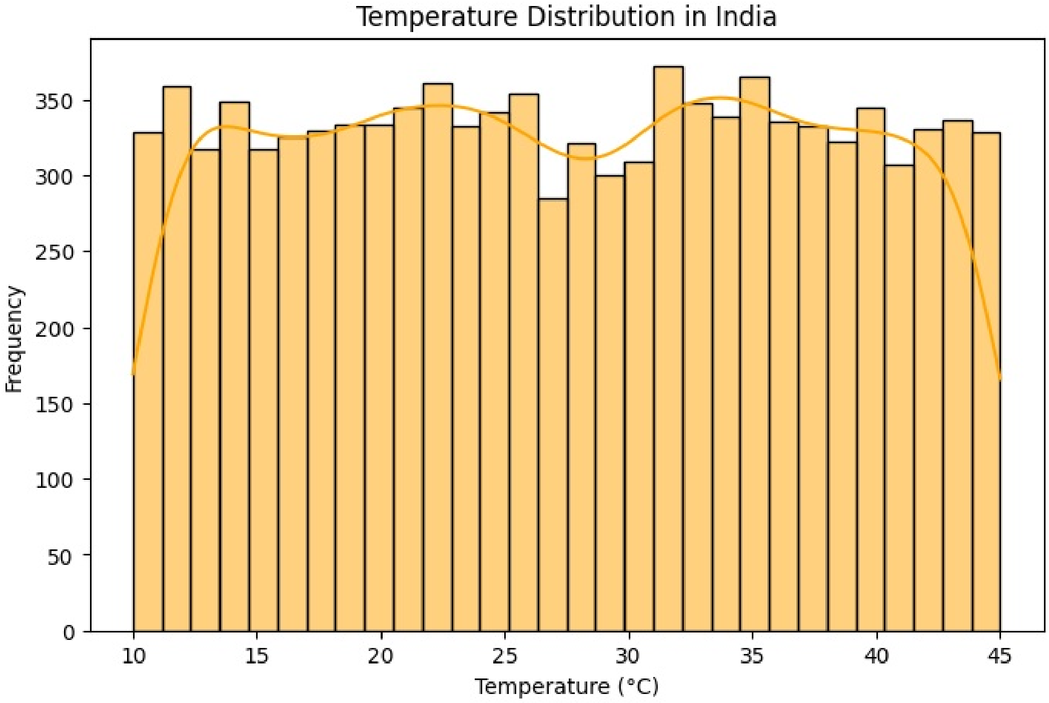
```
In [55]: best_farming_cities = df.groupby("City").agg({
             "Temperature (°C)": "mean",
             "Rainfall (mm)": "sum",
             "Humidity (%)": "mean"
         }).round(1).sort_values(by=["Rainfall (mm)"], ascending=False)
```

```
print(best_farming_cities.head(10))  # Top 10 cities with high rainfall
```

```
               Temperature (°C)  Rainfall (mm)  Humidity (%)
City
Delhi                      27.5        28633.3          59.2
Jaipur                     26.6        28340.9          59.8
Hyderabad                  26.6        27810.2          57.8
Coimbatore                 27.9        27249.9          60.1
Pune                       27.4        26296.8          59.7
Ahmedabad                  28.0        26240.5          59.2
Bhopal                     27.6        26057.9          60.6
Visakhapatnam              28.0        25986.3          58.0
Lucknow                    27.2        25352.8          59.4
Kolkata                    28.2        25263.5          59.0
```

In [56]:

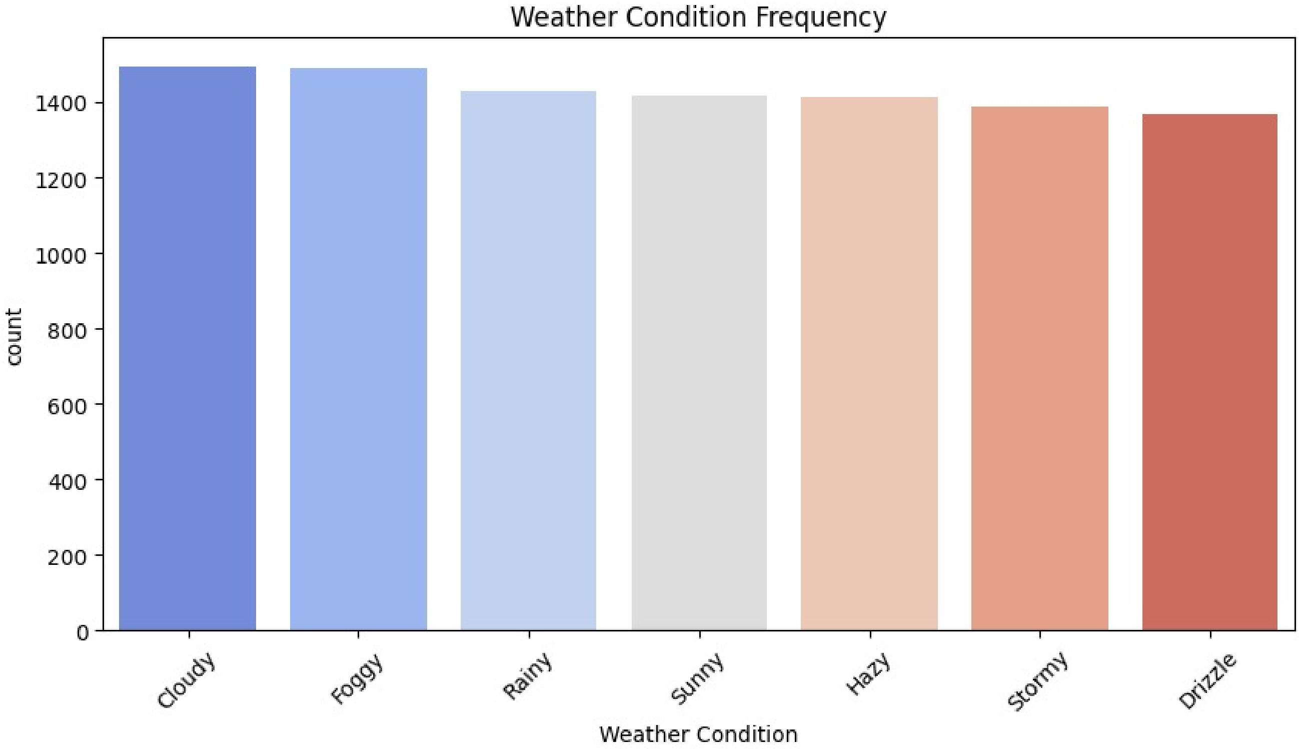
```
import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(8,5))
sns.histplot(df["Temperature (°C)"], bins=30, kde=True, color='orange')
plt.title("Temperature Distribution in India")
plt.xlabel("Temperature (°C)")
plt.ylabel("Frequency")
plt.show()
```



In [57]:

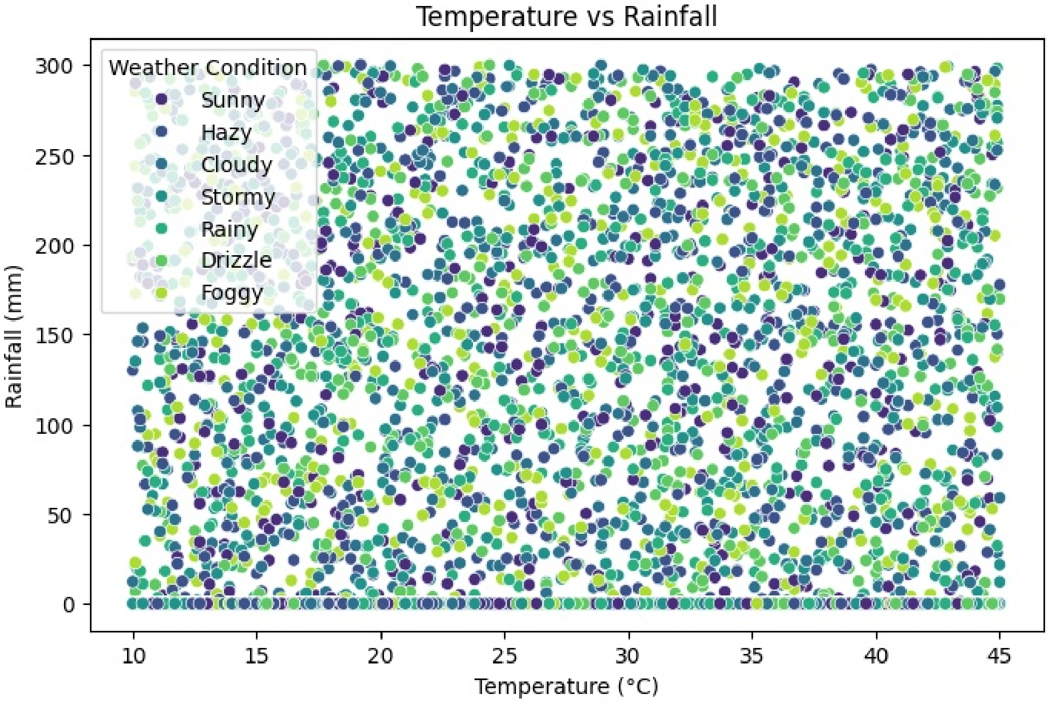
```
plt.figure(figsize=(10,5))
sns.countplot(data=df,x="Weather Condition",palette="coolwarm",order=df["Weather Condition"].value_counts().ind
plt.xticks(rotation=45)
plt.title("Weather Condition Frequency")
plt.show()
```

```
C:\Users\sunit\AppData\Local\Temp\ipykernel_14816\1954278056.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df,x="Weather Condition",palette="coolwarm",order=df["Weather Condition"].value_counts().in
dex)
```



In [58]:

```
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df, x="Temperature (°C)", y="Rainfall (mm)", hue="Weather Condition", palette="viridis")
plt.title("Temperature vs Rainfall")
plt.show()
```

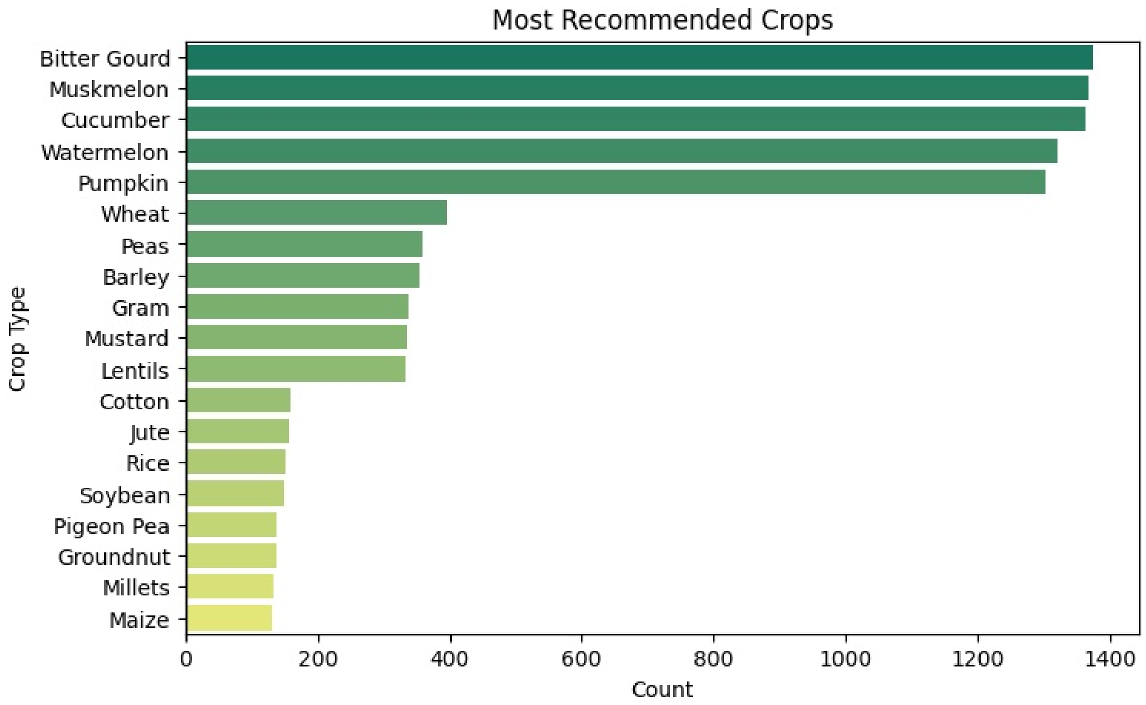


In [59]:

```
plt.figure(figsize=(8, 5))
```

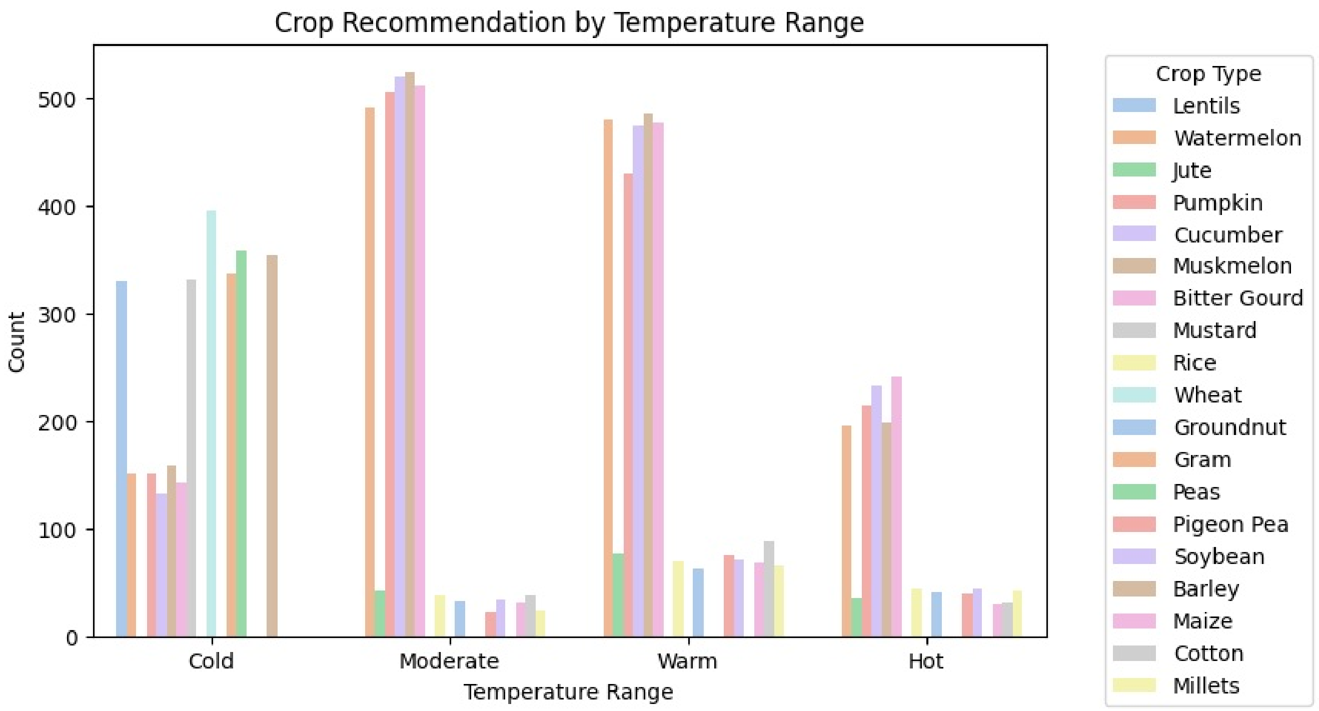
```
sns.countplot(data=df, y="Recommended Crop", order=df["Recommended Crop"].value_counts().index, palette="summer
plt.title("Most Recommended Crops")
plt.xlabel("Count")
plt.ylabel("Crop Type")
plt.show()
```

```
C:\Users\sunit\AppData\Local\Temp\ipykernel_14816\225563852.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable
to `hue` and set `legend=False` for the same effect.

  sns.countplot(data=df, y="Recommended Crop", order=df["Recommended Crop"].value_counts().index, palette="summe
r")
```



In [60]:

```
df["Temp Range"] = pd.cut(df["Temperature (°C)"], bins=[10, 20, 30, 40, 50], labels=["Cold", "Moderate", "Warm"

plt.figure(figsize=(8, 5))
sns.countplot(data=df, x="Temp Range", hue="Recommended Crop", palette="pastel")
plt.title("Crop Recommendation by Temperature Range")
plt.xlabel("Temperature Range")
plt.ylabel("Count")
plt.legend(title="Crop Type", bbox_to_anchor=(1.05, 1), loc='upper left')
plt.show()
```



In [61]:

```
best_crops = df.groupby("City")["Recommended Crop"].agg(lambda x: x.value_counts().idxmax())
print(best_crops)
```

```
City
Ahmedabad           Muskmelon
Bangalore            Cucumber
Bhopal               Cucumber
Chandigarh            Pumpkin
Chennai              Cucumber
Coimbatore          Muskmelon
Delhi                Cucumber
Hyderabad        Bitter Gourd
Indore               Cucumber
Jaipur                Pumpkin
Kochi            Bitter Gourd
Kolkata            Watermelon
Lucknow          Bitter Gourd
Mumbai               Cucumber
Nagpur               Cucumber
Patna            Bitter Gourd
Pune             Bitter Gourd
Visakhapatnam    Bitter Gourd
Name: Recommended Crop, dtype: object
```

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js