In [6]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
import warnings
warnings.filterwarnings("ignore")
%matplotlib inline
```

In [7]:

```
df = pd.read_csv('unemployment_data.csv')
df
```

Out[7]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...	...
763	NaN	NaN	NaN	NaN	NaN	NaN	NaN
764	NaN	NaN	NaN	NaN	NaN	NaN	NaN
765	NaN	NaN	NaN	NaN	NaN	NaN	NaN
766	NaN	NaN	NaN	NaN	NaN	NaN	NaN
767	NaN	NaN	NaN	NaN	NaN	NaN	NaN

768 rows × 7 columns

In [8]:

```
df.head()
```

Out[8]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural

In [9]:

```
df.columns
```

Out[9]:

```
Index(['Region', ' Date', ' Frequency', ' Estimated Unemployment Rate
(%)',
       ' Estimated Employed', ' Estimated Labour Participation Rate (%)',
       'Area'],
      dtype='object')
```

In [10]:
```
df.columns=df.columns.str.strip()
df
```

Out[10]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...	...
763	NaN	NaN	NaN	NaN	NaN	NaN	NaN
764	NaN	NaN	NaN	NaN	NaN	NaN	NaN
765	NaN	NaN	NaN	NaN	NaN	NaN	NaN
766	NaN	NaN	NaN	NaN	NaN	NaN	NaN
767	NaN	NaN	NaN	NaN	NaN	NaN	NaN

768 rows × 7 columns

In [11]:
```
print(f"The dataframe has {df.shape[0]} rows and {df.shape[1]} columns")
```

```
The dataframe has 768 rows and 7 columns
```

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 7 columns):
 #   Column                                   Non-Null Count  Dtype  
---  ------                                   --------------  -----  
 0   Region                                   740 non-null    object 
 1   Date                                     740 non-null    object 
 2   Frequency                                740 non-null    object 
 3   Estimated Unemployment Rate (%)          740 non-null    float64
 4   Estimated Employed                       740 non-null    float64
 5   Estimated Labour Participation Rate (%)  740 non-null    float64
 6   Area                                     740 non-null    object 
dtypes: float64(3), object(4)
memory usage: 42.1+ KB
```

In [13]:

```
df.describe()
```

Out[13]:

	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)
count	740.000000	7.400000e+02	740.000000
mean	11.787946	7.204460e+06	42.630122
std	10.721298	8.087988e+06	8.111094
min	0.000000	4.942000e+04	13.330000
25%	4.657500	1.190404e+06	38.062500
50%	8.350000	4.744178e+06	41.160000
75%	15.887500	1.127549e+07	45.505000
max	76.740000	4.577751e+07	72.570000

In [14]:

```
print(df.isnull().sum())
```

```
Region                                      28
Date                                        28
Frequency                                   28
Estimated Unemployment Rate (%)             28
Estimated Employed                          28
Estimated Labour Participation Rate (%)     28
Area                                        28
dtype: int64
```

```
In [15]: df = df.dropna()
         df
```

Out[15]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area
0	Andhra Pradesh	31-05-2019	Monthly	3.65	11999139.0	43.24	Rural
1	Andhra Pradesh	30-06-2019	Monthly	3.05	11755881.0	42.05	Rural
2	Andhra Pradesh	31-07-2019	Monthly	3.75	12086707.0	43.50	Rural
3	Andhra Pradesh	31-08-2019	Monthly	3.32	12285693.0	43.97	Rural
4	Andhra Pradesh	30-09-2019	Monthly	5.17	12256762.0	44.68	Rural
...	...	...	...	...	...	...	...
749	West Bengal	29-02-2020	Monthly	7.55	10871168.0	44.09	Urban
750	West Bengal	31-03-2020	Monthly	6.67	10806105.0	43.34	Urban
751	West Bengal	30-04-2020	Monthly	15.63	9299466.0	41.20	Urban
752	West Bengal	31-05-2020	Monthly	15.22	9240903.0	40.67	Urban
753	West Bengal	30-06-2020	Monthly	9.86	9088931.0	37.57	Urban

740 rows × 7 columns

```
In [16]: print(df.isnull().sum())
```

```
Region                                      0
Date                                        0
Frequency                                   0
Estimated Unemployment Rate (%)             0
Estimated Employed                          0
Estimated Labour Participation Rate (%)     0
Area                                        0
dtype: int64
```

```
In [17]: print(df.duplicated().sum())
```

```
0
```

In [18]:
```
df.columns
```

Out[18]:
```
Index(['Region', 'Date', 'Frequency', 'Estimated Unemployment Rate (%)',
       'Estimated Employed', 'Estimated Labour Participation Rate (%)',
       'Area'],
      dtype='object')
```

In [19]:
```
df['Date'] = pd.to_datetime(df['Date'])
df['Day'] = df['Date'].dt.day
df['Month'] = df['Date'].dt.month_name()
df['Year'] = df['Date'].dt.year
import warnings
warnings.filterwarnings("ignore")
df
```

Out[19]:

	Region	Date	Frequency	Estimated Unemployment Rate (%)	Estimated Employed	Estimated Labour Participation Rate (%)	Area	Day	M
0	Andhra Pradesh	2019-05-31	Monthly	3.65	11999139.0	43.24	Rural	31	
1	Andhra Pradesh	2019-06-30	Monthly	3.05	11755881.0	42.05	Rural	30	
2	Andhra Pradesh	2019-07-31	Monthly	3.75	12086707.0	43.50	Rural	31	
3	Andhra Pradesh	2019-08-31	Monthly	3.32	12285693.0	43.97	Rural	31	Au
4	Andhra Pradesh	2019-09-30	Monthly	5.17	12256762.0	44.68	Rural	30	Septe
...	...	...	...	...	...	...	...	...	
749	West Bengal	2020-02-29	Monthly	7.55	10871168.0	44.09	Urban	29	Feb
750	West Bengal	2020-03-31	Monthly	6.67	10806105.0	43.34	Urban	31	M
751	West Bengal	2020-04-30	Monthly	15.63	9299466.0	41.20	Urban	30	
752	West Bengal	2020-05-31	Monthly	15.22	9240903.0	40.67	Urban	31	
753	West Bengal	2020-06-30	Monthly	9.86	9088931.0	37.57	Urban	30	

740 rows × 10 columns

In [20]:
```
df.columns
```

Out[20]:
```
Index(['Region', 'Date', 'Frequency', 'Estimated Unemployment Rate (%)',
       'Estimated Employed', 'Estimated Labour Participation Rate (%)', 'A
rea',
       'Day', 'Month', 'Year'],
      dtype='object')
```

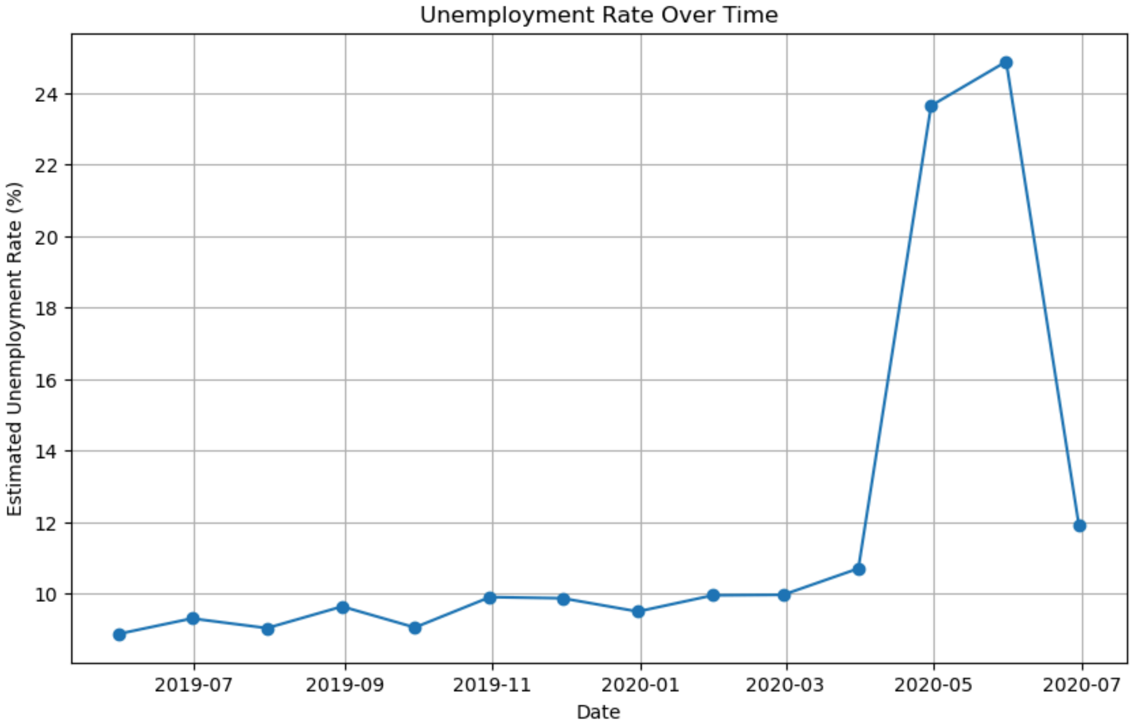
In [21]:

```
print(df.describe())

# Mean unemployment rate
mean_unemployment = df['Estimated Unemployment Rate (%)'].mean()
print(f'Mean Unemployment Rate: {mean_unemployment:.2f}%')

# Median unemployment rate
median_unemployment = df['Estimated Unemployment Rate (%)'].median()
print(f'Median Unemployment Rate: {median_unemployment:.2f}%')

# Standard deviation of unemployment rate
std_unemployment = df['Estimated Unemployment Rate (%)'].std()
print(f'Standard Deviation of Unemployment Rate: {std_unemployment:.2f}%')
```

```
       Estimated Unemployment Rate (%)  Estimated Employed  \
count                       740.000000        7.400000e+02   
mean                         11.787946        7.204460e+06   
std                          10.721298        8.087988e+06   
min                           0.000000        4.942000e+04   
25%                           4.657500        1.190404e+06   
50%                           8.350000        4.744178e+06   
75%                          15.887500        1.127549e+07   
max                          76.740000        4.577751e+07   

       Estimated Labour Participation Rate (%)         Day         Year  
count                               740.000000  740.000000   740.000000  
mean                                 42.630122   30.502703  2019.418919  
std                                   8.111094    0.627509     0.493716  
min                                  13.330000   29.000000  2019.000000  
25%                                  38.062500   30.000000  2019.000000  
50%                                  41.160000   31.000000  2019.000000  
75%                                  45.505000   31.000000  2020.000000  
max                                  72.570000   31.000000  2020.000000  
Mean Unemployment Rate: 11.79%
Median Unemployment Rate: 8.35%
Standard Deviation of Unemployment Rate: 10.72%
```

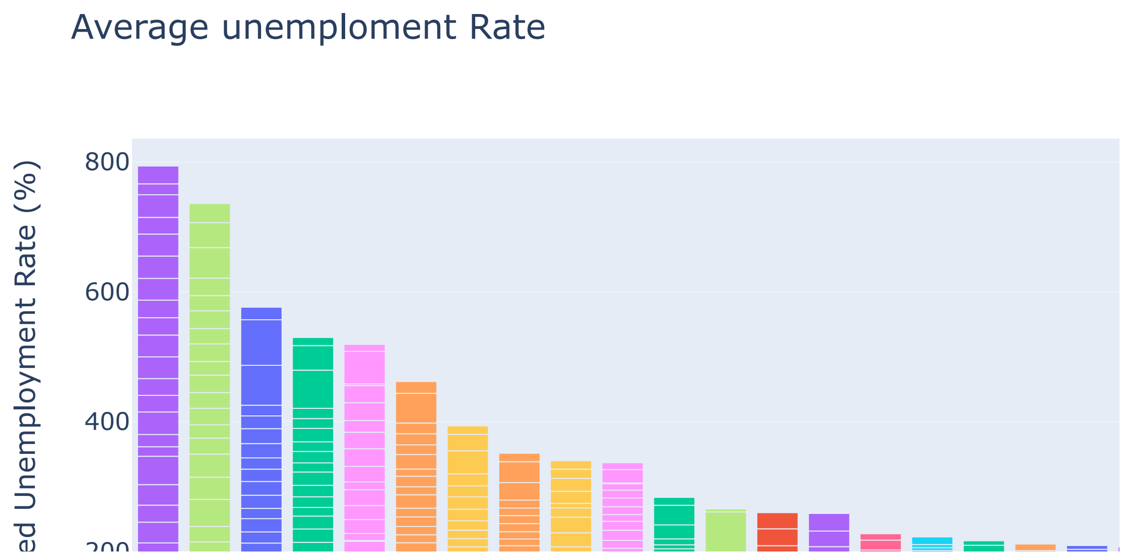
In [22]:

```
import matplotlib.pyplot as plt

# Group the data by Date and calculate the mean unemployment rate for each 
mean_unemployment_over_time = df.groupby('Date')['Estimated Unemployment Ra

# Create a line plot
plt.figure(figsize=(10, 6))
plt.plot(mean_unemployment_over_time.index, mean_unemployment_over_time.valu
plt.xlabel('Date')
plt.ylabel('Estimated Unemployment Rate (%)')
plt.title('Unemployment Rate Over Time')
plt.grid(True)

plt.show()
```

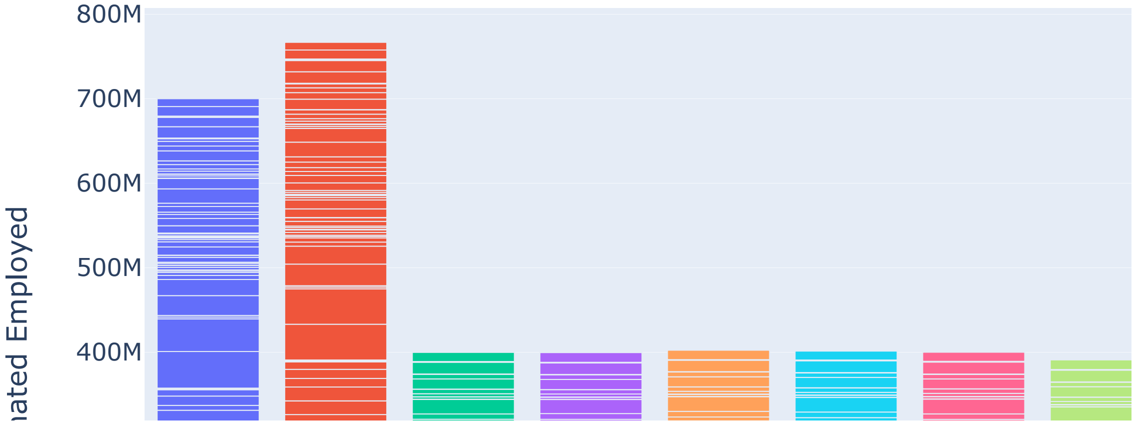


In [23]:

```
fig = px.bar(df, x = 'Region', y = "Estimated Unemployment Rate (%)", color
fig.update_layout(xaxis = {'categoryorder':'total descending'})
fig.show()
```



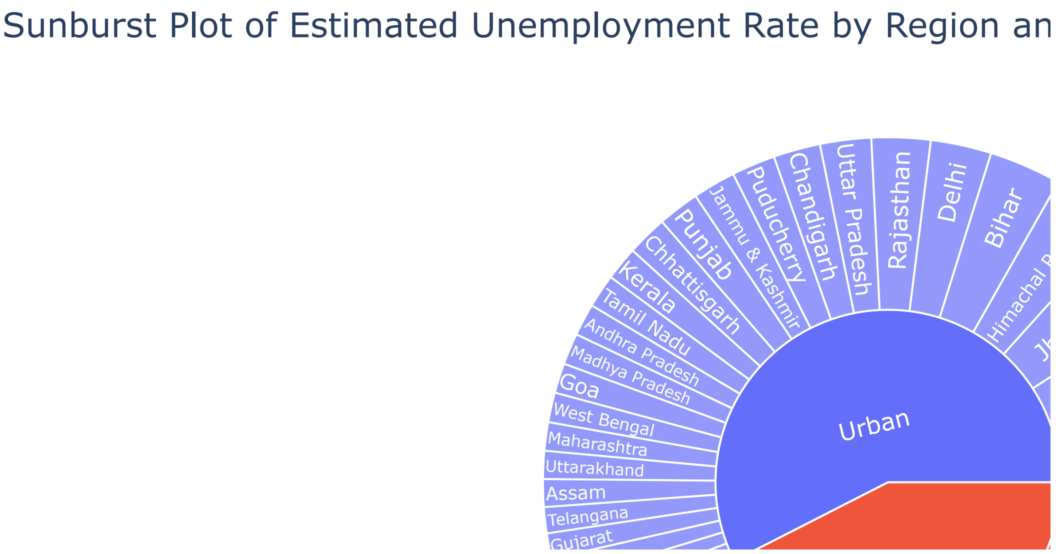
```
In [24]: fig = px.bar(df, x = 'Month', y = 'Estimated Employed', color = 'Month', ti
         fig.show()
```



Estimated Employed People

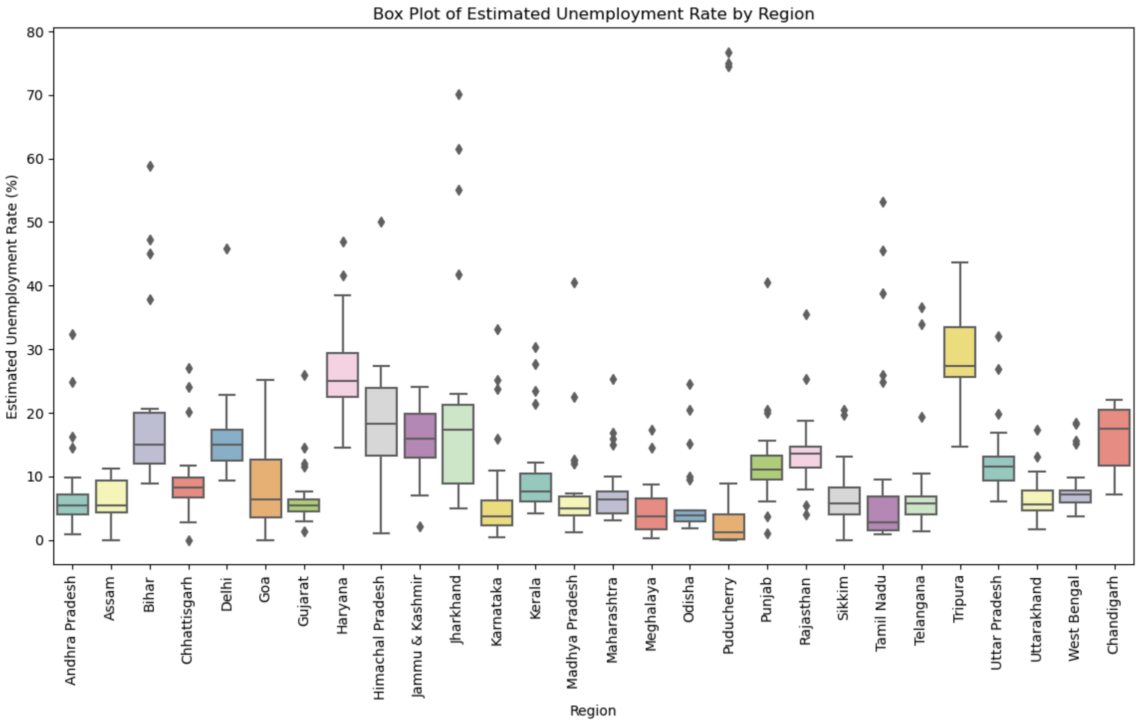
In [25]:

```
fig = px.sunburst(df, path=['Area', 'Region'], values='Estimated Unemploymen
fig.show()
```



In [26]:

```
plt.figure(figsize=(14, 7))
# Create the box plot with separate colors for each region
sns.boxplot(x='Region', y='Estimated Unemployment Rate (%)', data=df, palet
# Rotate x-axis labels for better readability
plt.xticks(rotation='vertical')
# Add title and axis labels
plt.title('Box Plot of Estimated Unemployment Rate by Region')
plt.xlabel('Region')
plt.ylabel('Estimated Unemployment Rate (%)')

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

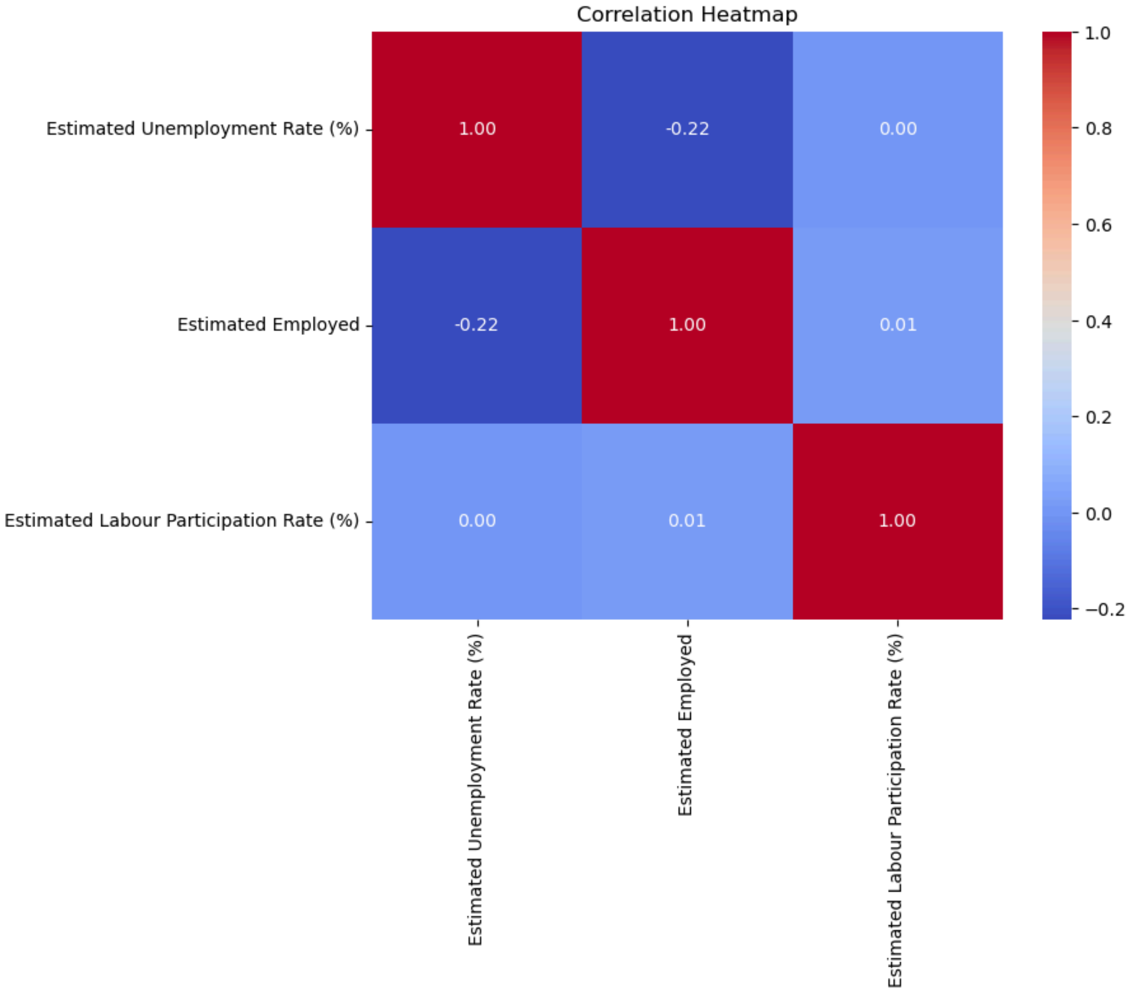


In [27]:

```
correlation = df[['Estimated Unemployment Rate (%)', 'Estimated Employed',
print(correlation)
```

```
                                         Estimated Unemployment Rate (%)  \
Estimated Unemployment Rate (%)                                 1.000000
Estimated Employed                                             -0.222876
Estimated Labour Participation Rate (%)                         0.002558

                                         Estimated Employed  \
Estimated Unemployment Rate (%)                   -0.222876
Estimated Employed                                 1.000000
Estimated Labour Participation Rate (%)            0.011300

                                         Estimated Labour Participation Ra
te (%)
Estimated Unemployment Rate (%)                                         0.
002558
Estimated Employed                                                      0.
011300
Estimated Labour Participation Rate (%)                                 1.
000000
```

In [28]:
```
correlation = df[['Estimated Unemployment Rate (%)', 'Estimated Employed',

# Plot heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(correlation, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap')
plt.show()
```

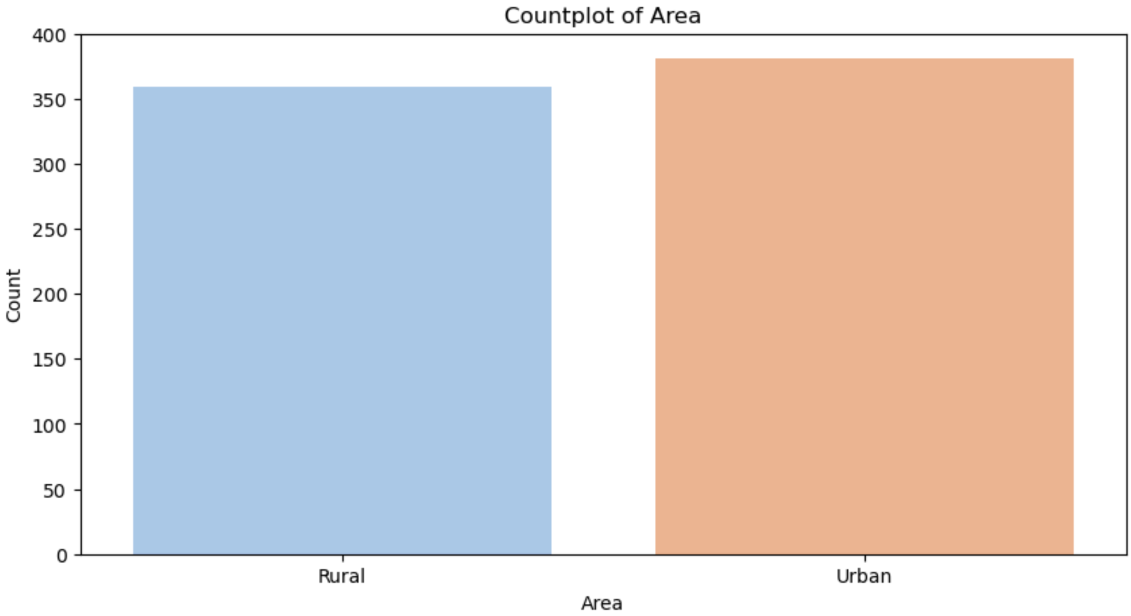


In [29]:
```
from scipy.stats import ttest_ind

urban_unemployment = df[df['Area'] == 'Urban']['Estimated Unemployment Rate
rural_unemployment = df[df['Area'] == 'Rural']['Estimated Unemployment Rate

t_stat, p_val = ttest_ind(urban_unemployment, rural_unemployment)
print(f'T-Statistic: {t_stat:.2f}')
print(f'P-Value: {p_val:.2f}')
```

```
T-Statistic: 3.63
P-Value: 0.00
```

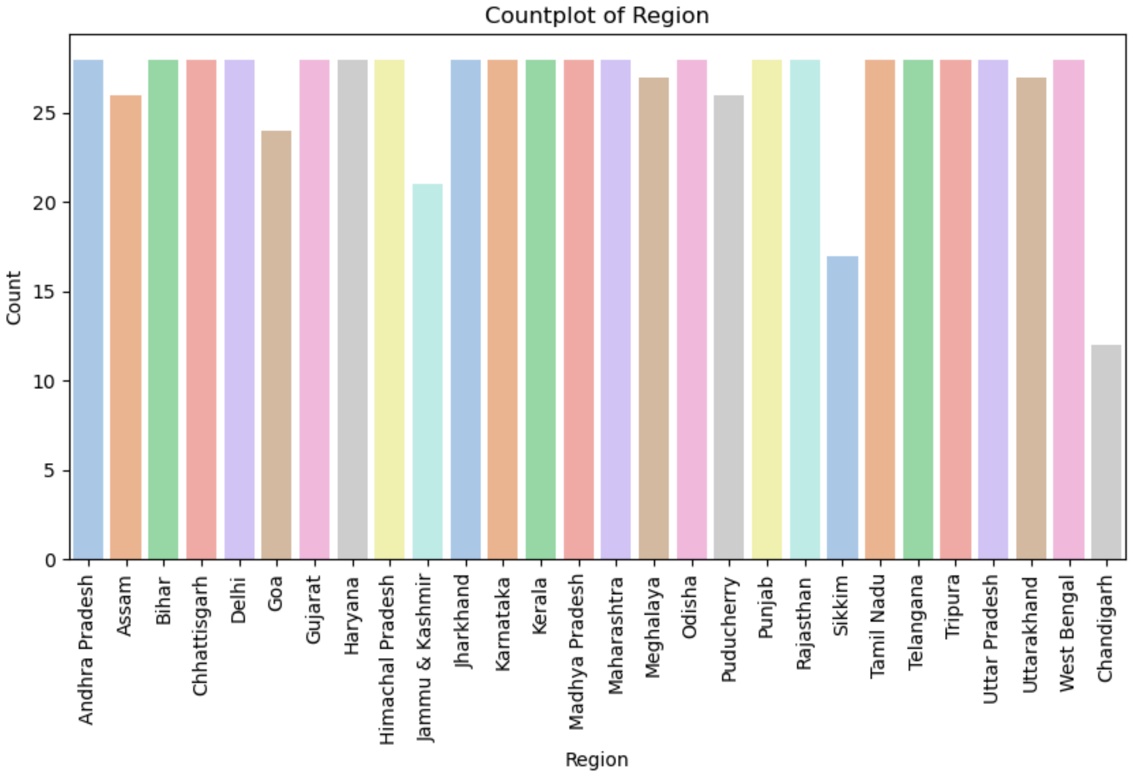
In [30]:

```
fig = plt.figure(figsize=(10, 5))
sns.countplot(x='Area', data=df,palette='pastel')
plt.title('Countplot of Area')
plt.xlabel('Area')
plt.ylabel('Count')
# save the plot
plt.savefig('countplot_area.png', dpi=300)
plt.show()
```



In [31]:

```
plt.figure(figsize=(10, 5))
sns.countplot(x='Region', data=df,palette='pastel')
plt.xticks(rotation ='vertical')
plt.title('Countplot of Region')
plt.xlabel('Region')
plt.ylabel('Count')
plt.show()
```



In [ ]: