Tarea 3 - Programación de aplicaciones en SIG

Introduction Data Set

To solve Task 3 of the class ***"Programación de aplicaciones en SIG"*** a data set have been choosen that contains daily data of the recent year 2023. Those data have their origin from weather stations of the west European countries such as Spain. In total the data set includes 188 465 data point, yet some of the stations are not located inside the Europa as they provide e.g. data from French oversee territory.

The collected data contain the following Information:

- Station Code: **STATION**
- Station Name: **NAME**
- City: **City**
- Country: **Country**
- Latitude: **LATITUDE**
- Longitude: **LONGITUDE**
- Elevation: **ELEVATION**
- Date: **DATE**
- Amount of precipitation: **PRCP**
- Amount of snow: **SNWD**
- Average air Temperature: **TAVG**
- Maximal temperature: **TMAX**
- Minimal temperature: **TMIN**

Those data have been provided in form of a CSV file by the National Oceanic and Atmospheric Administration (NOAA) (https://www.ncei.noaa.gov/cdo-web/), a in Washington, D.C. US based scientific research agency. The data are openly accessible and NOAA itself describes its mission as forecasting and sharing climate, weather, ocean, and coastal changes, and safeguard marine ecosystems and resources.

In [170]:
```
import pandas as pd
import plotly.express as px
from datetime import datetime
```

In [171]:
```
df = pd.read_csv("C:\SIG\DATA_TAREA_2B.csv")
```

In [172]:
```
#df.tail()
```

Overview of the Provided Data Frame

In the following the Date Range of the data is queried. Therefore the date format has to be defined first. As can be seen the data set contain data from the beginning of 2023 until the 27th of October 2023

In [173]:
```
df['DATE'] = pd.to_datetime(df['DATE'], format = "%d/%m/%Y").dt.date

earliest_date = df['DATE'].min()
latest_date = df['DATE'].max()

print("DATE RANGE:")
print("- Earliest Date:", earliest_date)
print("- Latest Date:", latest_date)
```

```
DATE RANGE:
- Earliest Date: 2023-01-01
- Latest Date: 2023-10-27
```

As several European countries have oversee territories and the station of those are as well part of the data set, those stations will be grouped as "Others" in the column of countries. This takes place as those oversee territories are often located in location with different climate conditions and weather patters. Anlaysing those data together with the on the European continent located can falsify results

In [174]:
```
values_to_group = ["FG", "FP", "GP", "MB", "MF", "SV", "SI", "SB"]

df['Country'] = df['Country'].replace(values_to_group, "Others")

stations_per_country = df.groupby('Country')['STATION'].nunique()
```

In [175]:
```
import plotly.graph_objects as go

Table1 = go.Figure(data=[go.Table(
    header=dict(values=['Country', 'Number of Stations'],
                fill_color='lightblue',
                align='center'),
    cells=dict(values=[stations_per_country.index, stations_per_country.values],
               fill_color='lightgrey',
               align='left'))
])

Table1.update_layout(title={'text':'Number of Stations per Country'},autosize=False, width=500, height=400)
Table1.show()
```

Number of Stations per Country

Country	Number of Stations
Belgium	1
France	68
Italy	35
Netherlands	340
Others	11
Portugal	6
Spain	171
Switzerland	9
UK	87

Analysis and Visualisation of the Data

Location of Weatherstations of Portugal and Switzerland

The two countries *Portugal* and *Switzerland* have been selected for further analysis. In Map 1 the Locations of the weather stations of both countries can be found.

In [176]:
```
from folium import Marker
import folium
import requests
```

In [177]:
```
selected_countries = ['Switzerland', 'Portugal']
df_selected_countries = df[df['Country'].isin(selected_countries)]


map_center = [df_selected_countries['LATITUDE'].mean(), df_selected_countries['LONGITUDE'].mean()]
m = folium.Map(location=map_center, zoom_start=4.5)

folium.TileLayer(
    tiles='CartoDB positron',
    name='CartoDB positron').add_to(m)

# ESRI NatGeo World Map
folium.TileLayer(
    tiles='http://services.arcgisonline.com/arcgis/rest/services/NatGeo_World_Map/MapServer/MapServer/tile/{z}/{y}/{x}
    name='NatGeo World Map',
    attr='ESRI NatGeo World Map').add_to(m)

# Make makers to layer
marker_layer = folium.FeatureGroup(name='Weather Stations')



for _, station in df_selected_countries.iterrows():
    folium.Marker([station['LATITUDE'], station['LONGITUDE']],
                  popup=f"{station['NAME']} - {station['City']}, {station['Country']}").add_to(marker_layer)

# Add Layer to map
marker_layer.add_to(m)


folium.LayerControl().add_to(m)

m
```

Out[177]:

Tables with Accumulated Rainfall 2023 per Station

In the following two tables the accumulated rainfall of 2023 of each stations in Switzerland and Portugal diveded by coutry is calculated.

In [178]:
```
import plotly.graph_objects as go

def create_accumulated_rainfall_table(df, country):

    df_country = df[df['Country'] == country]


    accumulated_rainfall = df_country.groupby('NAME')['PRCP'].sum().reset_index()
    accumulated_rainfall['PRCP'] = accumulated_rainfall['PRCP'].round(2)


    table = go.Figure(data=[go.Table(
        header=dict(values=['Station Name', 'Accumulated Rainfall (mm)'],
                    fill_color='lightblue',
                    align='center'),
        cells=dict(values=[accumulated_rainfall['NAME'], accumulated_rainfall['PRCP']],
                   fill_color='lightgrey',
                   align='left'))
    ])

    table.update_layout(title={'text': f'Accumulated Rainfall of Weather Stations in {country}'},
                        autosize=False, width=600, height=400)


    table.show()

create_accumulated_rainfall_table(df, 'Portugal')
create_accumulated_rainfall_table(df, 'Switzerland')
```

Accumulated Rainfall of Weather Stations in Portugal

Station Name	Accumulated Rainfall (mm)
BEJA,PO	0
FARO,PO	1.5
FLORES,PO	6.1
PONTADELGADA,PO	89.9
PORTOSANTO,PO	208.6
SANTAMARIA,PO	0

Accumulated Rainfall of Weather Stations in Switzerland

Station Name	Accumulated Rainfall (mm)
BASELBINNINGEN,SZ	584.9
COLDUGRANDSTBERNARD,SZ	986.3
GENEVECOINTRIN,SZ	657
LUGANO,SZ	1229
PAYERNE,SZ	556
SAENTIS,SZ	2557
SION2,SZ	505.4
STATIONMARIAVALMUESTAIR,SZ	597.8
ZUERICHFLUNTERN,SZ	813.9

Daily Average Precipitation per Country

From all stations of each countries the daily average precipitation data are going to be compared in the following interactive bar chart.

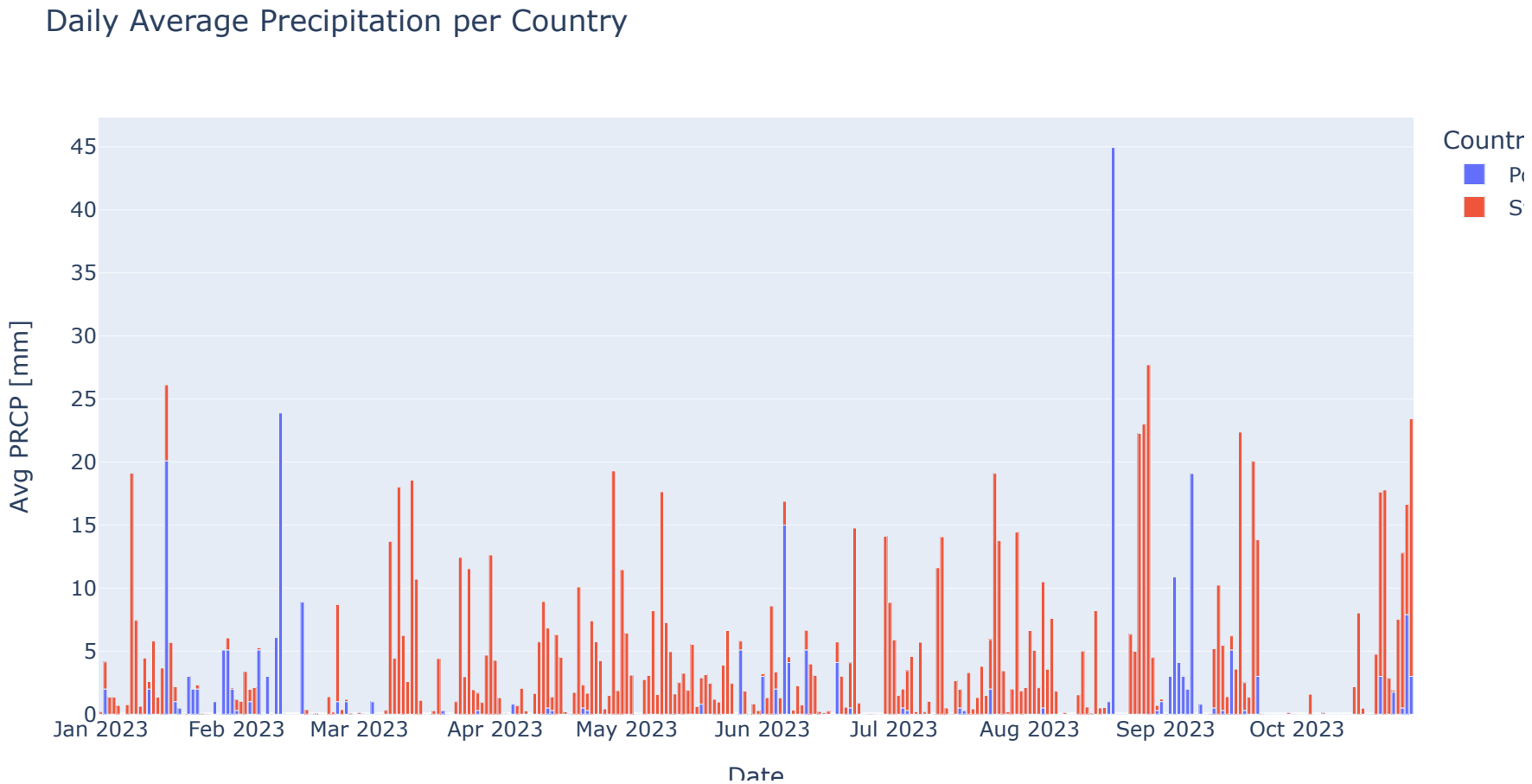
In [179]:
```
selected_countries = ['Portugal','Switzerland']
filtered_df = df[df['Country'].isin(selected_countries)]
#filtered_df.head()
```

In [180]:
```
filtered_df['DATE'] = pd.to_datetime(filtered_df['DATE'])
```

```
C:\Users\there\AppData\Local\Temp\ipykernel_20552\2803641203.py:1: SettingWithCopyWarning:


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
```

In [181]:
```
daily_avg_precip = filtered_df.groupby(['Country', 'DATE'])['PRCP'].agg('mean').reset_index()
```

In [182]:
```
fig1 = px.bar(
    daily_avg_precip,
    x='DATE',
    y='PRCP',
    color='Country',
    labels={'PRCP': 'Avg PRCP'},
    title='Daily Average Precipitation per Country')

fig1.update_xaxes(title_text='Date')
fig1.update_yaxes(title_text='Avg PRCP [mm]')

fig1.show()
```

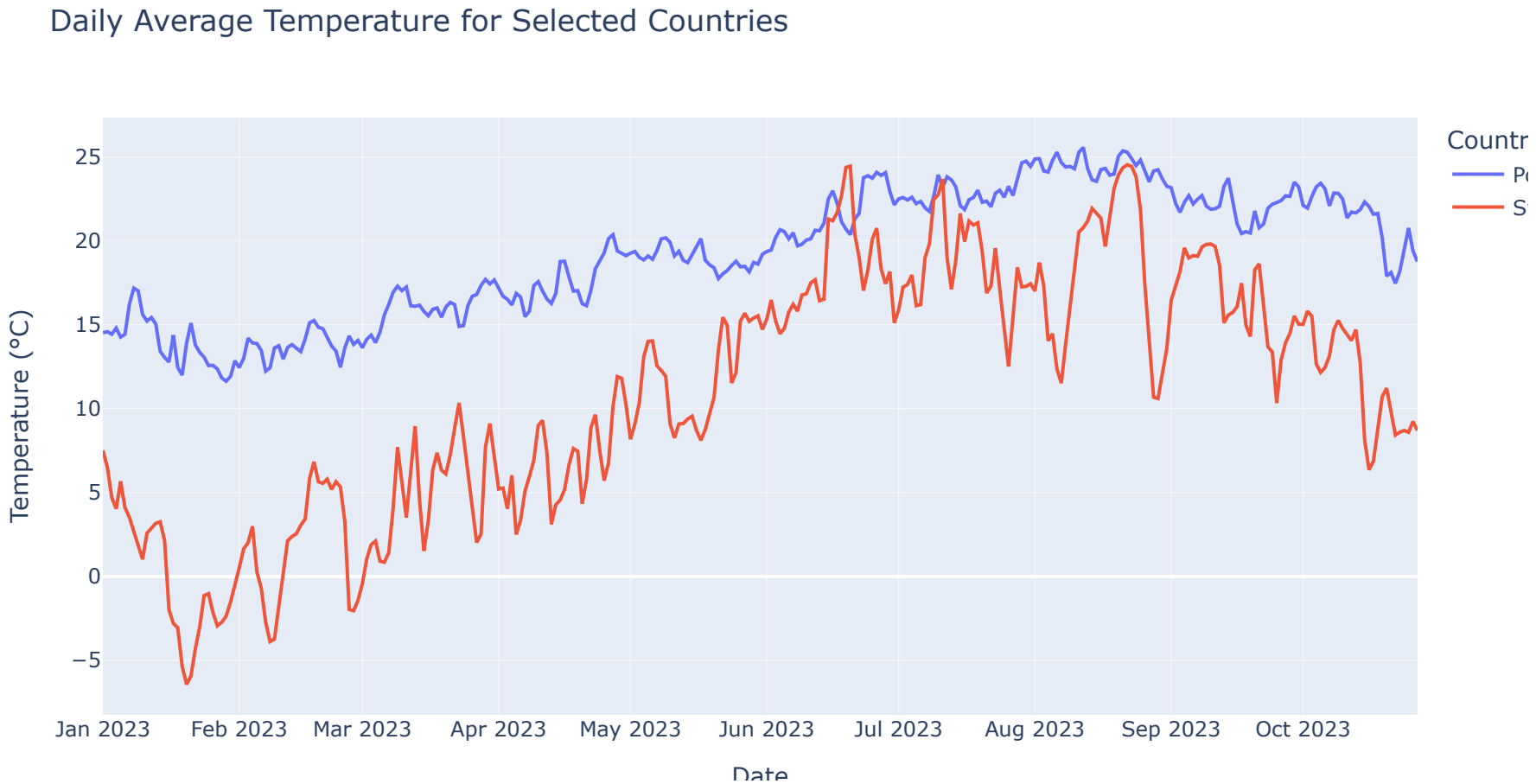


Daily Average Temperature for Selected Countries

Furthermore of the same countries the daily average min and max temperature is calculated and displayed in a line plot, to compare the temperatures of both countries over 2023.

In [183]:
```
daily_avg_TEMP = filtered_df.groupby(['Country', 'DATE'])['TAVG'].agg('mean').reset_index()
```

In [184]:
```
fig2 = px.line(
    daily_avg_TEMP,
    x='DATE',
    y=['TAVG'],
    color='Country',
    labels={'TAVG': 'Avg T'},
    title='Daily Average Temperature for Selected Countries')

fig2.update_layout(xaxis_title='Date', yaxis_title='Temperature (°C)')
fig2.update_traces(name='TAVG', selector=dict(name='TAVG'))

fig2.show()
```



Heatmap of Average Temperature for Selected Countries

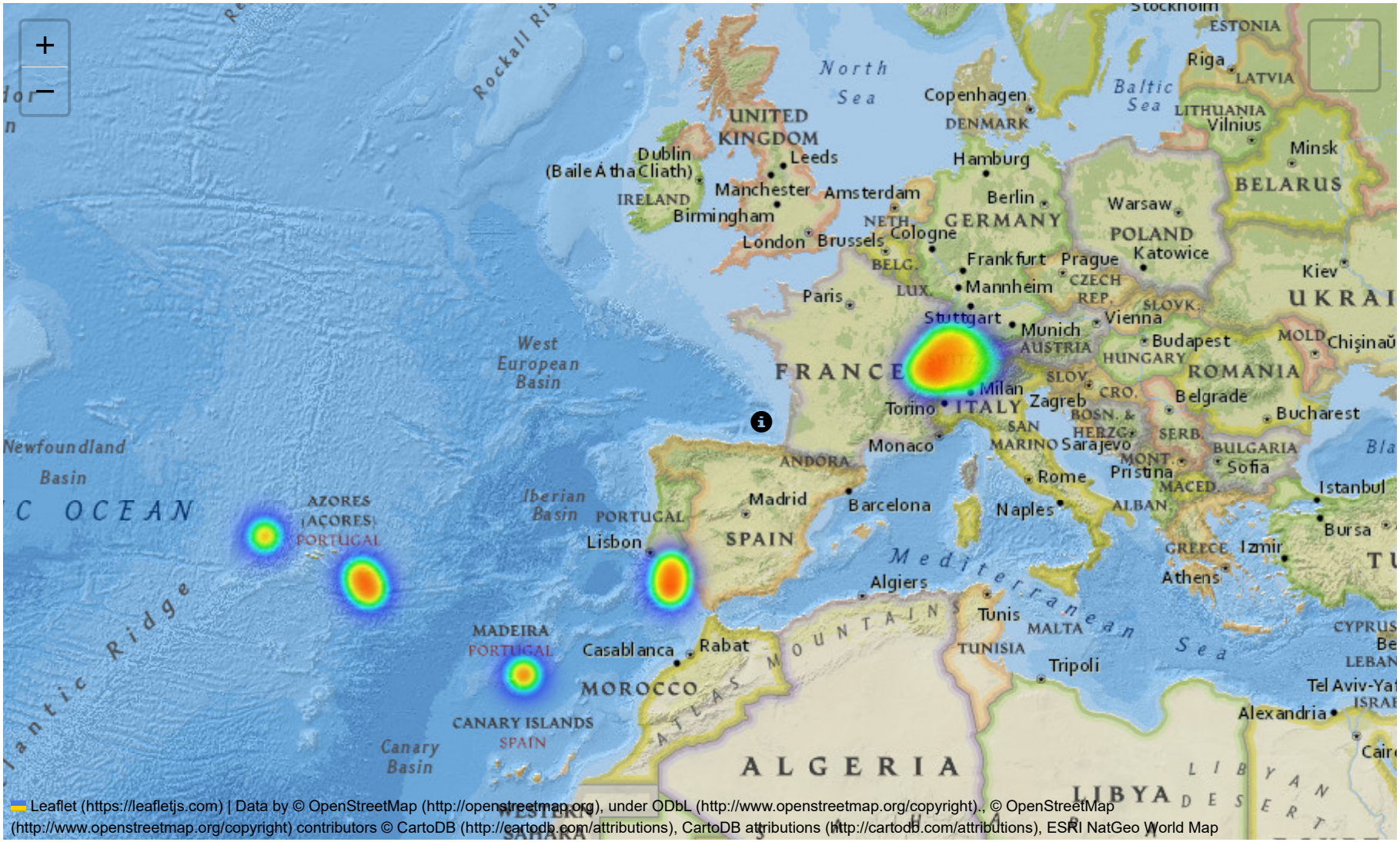
Furthermore of the same countries the average emperature is calculated for each station and displayed in Map 2.

In [185]:
```
Avg_TEMP = filtered_df.groupby(['City', 'LONGITUDE', 'LATITUDE'])['TAVG'].agg('mean').reset_index()
```

In [186]:
```
#Avg_TEMP.head()
```

In [187]:
```
import folium
from folium.plugins import HeatMap

# Folium map
map_center = [Avg_TEMP['LATITUDE'].mean(), Avg_TEMP['LONGITUDE'].mean()]
m = folium.Map(location=map_center, zoom_start=4)

folium.TileLayer(
    tiles='CartoDB positron',
    name='CartoDB positron').add_to(m)

# ESRI NatGeo World Map
folium.TileLayer(
    tiles='http://services.arcgisonline.com/arcgis/rest/services/NatGeo_World_Map/MapServer/MapServer/tile/{z}/{y}/{x}
    name='NatGeo World Map',
    attr='ESRI NatGeo World Map').add_to(m)

# HeatMap layer
heat_data = [[point['LATITUDE'], point['LONGITUDE'], point['TAVG']] for _, point in Avg_TEMP.iterrows()]
HeatMap(heat_data, radius=15, name='Heatmap').add_to(m)

# Marker with Information
label = 'Colors represent temperature values (TAVG)'
folium.Marker(
    location=[map_center[0], map_center[1]],
    popup=label,
    icon=folium.Icon(color='red', icon='info-sign')
).add_to(m)


folium.LayerControl().add_to(m)

m
```

Out[187]:



Conclusion

The selected choice of visualisation and analysis of the given data are only a few of a large varity of posiblilities. Further analysing and comparing those data can provide researchers, experts and decisionmaker useful information for their work. Those data can be e.g. compared with historical data to see changes over decades or can be decision base for investments e.g. in flood protection or prevention measurements.