Assignment 3: Interactive Dashboard - World Happiness Report

In [1]:
```
# Import Required Libraries
import pandas as pd
import numpy as np
import plotly.express as px
import plotly.graph_objects as go
from ipywidgets import interact, Dropdown
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
!pip install plotly
```

```
Requirement already satisfied: plotly in /opt/anaconda3/lib/python3.9/site-packages (5.9.0)
Requirement already satisfied: tenacity>=6.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from plotly) (8.0.1)

[notice] A new release of pip is available: 24.0 -> 25.0.1
[notice] To update, run: pip install --upgrade pip
```

In [3]:
```
!pip install ipywidgets
```

```
Requirement already satisfied: ipywidgets in /opt/anaconda3/lib/python3.9/site-packages (7.6.5)
Requirement already satisfied: ipykernel>=4.5.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (6.15.2)
Requirement already satisfied: ipython-genutils~=0.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: traitlets>=4.3.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.1.1)
Requirement already satisfied: nbformat>=4.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.5.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (3.5.2)
Requirement already satisfied: ipython>=4.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (7.31.1)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (1.0.0)
Requirement already satisfied: appnope in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (0.1.2)
Requirement already satisfied: debugpy>=1.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.1)
Requirement already satisfied: jupyter-client>=6.1.12 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(7.3.4)
Requirement already satisfied: matplotlib-inline>=0.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(0.1.6)
Requirement already satisfied: nest-asyncio in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.5)
Requirement already satisfied: packaging in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (21.3)
Requirement already satisfied: psutil in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (5.9.0)
Requirement already satisfied: pyzmq>=17 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (23.2.0)
Requirement already satisfied: tornado>=6.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (6.1)
Requirement already satisfied: setuptools>=18.5 in /opt/anaconda3/lib/python3.9/site-packages (from ipython>=4.0.0->ipywidgets) (66.1.1)
```

In [4]:
```
!pip install plotly
```

```
Requirement already satisfied: plotly in /opt/anaconda3/lib/python3.9/site-packages (5.9.0)
Requirement already satisfied: tenacity>=6.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from plotly) (8.0.1)

[notice] A new release of pip is available: 24.0 -> 25.0.1
[notice] To update, run: pip install --upgrade pip
```

In [5]:
```
!pip install ipywidgets
```

```
Requirement already satisfied: ipywidgets in /opt/anaconda3/lib/python3.9/site-packages (7.6.5)
Requirement already satisfied: ipykernel>=4.5.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (6.15.2)
Requirement already satisfied: ipython-genutils~=0.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: traitlets>=4.3.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.1.1)
Requirement already satisfied: nbformat>=4.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.5.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (3.5.2)
Requirement already satisfied: ipython>=4.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (7.31.1)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (1.0.0)
Requirement already satisfied: appnope in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (0.1.2)
Requirement already satisfied: debugpy>=1.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.1)
Requirement already satisfied: jupyter-client>=6.1.12 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(7.3.4)
Requirement already satisfied: matplotlib-inline>=0.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(0.1.6)
Requirement already satisfied: nest-asyncio in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.5)
Requirement already satisfied: packaging in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (21.3)
Requirement already satisfied: psutil in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (5.9.0)
Requirement already satisfied: pyzmq>=17 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (23.2.0)
Requirement already satisfied: tornado>=6.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (6.1)
Requirement already satisfied: setuptools>=18.5 in /opt/anaconda3/lib/python3.9/site-packages (from ipython>=4.0.0->ipywidgets) (66.1.1)
```

In [6]:
```
!pip install ipywidgets
```

```
Requirement already satisfied: ipywidgets in /opt/anaconda3/lib/python3.9/site-packages (7.6.5)
Requirement already satisfied: ipykernel>=4.5.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (6.15.2)
Requirement already satisfied: ipython-genutils~=0.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (0.2.0)
Requirement already satisfied: traitlets>=4.3.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.1.1)
Requirement already satisfied: nbformat>=4.2.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (5.5.0)
Requirement already satisfied: widgetsnbextension~=3.5.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (3.5.2)
Requirement already satisfied: ipython>=4.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (7.31.1)
Requirement already satisfied: jupyterlab-widgets>=1.0.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipywidgets) (1.0.0)
Requirement already satisfied: appnope in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (0.1.2)
Requirement already satisfied: debugpy>=1.0 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.1)
Requirement already satisfied: jupyter-client>=6.1.12 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(7.3.4)
Requirement already satisfied: matplotlib-inline>=0.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets)
(0.1.6)
Requirement already satisfied: nest-asyncio in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (1.5.5)
Requirement already satisfied: packaging in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (21.3)
Requirement already satisfied: psutil in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (5.9.0)
Requirement already satisfied: pyzmq>=17 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (23.2.0)
Requirement already satisfied: tornado>=6.1 in /opt/anaconda3/lib/python3.9/site-packages (from ipykernel>=4.5.1->ipywidgets) (6.1)
Requirement already satisfied: setuptools>=18.5 in /opt/anaconda3/lib/python3.9/site-packages (from ipython>=4.0.0->ipywidgets) (66.1.1)
```

In [7]:
```
%%bash
jupyter nbextension enable --py widgetsnbextension
```

```
Enabling notebook extension jupyter-js-widgets/extension...
      - Validating: OK
```

In [8]:
```
### Installation: If the package is not installed.
#```bash
#pip install plotly
#pip install ipywidgets
#```
```

Visualization Technique (25%)

In [9]:
```
### Step 1: Load Dataset
# Load Dataset
url = 'https://github.com/dharshana-rs/VisualDataExploration/blob/main/World%20Happiness%20Report/assets/happiness_report2021.csv?raw=true'
df = pd.read_csv(url)
```

In [10]:
```
#Display sample records
df.head()
```

Out[10]:

	Country_name	Regional_indicator	Ladder_score	Standard_error_of_ladder_score	upperwhisker	lowerwhisker	Logged_GDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_
0	Finland	Western Europe	7.842	0.032	7.904	7.780	10.775	0.954	72.0	
1	Denmark	Western Europe	7.620	0.035	7.687	7.552	10.933	0.954	72.7	
2	Switzerland	Western Europe	7.571	0.036	7.643	7.500	11.117	0.942	74.4	
3	Iceland	Western Europe	7.554	0.059	7.670	7.438	10.878	0.983	73.0	
4	Netherlands	Western Europe	7.464	0.027	7.518	7.410	10.932	0.942	72.4	

In [11]:
```
# Clean Column Names
df.columns = df.columns.str.replace(' ', '_')
df.rename(columns={'Country_name':'Country', 'Regional_indicator':'Region', 'Ladder_score':'Happiness_Score'}, inplace=True)
df.head()
```

Out[11]:

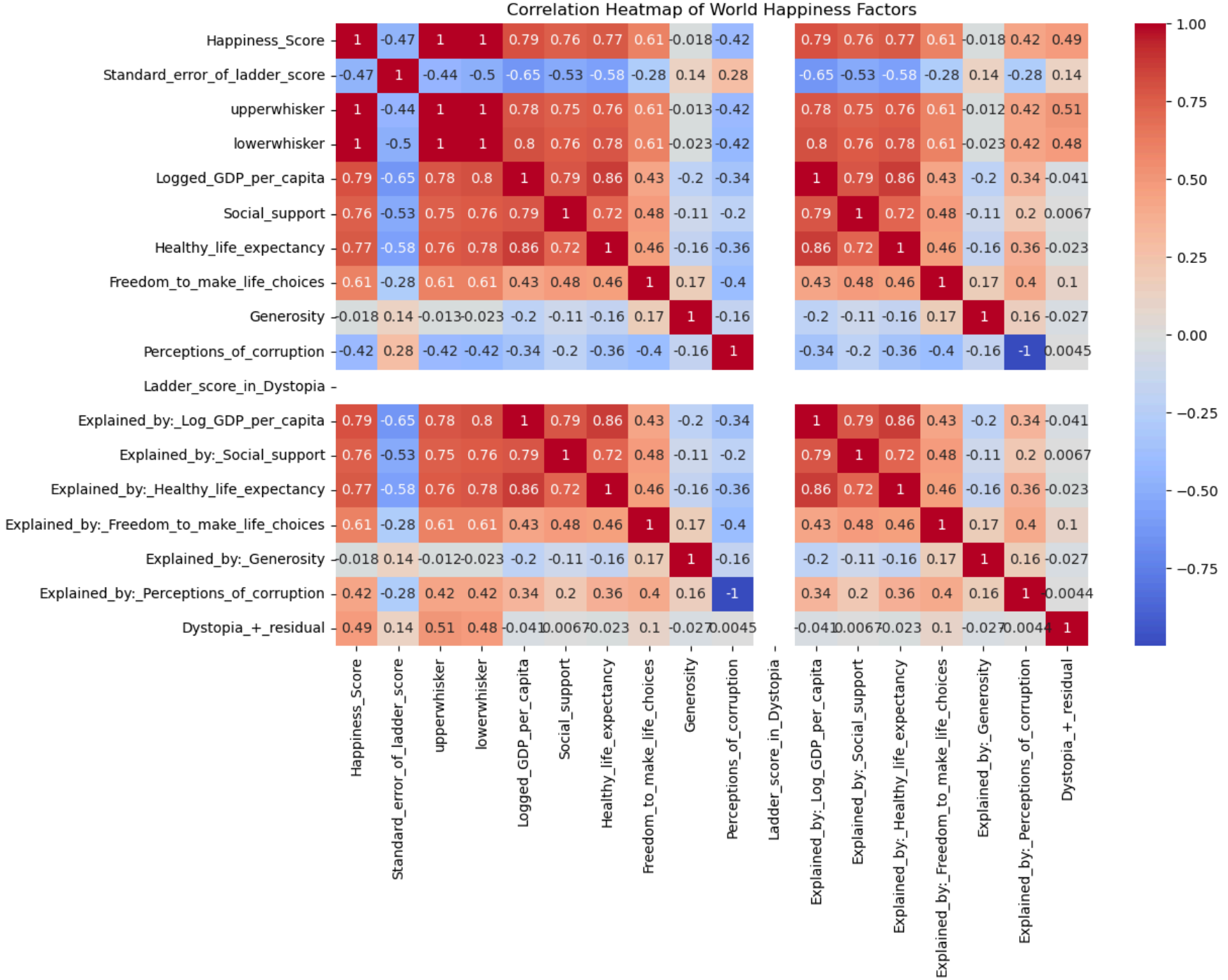
	Country	Region	Happiness_Score	Standard_error_of_ladder_score	upperwhisker	lowerwhisker	Logged_GDP_per_capita	Social_support	Healthy_life_expectancy	Freedom_to_make_life_
0	Finland	Western Europe	7.842	0.032	7.904	7.780	10.775	0.954	72.0	
1	Denmark	Western Europe	7.620	0.035	7.687	7.552	10.933	0.954	72.7	
2	Switzerland	Western Europe	7.571	0.036	7.643	7.500	11.117	0.942	74.4	
3	Iceland	Western Europe	7.554	0.059	7.670	7.438	10.878	0.983	73.0	
4	Netherlands	Western Europe	7.464	0.027	7.518	7.410	10.932	0.942	72.4	

Overview of Visualization Techniques Used:

Heatmap:

```
- Purpose:
    Provides a visual summary of the correlations between all numerical features in the dataset, highlighting relationships such as
how GDP, social support, or life expectancy correlate with happiness scores.

- Strengths:
    Quick identification of strong positive or negative relationships between variables.

- Use Case:
    Best for initial exploratory data analysis to detect trends or potential influences on happiness.
```

In [12]:
```
### Step 2: Heatmap of Correlation

# Shows relationships between numeric variables.

# Select only numeric columns
numeric_df = df.select_dtypes(include=np.number)

plt.figure(figsize=(12,8))
sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm')
plt.title('Correlation Heatmap of World Happiness Factors')
plt.show()
```


Correlation Heatmap of World Happiness Factors
Happiness_Score
Standard_error_of_ladder_score
upperwhisker
lowerwhisker
Logged_GDP_per_capita
Social_support
Healthy_life_expectancy
Freedom_to_make_life_choices
Generosity
Perceptions_of_corruption
Ladder_score_in_Dystopia
Explained_by:_Log_GDP_per_capita
Explained_by:_Social_support
Explained_by:_Healthy_life_expectancy
Explained_by:_Freedom_to_make_life_choices
Explained_by:_Generosity
Explained_by:_Perceptions_of_corruption
Dystopia_+_residual
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75

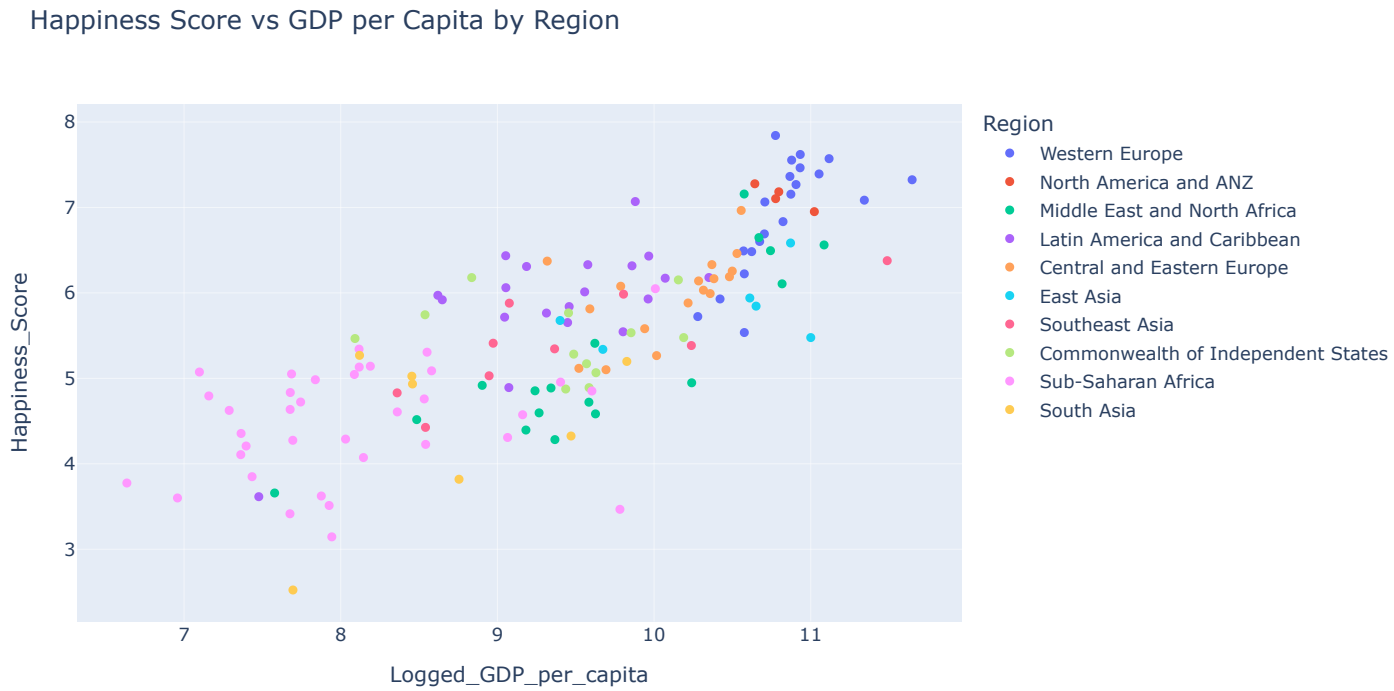
Scatter Plot:

- Purpose:
  Plots the relationship between two numeric variables, here between **Logged GDP per capita** and **Happiness Score**, colored by **Region**.

- Strengths:
  Highlights clusters and regional differences, showing how economic wealth correlates with happiness.

- Use Case:
  Useful for detecting patterns, trends, and outliers in pairwise numeric data.

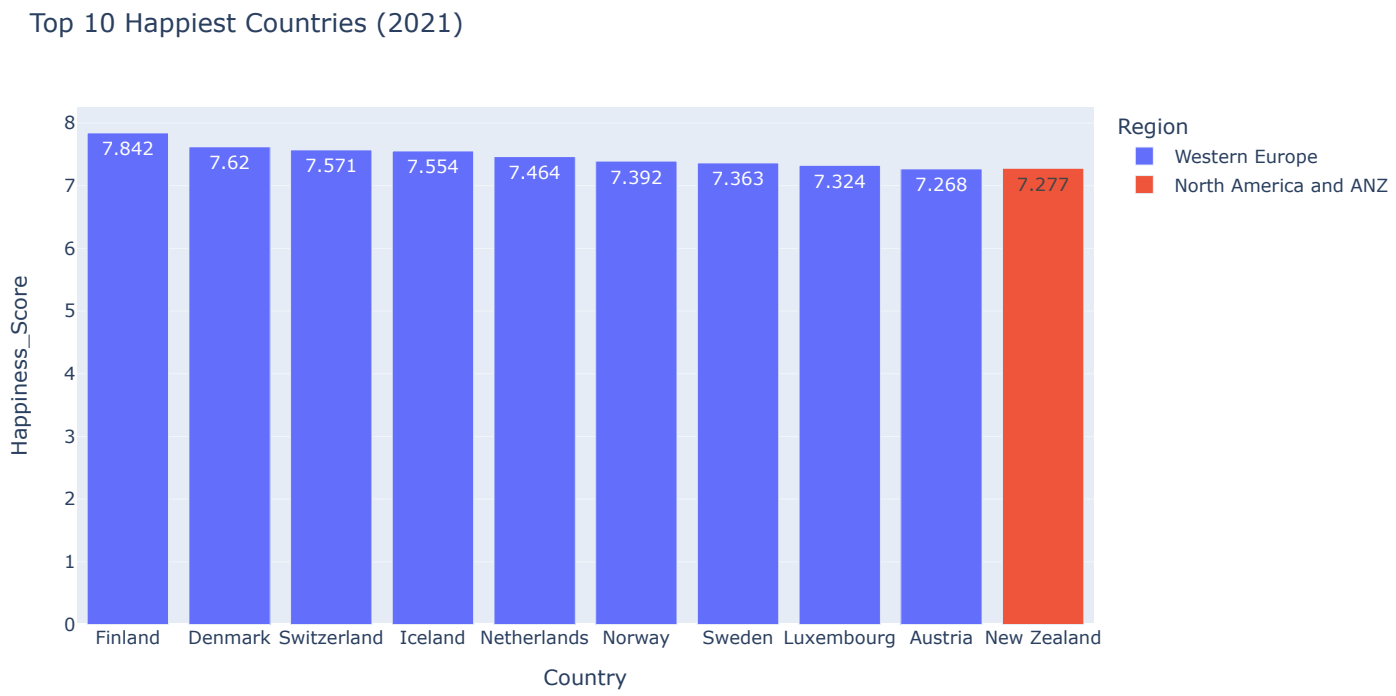
In [13]:
```
### Step 3: Scatter Plot – GDP vs Happiness Score

# Analyzes economic wealth vs happiness.


fig = px.scatter(df, x='Logged_GDP_per_capita', y='Happiness_Score',
                 color='Region', hover_name='Country',
                 title='Happiness Score vs GDP per Capita by Region')
fig.show()
```



Bar Chart:

- ***Purpose:***
  - Visualizes categorical comparisons by showing the **Top 10 Happiest Countries** based on their Happiness Scores.
- ***Strengths:***
  - Clear, easy-to-interpret comparison of categorical values.
- ***Use Case:***
  - Highlights the leading countries and makes ranking comparisons straightforward.

In [14]:
```
### Step 4: Bar Chart – Top 10 Happiest Countries
top10 = df.sort_values('Happiness_Score', ascending=False).head(10)
fig = px.bar(top10, x='Country', y='Happiness_Score', text_auto=True, color='Region',
             title='Top 10 Happiest Countries (2021)')
fig.show()
```



Box Plot:

- ***Purpose:***

  - Shows the distribution, median, quartiles, and outliers of **Happiness Scores** grouped by **Region**.

- ***Strengths:***

  - Reveals spread and potential disparities in happiness scores within different regions.

- ***Use Case:***

  - Best when comparing statistical distributions across categories.

Step 5: Box Plot - Happiness Score Distribution by Region

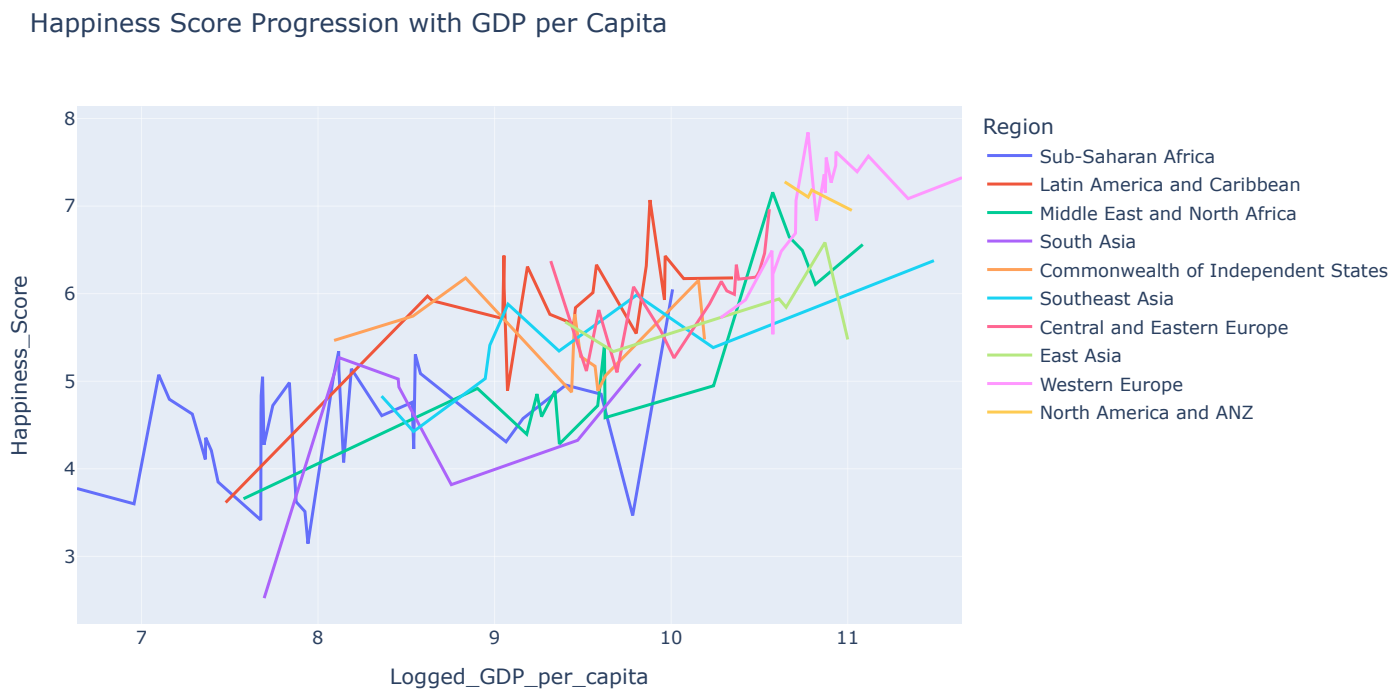
In [15]:
```
fig = px.box(df, x='Region', y='Happiness_Score', color='Region',
             title='Happiness Score Distribution by Region')
fig.update_layout(xaxis={'categoryorder':'total descending'})
fig.show()
```

Line Chart:

- **Purpose:**

  Illustrates how happiness scores vary across countries sorted by GDP per capita, giving a sense of economic disparity's influence.

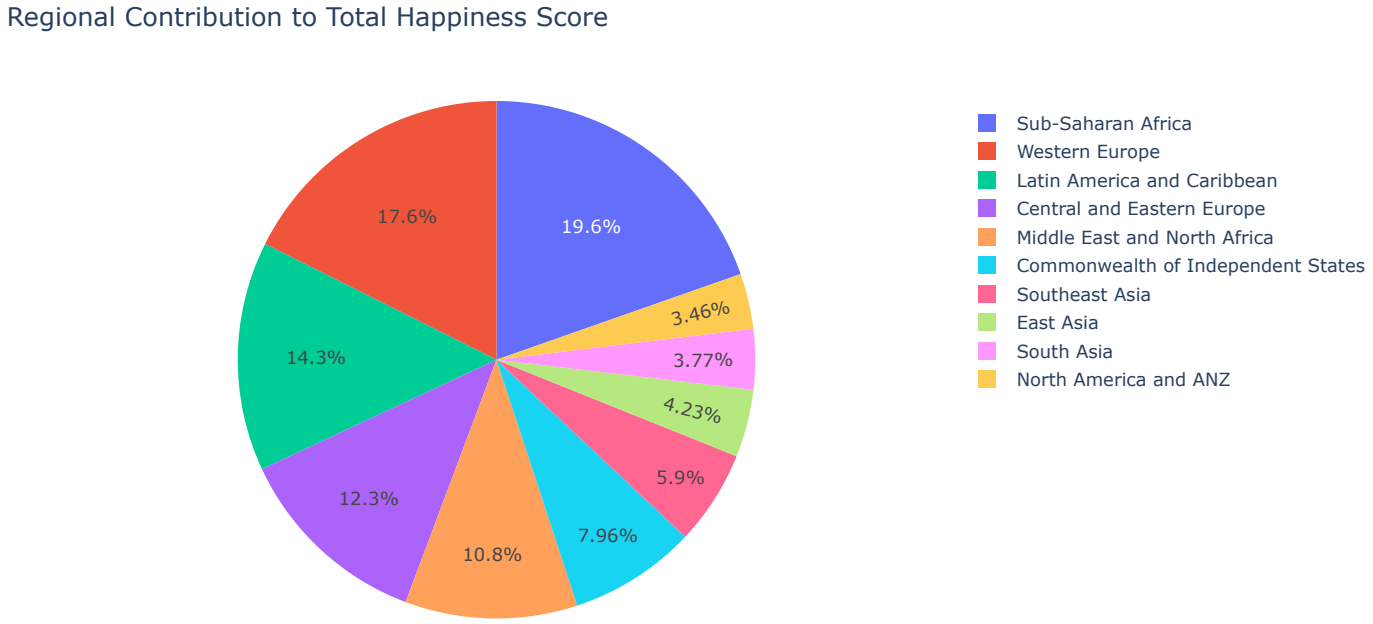
In [16]:
```
sorted_df = df.sort_values('Logged_GDP_per_capita')
fig = px.line(sorted_df,
              x='Logged_GDP_per_capita',
              y='Happiness_Score',
              hover_name='Country',
              color='Region',
              title='Happiness Score Progression with GDP per Capita')
fig.show()
```



Pie Chart:

- **Purpose:** Shows the percentage contribution of different regions to the total happiness score.

In [17]:
```
#df.groupby('Region')['Happiness_Score'].sum().
region_scores = df.groupby('Region')['Happiness_Score'].sum().reset_index()
fig = px.pie(region_scores,
             values='Happiness_Score',
             names='Region',
             title='Regional Contribution to Total Happiness Score')
fig.show()
```



How These Visualizations Complement Each Other:

- The **Heatmap** helps identify which factors (e.g., GDP, social support) influence happiness, guiding deeper visual analyses.
- The **Scatter Plot** offers a granular look at specific relationships discovered in the heatmap, with an emphasis on regional differences.
- The **Bar Chart** ranks the happiest countries, providing a clear, comparative snapshot of top performers.
- The **Box Plot** exposes the distribution and variability of happiness within regions, complementing the rankings.

Visualization Library (25%)

Library Chosen: Plotly

- **Description:**
  - Plotly is an **open-source**, highly interactive visualization library developed by **Plotly Inc**.
  - It supports multiple programming languages including Python, R, and JavaScript.

Is Plotly Open Source?

```
Yes! Plotly's core Python graphing libraries, including **Plotly.py** and **Plotly Express**, are **open source** and available und
er the **MIT license**. This ensures transparency, community contributions, and free use for personal and commercial purposes.
```

Who Created Plotly?

```
Plotly was created by **Plotly, Inc.**, a company based in Montreal, Canada. The company specializes in interactive data visualizat
ion and analytics software and also provides enterprise solutions like **Dash**, a framework for building full-fledged analytical w
eb applications.
```

Why Plotly?

- **Interactivity:**
  - Out-of-the-box interactive charts with tooltips, zooming, panning, and dropdown controls.
- **Ease of Use:**
  - High-level interface, especially `Plotly Express` , simplifies code while producing professional-quality visuals.
- **Declarative Approach:**
  - Users declare chart specifications without having to manage low-level drawing operations.
- **Integration:**
  - Seamless integration with **Jupyter Notebooks**.
- **Customization:**
  - Highly customizable with options to control colors, hover info, legends, and axes.

````
Installation:
Using pip
```bash
    pip install plotly
    pip install ipywidgets
    pip install ipywidgets jupyter nbextension enable --py widgetsnbextension
```
````

Limitations:

- For **very large datasets**, rendering performance might lag.
- Custom subplot arrangements and advanced customizations might require more verbose code compared to libraries like Matplotlib or Seaborn.

Overall Decision:

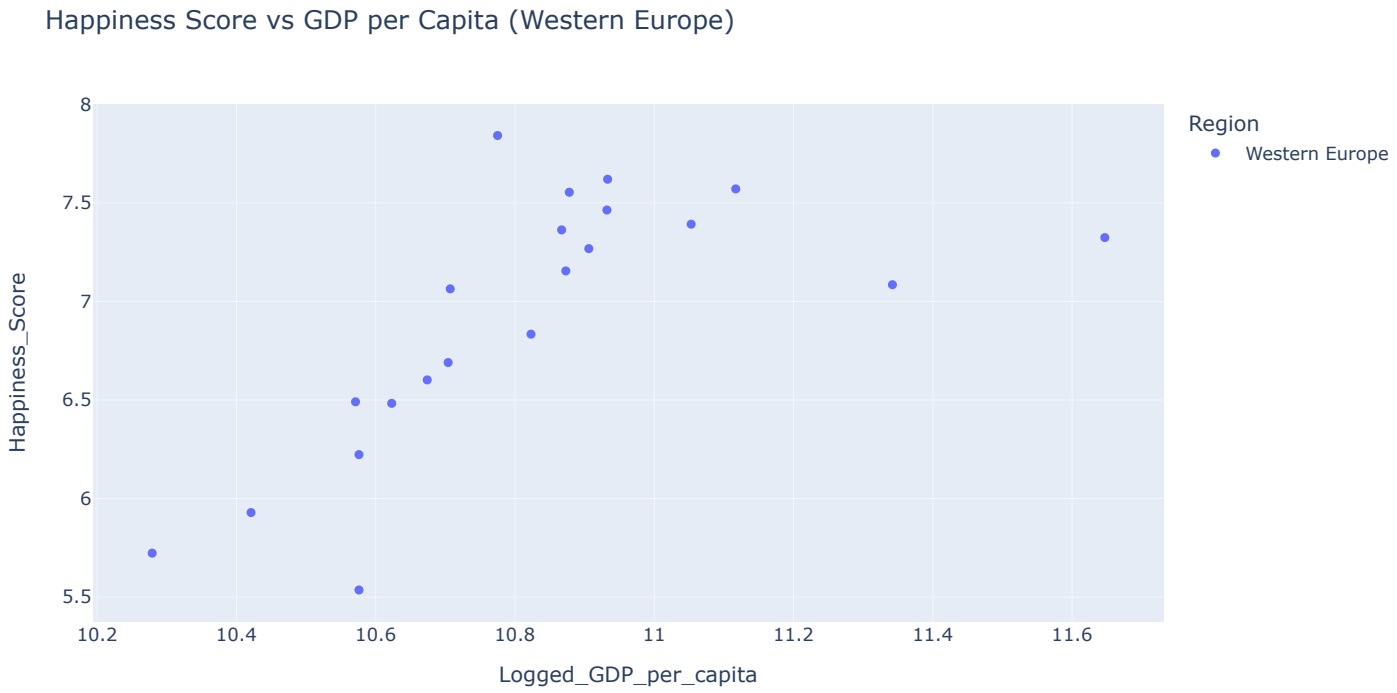
- Plotly offers the perfect balance between interactivity, simplicity, and aesthetics, making it ideal for dashboard-like notebooks.

Interactivity 1 : Filter by Region

In [18]:

```
from ipywidgets import interact

color = ['red','green','blue','grey','purple','yellow', 'orange'] * 3

@interact(region=df['Region'].unique())
def plot_by_region(region):
    filtered = df[df['Region'] == region]
    fig = px.scatter(filtered, x='Logged_GDP_per_capita', y='Happiness_Score',
                     hover_name='Country', color='Region',
                     title=f'Happiness Score vs GDP per Capita ({region})')
    fig.show()

#plot_by_region(region=Dropdown(options=df['Region'].unique(), description='Region'))
```

region: Western Europe



Interactivity 2: Slider to Adjust Minimum GDP Filter

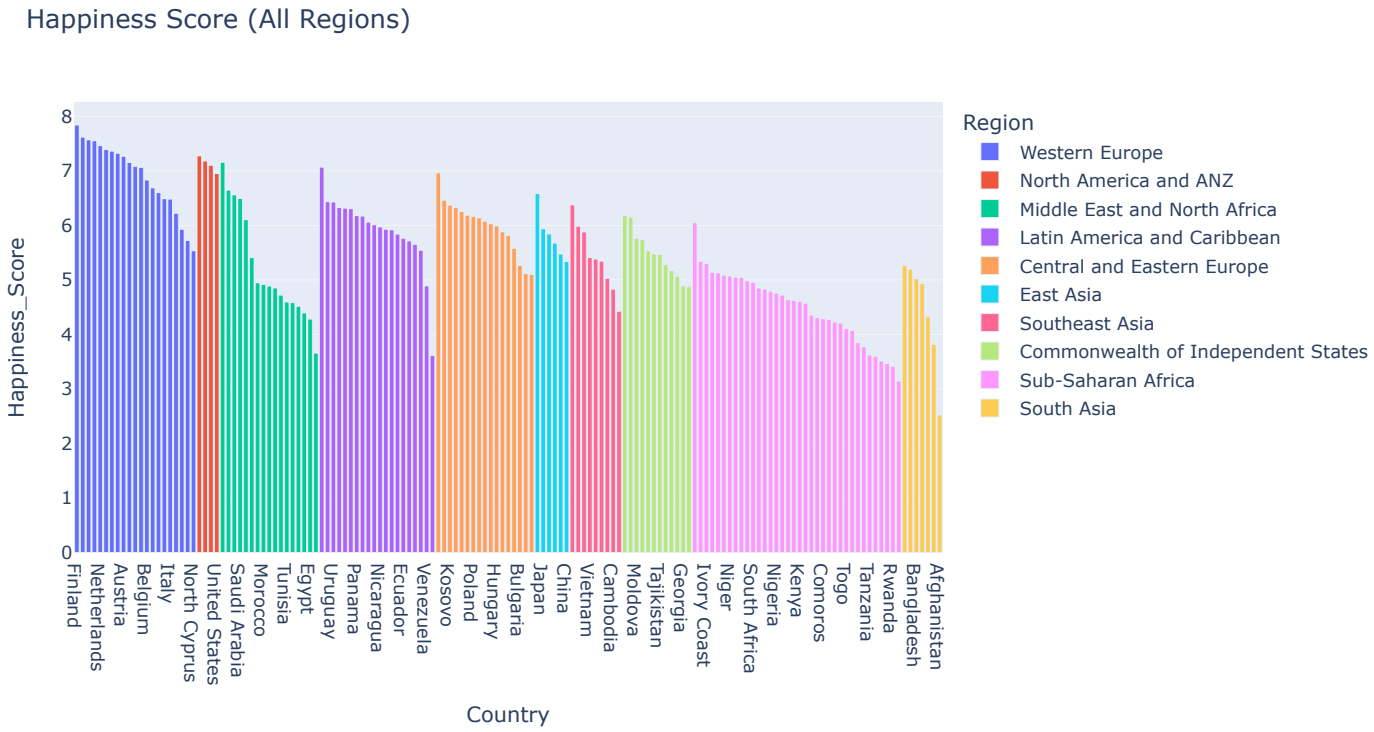
In [19]:
```
from ipywidgets import IntSlider
@interact(min_gdp=IntSlider(min=7, max=12, step=1, value=7, description='Min GDP'))
def plot_filtered_gdp(min_gdp):
    filtered = df[df['Logged_GDP_per_capita'] >= min_gdp]
    fig = px.scatter(filtered, x='Logged_GDP_per_capita', y='Happiness_Score', color='Region', hover_name='Country', title=f'Countries with
    fig.show()
```

Min GDP 7

Interactivity 3: Checkbox to Toggle Display of Western Europe Only

In [20]:
```
from ipywidgets import Checkbox
@interact(western_only=Checkbox(description='Show Western Europe Only'))
def plot_toggle(western_only):
    if western_only:
        filtered = df[df['Region'] == 'Western Europe']
        title = 'Happiness Score (Western Europe)'
    else:
        filtered = df
        title = 'Happiness Score (All Regions)'
    fig = px.bar(filtered, x='Country', y='Happiness_Score', color='Region', title=title)
    fig.show()
```

☐ Show Western Europe Only



Story and Insights:

Our analysis starts with the **Heatmap**, revealing that economic indicators like **GDP per capita**, **social support**, and **life expectancy** are strongly correlated with happiness. However, we also note that factors such as **perception of corruption** and **generosity** show weaker correlations.

Moving to the **Scatter Plot**, we can clearly observe that countries in regions like **Western Europe** and **North America** cluster towards higher GDP and happiness scores. Conversely, regions such as **Sub-Saharan Africa** show lower scores, highlighting economic disparities.

The **Bar Chart** emphasizes which countries (like **Finland**, **Denmark**, and **Switzerland**) consistently top happiness rankings. This naturally transitions to the **Box Plot**, which exposes the spread of happiness within different regions — with regions like **Western Europe** showing high median happiness and low variability.

The **Line Chart** further uncovers a smooth upward trend, reinforcing that wealthier countries generally report higher happiness.

Lastly, the **Pie Chart** gives a macro perspective, showing how total happiness is distributed regionally, offering insights into where global happiness is concentrated.

To make exploration dynamic, **dropdowns**, **sliders**, and **checkboxes** are introduced — allowing users to filter and interactively customize views based on region, GDP thresholds, and specific groups.

Conclusion:

By combining heatmaps, scatter plots, bar charts, box plots, line charts, and pie charts, this interactive dashboard provides a holistic and engaging view of global happiness trends. Users can explore how wealth, social factors, and regional differences influence happiness worldwide. The interactivity allows deep exploration by region, making it a powerful tool for policymakers, researchers, and global citizens.

Troubleshooting & Common Errors

1. `ParserError: Error tokenizing data. C error: Expected 1 fields in line 42, saw 39`

- Cause: Gitlab file not found.
- Fix: add ?raw=true at the end.

2. `ValueError: could not convert string to float`

- Cause: Non-numeric column included in correlation matrix.
- Fix: Use `df.select_dtypes(include=np.number)` before `.corr()`.

3. `Error displaying widget`

- Cause: Missing ipywidgets extension in Jupyter.
- Fix:
  - Run `pip install ipywidgets`
  - Then:
    ```
    jupyter nbextension enable --py widgetsnbextension
    ```

4. `NameError: name 'IntSlider' is not defined`

- Fix: Ensure `IntSlider` is imported:
  ```
  from ipywidgets import IntSlider
  ```

In [ ]: