**Step-by-Step Breakdown of What I Did**

**1️⃣ Installed Necessary Library**

Since I wanted to create an **interactive world map**, I used a library called plotly.express. It’s very beginner-friendly and perfect for maps and animations.

python

!pip install plotly

**Why Plotly?**  
It lets you build beautiful animated maps in just a few lines. Much easier than doing the same thing in matplotlib or seaborn.

**2️⃣ Imported Python Libraries**

import pandas as pd

import plotly.express as px

import glob

import re

* pandas – to read and clean the data
* plotly.express – to build the map
* glob – to automatically find all the .csv files
* re – to filter files by name (e.g. only 2015.csv, 2016.csv, etc.)

**3️⃣ Loaded Multiple CSV Files Automatically**

I had five CSV files: 2015.csv to 2019.csv. Each file had a list of countries with happiness scores for that year. But — different years had slightly different column names (e.g., Happiness.Score, Happiness Score, or Score).

So I wrote code to:

* Loop through each CSV
* Standardize the column names
* Extract just the necessary columns: Country, Score, and Year

files = [f for f in glob.glob("\*.csv") if re.match(r"^\d{4}\.csv$", f)]

all\_data = []

for file in files:

year = int(file.split('.')[0])

df = pd.read\_csv(file)

# Standardize column names

df.columns = [col.strip().replace(' ', '\_') for col in df.columns]

# Find the correct score column

for possible\_col in ['Happiness\_Score', 'Happiness.Score', 'Score']:

if possible\_col in df.columns:

score\_col = possible\_col

break

# Get the correct country column

if 'Country' in df.columns:

country\_col = 'Country'

elif 'Country\_or\_region' in df.columns:

country\_col = 'Country\_or\_region'

else:

continue

temp\_df = df[[country\_col, score\_col]].copy()

temp\_df.columns = ['Country', 'Score']

temp\_df['Year'] = year

all\_data.append(temp\_df)

# Combine all years into one DataFrame

happiness\_df = pd.concat(all\_data, ignore\_index=True)

**4️⃣ Verified the Data Was Loaded Correctly**

I checked how many countries had happiness scores per year, using:

happiness\_df.groupby("Year")["Score"].count()

This showed:

Year

2015 158

2016 157

2017 155

2018 156

2019 156

✅ That confirmed that **data for all 5 years was successfully loaded**.

**5️⃣ Cleaned and Sorted the Data**

To make sure there were no missing values or issues:

query\_df = happiness\_df[['Country', 'Year', 'Score']].dropna()

query\_df = query\_df.sort\_values(['Year', 'Country'])

**6️⃣ Plotted the World Happiness Index Map**

Finally, I used plotly.express.choropleth() to create the animated color map of the world, with one frame per year:

fig = px.choropleth(

query\_df,

locations='Country',

locationmode='country names',

color='Score',

hover\_name='Country',

animation\_frame='Year',

color\_continuous\_scale='Viridis',

title='World Happiness Index (2015–2019)',

height=600

)

fig.show()

This created a beautiful animated map that shows how the happiness score changes over time for each country from 2015 to 2019.

**What I Learned**

* Plotly makes interactive data visualizations simple, even for beginners.
* Real-world datasets often have inconsistencies (column names, missing values).
* Grouping and checking your data using groupby() and dropna() helps avoid issues later on.
* Adding the animation\_frame argument in Plotly makes visual storytelling much easier!