Submission

Put the ipynb file and html file in the github branch you created in the last assignment and submit the link to the commit in brightspace

In [1]:

```
from plotly.offline import init_notebook_mode
import plotly.io as pio
import plotly.express as px

init_notebook_mode(connected=True)
pio.renderers.default = "plotly_mimetype+notebook"
```

In [2]:

```
#load data
df = px.data.gapminder()
df.head()
```

Out[2]:

	country	continent	year	lifeExp	pop	gdpPercap	iso_alpha	iso_num
0	Afghanistan	Asia	1952	28.801	8425333	779.445314	AFG	4
1	Afghanistan	Asia	1957	30.332	9240934	820.853030	AFG	4
2	Afghanistan	Asia	1962	31.997	10267083	853.100710	AFG	4
3	Afghanistan	Asia	1967	34.020	11537966	836.197138	AFG	4
4	Afghanistan	Asia	1972	36.088	13079460	739.981106	AFG	4

Question 1:

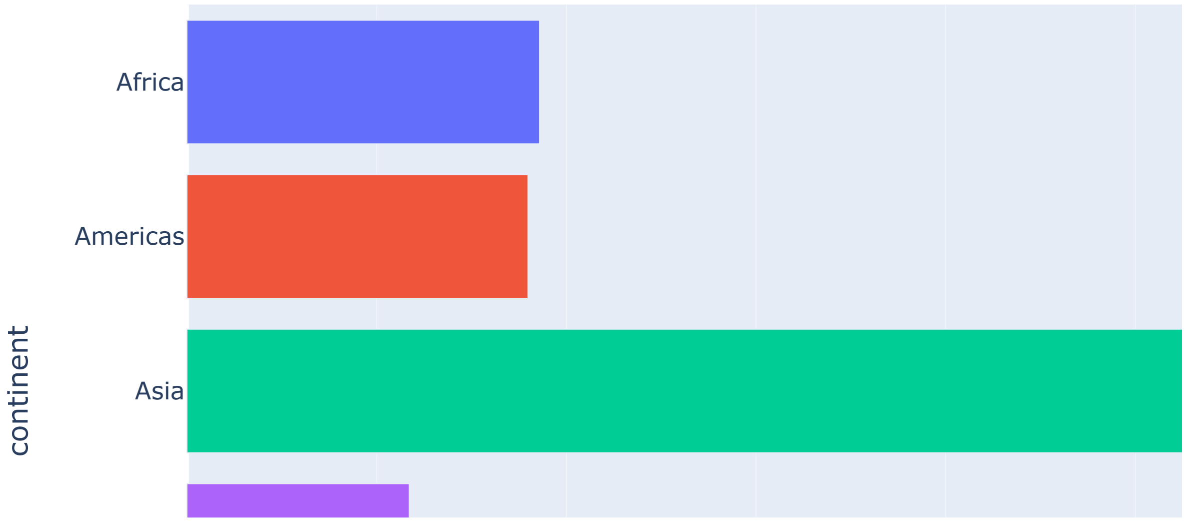
Recreate the barplot below that shows the population of different continents for the year 2007.

Hints:

- Extract the 2007 year data from the dataframe. You have to process the data accordingly
- use plotly bar (https://plotly.com/python-api-reference/generated/plotly.express.bar)
- Add different colors for different continents
- Sort the order of the continent for the visualisation. Use axis layout setting (https://plotly.com/python/reference/layout/xaxis/)
- Add text to each bar that represents the population

In [25]:

```
# YOUR CODE HERE
df_2007 = df.query('year == 2007')
df_2007_new = df_2007.groupby('continent').sum()
fig = px.bar(df_2007_new, x = "pop", orientation = "h", color = df_2007_new.index)
fig.show()
```



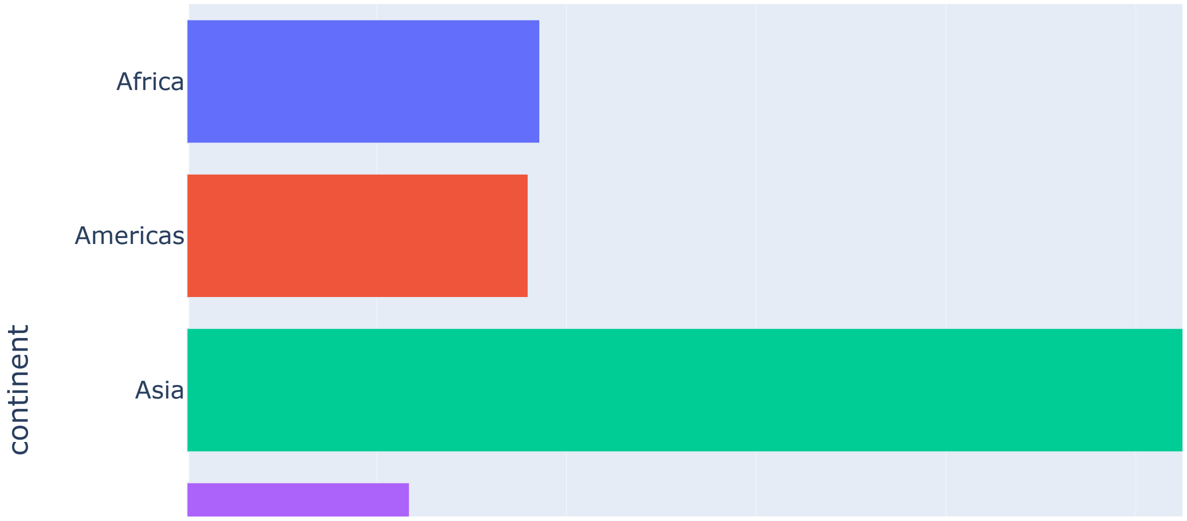
Question 2:

Sort the order of the continent for the visualisation

Hint: Use [axis layout setting (https://plotly.com/python/reference/layout/xaxis/)](https://plotly.com/python/reference/layout/xaxis/)

In [24]:

```
# YOUR CODE HERE
df_2007 = df.query('year == 2007')
df_2007_new = df_2007.groupby('continent').sum()
fig = px.bar(df_2007_new, x = "pop", orientation = "h", color = df_2007_new.index)
fig.update_yaxes(categoryorder = 'category descending')
fig.show()
```

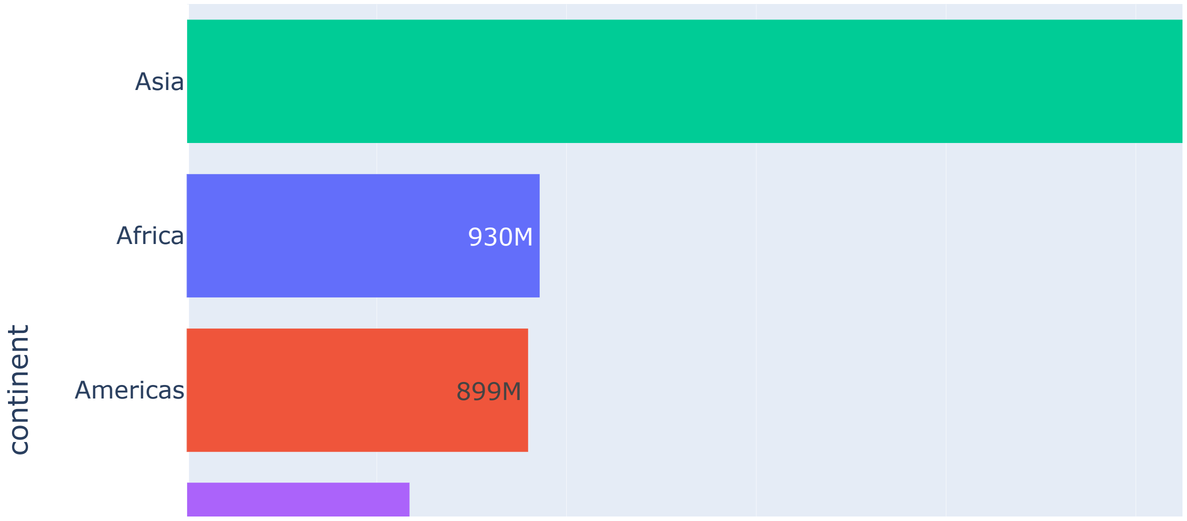


Question 3:

Add text to each bar that represents the population

In [10]:

```
# YOUR CODE HERE
df_2007 = df.query('year == 2007')
df_2007_new = df_2007.groupby('continent').sum()
fig = px.bar(df_2007_new, x = "pop", orientation = "h", color = df_2007_new.index, text_aut
fig.update_yaxes(categoryorder = 'total ascending')
fig.update_layout(barmode='stack')
fig.show()
```

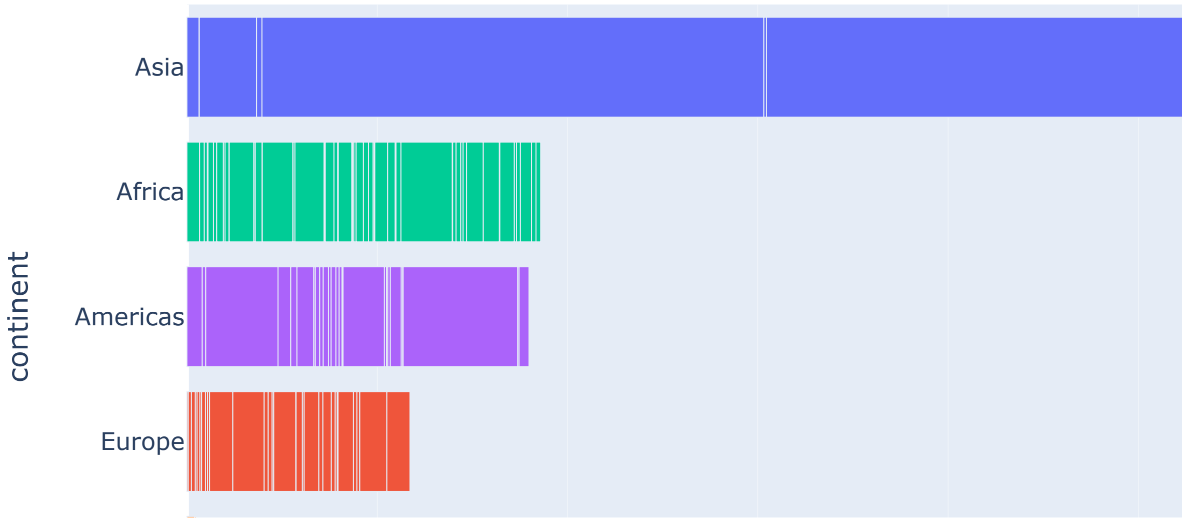


Question 4:

Thus far we looked at data from one year (2007). Lets create an animation to see the population growth of the continents through the years

In [17]:

```
# YOUR CODE HERE
df = px.data.gapminder()
fig = px.bar(df, x = 'pop', y = 'continent', animation_frame = 'year', color = 'continent',
fig.update_yaxes(categoryorder = 'total ascending')
fig.show()
```

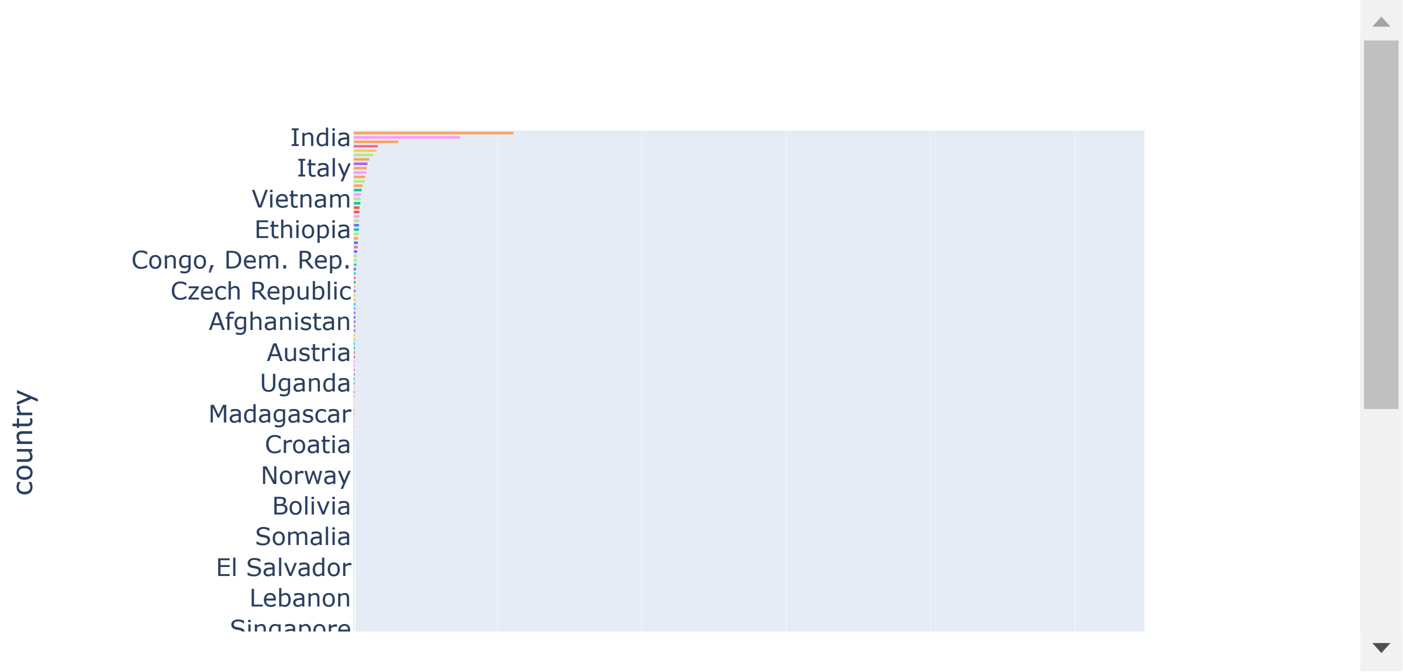


Question 5:

Instead of the continents, lets look at individual countries. Create an animation that shows the population growth of the countries through the years

In [22]:

```
# YOUR CODE HERE
fig = px.bar(df,
             y='country',
             x='pop',
             color = 'country',
             animation_frame = 'year',
             animation_group = 'country',
             range_x = [0,4000000000])
fig.update_yaxes(categoryorder = 'sum ascending')
fig.show()
```

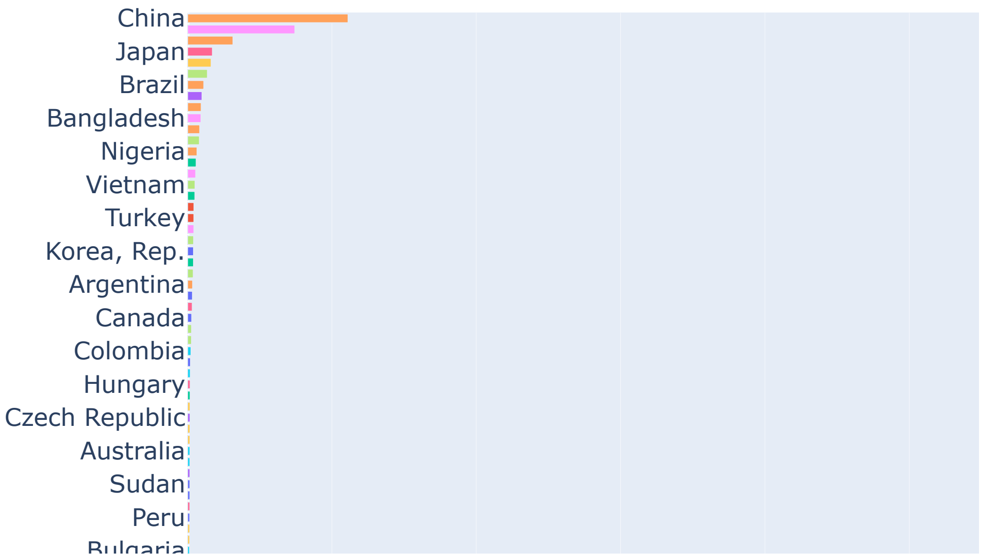


Question 6:

Clean up the country animation. Set the height size of the figure to 1000 to have a better view of the animation

In [23]:

```
# YOUR CODE HERE
fig = px.bar(df,
             y='country',
             x='pop',
             color = 'country',
             animation_frame = 'year',
             animation_group = 'country',
             range_x = [0,4000000000],
            height = 1000)
fig.update_yaxes(categoryorder = 'sum ascending')
fig.show()
```



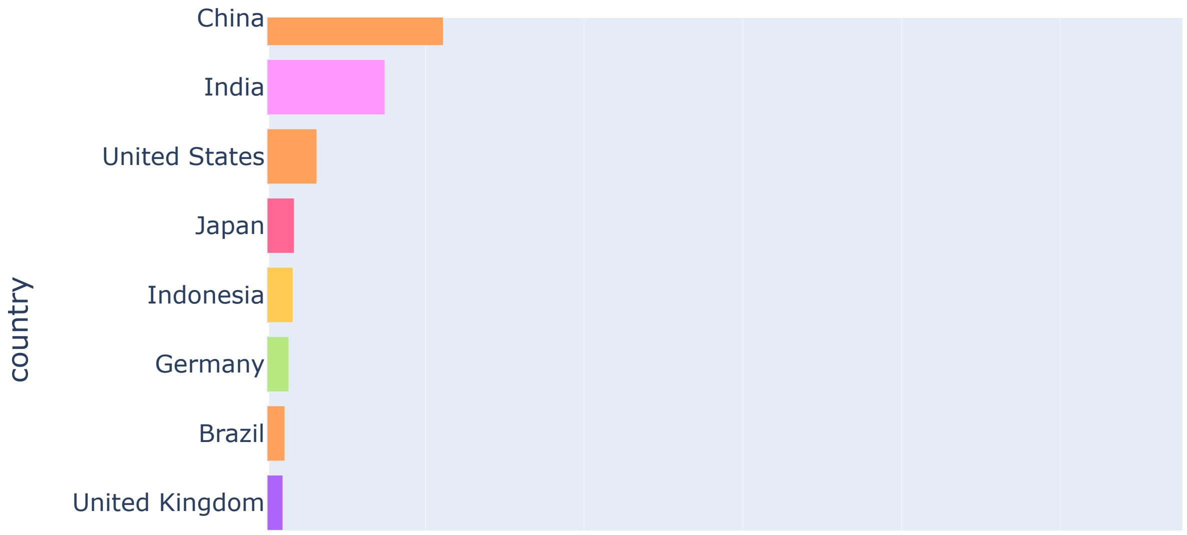
Question 7:

Show only the top 10 countries in the animation

Hint: Use the axis limit to set this.

In [21]:

```
# YOUR CODE HERE
fig = px.bar(df,
             y='country',
             x='pop',
             color = 'country',
             animation_frame = 'year',
             animation_group = 'country',
             range_x = [0,4000000000],
             range_y = [132,141])
fig.update_yaxes(categoryorder = 'sum ascending')
fig.show()
```



In [ ]:

```

```