COVID-19 VACCINES ANALYSIS

dac-phase5

November 1, 2023

Project Outline:

1. Objective:

The objective of this project is to analyze COVID-19 vaccination data with a focus on the USA. The analysis aims to provide insights into the vaccination progress, including total vaccinations, daily vaccinations, and monthly trends in 2021. The project also involves data visualization using Python and possibly IBM Cognos.

2. Design Thinking Process:

The design thinking process involves several key steps:

a. Empathize:
Understanding the need for COVID-19 vaccination data analysis, considering the context of a pandemic, and the importance of tracking vaccination progress.

b. Define:
Defining the specific analysis objectives, which include tracking total vaccinations, daily vaccinations, and monthly trends in the USA for 2021.

c. Ideate:
Generating ideas for data analysis and visualization methods, such as line plots, pie charts, and bar graphs, as demonstrated in the Python code.

d. Prototype:
Developing the Python code for data analysis and visualization and exploring the potential use of IBM Cognos for more advanced visualization.

e. Test:
Testing the code and visualizations to ensure they accurately represent the vaccination data.

3. Development Phases:

The development of this project can be broken down into several phases:

a. Data Collection:
- Importing necessary Python libraries for data analysis, such as NumPy, Pandas, Seaborn, and Matplotlib.
- Reading the COVID-19 vaccination dataset from a CSV file using Pandas.
- Cleaning and preparing the data, including handling missing values.

b. Data Analysis:
- Splitting the data by year to focus on 2020, 2021, and 2022.
- Filtering data for the USA to create a separate dataset for analysis.
- Calculating and visualizing total vaccinations and daily vaccinations trends in the USA.

c.Data Visualization:
- Creating line plots to visualize trends in total vaccinations and daily vaccinations in the USA.
- Calculating and visualizing monthly total vaccinations in 2021 using a pie chart and a bar graph.

d. Integration with IBM Cognos (Optional):
- Considering the use of IBM Cognos for more advanced and interactive visualizations.

- Integrating Python code with IBM Cognos if necessary to provide a comprehensive dashboard for users.

4. Improving User Experience:

The insights gained from this analysis can help website owners improve the user experience in the following ways:

- Providing clear and up-to-date information on COVID-19 vaccination progress.
- Offering interactive visualizations to engage users and help them understand the data better.
- Allowing users to explore trends and patterns in vaccination data, which can help them make informed decisions about vaccination.
- Enabling website owners to tailor their content and resources to address user needs and concerns related to COVID-19 vaccination.
- Enhancing transparency and trust by providing reliable and easily accessible information.

[1]:
```
# importing the required python libraries
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import plotly.express as px
%matplotlib inline
```

[4]:
```
#using the read_cv method to read the dataset
df=pd.read_csv("D:\ss .nm4\country_vaccinations.csv")
df.head()
```

[4]:

	country	iso_code	date	total_vaccinations	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw	daily_vaccinations
0	Afghanistan	AFG	2021-02-22	0.0	0.0	NaN	NaN	NaN
1	Afghanistan	AFG	2021-02-23	NaN	NaN	NaN	NaN	1367.0
2	Afghanistan	AFG	2021-02-24	NaN	NaN	NaN	NaN	1367.0
3	Afghanistan	AFG	2021-02-25	NaN	NaN	NaN	NaN	1367.0
4	Afghanistan	AFG	2021-02-26	NaN	NaN	NaN	NaN	

```
4                   NaN                   NaN            1367.0
   total_vaccinations_per_hundred people_vaccinated_per_hundred \
0                             0.0                           0.0
2                             NaN                           NaN
3                             NaN                           NaN
4                             NaN                           NaN

   people_fully_vaccinated_per_hundred daily_vaccinations_per_million \
0                                  NaN                            NaN
2                                  NaN                           34.0
3                                  NaN                           34.0
4                                  NaN                           34.0
                                              vaccines \
0  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
1  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
2  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
3  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
4  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…

                source_name         source_website
0  World Health Organization https://covid19.who.int/
1  World Health Organization https://covid19.who.int/
2  World Health Organization https://covid19.who.int/
3  World Health Organization https://covid19.who.int/
4  World Health Organization https://covid19.who.int/
```

[15]:
```
#spliting the years from the date

# Assuming that your date column is named 'date', create a new column for the
 ↪year
df['year'] = pd.to_datetime(df['date']).dt.year

# Display the DataFrame with the 'year' column
df.head()
```

[15]:
```
       country iso_code        date  total_vaccinations  people_vaccinated  \
0 Afghanistan     AFG 2021-02-22                 0.0                 0.0
1 Afghanistan     AFG 2021-02-23                 NaN                 NaN
2 Afghanistan     AFG 2021-02-24                 NaN                 NaN
3 Afghanistan     AFG 2021-02-25                 NaN                 NaN
4 Afghanistan     AFG 2021-02-26                 NaN                 NaN
 people_fully_vaccinated daily_vaccinations_raw daily_vaccinations \
0                     NaN                   NaN                NaN
1                     NaN                   NaN             1367.0
2                     NaN                   NaN             1367.0
3                     NaN                   NaN             1367.0
4                     NaN                   NaN             1367.0
```

```
   total_vaccinations_per_hundred people_vaccinated_per_hundred \
0     0.0   0.0 1 NaN   NaN
2                          NaN    NaN
3                          NaN    NaN
4                          NaN    NaN
                           people_fully_vaccinated_per_hundred
                           daily_vaccinations_per_million \
0                              NaN                          NaN
1                              NaN                         34.0
2                              NaN                         34.0
3                              NaN                         34.0
4                              NaN                         34.0

                                        vaccines \
0  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
1  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
2  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
3  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…
4  Johnson&Johnson, Oxford/AstraZeneca, Pfizer/Bi…

              source_name         source_website year
0  World Health Organization https://covid19.who.int/
   2021
1  World Health Organization https://covid19.who.int/
   2021
2  World Health Organization https://covid19.who.int/
   2021
3  World Health Organization https://covid19.who.int/
   2021
4  World Health Organization https://covid19.who.int/
   2021
```

[17]:
```
pr=df[['country','iso_code','total_vaccinations','total_vaccinations_per_hundred','year']]
pr
```

[ ]:

[17]:
```
        country iso_code total_vaccinations \
0     Afghanistan    AFG                0.0
1     Afghanistan    AFG                NaN
2     Afghanistan    AFG                NaN
3     Afghanistan    AFG                NaN
4     Afghanistan    AFG                NaN
```

```
…          …      …              …
86507   Zimbabwe     ZWE         8691642.0
86508   Zimbabwe     ZWE         8791728.0
86509   Zimbabwe     ZWE         8845039.0
86510   Zimbabwe     ZWE         8934360.0
86511   Zimbabwe     ZWE         9039729.0

      total_vaccinations_per_hundred year
0                               0.00 2021
1                                NaN 2021
2                                NaN 2021
3                                NaN 2021
4                                NaN 2021
…                               … …
86507                           57.59 2022
86508                           58.25 2022
86509                           58.61 2022
86510                           59.20 2022
86511                           59.90 2022

86512                           rows x 5 columns]
```

[19]:
```
#isnull() will display whether there is any null data values of the dataset

pr.isnull()
```

[19]:
```
  country iso_code total_vaccinations total_vaccinations_per_hundred \
0       False    False            False                         False
1       False    False             True                          True
2       False    False             True                          True
3       False    False             True                          True
4       False    False             True                          True
…       …      …                 …                             …
86507   False    False            False                         False
86508   False    False            False                         False
86509   False    False            False                         False
86510   False    False            False                         False
86511   False    False            False                         False
```

```
       year
0     False
1     False
2     False
3     False
4     False
…     …
86507 False
86508 False
86509 False
86510 False
86511 False

[86512 rows x 5 columns]
```

[22]:
```
df.info()
```

```
<class
'pandas.core.frame.DataFrame'>
RangeIndex: 86512 entries, 0 to
86511 Data columns (total 16
columns):
#   Column                                Non-Null Count
                                                  Dtype
--- ------                            -------------- ----
                                                    -
0   country                                86512 non-null
                                                   object
 1  iso_code                           86512      non-null
                                       object
 2  date                               86512      non-null
                                       object
 3 total_vaccinations                  43607      non-null
                                       float64
 4 people_vaccinated                   41294      non-null
                                       float64
 5 people_fully_vaccinated             38802      non-null
                                       float64
 6 daily_vaccinations_raw              35362      non-null
                                       float64
 7 daily_vaccinations                  86213      non-null
                                       float64
 8 total_vaccinations_per_hundred      43607      non-null
                                       float64
 9 people_vaccinated_per_hundred       41294      non-null
                                       float64
10 people_fully_vaccinated_per_hundred 38802 non-null float64
```

```
 11 daily_vaccinations_per_million   86213       non-null
                                     float64
 12 vaccines                         86512       non-null
                                     object
 13 source_name                      86512       non-null
                                     object
 14 source_website                   86512       non-null
                                     object
 15 year                             86512 non-null
dtypes: float64(9), int32(1),        int32
object(6)
memory usage: 10.2+ MB
```

[25]:
```
#the date given in the dataset is converted into specified format
df['date']=pd.to_datetime(df['date'],format='%Y-%m-%d')
```

[29]:
```
#Now we are performing the covid-19 vaccine analysis on the
country USA df_USA=df[df["iso_code"]=='USA'].copy()
df_USA.head(3)
```

[29]:

	country	iso_code	date	total_vaccinations
82360	United States	USA	2020-12-13	30288.0
82361	United States	USA	2020-12-14	34867.0
82362	United States	USA	2020-12-15	84638.0

	people_vaccinated	people_fully_vaccinated	daily_vaccinations_raw
82360	25125.0	5897.0	NaN
82361	29543.0	6017.0	4579.0
82362	76984.0	6281.0	49771.0

	daily_vaccinations	total_vaccinations_per_hundred
82360	NaN	0.01
82361	4579.0	0.01
82362	27175.0	0.03

	people_vaccinated_per_hundred	people_fully_vaccinated_per_hundred
82360	0.01	0.0
82361	0.01	0.0
82362	0.02	0.0

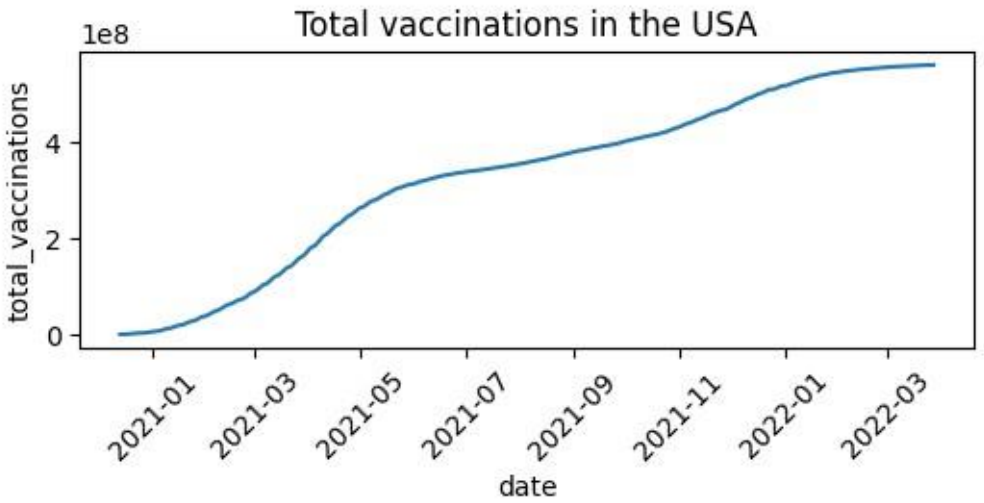
	daily_vaccinations_per_million
82360	NaN
82361	14.0
82362	82.0

	vaccines
82360	Johnson&Johnson, Moderna, Pfizer/BioNTech
82361	Johnson&Johnson, Moderna, Pfizer/BioNTech
82362	Johnson&Johnson, Moderna, Pfizer/BioNTech

```
                                    source_name \
82360 Centers for Disease Control and Prevention
82361 Centers for Disease Control and Prevention
82362 Centers for Disease Control and Prevention

                                    source_website year
82360 https://data.cdc.gov/Vaccinations/COVID-19-Vac… 2020
82361 https://data.cdc.gov/Vaccinations/COVID-19-Vac… 2020
82362 https://data.cdc.gov/Vaccinations/COVID-19-Vac… 2020
```

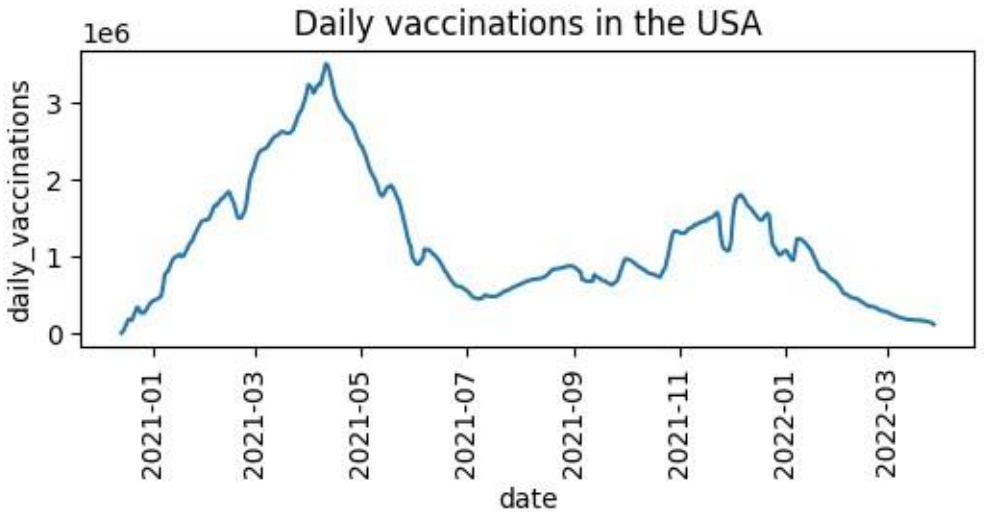
[30]:
```
df_USA.drop(df_USA.index[df_USA['total_vaccinations']==0],inplace=True)
```

[33]:
```
# Virtualization analysis of Total Vaccinations in the USA
plt.figure(figsize=(6,2))
sns.lineplot(data=df_USA,x="date",y="total_vaccinations")
plt.title("Total vaccinations in the USA ")
plt.xticks(rotation=45)
plt.show()
```



[35]:
```
# Virtualization analysis of Total Vaccinations in the USA
plt.figure(figsize=(6,2))
```

```
sns.lineplot(data=df_USA,x="date",y="daily_vaccinations")
plt.title("Daily vaccinations in the USA ")
plt.xticks(rotation=90)
plt.show()
```



[37]: *# number of total vaccinations for the years 2020, 2021, and 2022*

```
# Assuming that your date column is named 'date', create a new
column for the␣
 ↪year df['year'] =
pd.to_datetime(df['date']).dt.year

# Filter the DataFrame to select data for the years 2020, 2021, and
2022 selected_years = [2020, 2021, 2022]
filtered_df = df[df['year'].isin(selected_years)]

# Calculate the total vaccinations for each year
total_vaccinations_by_year =
filtered_df.groupby('year')['total_vaccinations']. ↪sum()

# Display the total vaccinations for each year
print(total_vaccinations_by_year)
```

```
year
2020    5.406426e+07
2021    1.217585e+12
2022    7.852151e+11
```

```
Name: total_vaccinations, dtype: float64
```

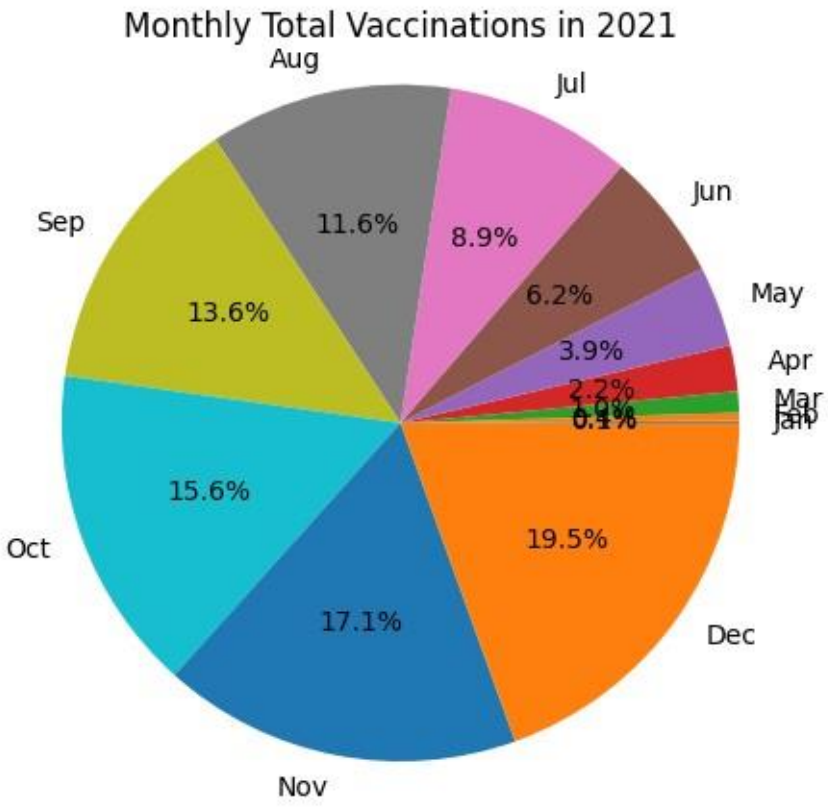
[40]:
```
#montly number of total vaccination from jan to dec in 2021

# Assuming that your date column is named 'date', create a new column for the␣
 ↪year and month df['year'] =
pd.to_datetime(df['date']).dt.year
df['month'] =
pd.to_datetime(df['date']).dt.month

# Filter the DataFrame to select data for the year 2021
filtered_df = df[df['year'] == 2021]

# Group and sum total vaccinations by month
monthly_total_vaccinations =
filtered_df.groupby('month')['total_vaccinations']. ↪sum()

# Display the monthly total vaccinations for 2021
monthly_total_vaccinations
```

[40]:
```
month
1
1.368363e+09 2
4.511692e+09 3
1.237050e+10 4
2.663815e+10 5
4.693966e+10 6
7.500972e+10 7
1.084767e+11 8
1.410912e+11 9
1.652023e+11
10   1.900397e+11
11   2.084644e+11
12   2.374725e+11
Name: total_vaccinations, dtype: float64
```

[43]:
```
# pie chart montly number of total vaccination from jan to dec in 2021

# Assuming that your date column is named 'date', create a new column for the␣
 ↪year and month df['year'] =
pd.to_datetime(df['date']).dt.year
df['month'] =
pd.to_datetime(df['date']).dt.month
```

```
# Filter the DataFrame to select data for the year 2021
filtered_df = df[df['year'] == 2021]

# Group and sum total vaccinations by month
monthly_total_vaccinations =
filtered_df.groupby('month')['total_vaccinations']. ↪sum()

# Define the labels for the pie chart
```

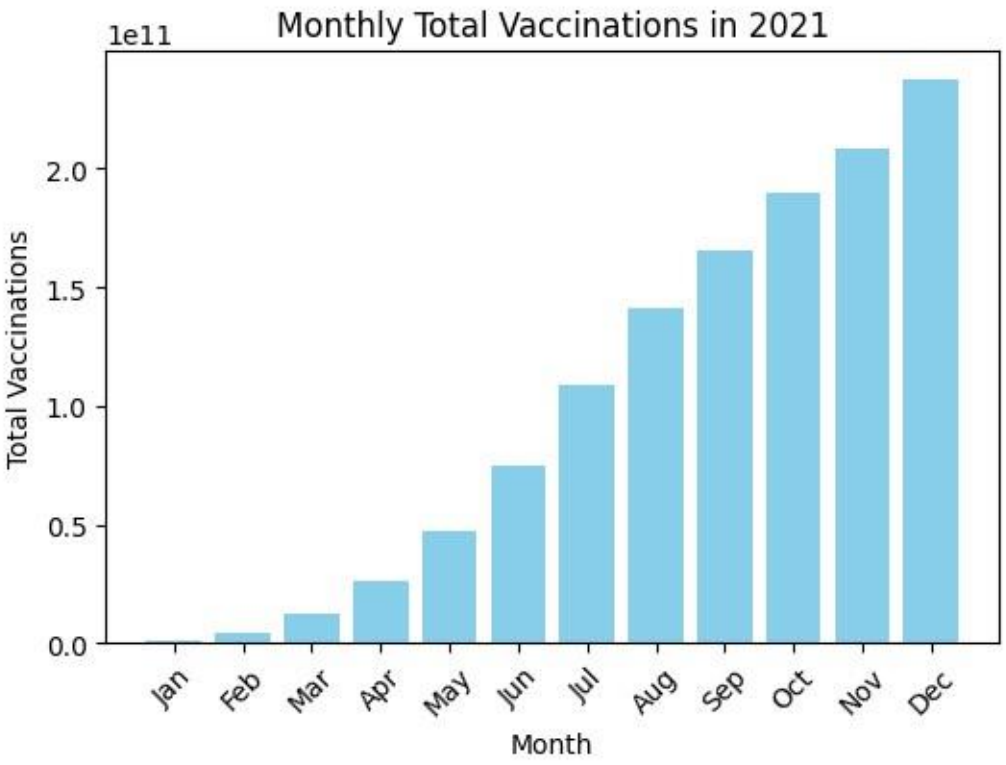
```
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',␣
 ↪'Nov', 'Dec']

# Create a pie chart
plt.figure(figsize=(5, 5))
plt.pie(monthly_total_vaccinations, labels=months, autopct='%1.1f%%')
plt.title('Monthly Total Vaccinations in 2021 ')
plt.axis('equal')  # Equal aspect ratio ensures that the pie chart is circular.

# Show the pie chart
plt.show()
```



[47]: # bar graph montly number of total vaccination from jan to dec in 2021

```
# Assuming that your date column is named 'date', create a new column for the␣
 ↪year and month df['year'] =
pd.to_datetime(df['date']).dt.year
df['month'] =
pd.to_datetime(df['date']).dt.month

# Filter the DataFrame to select data for the year 2021
filtered_df = df[df['year'] == 2021]
```

```
# Group and sum total vaccinations by month
monthly_total_vaccinations = filtered_df.groupby('month')['total_vaccinations'].
 ↪sum()

# Define the labels for the x-axis (months)
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct',␣
 ↪'Nov', 'Dec']

# Create a bar graph
plt.figure(figsize=(6, 4))
plt.bar(months, monthly_total_vaccinations, color='skyblue')
plt.title('Monthly Total Vaccinations in 2021 ')
plt.xlabel('Month')
plt.ylabel('Total Vaccinations')
plt.xticks(rotation=45)  # Rotate x-axis labels for better readability

# Show the bar graph
plt.show()
```



Conclusion

This project's design thinking process involves understanding the user's needs, defining objectives, generating ideas, developing and testing Python code for data analysis and visualization, and potentially integrating with IBM Cognos. The insights from this analysis can help website owners provide valuable information and improve the user experience regarding COVID-19 vaccination.