TG03 COVID 19 VACCINE ANALYSIS

PHASE 5

PROJECT OVERVIEW:

Problem Definition and Design Thinking

In this part you will need to understand the problem statement and create a document on what have you understood and how will you proceed ahead with solving the problem. Please think on a design and present in form of a document.

Innovation

The COVID-19 Analysis Project aims to leverage advanced data science techniques to gain comprehensive insights into various aspects of the pandemic. Our approach encompasses a range of innovative methods, each tailored to address specific facets of the COVID-19 crisis. Let's delve into the core components of this project.

In this phase, We are aiming to prepare our data set for further analysis phases of the project.

DATA PREPARATION

- **Data Pre-processing**
- **Data visualization**

For Data Pre-Processing, we have documented the most crucial area of work for our project type that is data preparation which will help in ready extraction and quick access to the precise data set.

For Data Visualization, We have planned to make all the required visual models for various accepts of Covid-19 vaccination progress in the form of various graphical and pictorial representation such as bar chart, line graph, scatter plot and pie chart.

The next stage of our project is arriving at

- **Feature Engineering**
- **Data Model Training**
- **Model Evaluation**

Where the preprocessed data on which earlier stage preliminary fundamental analytical tests have been performed is taken and further put to deeper understanding of the data information, clearer information processing and crisp on-point analysis that are advanced and extensively efficient.

Problem Definition:

The COVID-19 pandemic has significantly impacted public health, economies, and daily life worldwide. Data science can play a crucial role in understanding and mitigating its effects. The problem at hand is to conduct a data-driven analysis of COVID-19 data to gain insights into infection rates, vaccination trends, and their impact on healthcare systems. This analysis aims to inform decision-making, resource allocation, and public health strategies in the ongoing battle against the pandemic.

With reference to the link: https://www.kaggle.com/datasets/gpreda/covid-world-vaccination-progress

Design Thinking:

1. Data Collection:

Collecting data on COVID-19 vaccinations involves gathering information on variousaspects of the vaccination campaign.Some of the data points inclue:

- □ Number of doses administered(first dose , second dose)
- □ Type of vaccine administered(Covaxin,Covishield)
- □ Demographic information (age, gender)
- □ Location of vaccination sites
- □ Vaccination dates

2. Data Preprocessing:

Data Processing for COVID-19 Analysis involves several key steps to ensure that thedata is cleaned, prepared, and structured for meaningful analysis. This include:

Data collection: Gather reliable COVID-19 data from reputable sources.

- Data Cleaning: Handle missing values and outliers for accurate analysis.
- Integration and Transformation: Combine and format data for consistent analysis.
- Feature Engineering: Create new variables for deeper insights.
- Temporal and Spatial Aggregation: Analyze trends over time and by geographic regions.
- Ethical Considerations: Ensure data privacy and compliance with ethical guidelines.
- Documentation and Storage: Maintain clear records and store data securely.
- Validation and Quality Assurance: Verify data accuracy and integrity.

3.Exploratory Data Analysis:

By conducting EDA, we gain valuable insights into the COVID-19 pandemic, whichinforms public health strategies and policy decisions.

- Summarize Stats: Overview of cases, deaths, recoveries.
- Time Trends: Visualize cases, deaths, recoveries over time.
- Geospatial Patterns: Identify hotspots on maps.
- Correlations: Relationships between variables.
- Demographics Impact: Age, gender influence on infection rates.
- Vaccination Impact: Analyze vaccination rates on cases.
- Severity Distribution: Mild vs. severe cases.
- Epidemiological Metrics: R0, CFR, attack rates.
- Comparative Analysis: Regional, country-wise trends.
- Time to Event: Duration analysis.
- Visual Representation: Graphs, plots, heatmaps.
- Anomaly Detection: Identify unusual patterns.

4.Statistical Analysis:

Statistical analysis in COVID-19 involves applying various quantitative techniques to understand and draw insights from the data related to the pandemic. Here are some key statisticalanalyses commonly used in COVID-19 research:

- Hypothesis Testing: Evaluate significance of interventions.
- Regression Models: Predict cases, deaths, and trends.
- Time Series Analysis: Understand temporal patterns.
- Correlation Analysis: Examine relationships between variables.
- ANOVA: Compare means across different groups.
- Chi-Square Tests: Analyze categorical data associations.
- Survival Analysis: Study time-to-event outcomes.
- Bayesian Inference: Assess uncertainty in predictions.
- Machine Learning Models: Predict outcomes and inform policy.
- Monte Carlo Simulations: Evaluate scenarios and intervention.

5.Visualization:

Visualization aids in communicating COVID-19 trends effectively to inform decision-making and public health strategies.

- Time Series Plots: Track cases, recoveries, and deaths over time.
- Heatmaps: Visualize regional hotspots and trends.
- Bar Charts: Compare metrics like cases or vaccinations across regions.
- Pie Charts: Illustrate proportions of cases, recoveries, etc.
- Stacked Area Charts: Show cumulative trends over time.
- Epidemiological Curve: Plot cases by date of onset to understand disease spread.
- Dashboard Interfaces: Combine multiple visualizations for comprehensive insights.
- Box Plots: Analyze distributions and variability in data.
- Correlation Matrices: Visualize relationships between variables.
- Treemaps: Represent hierarchical data, e.g., cases by region.

6.Insights and Recommendations:

Insights:

- Track temporal trends.
- Identify hotspots.
- Analyze demographics, vaccinations, and metrics.
- Evaluate interventions and variants.

Recommendations:

- Prioritize vaccinations.
- Promote preventive measures.
- Target interventions.
- Plan healthcare capacity.
- Enhance testing and tracing.
- Monitor variants.
- Engage communities.
- Foster research and development.
- Support healthcare workers.

1. Real-time Data Integration:

We begin by ensuring that our analyses are based on the most current and accurate information available. This is achieved through a process of real-time data integration. By regularly collecting and updating data from trusted sources such as the World Health Organization (WHO), Centers for Disease Control and Prevention (CDC), and government health agencies, we ensure that our insights are always based on the latest information. For example, we fetch and update data on confirmed cases, recoveries, vaccinations, and preferences every 24 hours.

2. Geographical Visualizations:

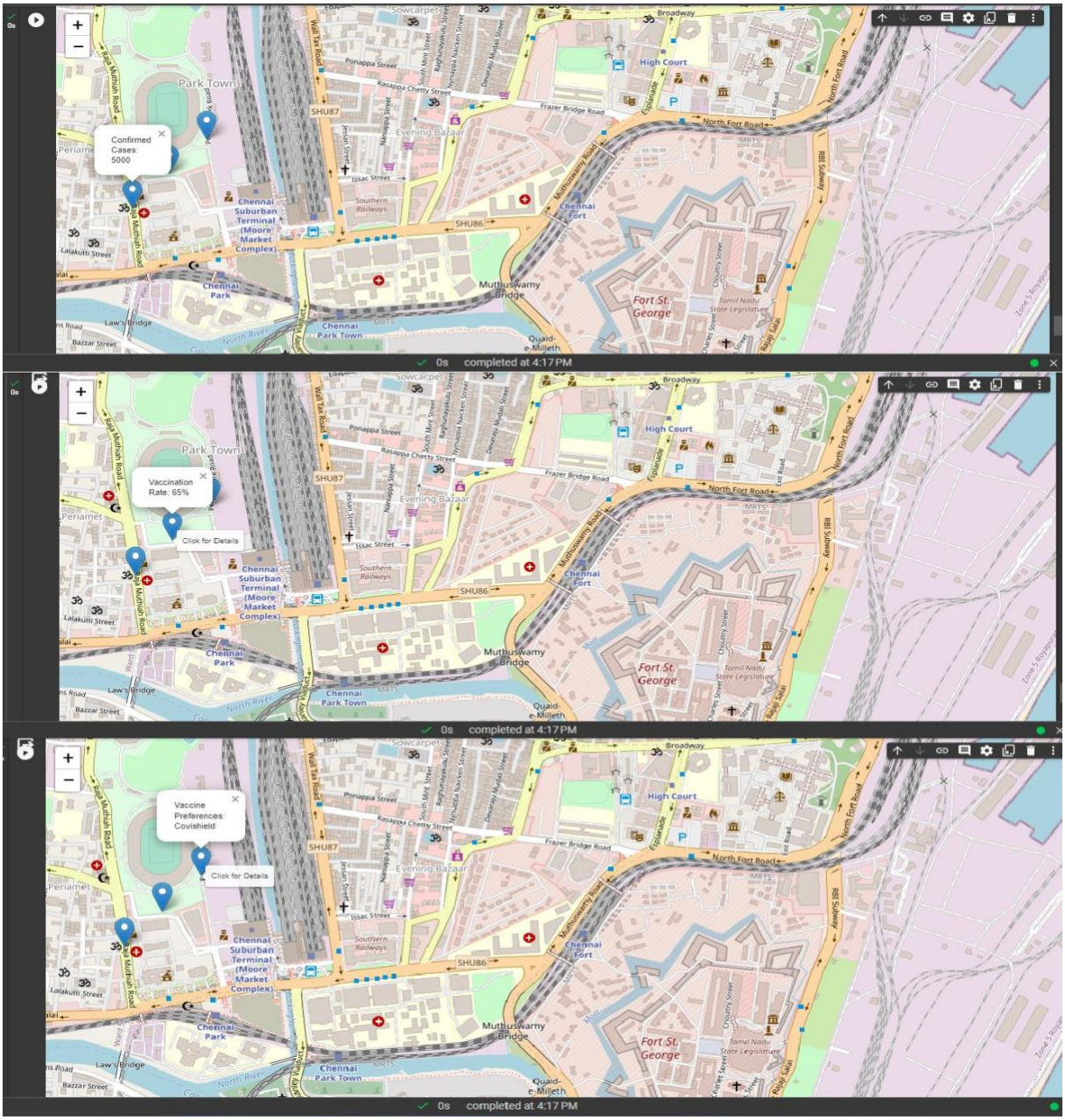
Visualizing data across different geographical regions is pivotal in understanding the spread and impact of COVID-19. To achieve this, we employ advanced geospatial analysis techniques. Through interactive maps, we provide a visual representation of critical metrics including confirmed cases, vaccination rates, and preferences. These heatmaps serve as powerful tools, enabling us to quickly identify trends and patterns across various states and regions."

Sample code:

Heatmap code for covid analysis in chennai

```
import folium

# Create a base map centered around Chennai, India
m = folium.Map(location=[13.0827, 80.2707], zoom_start=12)

# Add markers for confirmed cases, vaccination rates, and preferences
folium.Marker([13.0827, 80.2707], popup='Confirmed Cases: 5000', tooltip='Click for Details').add_to(m)
folium.Marker([13.0837, 80.2717], popup='Vaccination Rate: 65%', tooltip='Click for Details').add_to(m)
folium.Marker([13.0847, 80.2727], popup='Vaccine Preferences: Covishield', tooltip='Click for Details').add_to(m)

# Display the map
m
```

Output:



3. Temporal Trends and Forecasting:

Analyzing the temporal dimension of the pandemic allows us to identify trends and make informed projections. We employ time series analysis and visualization techniques to understand the trajectory of COVID-19 cases, recoveries, and vaccinations. Furthermore, we utilize state-of-the-art time series models to forecast future trends. This enables us to anticipate and prepare for potential shifts in the pandemic's course.

4. Vaccination Preferences Analysis:

Understanding public sentiment towards different vaccines is crucial in shaping effective vaccination campaigns. To achieve this, we conduct sentiment analysis on social media data, particularly on platforms like Twitter. This allows us to gauge public sentiment towards Covishield and Covivaccine. By analyzing trends over time, we gain valuable insights into how the public perceives and discusses these vaccines.

5. Risk Assessment and Allocation Model:

Optimizing vaccine allocation is a critical aspect of our approach. To achieve this, we employ machine learning models to assess COVID-19 risk. This involves considering a range of factors such as infection rates, population density, and healthcare capacity. By accurately predicting high-risk areas, we can strategically allocate vaccines to maximize their impact.

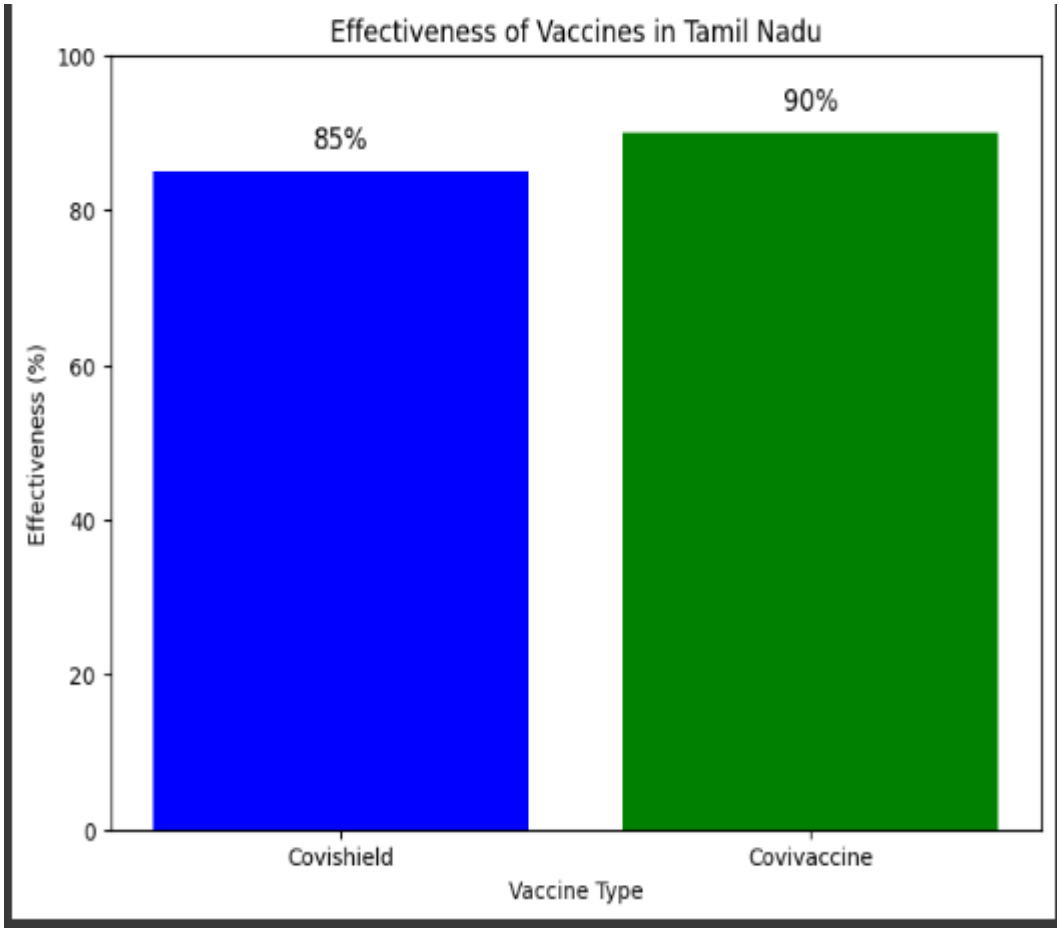
6. Effectiveness Comparison:

Evaluating the effectiveness of different vaccines is paramount in ensuring the success of vaccination campaigns. Through rigorous comparative studies, we quantitatively assess the effectiveness rates of both Covishield and Covivaccine. Visualizing these results using clear and informative visualizations, such as bar charts or radar plots, allows us to communicate these findings effectively.

Sample code we used -To analyse the rates of Covishield and Covivaccine in tamilnadu:

```
import matplotlib.pyplot as plt

# Example Data
vaccines = ['Covishield', 'Covivaccine']
effectiveness = [85, 90]  # Example effectiveness rates in percentage

# Create a bar chart
plt.figure(figsize=(8, 6))
plt.bar(vaccines, effectiveness, color=['blue', 'green'])
plt.xlabel('Vaccine Type')
plt.ylabel('Effectiveness (%)')
plt.title('Effectiveness of Vaccines in Tamil Nadu')
plt.ylim(0, 100)

# Add data labels
for i, value in enumerate(effectiveness):
    plt.text(i, value + 2, f'{value}%', ha='center', va='bottom', fontsize=12, color='black')

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

Output:



7.Demographic Insights:

Understanding how COVID-19 affects different demographic groups is crucial for targeted intervention strategies. We conduct a detailed analysis, breaking down infection rates and vaccination coverage by age, gender, and other relevant factors. This information is then presented in easily interpretable stacked bar charts, providing a comprehensive view of the pandemic's impact on various segments of the population.

1.DATA PREPARATION

- **Data Pre-processing**
- **Data visualization**

For Data Pre-Processing, we have documented the most crucial area of work for our project type that is data preparation which will help in ready extraction and quick access to the precise data set.

For Data Visualization, We have planned to make all the required visual models for various accepts of Covid-19 vaccination progress in the form of various graphical and pictorial representation such as bar chart, line graph, scatter plot and pie chart.

2.DATA PRE-PROCESSING:

Data preprocessing is an important step in the data science process. It refers to the cleaning, transforming, and integrating of data in order to make it ready for analysis.

The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific data science analysis task.

Data Cleaning:

- Handle missing values.
- Validate data for anomalies, outliers, and errors.
- Correct any inconsistent or erroneous entries.

In our project, we utilized this data cleaning process involved removing rows where critical vaccination information was entirely absent - such as 'total_vaccinations', 'people_vaccinated', and 'people_fully_vaccinated'.

The number of removed rows was then calculated, ensuring that our analysis was based on complete and reliable data.

In [1]:
```
import pandas as pd

df = pd.read_csv('country_vaccinations.csv')

num_rows_before = df.shape[0]

df = df.dropna(subset=['total_vaccinations', 'people_vaccinated', 'people_fully_vaccinated'], how='all')

num_rows_after = df.shape[0]

rows_removed = num_rows_before - num_rows_after

print(f"Number of rows removed: {rows_removed}")
```

```
Number of rows removed: 42483
```

In [2]:
```
import pandas as pd

df = pd.read_csv('country_vaccinations_by_manufacturer.csv')

num_rows_before = df.shape[0]

df = df[(df['total_vaccinations'].notna()) & (df['total_vaccinations'] != 0)]

num_rows_after = df.shape[0]

rows_removed = num_rows_before - num_rows_after

print(f"Number of rows removed: {rows_removed}")
```

```
Number of rows removed: 1482
```

Data Integration:

Merge or concatenate multiple datasets if applicable, based on common identifiers.
By using the integration technique ,we merged two datasets—vaccination records and vaccine manufacturer information—based on the common field 'total_vaccinations'. This integration allows us to analyze vaccination data alongside details about the manufacturers involved.

In [3]:
```
import pandas as pd

df_vaccinations = pd.read_csv('country_vaccinations.csv')
df_manufacturer = pd.read_csv('country_vaccinations_by_manufacturer.csv')
merged_df = pd.merge(df_vaccinations, df_manufacturer, on='total_vaccinations')
print(merged_df)
```

```
            country iso_code      date_x  total_vaccinations  \
0       Afghanistan      AFG  2021-02-22                 0.0
1       Afghanistan      AFG  2021-02-22                 0.0
2       Afghanistan      AFG  2021-02-22                 0.0
3       Afghanistan      AFG  2021-02-22                 0.0
4       Afghanistan      AFG  2021-02-22                 0.0
...             ...      ...         ...                 ...
187750     Zimbabwe      ZWE  2021-02-18                39.0
187751     Zimbabwe      ZWE  2021-02-18                39.0
187752     Zimbabwe      ZWE  2021-02-18                39.0
187753     Zimbabwe      ZWE  2021-03-09             36307.0
187754     Zimbabwe      ZWE  2021-03-23             45743.0

        people_vaccinated  people_fully_vaccinated  daily_vaccinations_raw  \
0                     0.0                      NaN                     NaN
1                     0.0                      NaN                     NaN
2                     0.0                      NaN                     NaN
3                     0.0                      NaN                     NaN
4                     0.0                      NaN                     NaN
```

Data Transformation:

- Normalize or standardize numerical features.
- Encode categorical variables.
- Handle date and time data.

We selected the first five rows of the dataset using the **.head(5)** method to get a quick overview of the data. We replaced any missing values in the DataFrame with zeros using **df.fillna(0, inplace=True)**. This ensures that missing data doesn't affect our analysis.

```
In [4]: df = pd.read_csv('country_vaccinations.csv').head(5)
        df['date'] = pd.to_datetime(df['date'])
        df.fillna(0, inplace=True)
        df
```

Out[4]:

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

3.DATA VISUALISATION:

Data visualization is transforming data or information into graphics to make it easier for the human brain to comprehend and get insights.

The purpose of data visualization projects is to identify patterns, trends, and anomalies or deviations in large datasets/big data (the main data for visualization projects); that otherwise would have been impossible. This is the final step of the data science process and data presentation architecture (DPA) .

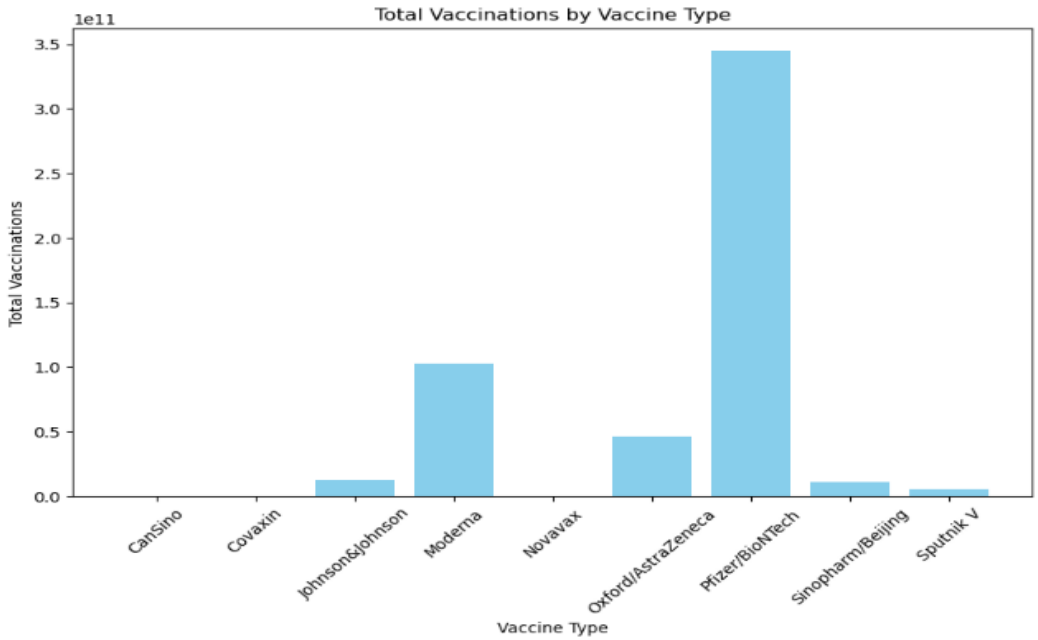
The raw dataset for “Covid -19 Vaccine Analysis“ provided has been extracted from kaggle, Loaded into Anaconda Jupyter notebook pre-processed and prepared using python(Pandas) for visualisation.

1.Bar graph:

A bar graph (or bar chart) is a graphical representation of data in which rectangular bars are used to compare the values of different categories**.**

We loaded a dataset containing manufacturer-specific vaccination information. After specifying key vaccines of interest, we filtered and summed the total vaccinations for each. This bar graph visually represents the total vaccinations for selected vaccine types, aiding in a clear comparison of their distribution.

In [5]:
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('country_vaccinations_by_manufacturer.csv')

additional_vaccines = ['CanSino', 'Pfizer/BioNTech', 'Johnson&Johnson', 'Covaxin', 'Novavax']

selected_vaccines = ['Moderna', 'Oxford/AstraZeneca', 'Sinopharm/Beijing', 'Sputnik V'] + additional_vaccines

df_selected_vaccines = df[df['vaccine'].isin(selected_vaccines)]

vaccine_totals = df_selected_vaccines.groupby('vaccine')['total_vaccinations'].sum()

plt.figure(figsize=(10,6))
plt.bar(vaccine_totals.index, vaccine_totals.values, color='skyblue')
plt.xlabel('Vaccine Type')
plt.ylabel('Total Vaccinations')
plt.title('Total Vaccinations by Vaccine Type')
plt.xticks(rotation=45)
plt.show()
```

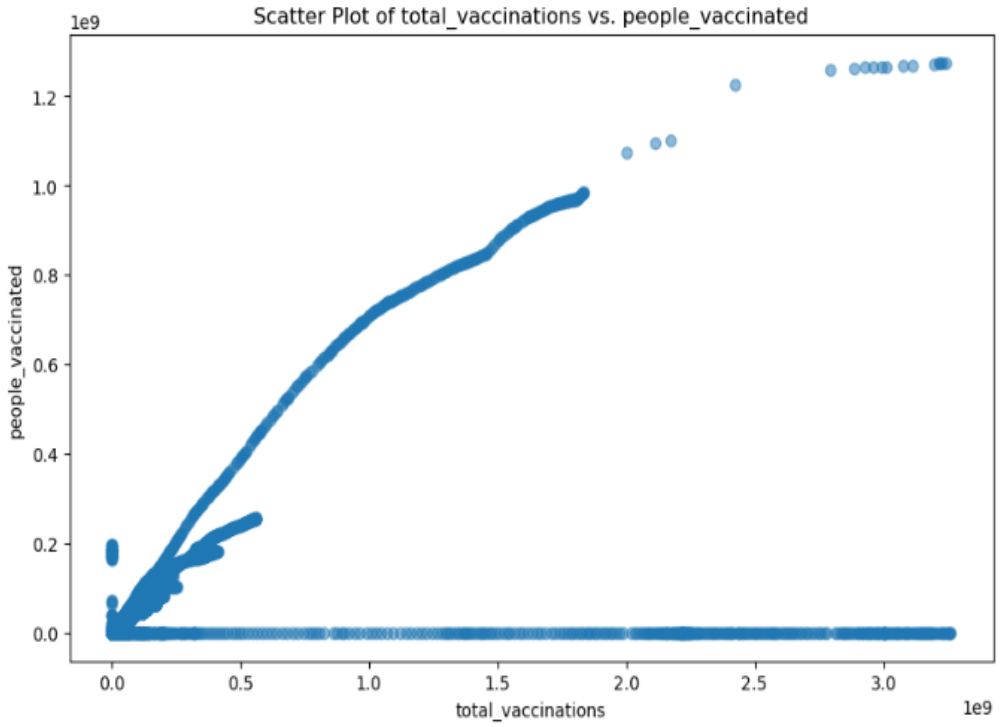


2.Scatter-Plot:

The purpose of the scatter plot is to display what happens to one variable when another variable is changed. The scatter plot is used to test a theory that the two variables are related.

The vaccination progress has been taken as the base theory involved in analysis and the two variable involved in the below scatter plot is 'total_vaccinations' and 'people_vaccinated' respectively.

In [6]:
```
import pandas as pd
import matplotlib.pyplot as plt

df = pd.read_csv('country_vaccinations.csv')

df['date'] = pd.to_datetime(df['date'])

df.fillna(0, inplace=True)

df = pd.get_dummies(df, columns=['country', 'iso_code'], drop_first=True)

plt.figure(figsize=(10, 6))

x_variable = 'total_vaccinations'
y_variable = 'people_vaccinated'

plt.scatter(df[x_variable], df[y_variable], alpha=0.5)

plt.xlabel(x_variable)
plt.ylabel(y_variable)

plt.title(f'Scatter Plot of {x_variable} vs. {y_variable}')

plt.show()
```



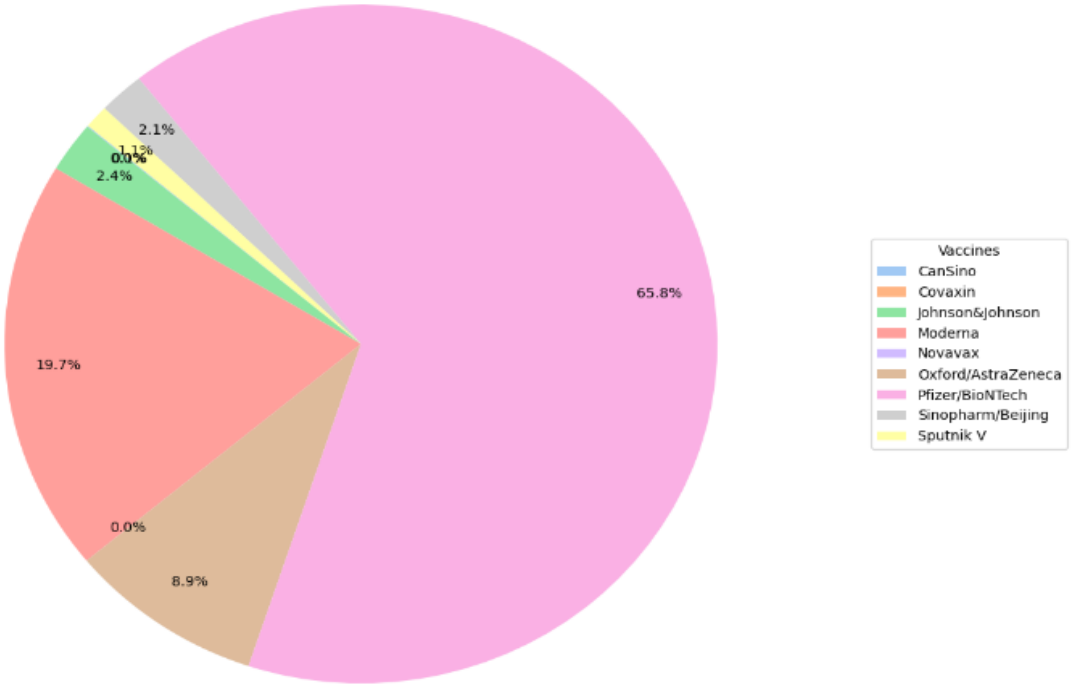
3.Pie-chart:

Determine the ratio or percentage that each component takes up out of the whole. The total sum of percentages should sum to 100%. Divide the circle into proportional sectors

We made a pie chart below to show how many people received each type of vaccine. Each slice represents a different vaccine, and the numbers inside the slices tell us the percentage. We used different colors to make each slice stand out. The key on the side tells us which color goes with each vaccine.

Program code and Output:

```
In [10]: import seaborn as sns

colors = sns.color_palette('pastel', len(vaccine_totals))

plt.figure(figsize=(12, 8))

wedges, texts, autotexts = plt.pie(vaccine_totals, autopct='%1.1f%%', startangle=140, colors=colors, pctdistance=0.85)

plt.axis('equal')

plt.legend(wedges, vaccine_totals.index, title="Vaccines", loc="center left", bbox_to_anchor=(1, 0, 0.5, 1))

plt.tight_layout()

plt.show()
```



4.Line Plot:

Line plot or a line chart—is a graph that uses lines to connect individual data points. A line graph displays quantitative values over a specified time interval.

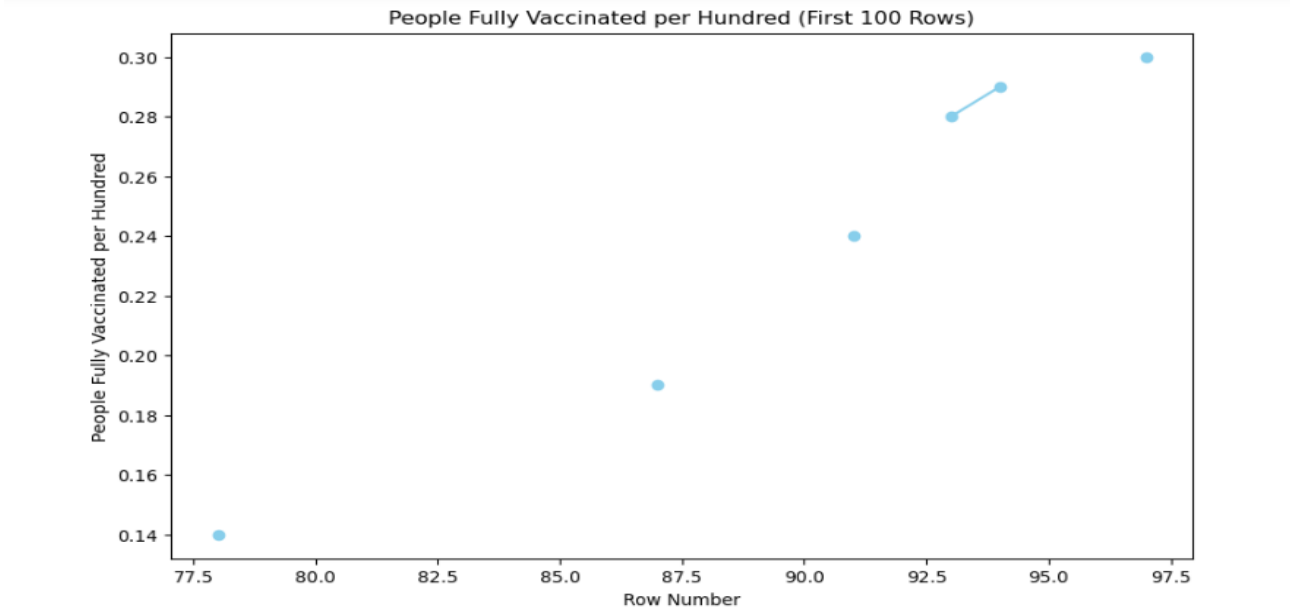
The **Covid 19 Vaccine Analysis** has various content fields containing dynamic and vast range of data entries and it will be highly challenging for the data scientist to compare every single entry. This is where data visualization using line charts comes handy.

Creating multiple line plots for individual fields will aid in quickly identifying and arriving at reliable insights-based decisions.

We made a line plot to show the people fully vaccinated per hundred.

Program Code and Output:

In [11]:
```
import pandas as pd

import matplotlib.pyplot as plt

df = pd.read_csv('country_vaccinations.csv').head(100)

plt.figure(figsize=(10, 6))

plt.plot(df['people_fully_vaccinated_per_hundred'], marker='o', color='skyblue', linestyle='-')

plt.xlabel('Row Number')

plt.ylabel('People Fully Vaccinated per Hundred')

plt.title('People Fully Vaccinated per Hundred (First 100 Rows)')

plt.show()
```



1.FEATURE ENGINEERING:

•Feature engineering is the process of selecting, transforming, or creating data attributes (features) to enhance the performance of machine learning models.

•It involves tasks such as feature selection, handling missing data, scaling, encoding categorical variables, and creating new informative features.

•Effective feature engineering can significantly impact a model's accuracy and predictive power. It requires domain knowledge and a balance between simplification and information retention.

2.TIME SERIES ANALYSIS:

Time series analysis is a statistical method for studying data collected over time. It involves identifying patterns, trends, and seasonality within the data, making predictions, and detecting anomalies.

Common techniques include decomposition, forecasting models like ARIMA, and visualizing data for better insights into temporal relationships.

We performed this analysis in our code below to find percentage of a country fully vaccinated month wise,

In [2]:
```
import pandas as pd

df = pd.read_csv('country_vaccinations.csv')

# Convert 'date' column to datetime format
df['date'] = pd.to_datetime(df['date'])

# Extract year and month
df['year'] = df['date'].dt.year
df['month'] = df['date'].dt.month

# Calculate the percentage
df['percentage_fully_vaccinated'] = (df['people_fully_vaccinated'] / df['total_vaccinations']) * 100

# Sort the DataFrame by country, year, and month
df_sorted = df.sort_values(by=['country', 'year', 'month'])

# Group by country, year, and month, then calculate the average percentage for each month
result = df_sorted.groupby(['country', 'year', 'month'])['percentage_fully_vaccinated'].mean()

# Reset the index to get the DataFrame format
result_df = result.reset_index()

# Print the result DataFrame
print(result_df)
```

	country	year	month	percentage_fully_vaccinated
0	Afghanistan	2021	2	NaN
1	Afghanistan	2021	3	NaN
2	Afghanistan	2021	4	NaN
3	Afghanistan	2021	5	16.675279
4	Afghanistan	2021	6	23.651193
...	...	...	...	...
3013	Zimbabwe	2021	11	43.366796
3014	Zimbabwe	2021	12	42.953565
3015	Zimbabwe	2022	1	43.225408
3016	Zimbabwe	2022	2	43.138566
3017	Zimbabwe	2022	3	41.875884

[3018 rows x 4 columns]

•Initially, we load the dataset and ensure the dates are in the right format for accurate analysis. Then, we extract the year and month, which will be crucial for understanding trends over time.

•Moving forward, we calculate the percentage of individuals who are fully vaccinated. To keep things organized, we sort the data by country, year, and month.

Following that, we group the data and compute the average percentage of fully vaccinated individuals for each month

3.GEOSPATIAL ANALYSIS:

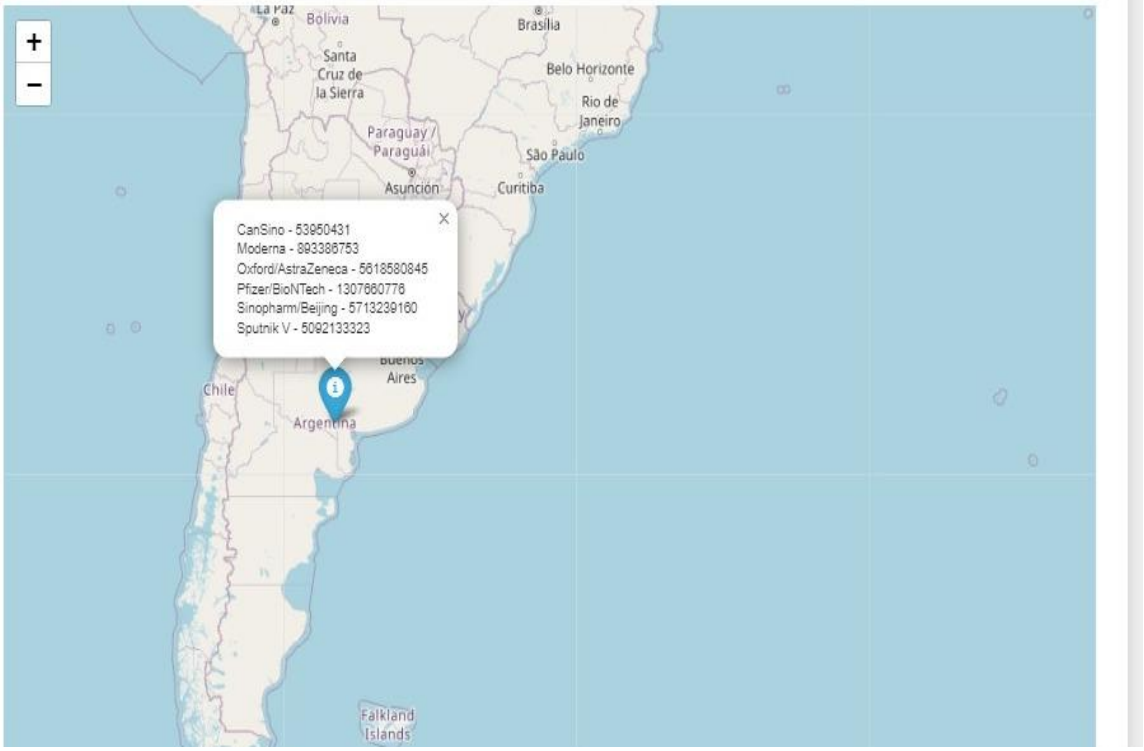
Geospatial analysis is the process of examining and interpreting data that is linked to geographic locations. It involves using geographic information systems (GIS) and other tools to analyze spatial patterns, relationships, and trends.

Geospatial analysis is vital in fields like urban planning, environmental management, logistics, and disaster response for informed decision-making and problem-solving based on location-based data.We performed this analysis in our code below to examine a visualize about total vaccination

In [3]:

```
import pandas as pd
import folium

# Load the dataset with total vaccinations by manufacturer
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')

# Filter data for Argentina,we can give any country we want.
argentina_data = data[data['location'] == 'Argentina']
total_vaccinations_by_vaccine = argentina_data.groupby('vaccine')['total_vaccinations'].sum().reset_index()

# Get the latitude and longitude for Argentina
argentina_lat = -38.4161
argentina_lon = -63.6167

# Create a base map centered on Argentina
map = folium.Map(location=[argentina_lat, argentina_lon], zoom_start=4)

# Generate popup content with vaccine names and total vaccinations
popup_content = '<br>'.join(f"{row['vaccine']} - {row['total_vaccinations']}" for _, row in total_vaccinations_by_vaccine.ite

# Add a marker with the popup content
folium.Marker(
    location=[argentina_lat, argentina_lon],
    popup=folium.Popup(popup_content, max_width=300),
    icon=folium.Icon(color='blue')
).add_to(map)

# Show the map
map
```

Out[3]:



•Firstly, we load a dataset containing information about total vaccinations by manufacturer. We then filter this data specifically for Argentina to focus our analysis.

•After that, we aggregate the total vaccinations for each vaccine type.Next, we gather the latitude and longitude coordinates for Argentina. Using these coordinates, we create a map centered on the country.

•Now, for each vaccine, we generate a popup displaying its name along with the total vaccinations administered. To visualize this on the map, we add a marker precisely at the center of Argentina.

•Clicking on this marker will provide detailed information about the vaccines and their respective totals. Finally, we display the map to see the distribution of vaccinations across different vaccine types.

4.DERIVED METRICS:

Derived metrics are calculated or derived from existing data to provide additional insights. These metrics help to understand underlying patterns, relationships, and trends.

They often involve mathematical operations or transformations on the original data, such as calculating percentages, averages, or ratios. Derived metrics are valuable in analytics and reporting for making data-driven decisions and gaining a deeper understanding of complex datasets

In [4]:
```
import pandas as pd

# Load the dataset
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')

# Calculate vaccination rate by vaccine type
vaccination_rate_by_vaccine = data.groupby('vaccine')['total_vaccinations'].sum() / data.groupby('vaccine').size()

# Create a new DataFrame to store the result
result_df = pd.DataFrame({'vaccination_rate_by_vaccine': vaccination_rate_by_vaccine})

# Merge the result DataFrame with the original data on 'vaccine'
data = data.merge(result_df, on='vaccine', how='left')

# Show the result
print(data)
```

	location	date	vaccine	total_vaccinations	vaccination_rate_by_vaccine
0	Argentina	2020-12-29	Moderna	2	1.552058e+07
1	Argentina	2020-12-29	Oxford/AstraZeneca	3	7.003092e+06
2	Argentina	2020-12-29	Sinopharm/Beijing	1	7.241682e+06
3	Argentina	2020-12-29	Sputnik V	20481	5.923586e+06
4	Argentina	2020-12-30	Moderna	2	1.552058e+07
...	...	...	...	...	...
35618	European Union	2022-03-29	Oxford/AstraZeneca	67403106	7.003092e+06
35619	European Union	2022-03-29	Pfizer/BioNTech	600519998	3.879792e+07
35620	European Union	2022-03-29	Sinopharm/Beijing	2301516	7.241682e+06
35621	European Union	2022-03-29	Sinovac	1809	5.556222e+06
35622	European Union	2022-03-29	Sputnik V	1845103	5.923586e+06

[35623 rows x 5 columns]

5.MODEL TRAINING:

Model training is the phase in the data science development lifecycle where practitioners try to fit the best combination of weights and bias to a machine learning algorithm to minimize a loss function over the prediction range.

A linear regression model can be trained using the optimization algorithm gradient descent by iteratively modifying the model's parameters to reduce the mean squared error (MSE) of the model on a training dataset. To update θ_1 and θ_2 values in order to reduce the Cost function (minimizing RMSE value) and achieve the best-fit line the model uses Gradient Descent.

The idea is to start with random θ_1 and θ_2 values and then iteratively update the values, reaching minimum cost. A gradient is nothing but a derivative that defines the effects on outputs of the function with a little bit of variation in inputs

Let's differentiate the cost function(J) with respect to θ_1

$$
\begin{aligned}
J'_{\theta_1} &= \frac{\partial J(\theta_1, \theta_2)}{\partial \theta_1} \\
&= \frac{\partial}{\partial \theta_1} \left[\frac{1}{n} \left(\sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(\frac{\partial}{\partial \theta_1} (\hat{y}_i - y_i) \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(\frac{\partial}{\partial \theta_1} (\theta_1 + \theta_2 x_i - y_i) \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(1 + 0 - 0 \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} (\hat{y}_i - y_i) \left(2 \right) \right] \\
&= \frac{2}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)
\end{aligned}
$$

Let's differentiate the cost function(J) with respect to

$$
\begin{aligned}
J'_{\theta_2} &= \frac{\partial J(\theta_1, \theta_2)}{\partial \theta_2} \\
&= \frac{\partial}{\partial \theta_2} \left[\frac{1}{n} \left(\sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(\frac{\partial}{\partial \theta_2} (\hat{y}_i - y_i) \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(\frac{\partial}{\partial \theta_2} (\theta_1 + \theta_2 x_i - y_i) \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} 2(\hat{y}_i - y_i) \left(0 + x_i - 0 \right) \right] \\
&= \frac{1}{n} \left[\sum_{i=1}^{n} (\hat{y}_i - y_i) \left(2x_i \right) \right] \\
&= \frac{2}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i) \cdot x_i
\end{aligned}
$$

Finding the coefficients of a linear equation that best fits the training data is the objective of linear regression. By moving in the direction of the Mean Squared Error negative gradient with respect to the coefficients, the coefficients can be changed. And the respective intercept and coefficient of X will be if is the learning rate.

Parameter Tuning

Selecting the correct parameter that will be modified to influence the ML model is key to attaining accurate correlation. The set of parameters that are selected based on their influence on the model architecture are called hyperparameters. The process of identifying the hyperparameters by tuning the model is called parameter tuning. The parameters for correlation should be clearly defined in a manner in which the point of diminishing returns for validation is as close to 100% accuracy as possible.

6.LINEAR REGRESSION:

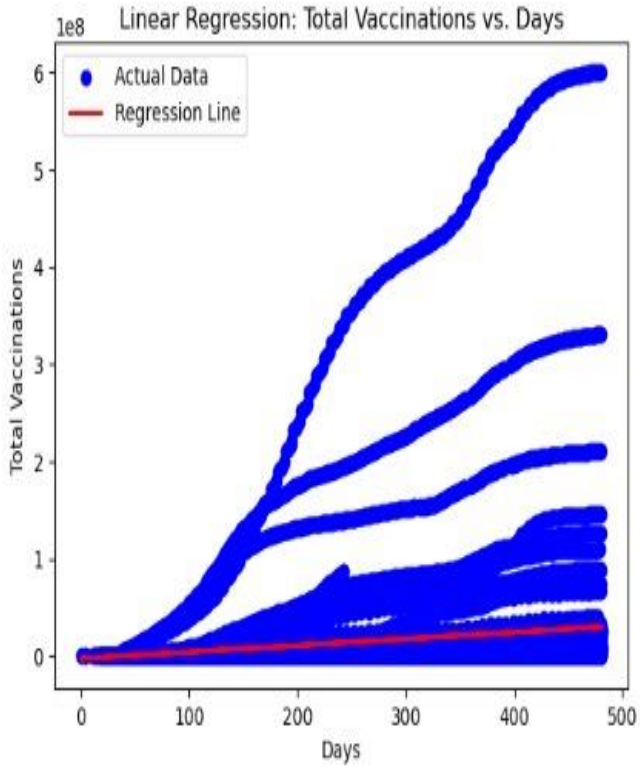
Linear regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables by fitting a linear equation. It aims to predict the dependent variable based on the values of the independent variables.

The model assumes a linear relationship, with coefficients representing the impact of each independent variable on the dependent variable. Linear regression is widely used for predictive modeling and understanding the strength and direction of relationships between variables.

In the code below.we, are training the model using the linear regression technique to predict the vaccinations rate by the number of vaccinations obtained in certain days .if thr coefficient is positive, it means that as the number of days increases, the total vaccinations are expected to increase as well.

In [6]:
```
import pandas as pd
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt+
# Load the dataset
data = pd.read_csv('country_vaccinations_by_manufacturer.csv')

# Convert 'date' to numerical format (number of days since the start date)
data['date'] = (pd.to_datetime(data['date']) - pd.to_datetime(data['date']).min()).dt.days

# Prepare the feature matrix X and the target variable y
X = data[['date']]
y = data['total_vaccinations']

# Initialize the Linear Regression model
model = LinearRegression()
# Train the model
model.fit(X, y)

# Print the coefficients
print('Coefficient:', model.coef_[0])
print('Intercept:', model.intercept_)

# Predict total vaccinations
predictions = model.predict(X)

# Create a scatter plot of the actual data
plt.scatter(X, y, color='blue', label='Actual Data')

# Plot the regression line
plt.plot(X, predictions, color='red', linewidth=2, label='Regression Line')

# Add labels and title
plt.xlabel('Days')
plt.ylabel('Total Vaccinations')
plt.title('Linear Regression: Total Vaccinations vs. Days')
# Add legend
plt.legend()
# Show the plot
plt.show()
```

```
Coefficient: 67905.85580252811
Intercept: -2954051.2822799943
```



We're applying Linear Regression to analyze COVID-19 vaccination data. Firstly, we load a dataset .Then, to make the 'date' column usable for the regression model, we convert it into a numerical format representing the number of days since the start date. Next, we prepare the feature matrix 'X', containing the numerical representation of dates, and the target variable 'y', representing the total vaccinations

- . We then initialize a Linear Regression model. Moving on, we train the model on the given data. After training, we print out the coefficients, providing insights into how the model has learned from the data.

We then use the trained model to predict total vaccinations. For visualization, we create a scatter plot of the actual data points in blue. Additionally, we plot the regression line in red, which represents the model's predictions. This line showcases the relationship between days and total vaccinations as predicted by the model.

Coefficient: 67905.85580252811

This means that for every additional day, the total number of vaccinations is expected to increase by approximately 67906. This is the slope of the regression line.

7.MODEL EVALUATION:

Model evaluation is the process of assessing the performance and effectiveness of a machine learning model. It involves techniques like splitting data into training and testing sets, cross-validation, and various metrics (e.g., accuracy, precision, recall, F1-score) to measure a model's predictive quality. The goal is to ensure the model generalizes well to new, unseen data and to make informed decisions about its suitability for a specific task.

```
In [7]: import pandas as pd
        from sklearn.linear_model import LinearRegression
        from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
        import numpy as np

        # Load the dataset
        data = pd.read_csv('country_vaccinations_by_manufacturer.csv')

        # Convert 'date' to numerical format (number of days since the start date)
        data['date'] = (pd.to_datetime(data['date']) - pd.to_datetime(data['date']).min()).dt.days

        # Prepare the feature matrix X and the target variable y
        X = data[['date']]
        y = data['total_vaccinations']

        # Initialize the Linear Regression model
        model = LinearRegression()

        # Training the model x,y
        model.fit(X, y)

        # Predicting total vaccinations
        predictions = model.predict(X)

        # Calculate evaluation metrics
        rmse = np.sqrt(mean_squared_error(y, predictions))
        mae = mean_absolute_error(y, predictions)
        r_squared = r2_score(y, predictions)

        # Printing the evaluation metrics
        print(f'Root Mean Squared Error (RMSE): {rmse:.2f}')
        print(f'Mean Absolute Error (MAE): {mae:.2f}')
        print(f'R-squared (R2): {r_squared:.2f}')
```

```
Root Mean Squared Error (RMSE): 51122830.05
Mean Absolute Error (MAE): 20286696.16
R-squared (R2): 0.03
```

Moving on to model evaluation, we calculate three important metrics. These metrics help to assess how well the model is performing in terms of predicting the total vaccinations based on the number of days since the start date.:

- Root Mean Squared Error (RMSE): This measures the average error between predicted and actual values, with higher values indicating greater deviation.
- Mean Absolute Error (MAE): It computes the average absolute difference between predicted and actual values, providing another perspective on the model's performance.
- R-squared (R2) Score: This metric assesses how well the model's predictions align with the actual data. An R2 score of 1 indicates a perfect fit.

Finally, we print out these evaluation metrics, offering a comprehensive view of the model's performance. These metrics are crucial in assessing the accuracy and effectiveness of the Linear Regression model.

PROJECT OUTCOMES:

To bbegin with the fundamental idea of the project we use the most basic problem solving concepts such as clear problem definition to gain knowledge about the problem from various dimensions and design thinking to get a hold on the multiple possibilities or routes to arrive at the most optimal solution to the project

We have removed missing values and cleaned(data cleaning) clumsy and vast data by removal of redundant dataset entries while also working on data integration to avoid complete removal of repetitive data and retain the entry by combining them.

Also we have displayed all visual elements that will provide the project's analysis phase an immense grip and clarity .

- to gain a quick and comparative knowledge of the world wide progress
- leading to clear and comprehensive analysis
- summing up to a more efficient and data driven decisions in various aspects.

Feature Engineering

- Time series analysis
- Geospatial Analysis
- Derived Metrics

Model Training and Machine Learning

- Linear Regression Training
- History Based Learning

The model evaluation of our data science model's have been carried out using a new-to-tech methodology involving Machine Learning which evaluates the model's based on training memoryto compare and improve the models error free efficiency and overall analysis success.

CONCLUSION:

The COVID-19 Vaccine dataset with various fields has been successfully analyzed preprocessed and multiple forms of evaluative insights using various techniques have been withdrawn to arrive at the final project completion to improve the vaccination implementation across the world.