PROFESSIONAL TRAINING REPORT
at

Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

Kolla Om Vivek
REG. NO. 39110511

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI – 600119, TAMILNADU

NOVEMBER 2021

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

(Established under Section 3 of UGC Act, 1956)

JEPPIAAR NAGAR, RAJIV GANDHI SALAI

CHENNAI– 600119

www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Kolla Om Vivek (Reg. No: 39110511)** who carried out the project entitled "**Covid-19 day to day analysis and prediction** " under my supervision from June 2021 to November 2021.

Internal Guide

Dr. Joshila Grace

Head of the Department

Submitted for Viva voce Examination held on____________________

InternalExaminer **ExternalExaminer**

DECLARATION

I, **Kolla Om Vivek** hereby declare that the project report entitled **Covid-19 day to day analysis and prediction** done by me under the guidance of **Dr. Joshila Grace** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

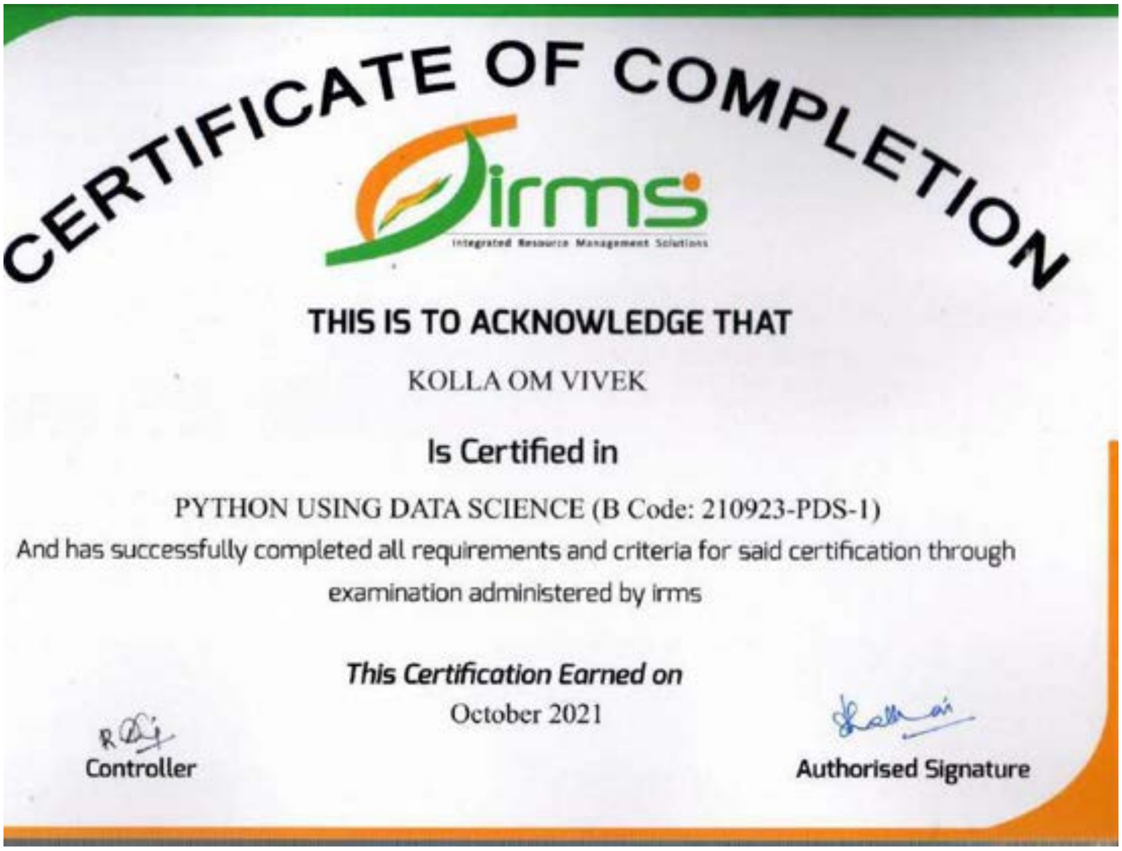
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D**, **Dean**, School of Computing, **Dr. S. Vigneshwari, M.E., Ph.D. and Dr. L. Lakshmanan, M.E., Ph.D., Heads of the Department** of **Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. joshila grace** for his valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



CERTIFICATE OF COMPLETION

irms

Integrated Resource Management Solutions

THIS IS TO ACKNOWLEDGE THAT

KOLLA OM VIVEK

Is Certified in

PYTHON USING DATA SCIENCE (B Code: 210923-PDS-1)

And has successfully completed all requirements and criteria for said certification through examination administered by irms

This Certification Earned on

October 2021

[illegible]

Controller

[illegible]

Authorised Signature

TABLE OF CONTENTS

ABSTRACT

The world is suffering from a pandemic called COVID-19, caused by the SARS-CoV-2 virus. So, I decided to do a project on visualizing the data of Covid-19.

In Covid-19 day to day analysis application one can be able to understand the trends of corona virus by data visualization for all the different countries individually, all the states of India and they can also able to find the prediction of next day Covid-19 cases.

Data science has attracted a lot of attention, promising to turn vast amounts of data into useful predictions and insights. But here, in this project we're going to mainly focus on data visualization because, now a days data analysis is playing a major role in many different fields. These developments include investigating ways of applying visualization and systems for more efficient manipulation, interpretation and presentation of data.

So, here by using visualization we can communicate with the data and analyze the results. The research is to bring the real-world problems to the human capabilities and perception together with computer algorithms, using visualization.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS

ABBREVATION	FULL FORM
COVID -19	Coronavirus Disease 2019
WHO	World Health Organization
NumPy	Numerical Python
MERS	Middle East Respiratory Syndrome
SARS	Severe Acute Respiratory Syndrome
GPL	General Public License
PHP	Hypertext Preprocessor
CPU	Central Processing Unit
GPU	Graphics Processing Unit
RAM	Random Access Memory
SUM	Summation

1. INTRODUCTION

1.1 Covid-19 INTRODUCTION

The World Health Organization (WHO) has declared the coronavirus disease 2019 (COVID-19) a pandemic. A global coordinated effort is needed to stop the further spread of the virus. A pandemic is defined as "occurring over a wide geographic area and affecting an exceptionally high proportion of the population." The last pandemic reported in the world was the H1N1 flu pandemic in 2009.

On 31 December 2019, a cluster of cases of pneumonia of unknown cause, in the city of Wuhan, Hubei province in China, was reported to the World Health Organization. In January 2020, a previously unknown new virus was identified, subsequently named the 2019 novel coronavirus, and samples obtained from cases and analysis of the virus' genetics indicated that this was the cause of the outbreak. This novel coronavirus was named Coronavirus Disease 2019 (COVID-19) by WHO in February 2020. The virus is referred to as SARS -CoV-2 and the associated disease is COVID-19.

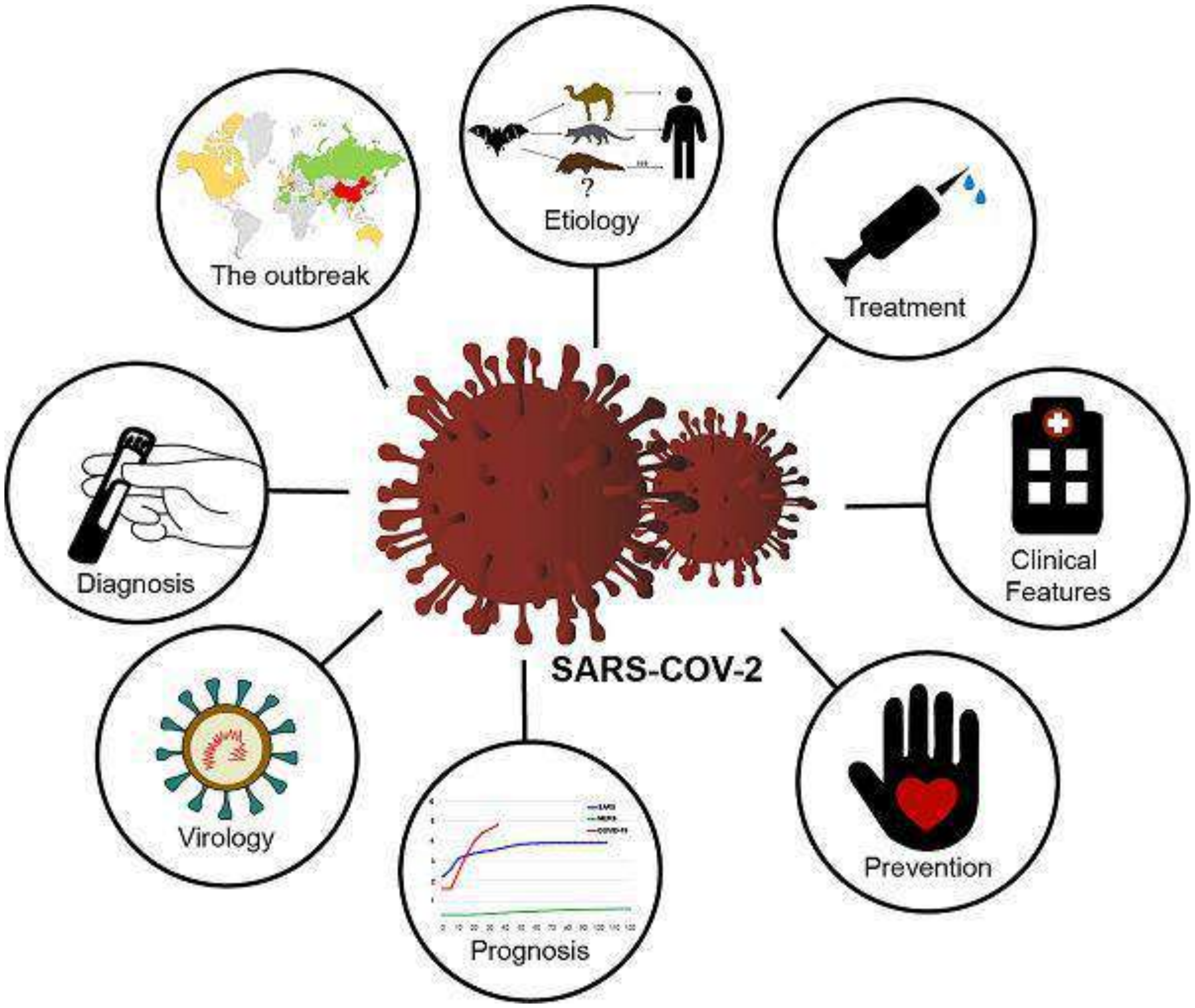


Fig 1.1. SARS-COV-2

Symptoms may include:

- Respiratory symptoms
- Fever
- Cough
- Shortness of breath
- Breathing difficulties
- Fatigue
- Sore throat

Preventions for covid:

- Wear a mask properly
- Make your environment safer
- Keep good hygiene
- Wash your hands properly and regularly

What to do if you feel unwell:

If you feel unwell, here’s what to do.

- If you have a fever, cough and difficulty breathing, seek medical attention immediately. Call by telephone first and follow the directions of your local health authority.
- Know the full range of symptoms of COVID-19. The most common symptoms of COVID-19 are fever, dry cough, tiredness and loss of taste or smell. Less common symptoms include aches and pains, headache, sore throat, red or irritated eyes, diarrhoea, a skin rash or discolouration of fingers or toes.
- Stay home and self-isolate for 10 days from symptom onset, plus three days after symptoms cease. Call your health care provider or hotline for advice. Have someone bring you supplies. If you need to leave your house or have someone near you, wear a properly fitted mask to avoid infecting others.
- Keep up to date on the latest information from trusted sources, such as WHO or your local and national health authorities. Local and national authorities and

public health units are best placed to advise on what people in your area should be doing to protect themselves

1.2 PYTHON INTRODUCTION

Python is primarily a programming language but is used in the field of data due to its versatility of functionalities from a mathematical and statistical perspective. Along with this, its processing speed, accessibility and syntax have made it super popular. It is not only a free software, but it also allows users to develop their own packages and libraries that others can reuse. And even if an individual has not coded in their entire life, they would be able to pick up Python in no time. The syntax used in this language is intuitive and easy to understand.

Python is a powerful tool to perform data science. From cleaning null values to wrangling data for a detailed analysis to visualizing algorithms, this tool is unlike any other.

Python for data visualisation – One of the most important tasks for a data scientist is to create visual representations of the analysis that has been done. It helps understand and communicate the results better. Matplotlib package in Python is the genie for all data visualisations.

1.3 PYTHON LIBRARIES

Pandas is designed for quick and easy data manipulation, reading, aggregation, and visualization.

- Index, manipulate, rename, sort, merge data frames
- Update, Add, Delete columns from a data frame
- Impute missing files, handle missing data or NANs
- Plot data with histogram or box plot

NumPy is used to facilitate math operations on arrays and their vectorisation. This significantly enhances performance and speeds up the execution time on a dataset. With this library.

- Perform basic array operations like add, multiply, slice, flatten, reshape, index arrays
- Perform advanced array operations like stack arrays, split into sections, broadcast arrays
- Perform basic slicing and advanced indexing

Matplotlib is to create stories with visualisations. Plotting 2D figures of Histogram, bar plots, scatter plots, area plot to pie plot, Matplotlib can depict a wide range of visualisations. This is what makes it a versatile library.

Seaborn is an extension of Matplotlib with advanced features. It is different from Matplotlib for its variety of visualisation patterns and lesser syntax.

- Determine relationships between multiple variables
- Observe categorical variables for aggregate statistics
- Analyse univariate or bivariate distributions, plot regression models
- Provide high-level abstractions, multi-plot grids

TensorFlow is an AI library that helps developers to create large-scale neural networks with many layers using data flow graphs.

- Voice/sound recognition
- Sentiment analysis on CRM or CX data
- Face Recognition
- Time series analysis on datasets from organisations like Amazon and Google

Keras, like TensorFlow, is for building and training deep neural network code; but here statistical modelling and working with images and text is a lot easier. Keras is exclusively for neural networks whereas TensorFlow is for various machine learning tasks.

Regular Expressions is a special sequence of characters that uses a search pattern to find a string or set of strings

   – It can detect the presence or absence of a text by matching with a particular pattern, and also can split a pattern into one or more sub-patterns.
   – Python provides re module that supports the use of regex in Python.
   – Its primary function is to offer a search, where it takes a regular expression and a string.

Time module provides many ways of representing time in code, such as objects, numbers, and strings

   – It also provides functionality other than representing time
      - like waiting during code execution and measuring the efficiency of your code.

Pywebio   provides a diverse set of imperative functions to obtain user input and output content on the browser, turning the browser into a “rich text terminal”, and can be used to build simple web applications or browser-based GUI applications.

Pygame library is an open-source module for the Python programming language specifically intended to help you make games and other multimedia applications.

- Built on top of the highly portable SDL (Simple Direct Media Layer) development library, pygame can run across many platforms and operating systems.

Plotly library is an interactive, open-source plotting library that supports over 40 unique chart types covering a wide range of statistical, financial, geographic, scientific, and 3-dimensional use-cases.

1.4 DATA SCIENCE INTRODUCTION

Data Science for COVID-19 presents leading-edge research on data science techniques for the detection, mitigation, treatment and elimination of COVID-19. Sections provide an introduction to data science for COVID-19 research, considering past and future pandemics, as well as related Coronavirus variations. Other chapters cover a wide range of Data Science applications concerning COVID-19 research, including Image Analysis and Data Processing, Geoprocessing and tracking, Predictive Systems, Design Cognition, mobile technology, and telemedicine solutions.

- Provides a leading-edge survey of Data Science techniques and methods for research, mitigation and treatment of the COVID-19 virus
- Integrates various Data Science techniques to provide a resource for COVID-19 researchers and clinicians around the world, including both positive and negative research findings
- Provides insights into innovative data-oriented modeling and predictive techniques from COVID-19 researchers

1.5 DATA VISUALIZATION INTRODUCTION

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed.

Data visualization is the graphical representation of data in order to interactively and efficiently convey insights to clients, customers and stakeholders in general.it is a way to summarize your findings and display it in a form that facilitates interpretation and can help in identifying patterns or trends.

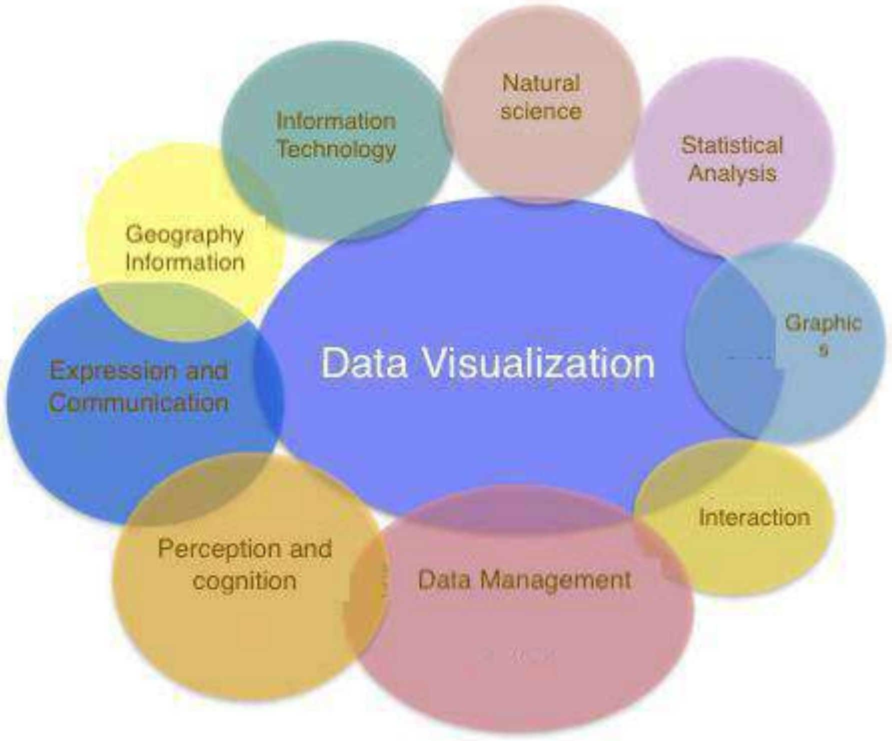


Fig 1.2. description for data visualization

Python offers multiple great graphing libraries that come packed with lots of different features. No matter if you want to create interactive, live or highly customized plots python has an excellent library.

- Matplotlib low level, provides lots of freedom
- Pandas easy to use interface, built on Matplotlib
- seaborn high-level interface, great default styles
- potly can create interactive plots

2. AIM AND SCOPE

2.1 AIM

During the current coronavirus pandemic, monitoring the evolution of COVID-19 cases is of utmost importance for the authorities to make informed policy decisions (e.g., lock-downs), and to raise awareness in the general public for taking appropriate public health measures. At the time of the pandemic outbreak, a lack of laboratory tests, materials, and human resources implied that the evolution of officially confirmed cases did not represent the total number of cases as our country is huge in terms of population, even government is unable to record the cases as a result many cases are out of the records. As the number of cases recorded are less than the actual number of cases there is no much awareness over the world, this costed huge human loss, because of improper records over the pandemic. In some cases health authorities are forced to make important decisions based on sub-optimal data.

To this end, I came up with this project the whole aim of this visualization is to give better understanding to the users that they can be ready for any other variants coming out from covid 19 virus, this visualization is done primarily on graphs and charts because those are best ways to represent huge amount of data, As this covid data is huge it is very much better to go with the visualization than any other techniques, this gives us the best way of visualizing the data.

Here we’re going to Visualize the data of covid-19 of all the countries around the world and all the states of India and also predicting tomorrows Covid-19 cases.

2.2 SCOPE

By the end of this project, you will get to know how the pandemic created great human loss and number of peoples died, and from this investigate you will get clear cut visualized graphs and tables bar plots and many other kinds of charts, and we can easily analyse things using this project we can have hundreds of comparisons between all the countries in the world. By using those comparisons we can get clear information about the countries which are affected by covid.

In this project we are going to use COVID19 dataset we have consisting of the data related to cumulative number of confirmed and deaths cases. we can also answer many questions like: how many countries were affected by covid?, How does the Global Spread of the virus look like?, How intensive the spread of the virus has been in the countries? Does covid19 national lockdowns and self-isolations in different countries have actually impact on COVID19 transmission? we are going to use Plotly, matplotlib, pandas, seaborn module, which is a great visualization tools in python, in order to plot some insightful and intuitive graphs to answer the questions.

we are ending up with best looking graphs and charts which provides the better analyzing capabilities to end users, we are getting those best analyzing graphs and charts by using inbuilt modules like matplotlib, pandas, seaborn

For predicting tomorrows Covid-19 cases we're going to train a machine learning algorithm which can predict whether they are going to have high or low cases than the previous day. This prediction can help government to make decisions.

3. METHOD AND MATERIALS

3.1 HARDWARE AND SOFTWARE REQUIREMENTS

3.1.1 HARDWARE REQUIREMENTS

- Desktop, Keyboard, Mouse, Screen, Hard Drive, GPU, CPU and RAM

3.1.2 SOFTWARE REQUIREMENTS

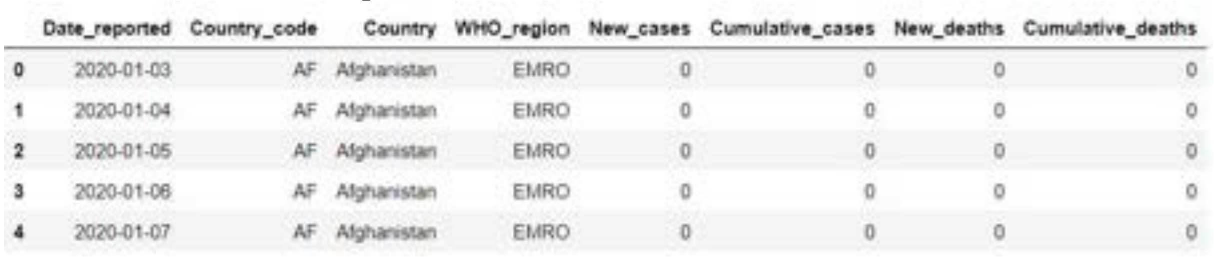
- Operating system: Linux- Ubuntu 16.04 to 17.10, or Windows 7 to 10, with 2GB RAM (4GB preferable)
- install Python 3.6 and related packages
- Jupyter Notebook, IDLE, Atom

3.2 DATA SETS

3.2.1 ALL THE COUNTRIES AROUND THE WORLD

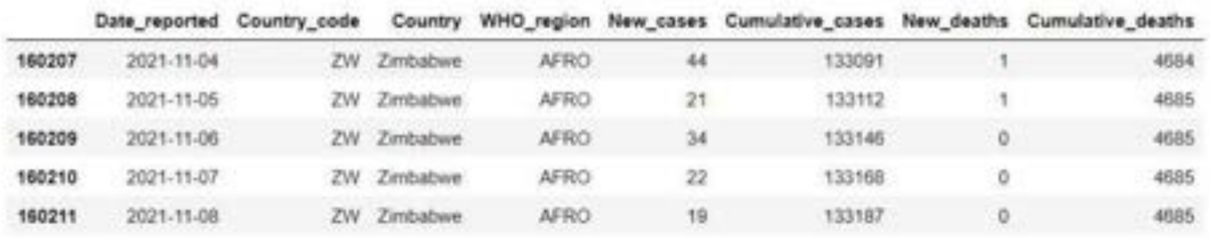
- Here we’re going to get Covid-19 data set from the WHO official website

The dataset contains 8 columns
In 1st column we’re going to have date
In 2nd column we’re having Country code
In 3rd column we’re having country
In 4th column we’re having WHO region
In 5th column we’re having new cases
In 6th column we’re having cumulative cases
In 7th column we’re having new deaths
In 8th column we’re having cumulative deaths



	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
0	2020-01-03	AF	Afghanistan	EMRO	0	0	0	0
1	2020-01-04	AF	Afghanistan	EMRO	0	0	0	0
2	2020-01-05	AF	Afghanistan	EMRO	0	0	0	0
3	2020-01-06	AF	Afghanistan	EMRO	0	0	0	0
4	2020-01-07	AF	Afghanistan	EMRO	0	0	0	0

Fig 3.1.data set head



	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
160207	2021-11-04	ZW	Zimbabwe	AFRO	44	133091	1	4684
160208	2021-11-05	ZW	Zimbabwe	AFRO	21	133112	1	4685
160209	2021-11-06	ZW	Zimbabwe	AFRO	34	133146	0	4685
160210	2021-11-07	ZW	Zimbabwe	AFRO	22	133168	0	4685
160211	2021-11-08	ZW	Zimbabwe	AFRO	19	133187	0	4685

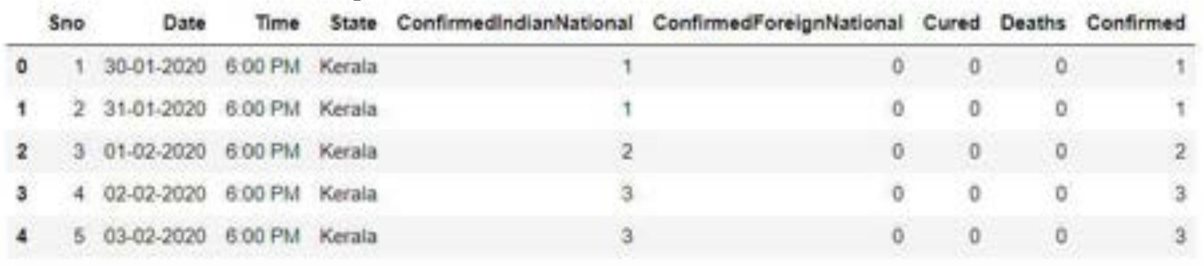
Fig 3.2.data set tail

3.2.2 ALL THE STATES OF INDIA

- Here we’re going to get Covid-19 data set from the kaggle official website

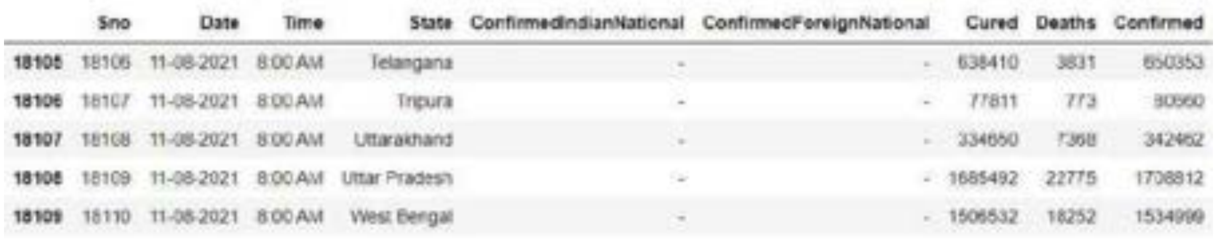
The dataset contains 9 columns
In 1st column we’re going to have sno
In 2nd column we’re having date
In 3rd column we’re having time
In 4th column we’re having state
In 5th column we’re having confirmed Indian national
In 6th column we’re having confirmed foreign national

In 7th column we're having cured
In 8th column we're having deaths
In 9th column we're having confirmed



	Sno	Date	Time	State	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
0	1	30-01-2020	6:00 PM	Kerala	1	0	0	0	1
1	2	31-01-2020	6:00 PM	Kerala	1	0	0	0	1
2	3	01-02-2020	6:00 PM	Kerala	2	0	0	0	2
3	4	02-02-2020	6:00 PM	Kerala	3	0	0	0	3
4	5	03-02-2020	6:00 PM	Kerala	3	0	0	0	3

Fig 3.3.data set head



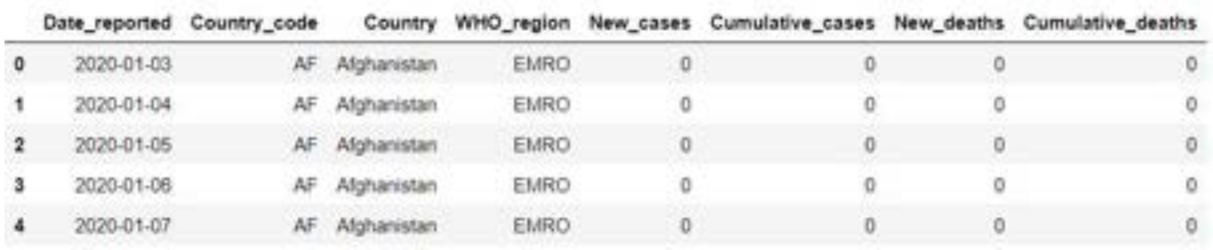
	Sno	Date	Time	State	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Confirmed
18105	18106	11-08-2021	8:00 AM	Telangana	-	-	638410	3831	650353
18106	18107	11-08-2021	8:00 AM	Tripura	-	-	77811	773	80660
18107	18108	11-08-2021	8:00 AM	Uttarakhand	-	-	334650	7368	342462
18108	18109	11-08-2021	8:00 AM	Uttar Pradesh	-	-	1685492	22775	1708812
18109	18110	11-08-2021	8:00 AM	West Bengal	-	-	1506532	18252	1534999

Fig 3.4.data set tail

3.2.3 PREDICTING TOMORROW CASES

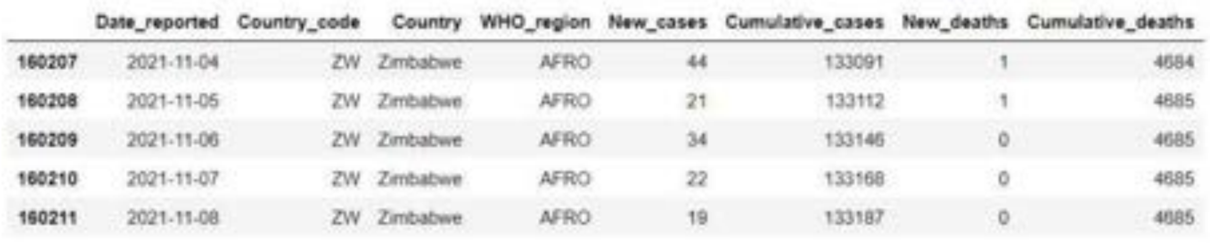
- Here we're going to get Covid-19 data set from the WHO official website

The dataset contains 8 columns
In 1st column we're going to have date
In 2nd column we're having Country code
In 3rd column we're having country
In 4th column we're having WHO region
In 5th column we're having new cases
In 6th column we're having cumulative cases
In 7th column we're having new deaths
In 8th column we're having cumulative deaths



	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
0	2020-01-03	AF	Afghanistan	EMRO	0	0	0	0
1	2020-01-04	AF	Afghanistan	EMRO	0	0	0	0
2	2020-01-05	AF	Afghanistan	EMRO	0	0	0	0
3	2020-01-06	AF	Afghanistan	EMRO	0	0	0	0
4	2020-01-07	AF	Afghanistan	EMRO	0	0	0	0

Fig 3.5.data set head



	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
160207	2021-11-04	ZW	Zimbabwe	AFRO	44	133091	1	4684
160208	2021-11-05	ZW	Zimbabwe	AFRO	21	133112	1	4685
160209	2021-11-06	ZW	Zimbabwe	AFRO	34	133146	0	4685
160210	2021-11-07	ZW	Zimbabwe	AFRO	22	133168	0	4685
160211	2021-11-08	ZW	Zimbabwe	AFRO	19	133187	0	4685

Fig 3.6.data set tail

3.3 DATA PROCESSING

Data processing occurs when data is collected and translated into usable information. Usually performed by a data scientist or team of data scientists, it is important for data processing to be done correctly as not to negatively affect the end product, or data output.
Data processing starts with data in its raw form and converts it into a more readable format (graphs, documents, etc.)

We’re going to use six stages for data processing

1. Data collection
2. Data preparation
3. Data input
4. Processing
5. Data output/interpretation
6. Data storage

Data collection: Collecting data is the first step in data processing. Data is pulled from available sources. It is important that the data sources available are trustworthy and well-built so the data collected (and later used as information) is of the highest possible quality.
data presentation: Once the data is collected, it then enters the data preparation stage. Data preparation, often referred to as “pre-processing” is the stage at which raw data is cleaned up and organized for the following stage of data processing. During preparation, raw data is diligently checked for any errors. The purpose of this step is to eliminate bad data (incomplete, or incorrect data) and begin to create high-quality data.

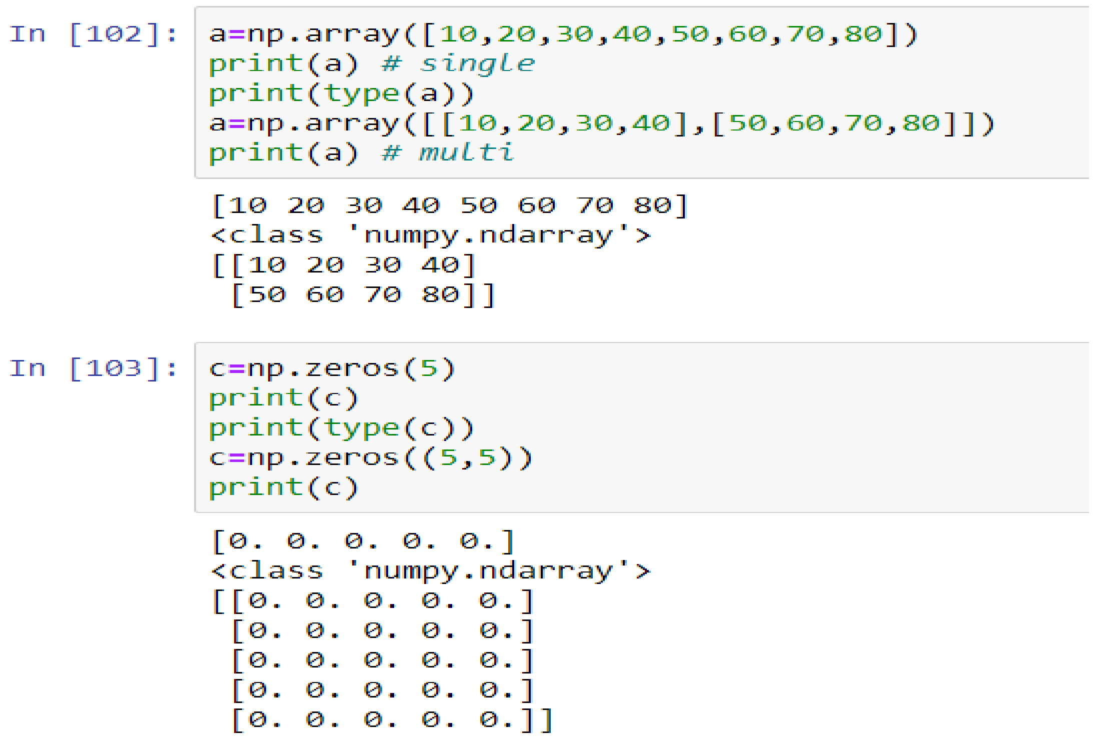
Data input: The clean data is then entered into its destination, and translated into a language that it can understand. Data input is the first stage in which raw data begins to take the form of usable information.

Processing: During this stage, the data inputted to the computer in the previous stage is actually processed for interpretation. Processing is done using machine learning algorithms, though the process itself may vary slightly depending on the source of data being processed (data lakes, social networks, connected devices etc.) and its intended use (examining advertising patterns, medical diagnosis from connected devices, determining customer needs, etc.).

Data output/interpretation: The output/interpretation stage is the stage at which data is finally usable to non-data scientists. It is translated, readable, and often in the form of graphs, videos, images, plain text, etc.). Members of the company or institution can now begin to self-service the data for their own data analytics projects.

Data storage: The final stage of data processing is storage. After all of the data is processed, it is then stored for future use. While some information may be put to use immediately, much of it will serve a purpose later on.

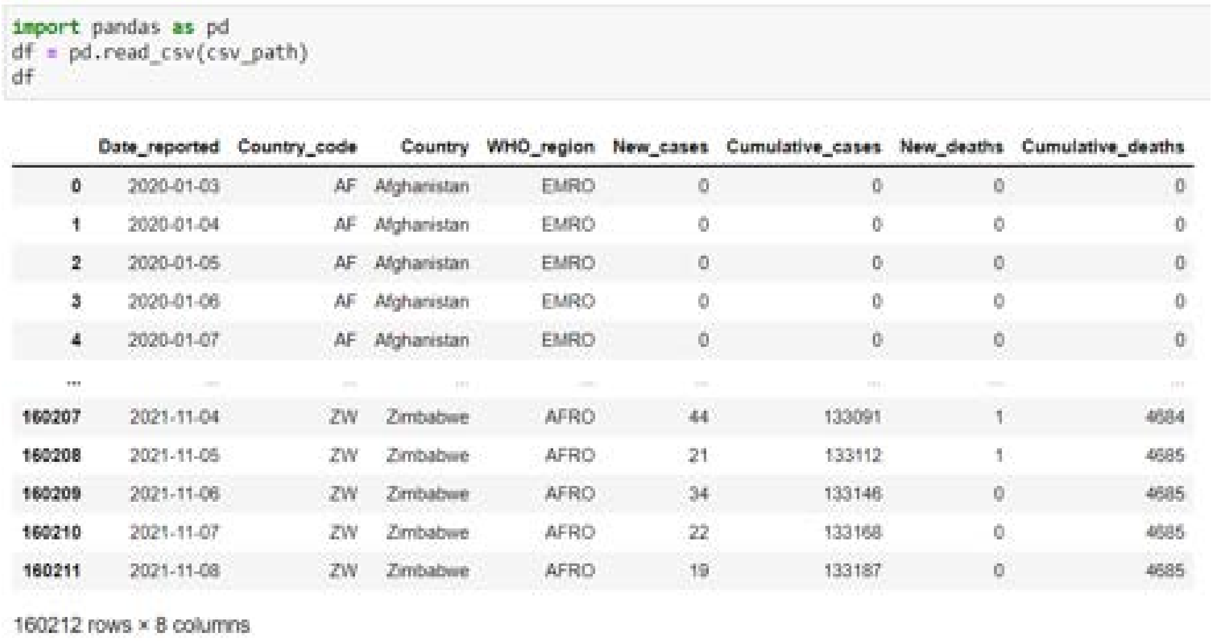
- **NumPy:** NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed. NumPy is a Python package. It stands for 'Numerical Python'. NumPy is used in many ways for visualizing data in which we can slice the list e.t.c.



```
In [102]: a=np.array([10,20,30,40,50,60,70,80])
          print(a) # single
          print(type(a))
          a=np.array([[10,20,30,40],[50,60,70,80]])
          print(a) # multi

          [10 20 30 40 50 60 70 80]
          <class 'numpy.ndarray'>
          [[10 20 30 40]
           [50 60 70 80]]

In [103]: c=np.zeros(5)
          print(c)
          print(type(c))
          c=np.zeros((5,5))
          print(c)

          [0. 0. 0. 0. 0.]
          <class 'numpy.ndarray'>
          [[0. 0. 0. 0. 0.]
           [0. 0. 0. 0. 0.]
           [0. 0. 0. 0. 0.]
           [0. 0. 0. 0. 0.]
           [0. 0. 0. 0. 0.]]
```

Fig 3.7. functionalities of NumPy

- **Pandas:** Pandas is an opensource Python package that is most widely used for data science/data analysis and machine learning tasks. It is built on top of another package named NumPy, which provides support for multi-dimensional arrays.



```python
import pandas as pd
df = pd.read_csv(csv_path)
df
```

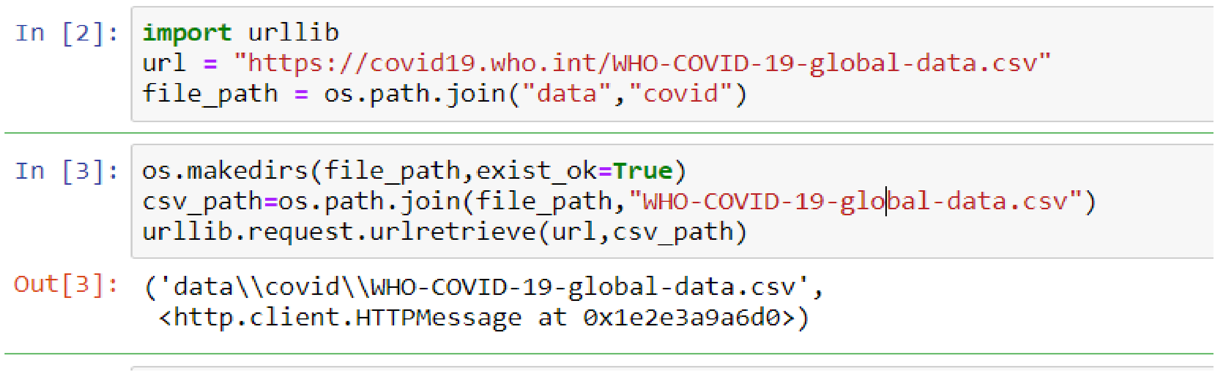
```
        Date_reported Country_code     Country WHO_region New_cases Cumulative_cases New_deaths Cumulative_deaths
     0     2020-01-03           AF Afghanistan       EMRO         0                0          0                 0
     1     2020-01-04           AF Afghanistan       EMRO         0                0          0                 0
     2     2020-01-05           AF Afghanistan       EMRO         0                0          0                 0
     3     2020-01-06           AF Afghanistan       EMRO         0                0          0                 0
     4     2020-01-07           AF Afghanistan       EMRO         0                0          0                 0
   ...            ...          ...         ...        ...       ...              ...        ...               ...
160207     2021-11-04           ZW    Zimbabwe       AFRO        44           133091          1              4684
160208     2021-11-05           ZW    Zimbabwe       AFRO        21           133112          1              4685
160209     2021-11-06           ZW    Zimbabwe       AFRO        34           133146          0              4685
160210     2021-11-07           ZW    Zimbabwe       AFRO        22           133168          0              4685
160211     2021-11-08           ZW    Zimbabwe       AFRO        19           133187          0              4685

160212 rows × 8 columns
```

Table 3.8. Reads the csv file

3.4 DATA VISUALIZATION

Data visualization in python is perhaps one of the most utilized features for data science with python in today’s day and age. The libraries in python come with lots of different features that enable users to make highly customized, elegant, and interactive plots.

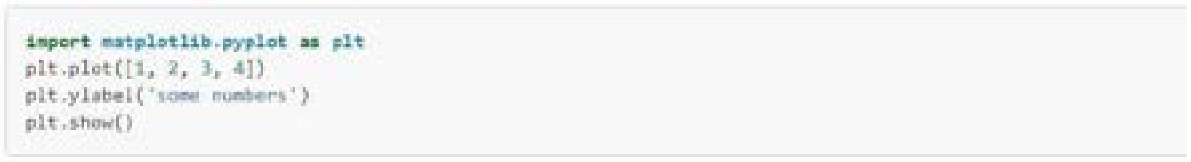
- **Urllib:** Urllib package is the URL handling module for python. It is used to fetch URLs (Uniform Resource Locators). It uses the url open function and is able to fetch URLs using a variety of different protocols.



```
In [2]: import urllib
        url = "https://covid19.who.int/WHO-COVID-19-global-data.csv"
        file_path = os.path.join("data","covid")

In [3]: os.makedirs(file_path,exist_ok=True)
        csv_path=os.path.join(file_path,"WHO-COVID-19-global-data.csv")
        urllib.request.urlretrieve(url,csv_path)

Out[3]: ('data\\covid\\WHO-COVID-19-global-data.csv',
         <http.client.HTTPMessage at 0x1e2e3a9a6d0>)
```

Fig 3.9. Functionalities of urllib module

- **Matplotlib:** Matplotlib is a visualization library in Python for 2D plots of arrays. Matplotlib is written in Python and makes use of the NumPy library. It can be used in Python and IPython shells, Jupyter notebook, and web application servers. Matplotlib comes with a wide variety of plots like line, bar, scatter, histogram, etc.



```
import matplotlib.pyplot as plt
plt.plot([1, 2, 3, 4])
plt.ylabel('some numbers')
plt.show()
```

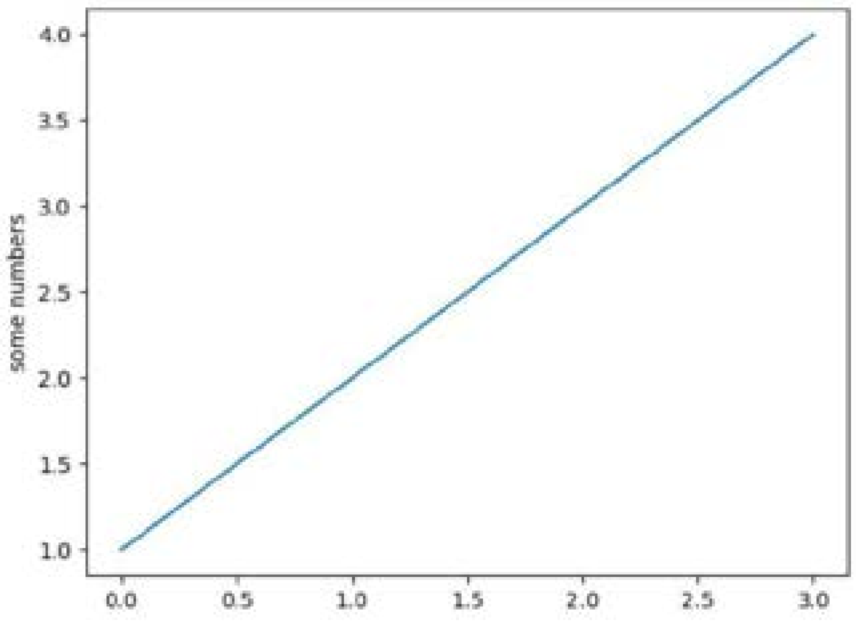


Fig 3.10. matplotlib functionalities

- **Seaborn:** Seaborn is a dataset-oriented library for making statistical representations in Python. It is developed atop matplotlib and to create different visualizations. It is integrated with pandas data structures. The library internally performs the required mapping and aggregation to create informative visuals It is recommended to use a Jupyter/IPython interface in matplotlib mode.

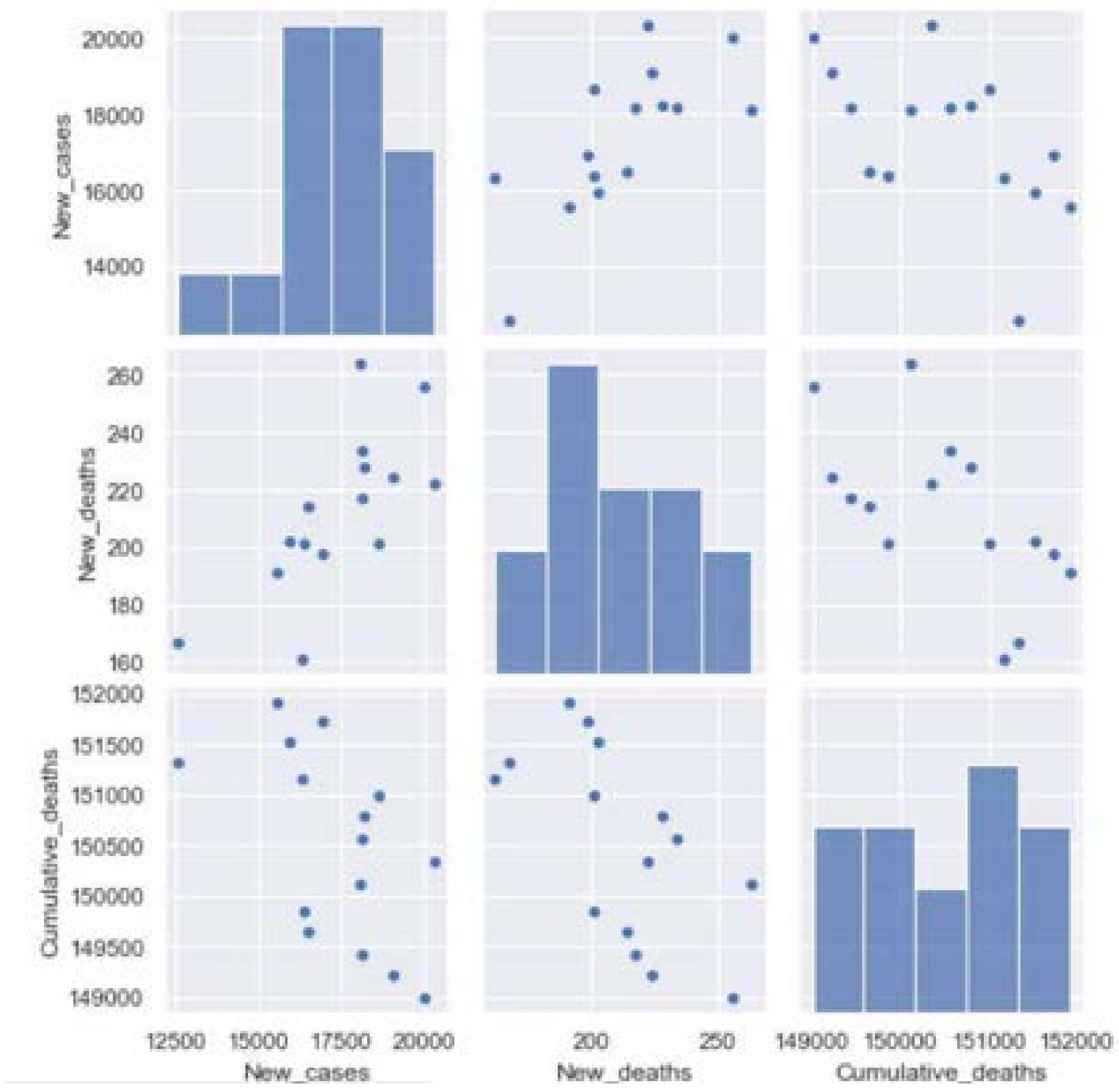


Fig 3.11. Functionalities of seaborn module

- **Plotly:** plotly.py is an interactive, open-source, high-level, declarative, and browser-based visualization library for Python. It holds an array of useful visualization which includes scientific charts, 3D graphs, statistical charts, financial charts among others. Plotly graphs can be viewed in Jupyter notebooks

3.5 reinforcement (ml algorithm)

Reinforcement learning is the training of machine learning models to make a sequence of decisions. The agent learns to achieve a goal in an uncertain, potentially complex environment. In reinforcement learning, an artificial intelligence faces a game-like situation. The computer employs trial and error to come up with a solution to the problem. To get the machine to do what the programmer wants, the artificial intelligence gets either rewards or penalties for the actions it performs. Its goal is to maximize the total reward. Although the designer sets the reward policy–that is, the rules of the game–he gives the model no hints or suggestions for how to solve the game. It's up to the model to figure out how to perform the task to maximize the reward, starting from totally random trials and finishing with sophisticated tactics and superhuman skills. By leveraging the power of search and many trials, reinforcement learning is currently the most effective way to hint machine's creativity. In contrast to human beings, artificial intelligence can gather experience from thousands of parallel gameplays if a reinforcement learning algorithm is run on a sufficiently powerful computer infrastructure.
Now, by using reinforcement ml algorithm we're going to train the machine

3.6 methodology

- project is carried out to achieve a set of goals with some conditions keeping in mind that it is easy to understand
- for easy understanding, I used graph
- plots have been used in the project for visualizing the collected data

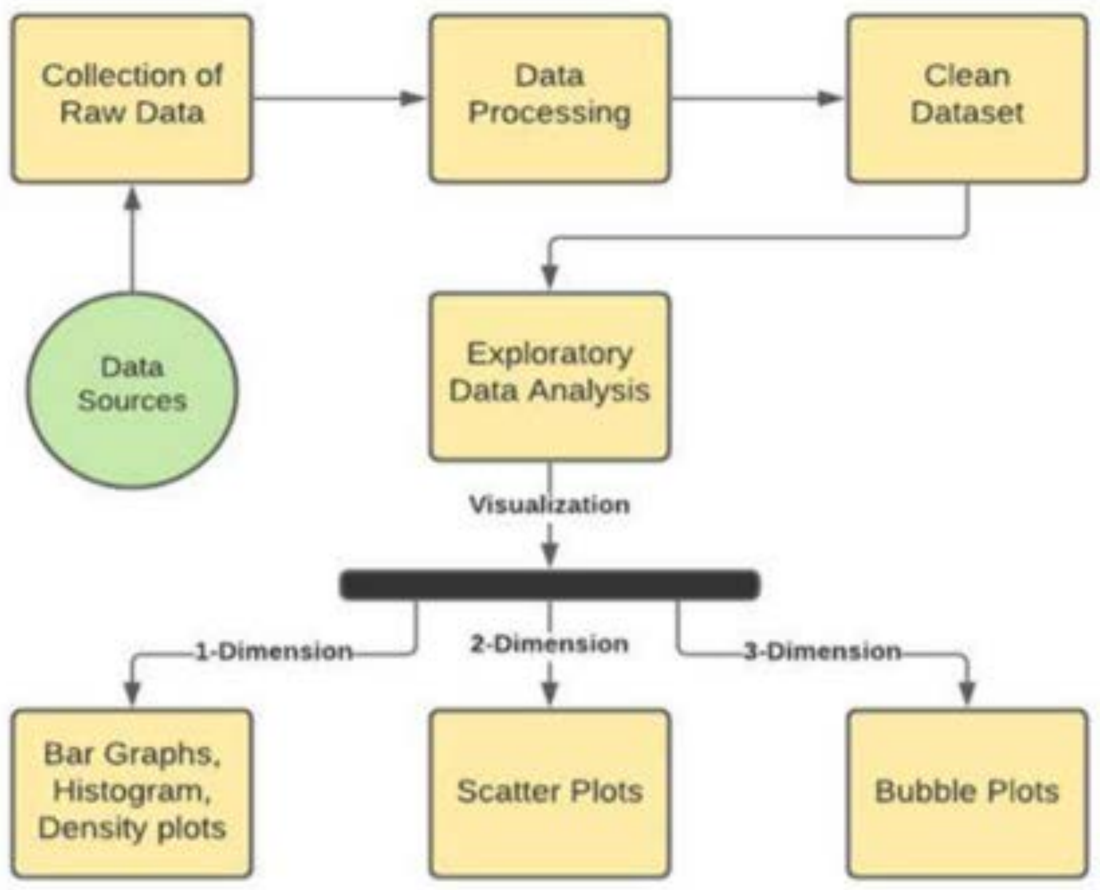


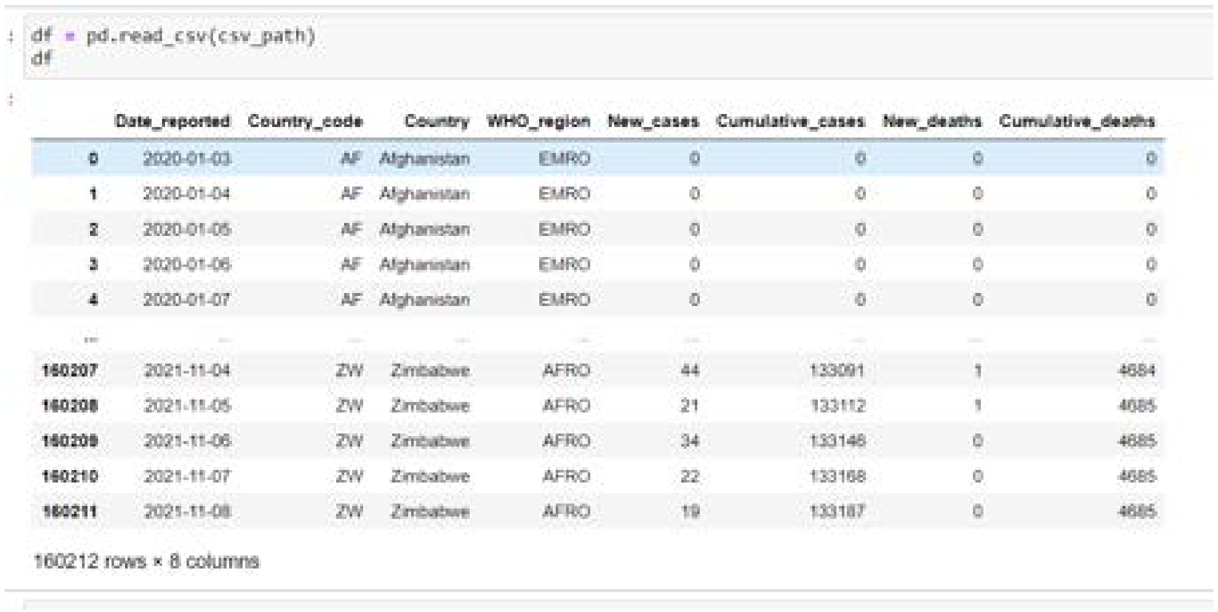
Fig 3.12: Data Visualization Methodology

- steps followed
  - **Step 1:** Import the needed modules.
  - **Step 2:** Get your dataset and read it.
  - **Step 3:** using numpy and pandas clean the data
  - **Step 4:** Once the data is clean now we are good to go with the visualization part.
  - **Step 5:** Start visualizing the data by considering different columns.
  - **Step 6:** Plot the graphs with the help of data and modules imported
  - **Step 7:** Generating the results in the pywebio application

- **Step 8:** Once visualizing the data is completed start analyzing the data.
- **Step 9:** predicting tomorrows Covid-19 cases

4. RESULTS AND DISCUSSION, PERFORMANCE ANALYSIS

The tabular format which is shown below, states that its reading the csv file using pandas and storing it in variable “df” and print the same



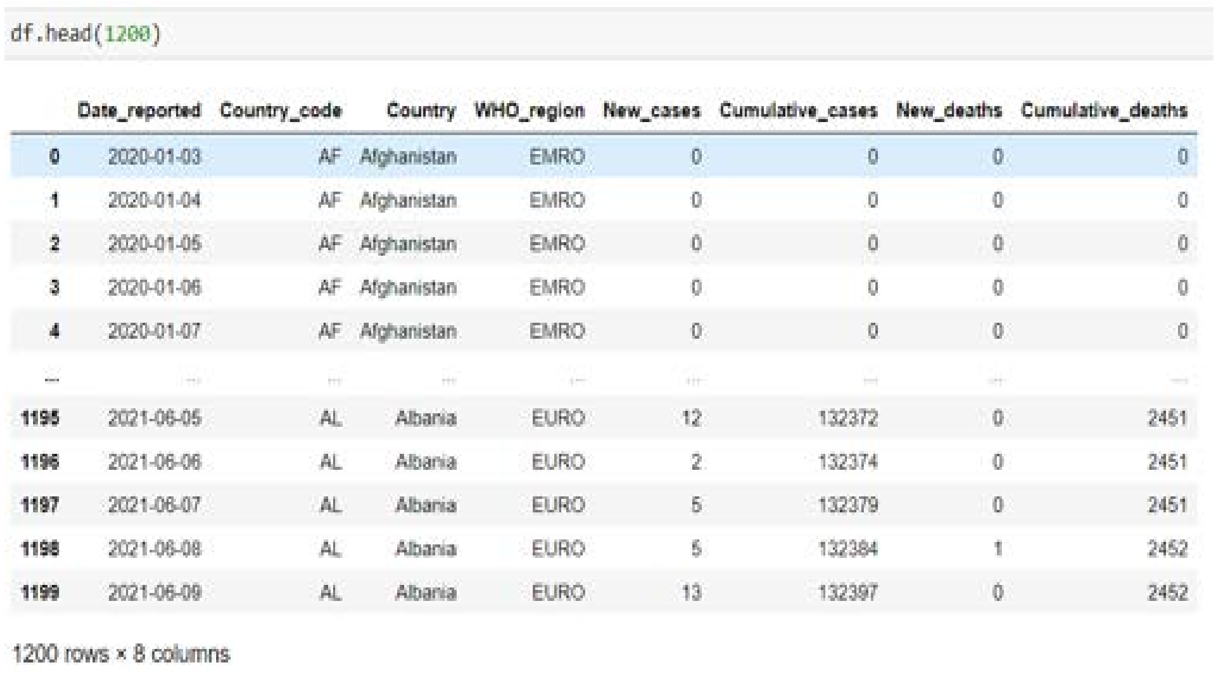
```
df = pd.read_csv(csv_path)
df
```

	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
0	2020-01-03	AF	Afghanistan	EMRO	0	0	0	0
1	2020-01-04	AF	Afghanistan	EMRO	0	0	0	0
2	2020-01-05	AF	Afghanistan	EMRO	0	0	0	0
3	2020-01-06	AF	Afghanistan	EMRO	0	0	0	0
4	2020-01-07	AF	Afghanistan	EMRO	0	0	0	0
...	...	...	...	...	...	...	...	...
160207	2021-11-04	ZW	Zimbabwe	AFRO	44	133091	1	4684
160208	2021-11-05	ZW	Zimbabwe	AFRO	21	133112	1	4685
160209	2021-11-06	ZW	Zimbabwe	AFRO	34	133146	0	4685
160210	2021-11-07	ZW	Zimbabwe	AFRO	22	133168	0	4685
160211	2021-11-08	ZW	Zimbabwe	AFRO	19	133187	0	4685

160212 rows × 8 columns

Table 4.1. Reads the csv file and print it out

The tabular form which is shown below, uses the inbuilt function head() and that will returns top n rows in the given dataset, where n is user input value



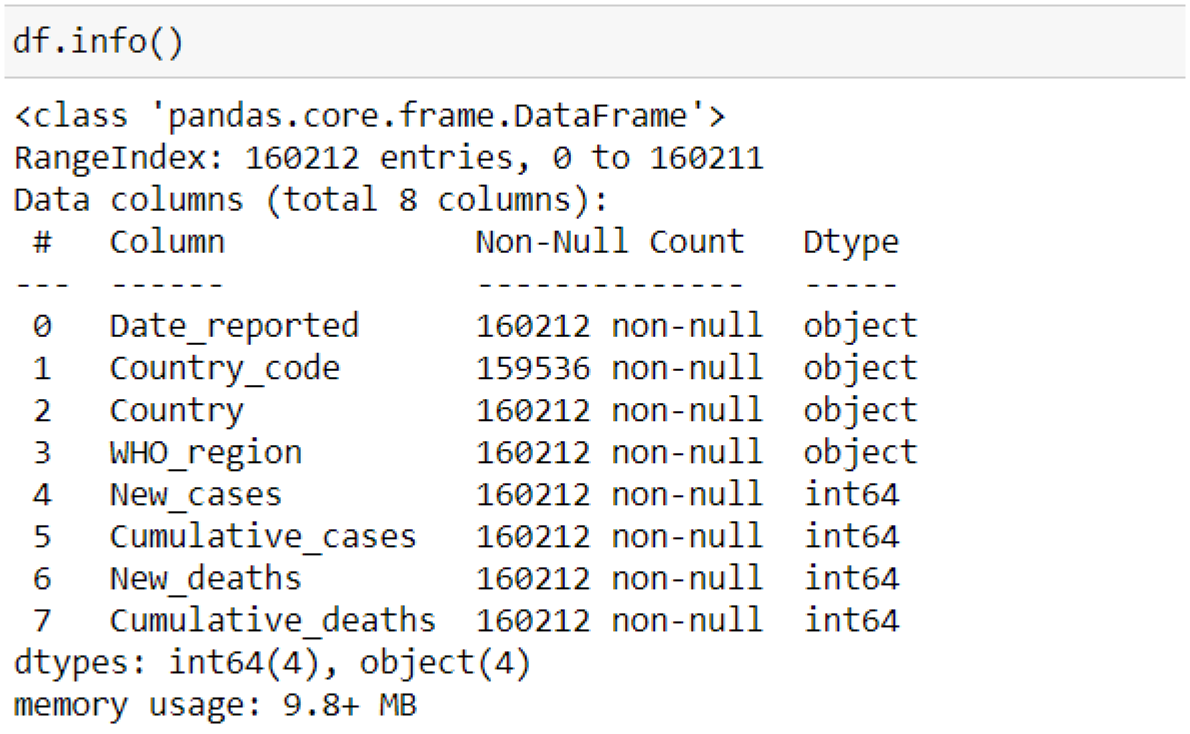
```
df.head(1200)
```

	Date_reported	Country_code	Country	WHO_region	New_cases	Cumulative_cases	New_deaths	Cumulative_deaths
0	2020-01-03	AF	Afghanistan	EMRO	0	0	0	0
1	2020-01-04	AF	Afghanistan	EMRO	0	0	0	0
2	2020-01-05	AF	Afghanistan	EMRO	0	0	0	0
3	2020-01-06	AF	Afghanistan	EMRO	0	0	0	0
4	2020-01-07	AF	Afghanistan	EMRO	0	0	0	0
...	...	...	...	...	...	...	...	...
1195	2021-06-05	AL	Albania	EURO	12	132372	0	2451
1196	2021-06-06	AL	Albania	EURO	2	132374	0	2451
1197	2021-06-07	AL	Albania	EURO	5	132379	0	2451
1198	2021-06-08	AL	Albania	EURO	5	132384	1	2452
1199	2021-06-09	AL	Albania	EURO	13	132397	0	2452

1200 rows × 8 columns

Table 4.2. Returns the top 1200 elements in the dataset

The tabular which is shown below, uses the inbuilt function info() which returns the datatypes of columns present in dataset

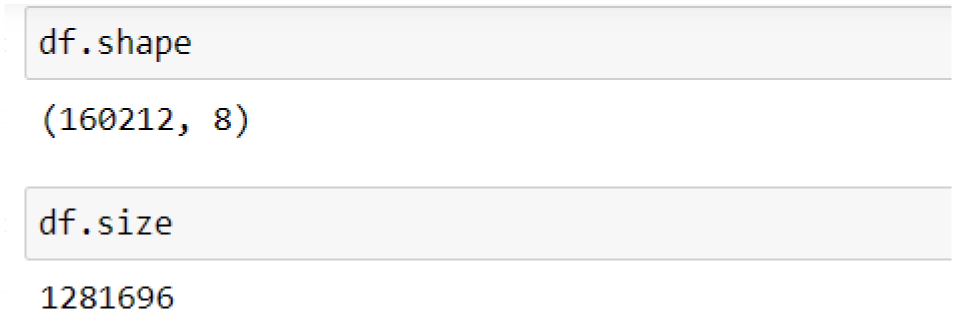


```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 160212 entries, 0 to 160211
Data columns (total 8 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   Date_reported      160212 non-null  object
 1   Country_code       159536 non-null  object
 2   Country            160212 non-null  object
 3   WHO_region         160212 non-null  object
 4   New_cases          160212 non-null  int64
 5   Cumulative_cases   160212 non-null  int64
 6   New_deaths         160212 non-null  int64
 7   Cumulative_deaths  160212 non-null  int64
dtypes: int64(4), object(4)
memory usage: 9.8+ MB
```

Fig 4.3. Information about dataset

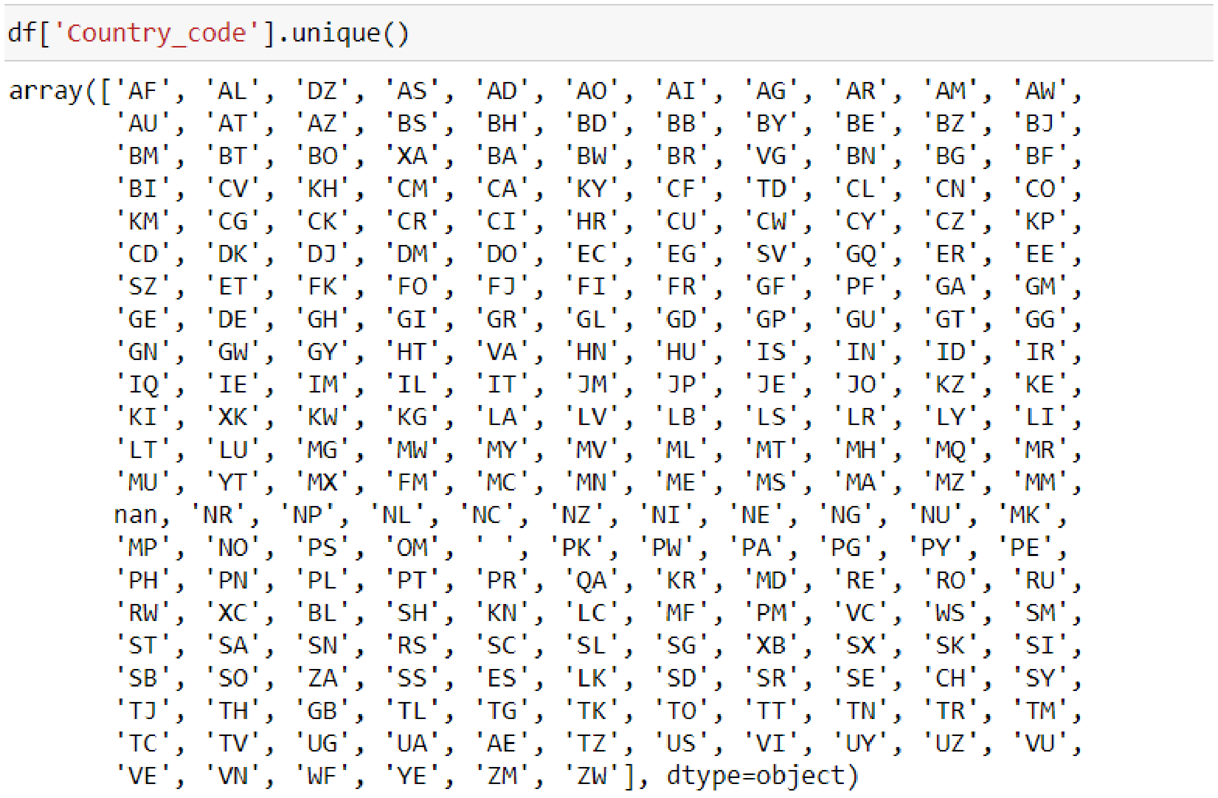
The diagram shown below, uses the functions shape and size to return the size and shape of the dataset



```
df.shape

(160212, 8)

df.size

1281696
```

Fig 4.4. returns the shape and size of dataset

The diagram shown below, uses the keyword “unique” in order to remove the duplicates in given dataset



```
df['Country_code'].unique()

array(['AF', 'AL', 'DZ', 'AS', 'AD', 'AO', 'AI', 'AG', 'AR', 'AM', 'AW',
       'AU', 'AT', 'AZ', 'BS', 'BH', 'BD', 'BB', 'BY', 'BE', 'BZ', 'BJ',
       'BM', 'BT', 'BO', 'XA', 'BA', 'BW', 'BR', 'VG', 'BN', 'BG', 'BF',
       'BI', 'CV', 'KH', 'CM', 'CA', 'KY', 'CF', 'TD', 'CL', 'CN', 'CO',
       'KM', 'CG', 'CK', 'CR', 'CI', 'HR', 'CU', 'CW', 'CY', 'CZ', 'KP',
       'CD', 'DK', 'DJ', 'DM', 'DO', 'EC', 'EG', 'SV', 'GQ', 'ER', 'EE',
       'SZ', 'ET', 'FK', 'FO', 'FJ', 'FI', 'FR', 'GF', 'PF', 'GA', 'GM',
       'GE', 'DE', 'GH', 'GI', 'GR', 'GL', 'GD', 'GP', 'GU', 'GT', 'GG',
       'GN', 'GW', 'GY', 'HT', 'VA', 'HN', 'HU', 'IS', 'IN', 'ID', 'IR',
       'IQ', 'IE', 'IM', 'IL', 'IT', 'JM', 'JP', 'JE', 'JO', 'KZ', 'KE',
       'KI', 'XK', 'KW', 'KG', 'LA', 'LV', 'LB', 'LS', 'LR', 'LY', 'LI',
       'LT', 'LU', 'MG', 'MW', 'MY', 'MV', 'ML', 'MT', 'MH', 'MQ', 'MR',
       'MU', 'YT', 'MX', 'FM', 'MC', 'MN', 'ME', 'MS', 'MA', 'MZ', 'MM',
       nan, 'NR', 'NP', 'NL', 'NC', 'NZ', 'NI', 'NE', 'NG', 'NU', 'MK',
       'MP', 'NO', 'PS', 'OM', ' ', 'PK', 'PW', 'PA', 'PG', 'PY', 'PE',
       'PH', 'PN', 'PL', 'PT', 'PR', 'QA', 'KR', 'MD', 'RE', 'RO', 'RU',
       'RW', 'XC', 'BL', 'SH', 'KN', 'LC', 'MF', 'PM', 'VC', 'WS', 'SM',
       'ST', 'SA', 'SN', 'RS', 'SC', 'SL', 'SG', 'XB', 'SX', 'SK', 'SI',
       'SB', 'SO', 'ZA', 'SS', 'ES', 'LK', 'SD', 'SR', 'SE', 'CH', 'SY',
       'TJ', 'TH', 'GB', 'TL', 'TG', 'TK', 'TO', 'TT', 'TN', 'TR', 'TM',
       'TC', 'TV', 'UG', 'UA', 'AE', 'TZ', 'US', 'VI', 'UY', 'UZ', 'VU',
       'VE', 'VN', 'WF', 'YE', 'ZM', 'ZW'], dtype=object)
```

Fig 4.5. usage of unique keyword

The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-30

Bar Graph

day's VS new_cases's and death's

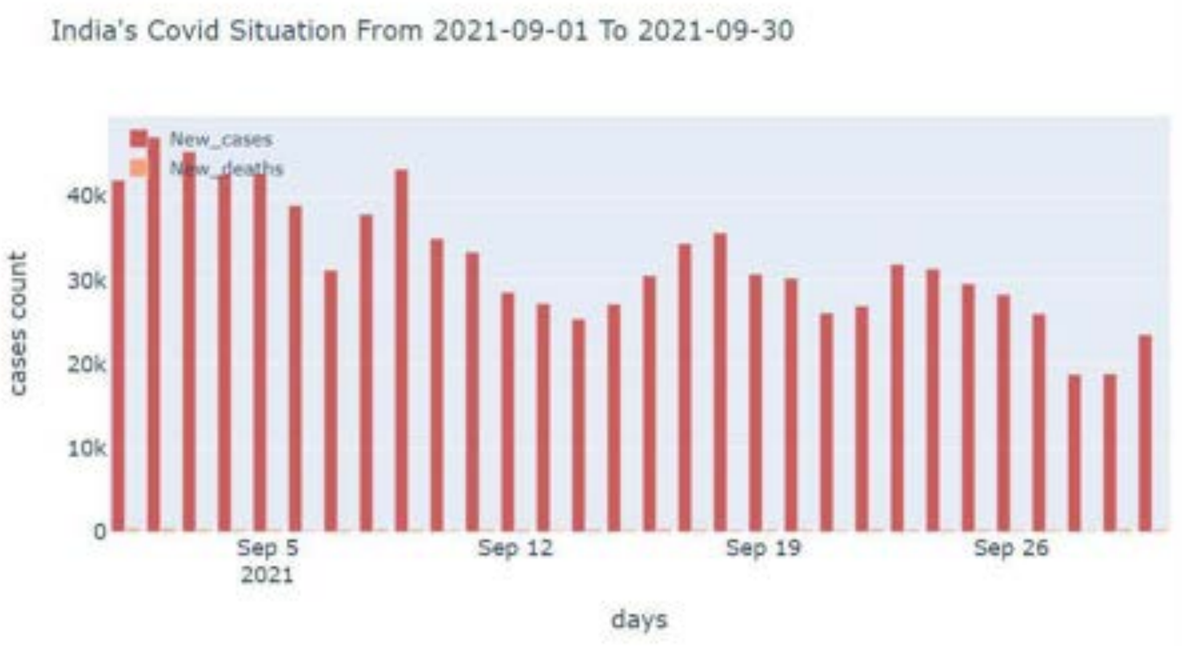


Fig 4.6. bar graph of days vs new cases and new deaths

The diagram shown below is the graph for days vs new cases from 2021-09-01 to2021-09-30

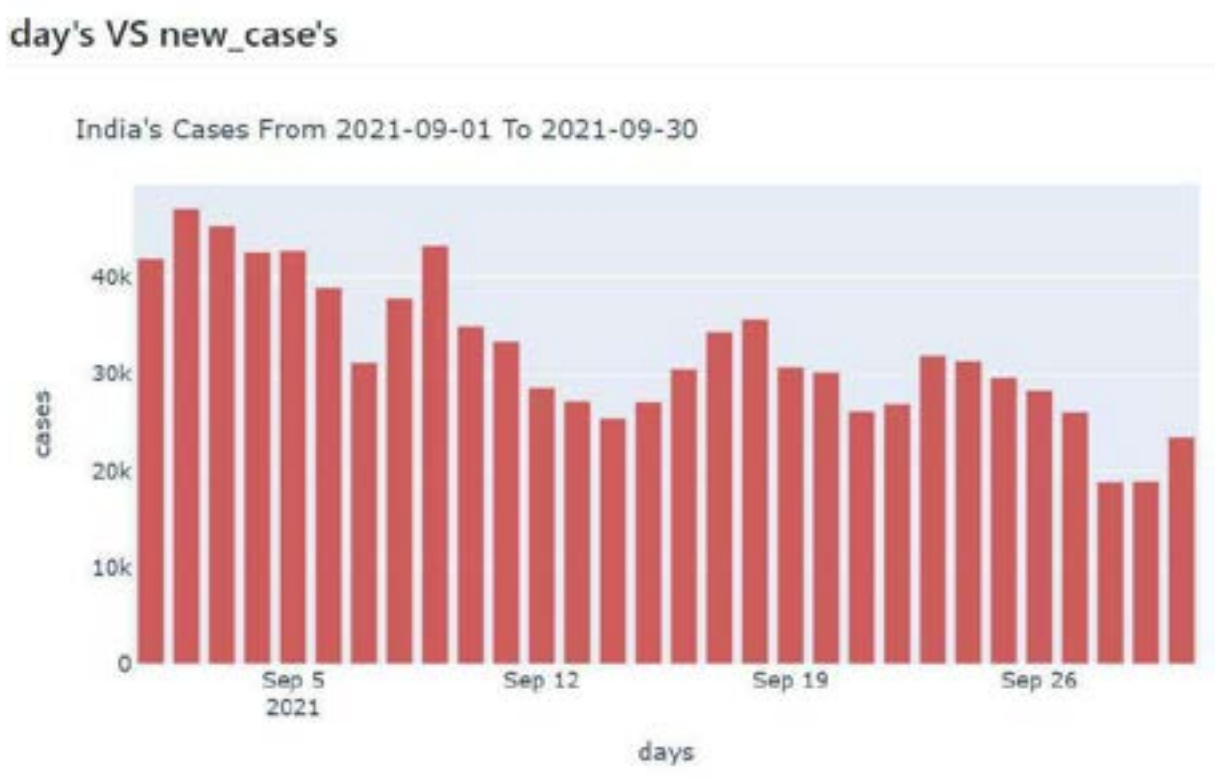


Fig 4.7.bar graph of days vs new cases

The diagram shown below is the graph for days vs new deaths from 2021-09-01 to 2021-09-30

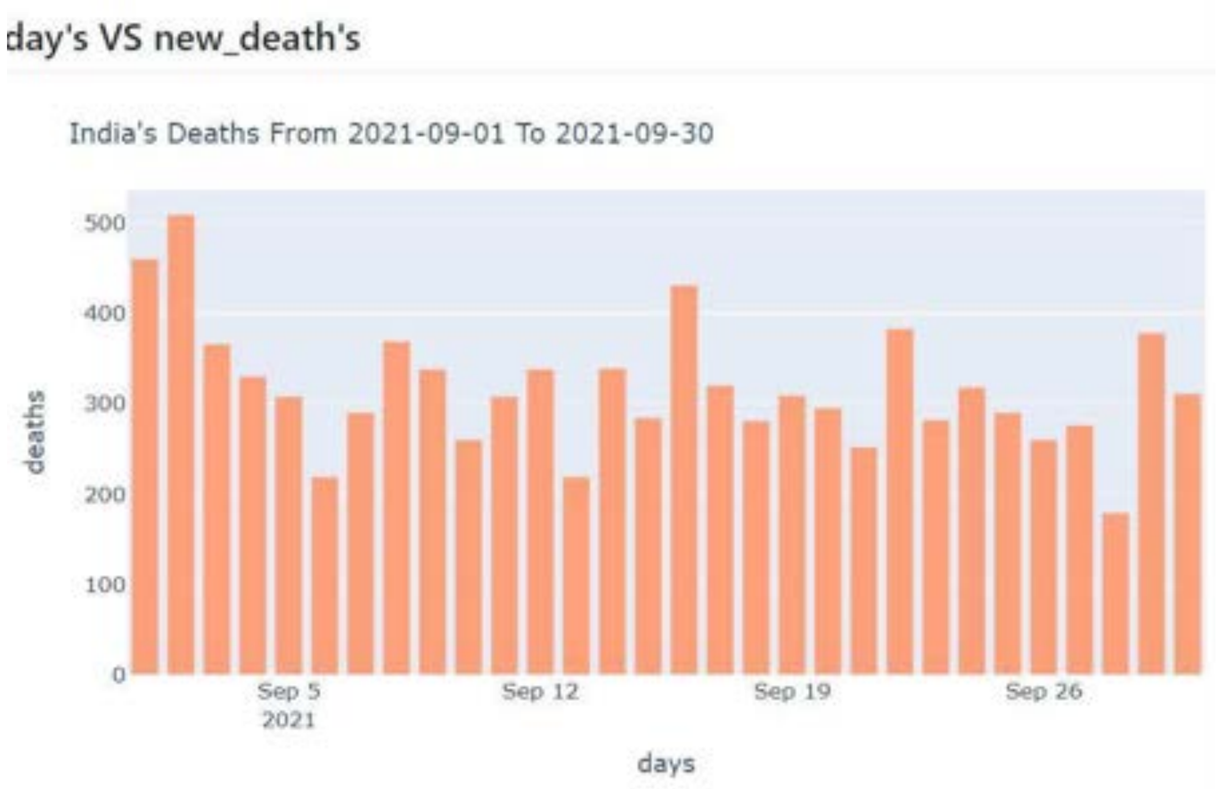


Fig 4.8. bar graph of days vs new deaths

- The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-30

Line Graph

day's VS new_cases's and death's

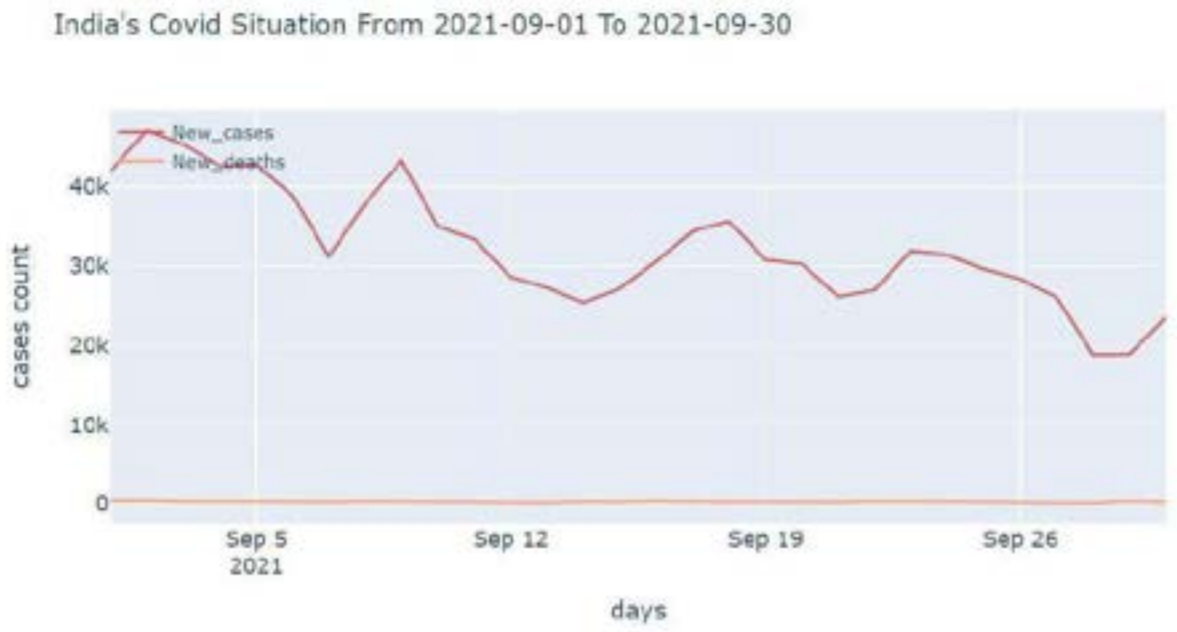


Fig 4.9. line graph of days vs new cases and new deaths

The diagram shown below is the graph for days vs new cases from 2021-09-01 to2021-09-30

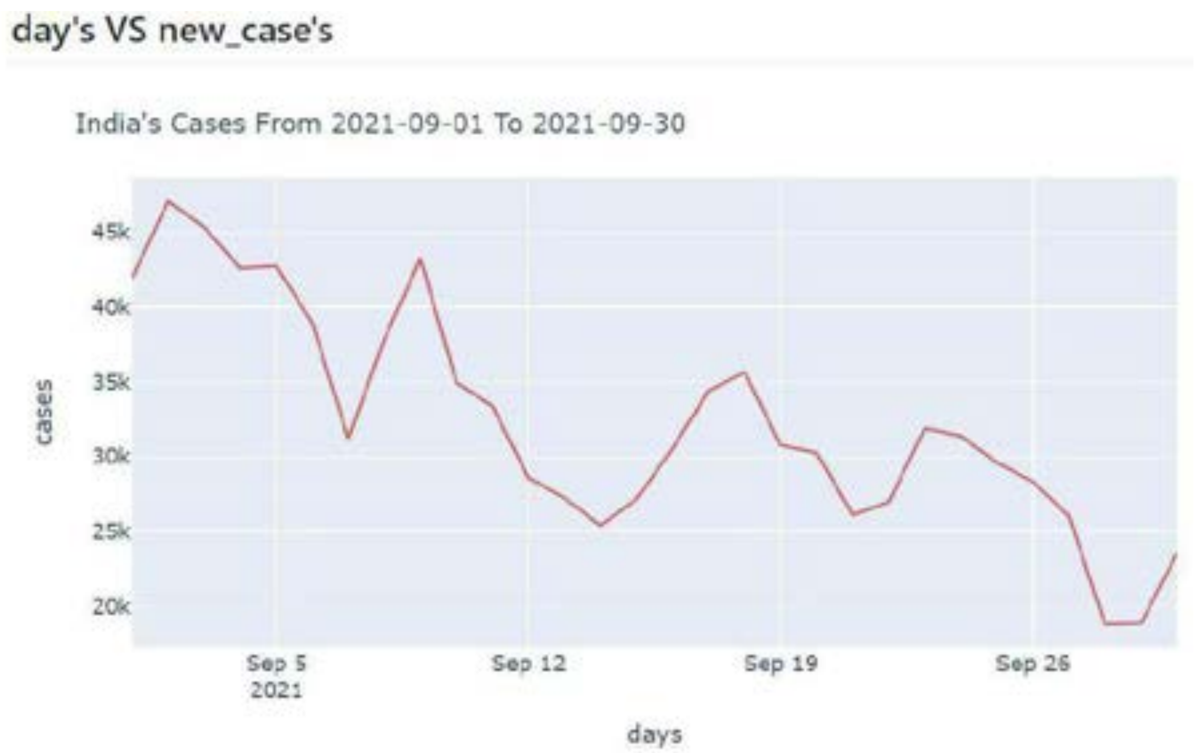


Fig 4.10. line graph of days vs new cases

- ➢ The diagram shown below is the graph for days vs new deaths from 2021-09-01 to 2021-09-30

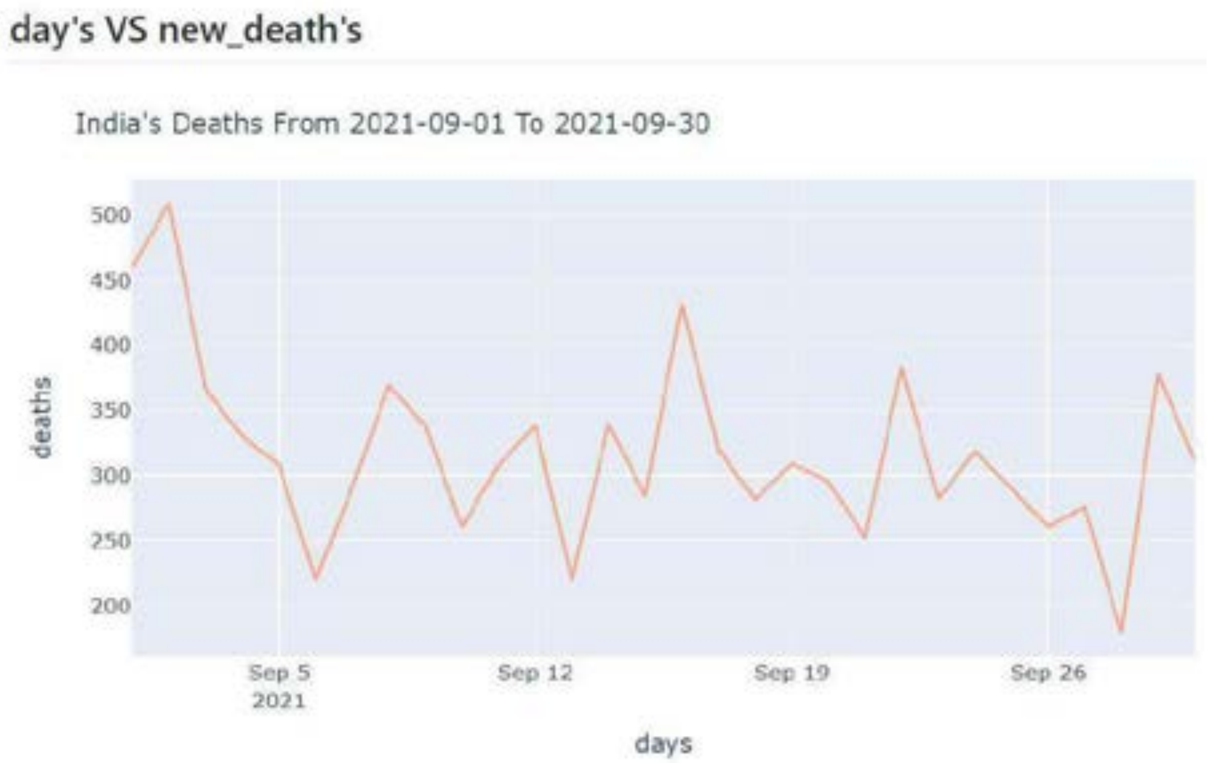


Fig 4.11. line graph of days vs new deaths

- The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-30

Pie Graph

day's VS new_cases's and death's

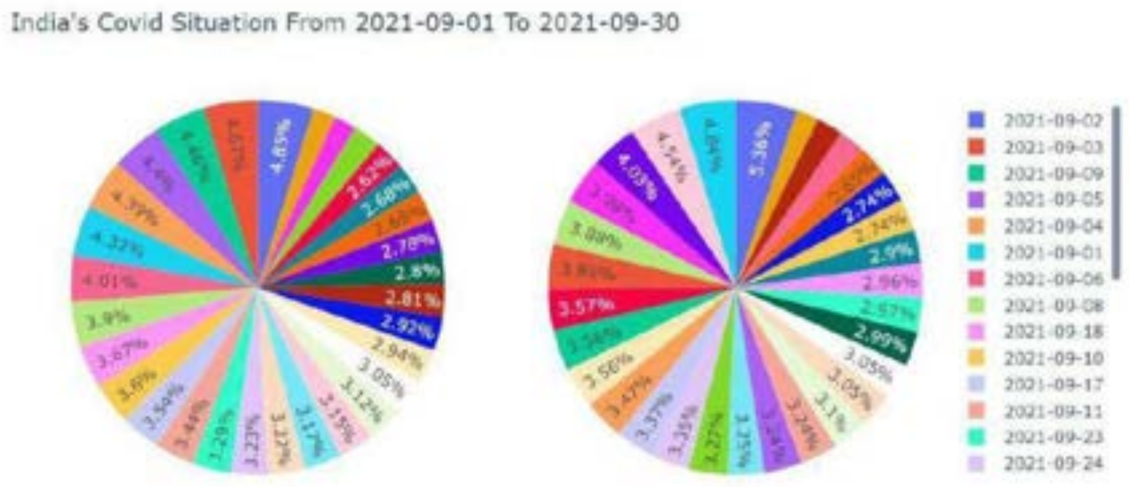


Fig 4.12. pie graph of days vs new cases and new deaths

The diagram shown below is the graph for days vs new cases from 2021-09-01 to2021-09-30

day's VS new_case's

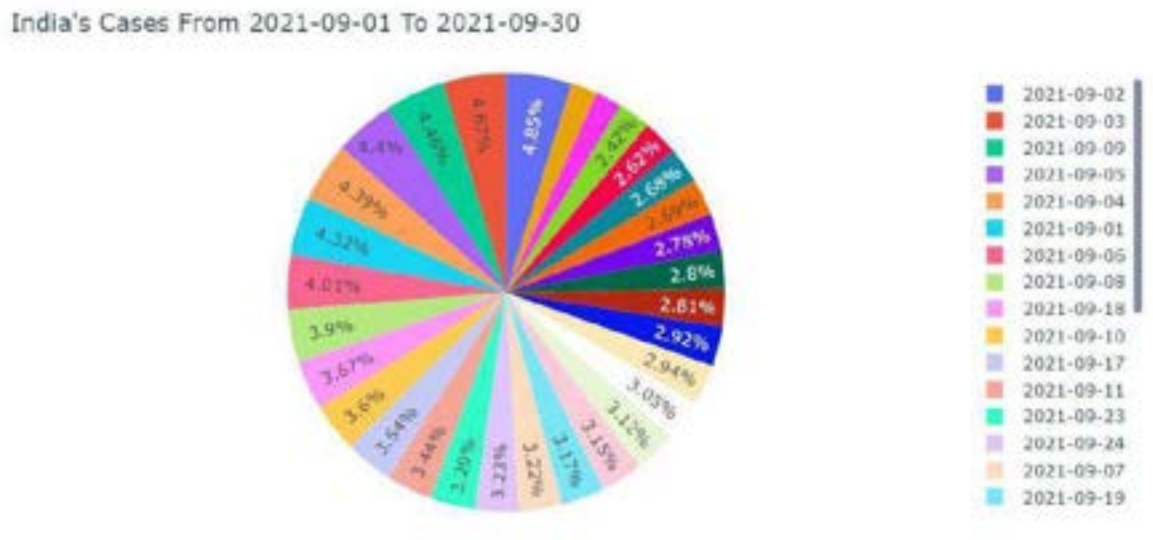


Fig 4.13. pie graph of days vs new cases

The diagram shown below is the graph for days vs new deaths from 2021-09-01 to 2021-09-30

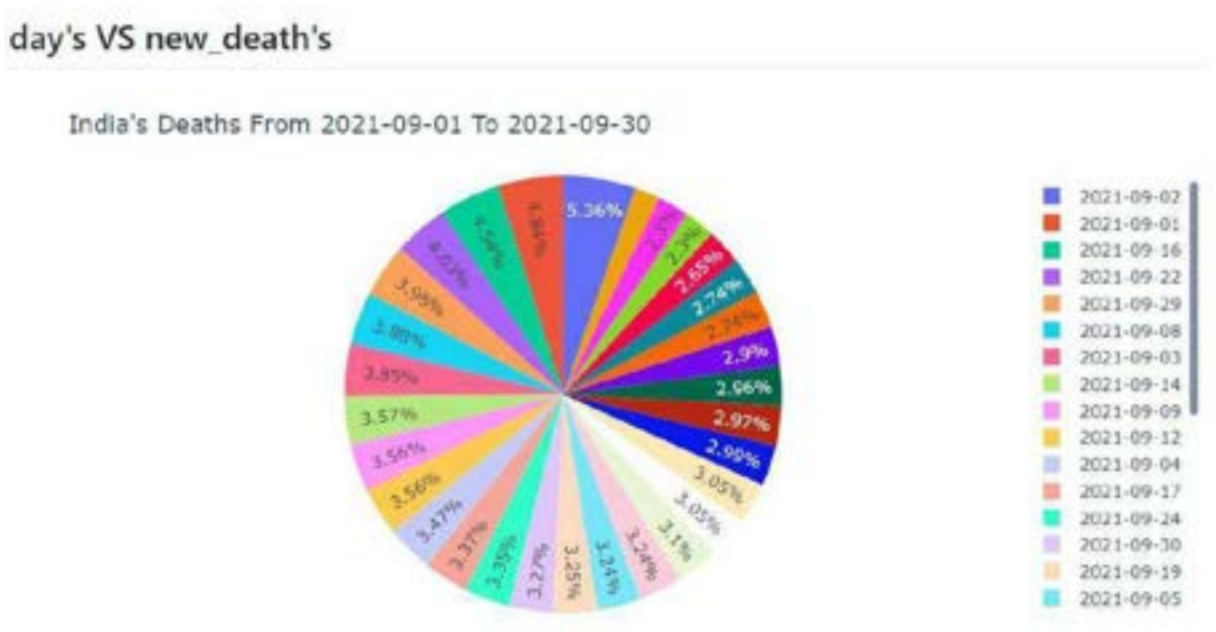


Fig 4.14. pie graph of days vs new deaths

- ➢ The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-30

Scatter Graph

day's VS new_cases's and death's

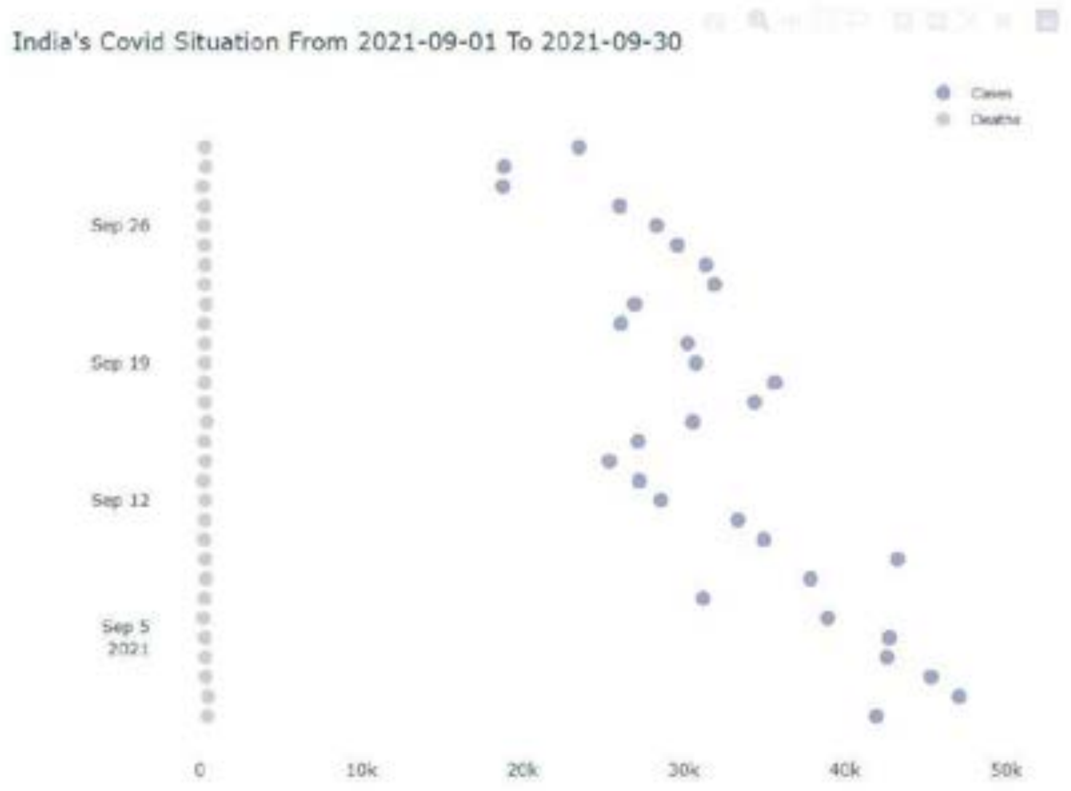


Fig 4.15. scatter graph of days vs new cases and new deaths

The diagram shown below is the graph for days vs new cases from 2021-09-01 to2021-09-30

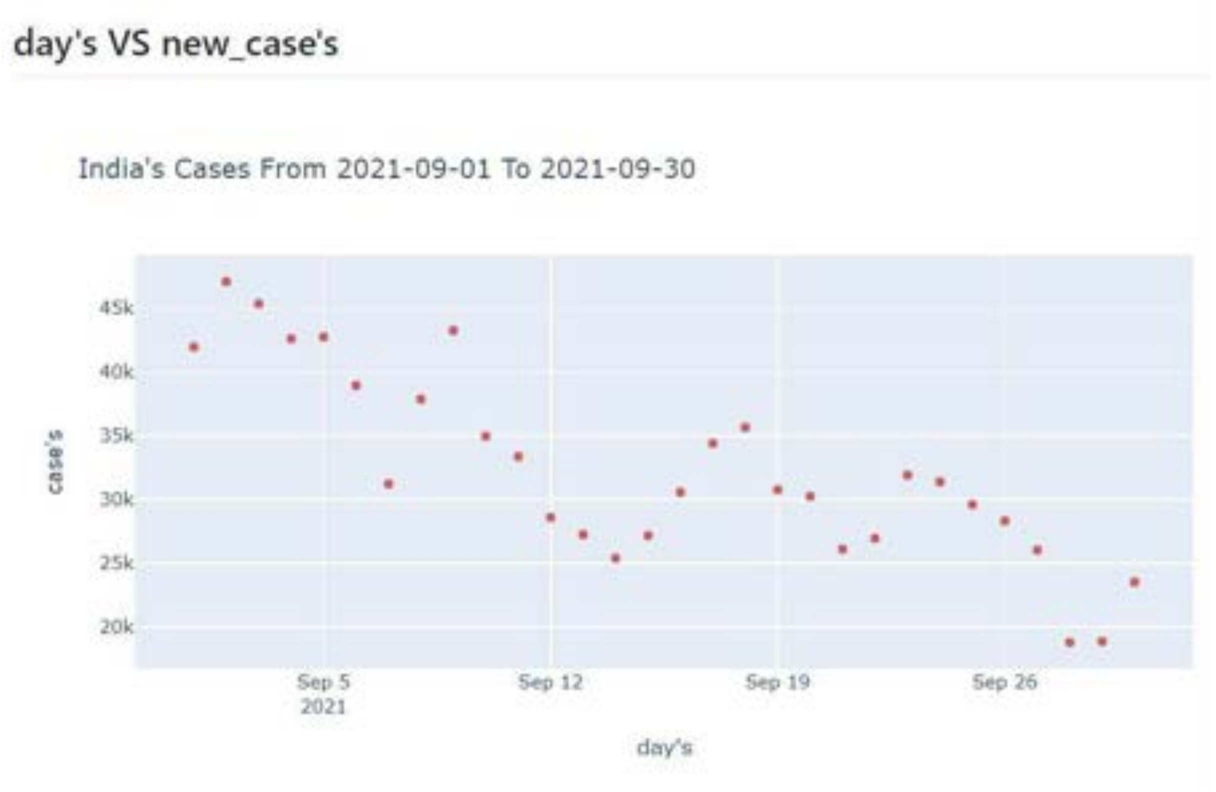


Fig 4.16. scatter graph of days vs new cases

- ➢ The diagram shown below is the graph for days vs new deaths from 2021-09-01 to 2021-09-30

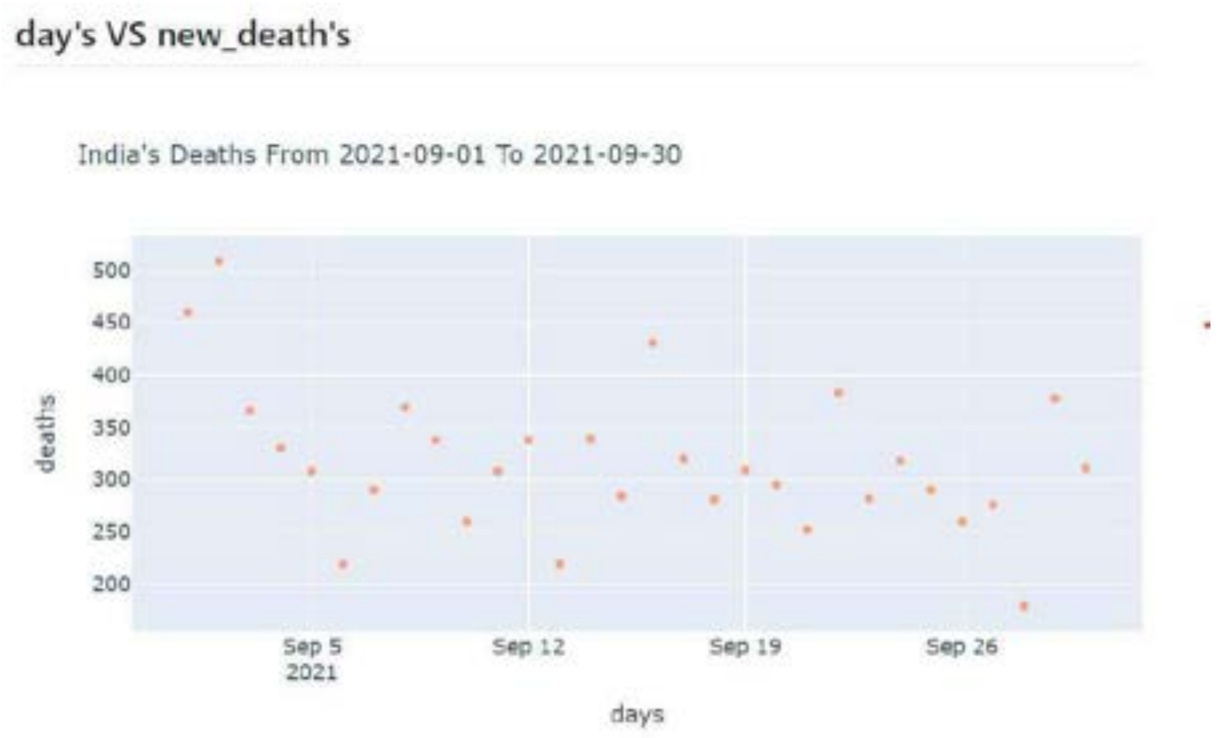


Fig 4.17. scatter graph of days vs new deaths

- The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-03

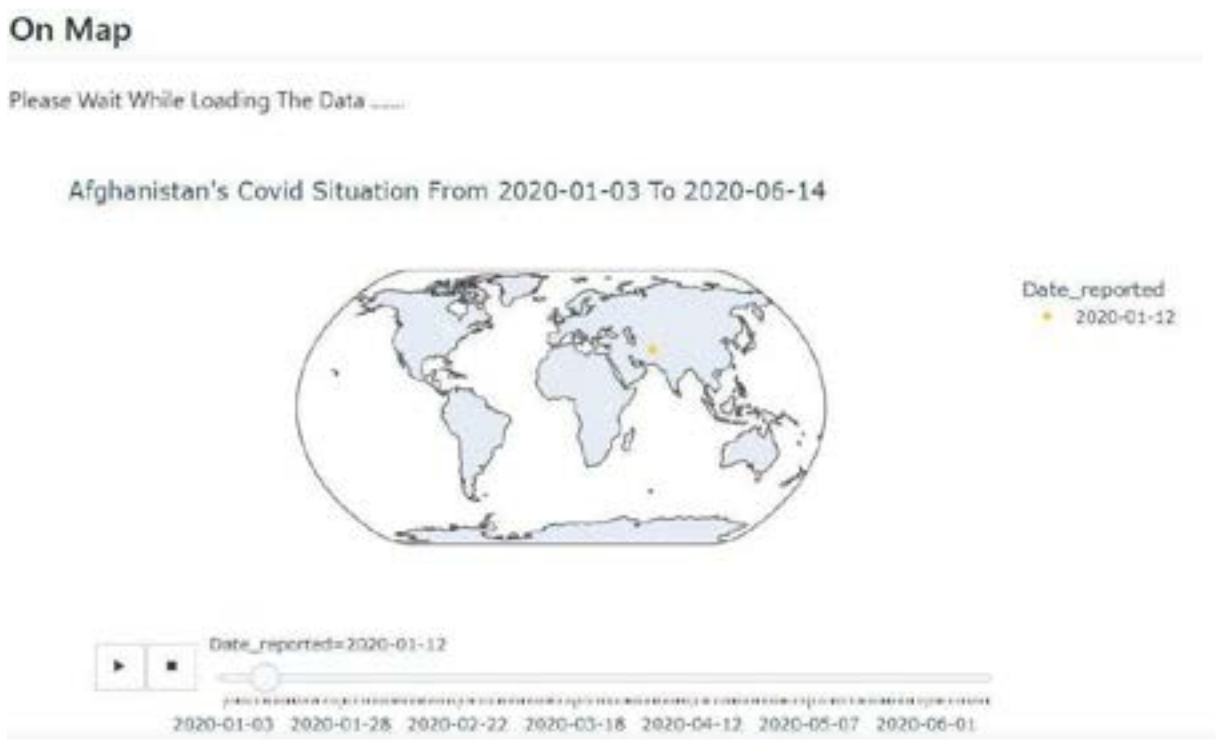


Fig 4.18. graph of new cases and new deaths on map

- The diagram shown below is the graph for days vs new cases and new deaths from 2021-09-01 to 2021-09-03

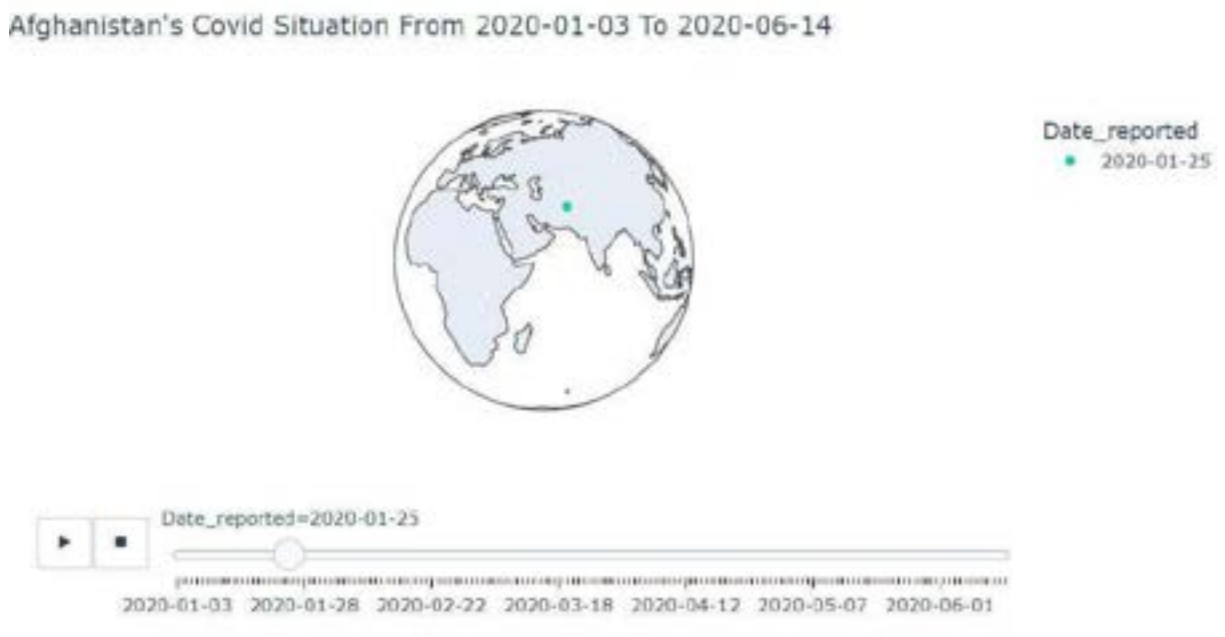


Fig 4.19. graph of new cases and new deaths on map

5. CONCLUSION

In this project we have provided a sample statistical visualization of the novel coronavirus (COVID -19) outbreak in the world. Using data of the daily and cumulative incidence in the world over approximately the first after the first case was confirmed in each respective countries. In addition, the result could be useful in contributing to health policy decisions or government interventions for the amount of human loss if in case of any third wave

By analysing these visualization techniques we can get proper idea about the pandemic that helps users to take much better safety precautions to survive from covid 19. The COVID-19 pandemic has affected the world in various ways. The deficiency of information, the need for accurate information, and the rapidity of its dissemination are important, as this pandemic requires the cooperation of entire populations.

REFERENCES

Dataset:

https://covid19.who.int/WHO-COVID-19-global-data.csv

https://www.kaggle.com/sudalairajkumar/covid19-in-india

Software:

https://docs.python.org/3/

https://atom.io/packages/ide-python

Modules:

https://www.w3schools.com/python/numpy/numpy_intro.asp

https://www.w3schools.com/python/pandas/default.asp

https://www.w3schools.com/python/python_regex.asp

https://www.w3schools.com/python/matplotlib_pyplot.asp

https://www.pyweb.io/tutorial.html

APPENDIX

a. source code:

```
from pywebio.input import * #input,
FLOAT,input_group,textarea,radio,select,DATE

from pywebio.output import *
#put_text,put_warning,put_success,put_info,toast,popup,put_image,put_markdo
wn,OutputPosition,put_processbar,set_processbar,clear

import urllib
import pandas as pd

import numpy as np

import os

import matplotlib.pyplot as plt

import re

import time

import pandas_datareader as web

import datetime as dt

import mplfinance as mpf

import matplotlib.pyplot as plt

from sklearn.preprocessing import MinMaxScaler

from tensorflow.keras.layers import Dense,Dropout,LSTM

from tensorflow.keras.models import Sequential

from pywebio.input import *

from pywebio.output import *

from pywebio import input as pin

from pywebio import output as pout

import plotly.graph_objects as go
```

```
#https://graphics.reuters.com/world-coronavirus-tracker-and-maps/countries-
and-territories/india/

def covid_app():

    import pandas as pd

    #put_processbar('bar')

    #for i in range(1, 11):

        #set_processbar('bar', i / 10)

    from pygame import mixer

    mixer.init()

    mixer.music.load("wait.mp3")

    mixer.music.play()

    put_text("Please Wait While Loading The Data .......")

    df = pd.read_csv('https://covid19.who.int/WHO-COVID-19-global-data.csv')

    country = df['Country'].unique()

    #time.sleep(2)

    #toast("Loading The Data .......")

    from pygame import mixer

    mixer.init()

    mixer.music.load("country.mp3")

    mixer.music.play()
    user_info=select("Choose The Country",country)#radio

    if user_info not in country:

        put_warning('Sorry ,once check the country name. Example : India')

    else:

        put_success("Congrats, You Can Go Further With The Detail's Of
"+user_info)

        date = df["Date_reported"].unique()

        #start =input("Start Date：(EX : 2021-09-01)")

        from pygame import mixer

        mixer.init()

        mixer.music.load("date.mp3")

        mixer.music.play()

        start = select("Start Date：",date)

        put_info('Start Date: ',start)

        #end = input("End Date：(EX : 2021-09-30)")

        from pygame import mixer

        mixer.init()

        mixer.music.load("date1.mp3")

        mixer.music.play()
        end = select("End Date： ",date)

        put_info('End Date: ',end)

        df_c = df[df.Country == user_info ]

        sep_c =  df_c.loc[(df_c["Date_reported"] >= start) & (df_c["Date_reported"]
<= end)]

        x = sep_c["Date_reported"]

        y = sep_c["New_cases"]

        z = sep_c["New_deaths"]

        from pygame import mixer

        mixer.init()

        mixer.music.load("graph.mp3")

        mixer.music.play()

        graphtype=select("Choose The Type Of
Graph",options=['Bar','Line','Pie','Scatter','On map','Table'])#radio

        #put_text(graphtype)

        #start of the bar graph

        if graphtype == 'Bar':

            from pygame import mixer

            mixer.init()

            mixer.music.load("bar.mp3")
        mixer.music.play()

        put_markdown('## Bar Graph')

        i=0

        while i==0:

            time.sleep(3)

            from pygame import mixer

            mixer.init()

            mixer.music.load("choose.mp3")

            mixer.music.play()

            tybar=select("Choose :",["day's VS new_cases's and death's","day's
VS new_case's","day's VS new_death's","end"])

            if tybar == "day's VS new_cases's and death's":

                put_markdown("## day's VS new_cases's and death's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cd.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.graph_objects as go

                fig = go.Figure()
```

```
fig.add_trace(go.Bar(x=x,y=y,name='New_cases',marker_color='indianred'))


fig.add_trace(go.Bar(x=x,y=z,name='New_deaths',marker_color='lightsalmon'))

                fig.update_layout(title=user_info+"'s Covid Situation From "+start+"
To "+end,

      #https://plotly.com/python/discrete-color/



                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title='cases count',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(
                x=0,

                y=1.0,

                bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'

                ),

                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

#https://pywebio.readthedocs.io/en/latest/

#https://pywebio-charts.pywebio.online/?app=plotly

                elif tybar == "day's VS new_case's":

                put_markdown("## day's VS new_case's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cdd.mp3")
                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

                col=[]

                for i in x:

                    col.append('indianred')

        #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

                fig = px.bar(x=x,y=y,color_discrete_sequence=col)

                fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

                fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

                fig.update_layout(title=user_info+"'s Cases From "+start+" To
"+end,

        #https://plotly.com/python/discrete-color/



                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,
                ),

                yaxis=dict(

                title='cases',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(

                x=0,

                y=1.0,

                bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'

                ),

                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0
```

```
        elif tybar == "day's VS new_death's":
            put_markdown("## day's VS new_death's")
            from pygame import mixer
            mixer.init()
            mixer.music.load("cdde.mp3")
            mixer.music.play()
            col=[]
            for i in x:
                col.append('lightsalmon')
    #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")
            fig = px.bar(x=x,y=z,color_discrete_sequence=col)
            fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
            fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')
            fig.update_layout(title=user_info+"'s Deaths From "+start+" To
"+end,
            xaxis_tickfont_size=14,
            xaxis=dict(
            title='days',
            titlefont_size=16,
            tickfont_size=14,

            ),

            yaxis=dict(

            title='deaths',

            titlefont_size=16,

            tickfont_size=14,

            ),

            legend=dict(

            x=0,

            y=1.0,

            bgcolor='rgba(255, 255, 255, 0)',

            bordercolor='rgba(255, 255, 255, 0)'

            ),

            barmode='group',

            bargap=0.15, # gap between bars of adjacent location coordinates.

            bargroupgap=0.1 # gap between bars of the same location
coordinate.

            )

            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)
```

```
                i=0

            elif tybar == "end":

                i=1

    elif graphtype == 'Line':

        from pygame import mixer

        mixer.init()

        mixer.music.load("line.mp3")

        mixer.music.play()

        put_markdown('## Line Graph')

        time.sleep(2)

        i=0

        while i==0:

            time.sleep(3)

            from pygame import mixer

            mixer.init()

            mixer.music.load("choose.mp3")

            mixer.music.play()

            tybar=select("Choose : ",["day's VS new_cases's and death's","day's
VS new_case's","day's VS new_death's","end"])

            if tybar == "day's VS new_cases's and death's":
                put_markdown("## day's VS new_cases's and death's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cd.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

                import plotly.graph_objects as go

        #df = px.data.gapminder().query("continent=='Oceania'")

        #fig = px.line(df,x=y, y=z,color_discrete_sequence=col )

                fig = go.Figure()


fig.add_trace(go.Line(x=x,y=y,name='New_cases',marker_color='indianred'))


fig.add_trace(go.Line(x=x,y=z,name='New_deaths',marker_color='lightsalmon'))

                fig.update_layout(title=user_info+"'s Covid Situation From "+start+"
To "+end,

                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',
                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title='cases count',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(x=0,y=1.0,bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'),

                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS new_case's":

                put_markdown("## day's VS new_case's")
```

```
               from pygame import mixer

               mixer.init()

               mixer.music.load("cdd.mp3")

               mixer.music.play()

               from pywebio.output import put_html

               import plotly.express as px

               col=[]

               for i in x:

                   col.append('indianred')

       #df = px.data.gapminder().query("continent=='Oceania'")

               fig = px.line(x=x, y=y,color_discrete_sequence=col )

               fig.update_layout(title=user_info+"'s Cases From "+start+" To
"+end,

       #https://plotly.com/python/discrete-color/



               xaxis_tickfont_size=14,

               xaxis=dict(

               title='days',

               titlefont_size=16,

               tickfont_size=14,
),
yaxis=dict(
title='cases',
titlefont_size=16,
tickfont_size=14,
),
legend=dict(
x=0,
y=1.0,
bgcolor='rgba(255, 255, 255, 0)',
bordercolor='rgba(255, 255, 255, 0)'
),
barmode='group',
bargap=0.15, # gap between bars of adjacent location coordinates.
bargroupgap=0.1 # gap between bars of the same location
coordinate.
)
html = fig.to_html(include_plotlyjs="require", full_html=False)
put_html(html)
i=0
```

```
            elif tybar == "day's VS new_death's":

                put_markdown("## day's VS new_death's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cdde.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

                col=[]

                for i in x:

                    col.append('lightsalmon')

        #df = px.data.gapminder().query("continent=='Oceania'")

                fig = px.line(x=x, y=z,color_discrete_sequence=col )

                fig.update_layout(title=user_info+"'s Deaths From "+start+" To
    "+end,

        #https://plotly.com/python/discrete-color/



                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',
                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title='deaths',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(

                x=0,

                y=1.0,

                bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'

                ),

                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
    coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)
```

```
                put_html(html)

                i=0

            elif tybar == 'end':

                i=1

    elif graphtype == 'Pie':

        from pygame import mixer

        mixer.init()

        mixer.music.load("pie.mp3")

        mixer.music.play()

        put_markdown('## Pie Graph')

        time.sleep(2)

        i=0

        while i==0:

            time.sleep(3)

            from pygame import mixer

            mixer.init()

            mixer.music.load("choose.mp3")

            mixer.music.play()

            tybar=select("Choose : ",["day's VS new_cases's and death's","day's
VS new_case's","day's VS new_death's","end"])
```

```
                if tybar == "day's VS new_cases's and death's":

                    put_markdown("## day's VS new_cases's and death's")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("cd.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.graph_objects as go

                    from plotly.subplots import make_subplots

                    fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},
{'type':'domain'}]])

                    fig.add_trace(go.Pie(labels=x, values=y, name="case's"),1, 1)

                    fig.add_trace(go.Pie(labels=x, values=z, name="death's"),1, 2)

                    fig.update_traces(textposition='inside')

                    fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                    fig.update_layout(title_text=user_info+"'s Covid Situation From
"+start+" To "+end)



        # Add annotations in the center of the donut pies.
```

```
                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS new_case's":

                put_markdown("## day's VS new_case's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cdd.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

        #df = px.data.gapminder().query("continent == 'Asia'")

                fig = px.pie(values=y, names=x)

                fig.update_traces(textposition='inside')

                fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                fig.update_layout(title=user_info+"'s Cases From "+start+" To
"+end)

                html = fig.to_html(include_plotlyjs="require", full_html=False)
                put_html(html)

                i=0

            elif tybar == "day's VS new_death's":

                put_markdown("## day's VS new_death's")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cdde.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

        #df = px.data.gapminder().query("continent == 'Asia'")

                fig = px.pie(values=z, names=x)

                fig.update_traces(textposition='inside')

                fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                fig.update_layout(title=user_info+"'s Deaths From "+start+" To
"+end)

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0
            elif tybar == "end":
                i=1


        elif graphtype == 'Scatter':
            from pygame import mixer
            mixer.init()
            mixer.music.load("scatter.mp3")
            mixer.music.play()
            put_markdown('## Scatter Graph')
            time.sleep(2)
            i=0
            while i==0:
                time.sleep(3)
                from pygame import mixer
                mixer.init()
                mixer.music.load("choose.mp3")
                mixer.music.play()
                tybar=select("Choose : ",["day's VS new_cases's and death's","day's
VS new_case's","day's VS new_death's","end"])
                if tybar == "day's VS new_cases's and death's":
```

```
put_markdown("## day's VS new_cases's and death's")

from pygame import mixer

mixer.init()

mixer.music.load("cd.mp3")

mixer.music.play()

from pywebio.output import put_html

import plotly.graph_objects as go

fig = go.Figure()

fig.add_trace(go.Scatter(x=y,y=x,name='Cases',

marker=dict(

color='rgba(156, 165, 196, 0.95)',

line_color='rgba(156, 165, 196, 1.0)',

)

))

fig.add_trace(go.Scatter(x=z, y=x,name='Deaths',

marker=dict(

color='rgba(204, 204, 204, 0.95)',

line_color='rgba(217, 217, 217, 1.0)'

)
```

```
                    ))

                    fig.update_traces(mode='markers', marker=dict(line_width=1,
symbol='circle', size=10))

                    fig.update_layout(
                    title=user_info+"'s Covid Situation From "+start+" To "+end,
                    margin=dict(l=140, r=40, b=50, t=80),
                    legend=dict(
                    font_size=10,
                    yanchor='middle',
                    xanchor='right',
                    ),
                    width=800,
                    height=600,
                    paper_bgcolor='white',
                    plot_bgcolor='white',
                    hovermode='closest',
                    )
                    html = fig.to_html(include_plotlyjs="require", full_html=False)
```

```
            put_html(html)

            i=0

        elif tybar == "day's VS new_case's":

            put_markdown("## day's VS new_case's")

            from pygame import mixer

            mixer.init()

            mixer.music.load("cdd.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.express as px

            import pandas as pd

#df = pd.DataFrame(dict(school=x, salary=y,gender=x))

            col=[]

            for i in x:

                col.append('indianred')

# Use column names of df for the different parameters x, y, color, ...

            fig = px.scatter(x=x, y=y,color_discrete_sequence=col,

             title=user_info+"'s Cases From "+start+" To "+end,

             labels={"x":"day's","y":"case's"} # customize axis label
        )



        html = fig.to_html(include_plotlyjs="require", full_html=False)

        put_html(html)

        i=0

    elif tybar == "day's VS new_death's":

        put_markdown("## day's VS new_death's")

        from pygame import mixer

        mixer.init()

        mixer.music.load("cdde.mp3")

        mixer.music.play()

        from pywebio.output import put_html

        import plotly.express as px

        import pandas as pd

#df = pd.DataFrame(dict(school=x, salary=y,gender=x))

        col=[]

        for i in x:

            col.append('lightsalmon')

# Use column names of df for the different parameters x, y, color, ...
```

```
                        fig = px.scatter(x=x, y=z,color_discrete_sequence=col,
                         title=user_info+"'s Deaths From "+start+" To "+end)
                        fig.update_layout(title=user_info+"'s Deaths From "+start+" To
"+end,
          #https://plotly.com/python/discrete-color/

                        xaxis_tickfont_size=14,
                        xaxis=dict(
                        title='days',
                        titlefont_size=16,
                        tickfont_size=14,
                        ),
                        yaxis=dict(
                        title='deaths',
                        titlefont_size=16,
                        tickfont_size=14,
                        ),
                        legend=dict(
                        x=0,
                        y=1.0,
                        bgcolor='rgba(255, 255, 255, 0)',
                        bordercolor='rgba(255, 255, 255, 0)'
                        ),
                        barmode='group',
                        bargap=0.15, # gap between bars of adjacent location coordinates.
                        bargroupgap=0.1 # gap between bars of the same location
              coordinate.
                        )

                        html = fig.to_html(include_plotlyjs="require", full_html=False)
                        put_html(html)
                        i=0
                    elif tybar == "end":
                        i=1
          elif graphtype == 'On map':
              from pygame import mixer
              mixer.init()
              mixer.music.load("onmap.mp3")
              mixer.music.play()
              put_markdown('## On Map')
              time.sleep(2)

              from pygame import mixer

              mixer.init()

              mixer.music.load("wait.mp3")

              mixer.music.play()

              put_text("Please Wait While Loading The Data .......")

              from pywebio.output import put_html

              import plotly.express as px

              import pandas as pd

              df = pd.read_csv('https://covid19.who.int/WHO-COVID-19-global-
  data.csv')

              df_c = df[df.Country == user_info]

              sep_c =  df_c.loc[(df_c["Date_reported"] >= start) &
  (df_c["Date_reported"] <= end)]

              x = sep_c["Date_reported"]

              y = sep_c["New_cases"]

              country=sep_c["Country"]



              df1 = px.data.gapminder()

              Country=df1['country'].unique()
```

```
from pygame import mixer

mixer.init()

mixer.music.load("country.mp3")

mixer.music.play()

user_info1=select("Choose The Country",Country)

df1=df1[['country', 'iso_alpha']]

try:

    df1=df1[df1.country == user_info1]

    df1=df1.reset_index()

    idd=df1["iso_alpha"][0]

    loc=[]

    for i in x:

        loc.append(idd)

    fig = px.scatter_geo(sep_c,locations=loc, color=x,

         hover_name=country,hover_data=['New_cases','New_deaths'],

         projection="natural earth",

         animation_frame=x,title=user_info+"'s Covid Situation From
"+start+" To "+end)



    html = fig.to_html(include_plotlyjs="require", full_html=False)
```

```
                put_html(html)

                fig = px.scatter_geo(sep_c,locations=loc, color=x,

                    hover_name=country,hover_data=['New_cases','New_deaths'],

                    projection="orthographic",

                    animation_frame=x,title=user_info+"'s Covid Situation From
"+start+" To "+end)

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

            except:

                toast("Something Went Wrong",color='red')




# Use column names of df for the different parameters x, y, color, ...






        elif graphtype == 'Table':

            from pygame import mixer

            mixer.init()

            mixer.music.load("table.mp3")
            mixer.music.play()

            put_markdown('## Table')

            time.sleep(2)

            from pywebio.output import put_html

            import plotly.graph_objects as go

            from plotly.colors import n_colors

            import numpy as np

            np.random.seed(1)



            colors = n_colors('rgb(255, 200, 200)', 'rgb(200, 0, 0)', 9, colortype='rgb')

            a = x

            b = y

            c = z

            headerColor = 'grey'

            col=[]

            for i in x:

                col.append('white')

            fig = go.Figure(data=[go.Table(

            header=dict(
            values=['<b>days</b>', '<b>cases</b>', '<b>deaths</b>'],

            line_color='darkslategray',fill_color='white',

            align='center',font=dict(color='black', size=12)

            ),

            cells=dict(

            values=[a, b, c],

            line_color='darkslategray',

            fill_color =col,

            align='center', font=dict(color='black', size=11)

            ))

            ])


            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)


def india_app():

    df = pd.read_csv('covid_19_india.csv')

    #df=df.replace(to_replace ="-",value ="0")

    df['Date']=pd.to_datetime(df.Date)
    state=df["State"].unique()

    from pygame import mixer

    mixer.init()

    mixer.music.load("state.mp3")

    mixer.music.play()

    state=select("Choose The State : ",state)

    put_success("Congrats, You Can Go Further With The Detail's Of "+state)

    time.sleep(3)

    from pygame import mixer

    mixer.init()

    mixer.music.load("choose.mp3")

    mixer.music.play()

    confirmed=select("Choose : ",["Cured vs Deaths","Cured vs
Confirmed","Deaths vs Confirmed"])

    #put_text(state)

    from pygame import mixer

    mixer.init()

    mixer.music.load("date.mp3")

    mixer.music.play()

    start = input("Start Date : ",type=DATE)
    put_info('Start Date: ',start)

#put_text(date)

#start=input("From : ",type=DATE)

#put_text(start)

    from pygame import mixer

    mixer.init()

    mixer.music.load("date1.mp3")

    mixer.music.play()

    end = input("End Date : ",type=DATE)

    put_info('End Date: ',end)

#end=input("End : ",type=DATE)

#put_text(end)

    df_c=df[df.State == state]

    sep_c =  df_c.loc[(df_c["Date"] >= start) & (df_c["Date"] <= end)]

    x = sep_c["Date"]

    if confirmed == "Cured vs Deaths":

        y = sep_c["Cured"]

        z = sep_c["Deaths"]

        from pygame import mixer
        mixer.init()

        mixer.music.load("graph.mp3")

        mixer.music.play()

        graphtype=select("Choose The Type Of
Graph",options=['Bar','Line','Pie','Scatter','Table'])#radio

        if graphtype == 'Bar':

            from pygame import mixer

            mixer.init()

            mixer.music.load("bar.mp3")

            mixer.music.play()

            put_markdown('## Bar Graph')

            time.sleep(2)

            i=0

            while i==0:

                time.sleep(3)

                from pygame import mixer

                mixer.init()

                mixer.music.load("choose.mp3")

                mixer.music.play()
```

```
                tybar=select("Choose :",["day's VS Cured and Deaths","day's VS
Cured","day's VS Deaths","end"])

                if tybar == "day's VS Cured and Deaths":

                    put_markdown("## day's VS Cured and Deaths")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("cur.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.graph_objects as go

                    fig = go.Figure()


fig.add_trace(go.Bar(x=x,y=y,name="Cured",marker_color='indianred'))


fig.add_trace(go.Bar(x=x,y=z,name="Deaths",marker_color='lightsalmon'))

                    fig.update_layout(title=state+"'s Cured and Death's From "+start+"
To "+end,

        #https://plotly.com/python/discrete-color/



                    xaxis_tickfont_size=14,

                    xaxis=dict(
                    title='days',
                    titlefont_size=16,
                    tickfont_size=14,
                    ),
                    yaxis=dict(
                    title="Cured and Death's",
                    titlefont_size=16,
                    tickfont_size=14,
                    ),
                    legend=dict(
                    x=0,
                    y=1.0,
                    bgcolor='rgba(255, 255, 255, 0)',
                    bordercolor='rgba(255, 255, 255, 0)'
                    ),
                    barmode='group',
                    bargap=0.15, # gap between bars of adjacent location coordinates.
                    bargroupgap=0.1 # gap between bars of the same location
coordinate.
                    )
```

```
                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

#https://pywebio.readthedocs.io/en/latest/

#https://pywebio-charts.pywebio.online/?app=plotly

                elif tybar == "day's VS Cured":

                    put_markdown("## day's VS Cured")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("cured.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.express as px

                    col=[]

                    for i in x:

                        col.append('indianred')

        #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

                    fig = px.bar(x=x,y=y,color_discrete_sequence=col)

                    fig.update_traces(textposition='outside')
                    fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

                    fig.update_layout(title=state+"'s Cured From "+start+" To "+end,

        #https://plotly.com/python/discrete-color/


                    xaxis_tickfont_size=14,

                    xaxis=dict(

                    title='days',

                    titlefont_size=16,

                    tickfont_size=14,

                    ),

                    yaxis=dict(

                    title="Cured",

                    titlefont_size=16,

                    tickfont_size=14,

                    ),

                    legend=dict(

                    x=0,

                    y=1.0,

                    bgcolor='rgba(255, 255, 255, 0)',
                    bordercolor='rgba(255, 255, 255, 0)'

                    ),

                    barmode='group',

                    bargap=0.15, # gap between bars of adjacent location coordinates.

                    bargroupgap=0.1 # gap between bars of the same location
        coordinate.

                    )

                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "day's VS Deaths":

                    put_markdown("## day's VS Deaths")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("death.mp3")

                    mixer.music.play()

                    col=[]

                    for i in x:

                        col.append('lightsalmon')
            #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
        and pop > 2.e6")

                    fig = px.bar(x=x,y=z,color_discrete_sequence=col)

                    fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

                    fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

                    fig.update_layout(title=state+"'s Death's From "+start+" To "+end,

                    xaxis_tickfont_size=14,

                    xaxis=dict(

                    title='days',

                    titlefont_size=16,

                    tickfont_size=14,

                    ),

                    yaxis=dict(

                    title="Deaths",

                    titlefont_size=16,

                    tickfont_size=14,

                    ),

                    legend=dict(

                    x=0,

                    y=1.0,
                    bgcolor='rgba(255, 255, 255, 0)',
                    bordercolor='rgba(255, 255, 255, 0)'
                    ),
                    barmode='group',
                    bargap=0.15, # gap between bars of adjacent location coordinates.
                    bargroupgap=0.1 # gap between bars of the same location
coordinate.
                    )
                    html = fig.to_html(include_plotlyjs="require", full_html=False)
                    put_html(html)
                    i=0
                elif tybar == "end":
                    i=1
        elif graphtype == 'Line':
            from pygame import mixer
            mixer.init()
            mixer.music.load("line.mp3")
            mixer.music.play()
            put_markdown('## Line Graph')
            time.sleep(2)
            i=0

            while i==0:

                time.sleep(3)

                from pygame import mixer

                mixer.init()

                mixer.music.load("choose.mp3")

                mixer.music.play()

                tybar=select("Choose : ",["day's VS Cured and Deaths","day's VS
       Cured","day's VS Deaths","end"])

                if tybar == "day's VS Cured and Deaths":

                    put_markdown("## day's VS Cured and Deaths")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("cur.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.express as px

                    import plotly.graph_objects as go

        #df = px.data.gapminder().query("continent=='Oceania'")

        #fig = px.line(df,x=y, y=z,color_discrete_sequence=col )
```

```
                fig = go.Figure()

fig.add_trace(go.Line(x=x,y=y,name='Cured',marker_color='indianred'))

fig.add_trace(go.Line(x=x,y=z,name='Deaths',marker_color='lightsalmon'))

                fig.update_layout(title=state+"'s Cured and Deaths From "+start+" To "+end,

                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title='Cured and Deaths',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(x=0,y=1.0,bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'),
                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS Cured":

                put_markdown("## day's VS Cured")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cured.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

                col=[]

                for i in x:

                    col.append('indianred')

    #df = px.data.gapminder().query("continent=='Oceania'")
                fig = px.line(x=x, y=y,color_discrete_sequence=col )

                fig.update_layout(title=state+"'s Cured From "+start+" To "+end,

    #https://plotly.com/python/discrete-color/



                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title='Cured',

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(

                x=0,

                y=1.0,

                bgcolor='rgba(255, 255, 255, 0)',
                    bordercolor='rgba(255, 255, 255, 0)'
                    ),
                    barmode='group',
                    bargap=0.15, # gap between bars of adjacent location coordinates.
                    bargroupgap=0.1 # gap between bars of the same location
    coordinate.
                    )
                    html = fig.to_html(include_plotlyjs="require", full_html=False)
                    put_html(html)
                    i=0
                elif tybar == "day's VS Deaths":
                    put_markdown("## day's VS Deaths")
                    from pygame import mixer
                    mixer.init()
                    mixer.music.load("death.mp3")
                    mixer.music.play()
                    from pywebio.output import put_html
                    import plotly.express as px
                    col=[]
                    for i in x:
```

```
        col.append('lightsalmon')

#df = px.data.gapminder().query("continent=='Oceania'")

      fig = px.line(x=x, y=z,color_discrete_sequence=col )

      fig.update_layout(title=state+"'s Deaths From "+start+" To "+end,

#https://plotly.com/python/discrete-color/



      xaxis_tickfont_size=14,

      xaxis=dict(

      title='days',

      titlefont_size=16,

      tickfont_size=14,

      ),

      yaxis=dict(

      title='Deaths',

      titlefont_size=16,

      tickfont_size=14,

      ),

      legend=dict(

      x=0,
                    y=1.0,

                    bgcolor='rgba(255, 255, 255, 0)',

                    bordercolor='rgba(255, 255, 255, 0)'

                    ),

                    barmode='group',

                    bargap=0.15, # gap between bars of adjacent location coordinates.

                    bargroupgap=0.1 # gap between bars of the same location
coordinate.

                    )

                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == 'end':

                    i=1

        elif graphtype == 'Pie':

            from pygame import mixer

            mixer.init()

            mixer.music.load("pie.mp3")

            mixer.music.play()

            put_markdown('## Pie Graph')
```

```
    time.sleep(2)

    i=0

    while i==0:

        time.sleep(3)

        from pygame import mixer

        mixer.init()

        mixer.music.load("choose.mp3")

        mixer.music.play()

        tybar=select("Choose : ",["day's VS Cured and Deaths","day's VS
Cured","day's VS Deaths","end"])

        if tybar == "day's VS Cured and Deaths":

            put_markdown("## day's VS Cured and Deaths")

            from pygame import mixer

            mixer.init()

            mixer.music.load("cur.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.graph_objects as go

            from plotly.subplots import make_subplots
```

```
                    fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},
{'type':'domain'}]])

                    fig.add_trace(go.Pie(labels=x, values=y, name="Cured"),1, 1)

                    fig.add_trace(go.Pie(labels=x, values=z, name="Deaths"),1, 2)

                    fig.update_traces(textposition='inside')

                    fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                    fig.update_layout(title_text=state+"'s Cured and Deaths From
"+start+" To "+end)



        # Add annotations in the center of the donut pies.



                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "day's VS Cured":

                    put_markdown("## day's VS Cured")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("cured.mp3")
                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.express as px

        #df = px.data.gapminder().query("continent == 'Asia'")

                    fig = px.pie(values=y, names=x)

                    fig.update_traces(textposition='inside')

                    fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                    fig.update_layout(title=state+"'s Cured From "+start+" To "+end)

                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "day's VS Deaths":

                    put_markdown("## day's VS Deaths")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("death.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.express as px
        #df = px.data.gapminder().query("continent == 'Asia'")

                    fig = px.pie(values=z, names=x)

                    fig.update_traces(textposition='inside')

                    fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                    fig.update_layout(title=state+"'s Deaths From "+start+" To "+end)

                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "end":

                    i=1



        elif graphtype == 'Scatter':

           from pygame import mixer

           mixer.init()

           mixer.music.load("scatter.mp3")

           mixer.music.play()

           put_markdown('## Scatter Graph')

           time.sleep(2)

           i=0
```

```
while i==0:

    time.sleep(3)

    from pygame import mixer

    mixer.init()

    mixer.music.load("choose.mp3")

    mixer.music.play()

    tybar=select("Choose : ",["day's VS Cured and Deaths","day's VS
Cured","day's VS Deaths","end"])

    if tybar == "day's VS Cured and Deaths":

        put_markdown("## day's VS Cured and Deaths")

        from pygame import mixer

        mixer.init()

        mixer.music.load("cur.mp3")

        mixer.music.play()

        from pywebio.output import put_html

        import plotly.graph_objects as go

        fig = go.Figure()

        fig.add_trace(go.Scatter(x=y,y=x,name='Cured',

        marker=dict(

        color='rgba(156, 165, 196, 0.95)',
        line_color='rgba(156, 165, 196, 1.0)',

        )

        ))

        fig.add_trace(go.Scatter(x=z, y=x,name='Deaths',

        marker=dict(

        color='rgba(204, 204, 204, 0.95)',

        line_color='rgba(217, 217, 217, 1.0)'

        )

        ))



        fig.update_traces(mode='markers', marker=dict(line_width=1,
symbol='circle', size=10))



        fig.update_layout(

        title=state+"'s Cured and Deaths From "+start+" To "+end,

        margin=dict(l=140, r=40, b=50, t=80),

        legend=dict(

        font_size=10,

        yanchor='middle',

        xanchor='right',
        ),

        width=800,

        height=600,

        paper_bgcolor='white',

        plot_bgcolor='white',

        hovermode='closest',

        )

        html = fig.to_html(include_plotlyjs="require", full_html=False)

        put_html(html)

        i=0

    elif tybar == "day's VS Cured":

        put_markdown("## day's VS Cured")

        from pygame import mixer

        mixer.init()

        mixer.music.load("cured.mp3")

        mixer.music.play()

        from pywebio.output import put_html

        import plotly.express as px
```

```
#df = pd.DataFrame(dict(school=x, salary=y,gender=x))
            col=[]
            for i in x:
                col.append('indianred')
# Use column names of df for the different parameters x, y, color, ...
            fig = px.scatter(x=x, y=y,color_discrete_sequence=col,
             title=state+"'s Cured From "+start+" To "+end,
             labels={"x":"day's","y":"Cured"} # customize axis label
            )


            html = fig.to_html(include_plotlyjs="require", full_html=False)
            put_html(html)
            i=0
        elif tybar == "day's VS Deaths":
            put_markdown("## day's VS Deaths")
            from pygame import mixer
            mixer.init()
            mixer.music.load("death.mp3")
            mixer.music.play()
```

```
          from pywebio.output import put_html

          import plotly.express as px



#df = pd.DataFrame(dict(school=x, salary=y,gender=x))

          col=[]

          for i in x:

             col.append('lightsalmon')

# Use column names of df for the different parameters x, y, color, ...

          fig = px.scatter(x=x, y=z,color_discrete_sequence=col,

           title=state+"'s Deaths From "+start+" To "+end)

          fig.update_layout(title=state+"'s Deaths From "+start+" To "+end,

#https://plotly.com/python/discrete-color/



          xaxis_tickfont_size=14,

          xaxis=dict(

          title='days',

          titlefont_size=16,

          tickfont_size=14,

          ),
          yaxis=dict(

          title='deaths',

          titlefont_size=16,

          tickfont_size=14,

          ),

          legend=dict(

          x=0,

          y=1.0,

          bgcolor='rgba(255, 255, 255, 0)',

          bordercolor='rgba(255, 255, 255, 0)'

          ),

          barmode='group',

          bargap=0.15, # gap between bars of adjacent location coordinates.

          bargroupgap=0.1 # gap between bars of the same location
          coordinate.

          )



          html = fig.to_html(include_plotlyjs="require", full_html=False)

          put_html(html)

          i=0
```

```
                elif tybar == "end":

                    i=1



    # Use column names of df for the different parameters x, y, color, ...






        elif graphtype == 'Table':

            from pygame import mixer

            mixer.init()

            mixer.music.load("table.mp3")

            mixer.music.play()

            put_markdown('## Table')

            time.sleep(2)

            from pywebio.output import put_html

            import plotly.graph_objects as go

            from plotly.colors import n_colors

            np.random.seed(1)
```

```
colors = n_colors('rgb(255, 200, 200)', 'rgb(200, 0, 0)', 9, colortype='rgb')

a = x

b = y

c = z

headerColor = 'grey'

col=[]

for i in x:

    col.append('white')

fig = go.Figure(data=[go.Table(

header=dict(

values=['<b>days</b>', '<b>Cured</b>', '<b>deaths</b>'],

line_color='darkslategray',fill_color='white',

align='center',font=dict(color='black', size=12)

),

cells=dict(

values=[a, b, c],

line_color='darkslategray',

fill_color =col,

align='center', font=dict(color='black', size=11)
```

```
            ))

            ])


            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)


    elif confirmed == "Cured vs Confirmed":

        y = sep_c["Cured"]

        z = sep_c["Confirmed"]

        from pygame import mixer

        mixer.init()

        mixer.music.load("graph.mp3")

        mixer.music.play()

        graphtype=select("Choose The Type Of
Graph",options=['Bar','Line','Pie','Scatter','Table'])#radio

        if graphtype == 'Bar':

            from pygame import mixer

            mixer.init()

            mixer.music.load("bar.mp3")

            mixer.music.play()
```

```
put_markdown('## Bar Graph')

time.sleep(2)

i=0

while i==0:

    time.sleep(3)

    from pygame import mixer

    mixer.init()

    mixer.music.load("choose.mp3")

    mixer.music.play()

    tybar=select("Choose :",["day's VS Cured and Confirmed","day's VS
Cured","day's VS Confirmed","end"])

    if tybar == "day's VS Cured and Confirmed":

        put_markdown("## day's VS Cured and Confirmed")

        from pygame import mixer

        mixer.init()

        mixer.music.load("cuco.mp3")

        mixer.music.play()

        from pywebio.output import put_html

        import plotly.graph_objects as go

        fig = go.Figure()
```

```
fig.add_trace(go.Bar(x=x,y=y,name="Cured",marker_color='indianred'))


fig.add_trace(go.Bar(x=x,y=z,name="Confirmed",marker_color='lightsalmon'))

                fig.update_layout(title=state+"'s Cured and Confirmed From
"+start+" To "+end,

      #https://plotly.com/python/discrete-color/



                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,

                ),

                yaxis=dict(

                title="Cured and Confirmed",

                titlefont_size=16,

                tickfont_size=14,

                ),

                legend=dict(
                x=0,

                y=1.0,

                bgcolor='rgba(255, 255, 255, 0)',

                bordercolor='rgba(255, 255, 255, 0)'

                ),

                barmode='group',

                bargap=0.15, # gap between bars of adjacent location coordinates.

                bargroupgap=0.1 # gap between bars of the same location
coordinate.

                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

#https://pywebio.readthedocs.io/en/latest/

#https://pywebio-charts.pywebio.online/?app=plotly

                elif tybar == "day's VS Cured":

                put_markdown("## day's VS Cured")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cured.mp3")
```

```
          mixer.music.play()

          from pywebio.output import put_html

          import plotly.express as px

          col=[]

          for i in x:

              col.append('indianred')

      #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

          fig = px.bar(x=x,y=y,color_discrete_sequence=col)

          fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

          fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

          fig.update_layout(title=state+"'s Cured From "+start+" To "+end,

      #https://plotly.com/python/discrete-color/



          xaxis_tickfont_size=14,

          xaxis=dict(

          title='days',

          titlefont_size=16,

          tickfont_size=14,

          ),
          yaxis=dict(

          title="Cured",

          titlefont_size=16,

          tickfont_size=14,

          ),

          legend=dict(

          x=0,

          y=1.0,

          bgcolor='rgba(255, 255, 255, 0)',

          bordercolor='rgba(255, 255, 255, 0)'

          ),

          barmode='group',

          bargap=0.15, # gap between bars of adjacent location coordinates.

          bargroupgap=0.1 # gap between bars of the same location
coordinate.

          )

          html = fig.to_html(include_plotlyjs="require", full_html=False)

          put_html(html)

          i=0

      elif tybar == "day's VS Confirmed":
          put_markdown("## day's VS Confirmed")

          from pygame import mixer

          mixer.init()

          mixer.music.load("con.mp3")

          mixer.music.play()

          col=[]

          for i in x:

              col.append('lightsalmon')

      #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

          fig = px.bar(x=x,y=z,color_discrete_sequence=col)

          fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

          fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

          fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

          xaxis_tickfont_size=14,

          xaxis=dict(

          title='days',

          titlefont_size=16,

          tickfont_size=14,

          ),
          yaxis=dict(
          title="Confirmed",
          titlefont_size=16,
          tickfont_size=14,
          ),
          legend=dict(
          x=0,
          y=1.0,
          bgcolor='rgba(255, 255, 255, 0)',
          bordercolor='rgba(255, 255, 255, 0)'
          ),
          barmode='group',
          bargap=0.15, # gap between bars of adjacent location coordinates.
          bargroupgap=0.1 # gap between bars of the same location
coordinate.
          )
          html = fig.to_html(include_plotlyjs="require", full_html=False)
          put_html(html)
          i=0
      elif tybar == "end":
```

```
            i=1
elif graphtype == 'Line':
    from pygame import mixer
    mixer.init()
    mixer.music.load("line.mp3")
    mixer.music.play()
    put_markdown('## Line Graph')
    time.sleep(2)
    i=0
    while i==0:
        time.sleep(3)
        from pygame import mixer
        mixer.init()
        mixer.music.load("choose.mp3")
        mixer.music.play()
        tybar=select("Choose : ",["day's VS Cured and Confirmed","day's VS Cured","day's VS Confirmed","end"])
        if tybar == "day's VS Cured and Confirmed":
            put_markdown("## day's VS Cured and Confirmed")
            from pygame import mixer
```

```
                mixer.init()

                mixer.music.load("cuco.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

                import plotly.graph_objects as go

        #df = px.data.gapminder().query("continent=='Oceania'")

        #fig = px.line(df,x=y, y=z,color_discrete_sequence=col )

                fig = go.Figure()


fig.add_trace(go.Line(x=x,y=y,name='Cured',marker_color='indianred'))


fig.add_trace(go.Line(x=x,y=z,name='Confirmed',marker_color='lightsalmon'))

                fig.update_layout(title=state+"'s Cured and Confirmed From
"+start+" To "+end,

                xaxis_tickfont_size=14,

                xaxis=dict(

                title='days',

                titlefont_size=16,

                tickfont_size=14,
                ),
                yaxis=dict(
                title='Cured and Confirmed',
                titlefont_size=16,
                tickfont_size=14,
                ),
                legend=dict(x=0,y=1.0,bgcolor='rgba(255, 255, 255, 0)',
                bordercolor='rgba(255, 255, 255, 0)'),
                barmode='group',
                bargap=0.15, # gap between bars of adjacent location coordinates.
                bargroupgap=0.1 # gap between bars of the same location
coordinate.
                )
                html = fig.to_html(include_plotlyjs="require", full_html=False)
                put_html(html)
                i=0
              elif tybar == "day's VS Cured":
                put_markdown("## day's VS Cured")
                from pygame import mixer
                mixer.init()
```

```
            mixer.music.load("cured.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.express as px

            col=[]

            for i in x:

                col.append('indianred')

#df = px.data.gapminder().query("continent=='Oceania'")

            fig = px.line(x=x, y=y,color_discrete_sequence=col )

            fig.update_layout(title=state+"'s Cured From "+start+" To "+end,

#https://plotly.com/python/discrete-color/



            xaxis_tickfont_size=14,

            xaxis=dict(

            title='days',

            titlefont_size=16,

            tickfont_size=14,

            ),

            yaxis=dict(
        title='Cured',
        titlefont_size=16,
        tickfont_size=14,
        ),
        legend=dict(
        x=0,
        y=1.0,
        bgcolor='rgba(255, 255, 255, 0)',
        bordercolor='rgba(255, 255, 255, 0)'
        ),
        barmode='group',
        bargap=0.15, # gap between bars of adjacent location coordinates.
        bargroupgap=0.1 # gap between bars of the same location
coordinate.
        )
        html = fig.to_html(include_plotlyjs="require", full_html=False)
        put_html(html)
        i=0
      elif tybar == "day's VS Confirmed":
        put_markdown("## day's VS Confirmed")
```

```
            from pygame import mixer

            mixer.init()

            mixer.music.load("con.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.express as px

            col=[]

            for i in x:

                col.append('lightsalmon')

#df = px.data.gapminder().query("continent=='Oceania'")

            fig = px.line(x=x, y=z,color_discrete_sequence=col )

            fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

#https://plotly.com/python/discrete-color/



            xaxis_tickfont_size=14,

            xaxis=dict(

            title='days',

            titlefont_size=16,

            tickfont_size=14,
            ),

            yaxis=dict(

            title='Confirmed',

            titlefont_size=16,

            tickfont_size=14,

            ),

            legend=dict(

            x=0,

            y=1.0,

            bgcolor='rgba(255, 255, 255, 0)',

            bordercolor='rgba(255, 255, 255, 0)'

            ),

            barmode='group',

            bargap=0.15, # gap between bars of adjacent location coordinates.

            bargroupgap=0.1 # gap between bars of the same location
            coordinate.

            )

            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)

            i=0
```

```
            elif tybar == 'end':
                i=1
        elif graphtype == 'Pie':
            from pygame import mixer
            mixer.init()
            mixer.music.load("pie.mp3")
            mixer.music.play()
            put_markdown('## Pie Graph')
            time.sleep(2)
            i=0
            while i==0:
                time.sleep(3)
                from pygame import mixer
                mixer.init()
                mixer.music.load("choose.mp3")
                mixer.music.play()
                tybar=select("Choose : ",["day's VS Cured and Confirmed","day's VS
Cured","day's VS Confirmed","end"])
                if tybar == "day's VS Cured and Confirmed":
                    put_markdown("## day's VS Cured and Confirmed")
                from pygame import mixer

                mixer.init()

                mixer.music.load("cuco.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.graph_objects as go

                from plotly.subplots import make_subplots

                fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},
{'type':'domain'}]])

                fig.add_trace(go.Pie(labels=x, values=y, name="Cured"),1, 1)

                fig.add_trace(go.Pie(labels=x, values=z, name="Confirmed"),1, 2)

                fig.update_traces(textposition='inside')

                fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                fig.update_layout(title_text=state+"'s Cured and Confirmed From
"+start+" To "+end)



        # Add annotations in the center of the donut pies.



                html = fig.to_html(include_plotlyjs="require", full_html=False)
                put_html(html)

                i=0

            elif tybar == "day's VS Cured":

                put_markdown("## day's VS Cured")

                from pygame import mixer

                mixer.init()

                mixer.music.load("cured.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

        #df = px.data.gapminder().query("continent == 'Asia'")

                fig = px.pie(values=y, names=x)

                fig.update_traces(textposition='inside')

                fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                fig.update_layout(title=state+"'s Cured From "+start+" To "+end)

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS Confirmed":
                put_markdown("## day's VS Confirmed")

                from pygame import mixer

                mixer.init()

                mixer.music.load("con.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

        #df = px.data.gapminder().query("continent == 'Asia'")

                fig = px.pie(values=z, names=x)

                fig.update_traces(textposition='inside')

                fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end)

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

             elif tybar == "end":

                i=1



        elif graphtype == 'Scatter':
```

```
from pygame import mixer

mixer.init()

mixer.music.load("scatter.mp3")

mixer.music.play()

put_markdown('## Scatter Graph')

time.sleep(2)

i=0

while i==0:

    time.sleep(3)

    from pygame import mixer

    mixer.init()

    mixer.music.load("choose.mp3")

    mixer.music.play()

    tybar=select("Choose : ",["day's VS Cured and Confirmed","day's VS
Cured","day's VS Confirmed","end"])

    if tybar == "day's VS Cured and Confirmed":

        put_markdown("## day's VS Cured and Confirmed")

        from pygame import mixer

        mixer.init()

        mixer.music.load("cuco.mp3")
        mixer.music.play()

        from pywebio.output import put_html

        import plotly.graph_objects as go

        fig = go.Figure()

        fig.add_trace(go.Scatter(x=y,y=x,name='Cured',

        marker=dict(

        color='rgba(156, 165, 196, 0.95)',

        line_color='rgba(156, 165, 196, 1.0)',

        )

        ))

        fig.add_trace(go.Scatter(x=z, y=x,name='Confirmed',

        marker=dict(

        color='rgba(204, 204, 204, 0.95)',

        line_color='rgba(217, 217, 217, 1.0)'

        )

        ))



        fig.update_traces(mode='markers', marker=dict(line_width=1,
symbol='circle', size=10))
```

```
    fig.update_layout(

    title=state+"'s Cured and Confirmed From "+start+" To "+end,

    margin=dict(l=140, r=40, b=50, t=80),

    legend=dict(

    font_size=10,

    yanchor='middle',

    xanchor='right',

    ),

    width=800,

    height=600,

    paper_bgcolor='white',

    plot_bgcolor='white',

    hovermode='closest',

    )

    html = fig.to_html(include_plotlyjs="require", full_html=False)

    put_html(html)

    i=0

elif tybar == "day's VS Cured":

    put_markdown("## day's VS Cured")
```

```
            from pygame import mixer

            mixer.init()

            mixer.music.load("cured.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.express as px



#df = pd.DataFrame(dict(school=x, salary=y,gender=x))

            col=[]

            for i in x:

                col.append('indianred')

# Use column names of df for the different parameters x, y, color, ...

            fig = px.scatter(x=x, y=y,color_discrete_sequence=col,

             title=state+"'s Cured From "+start+" To "+end,

             labels={"x":"day's","y":"Cured"} # customize axis label

            )



            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)
            i=0

        elif tybar == "day's VS Confirmed":

            put_markdown("## day's VS Confirmed")

            from pygame import mixer

            mixer.init()

            mixer.music.load("con.mp3")

            from pywebio.output import put_html

            import plotly.express as px


#df = pd.DataFrame(dict(school=x, salary=y,gender=x))

            col=[]

            for i in x:

                col.append('lightsalmon')

# Use column names of df for the different parameters x, y, color, ...

            fig = px.scatter(x=x, y=z,color_discrete_sequence=col,

             title=state+"'s Deaths From "+start+" To "+end)

            fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

#https://plotly.com/python/discrete-color/
```

```
xaxis_tickfont_size=14,

xaxis=dict(

title='days',

titlefont_size=16,

tickfont_size=14,

),

yaxis=dict(

title='Confirmed',

titlefont_size=16,

tickfont_size=14,

),

legend=dict(

x=0,

y=1.0,

bgcolor='rgba(255, 255, 255, 0)',

bordercolor='rgba(255, 255, 255, 0)'

),

barmode='group',

bargap=0.15, # gap between bars of adjacent location coordinates.
```

```
                    bargroupgap=0.1 # gap between bars of the same location
coordinate.
                    )


                    html = fig.to_html(include_plotlyjs="require", full_html=False)
                    put_html(html)
                    i=0
                elif tybar == "end":
                    i=1


# Use column names of df for the different parameters x, y, color, ...




        elif graphtype == 'Table':
            from pygame import mixer
            mixer.init()
            mixer.music.load("table.mp3")
            mixer.music.play()
            put_markdown('## Table')
```

```
time.sleep(2)

from pywebio.output import put_html

import plotly.graph_objects as go

from plotly.colors import n_colors

np.random.seed(1)



colors = n_colors('rgb(255, 200, 200)', 'rgb(200, 0, 0)', 9, colortype='rgb')

a = x

b = y

c = z

headerColor = 'grey'

col=[]

for i in x:

    col.append('white')

fig = go.Figure(data=[go.Table(

header=dict(

values=['<b>days</b>', '<b>Cured</b>', '<b>Confirmed</b>'],

line_color='darkslategray',fill_color='white',

align='center',font=dict(color='black', size=12)
```

```
            ),

            cells=dict(

            values=[a, b, c],

            line_color='darkslategray',

            fill_color =col,

            align='center', font=dict(color='black', size=11)

            ))

            ])



            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)

        elif confirmed == "Deaths vs Confirmed":

          y = sep_c["Deaths"]

          z = sep_c["Confirmed"]

          from pygame import mixer

          mixer.init()

          mixer.music.load("graph.mp3")

          mixer.music.play()

          graphtype=select("Choose The Type Of
    Graph",options=['Bar','Line','Pie','Scatter','Table'])#radio
          if graphtype == 'Bar':

            from pygame import mixer

            mixer.init()

            mixer.music.load("bar.mp3")

            mixer.music.play()

            put_markdown('## Bar Graph')

            time.sleep(2)

            i=0

            while i==0:

                time.sleep(3)

                from pygame import mixer

                mixer.init()

                mixer.music.load("choose.mp3")

                mixer.music.play()

                tybar=select("Choose :",["day's VS Deaths and Confirmed","day's VS
    Deaths","day's VS Confirmed","end"])

                if tybar == "day's VS Deaths and Confirmed":

                    put_markdown("## day's VS Deaths and Confirmed")

                    from pygame import mixer

                    mixer.init()
                    mixer.music.load("deco.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.graph_objects as go

                    fig = go.Figure()


    fig.add_trace(go.Bar(x=x,y=y,name="Deaths",marker_color='indianred'))


    fig.add_trace(go.Bar(x=x,y=z,name="Confirmed",marker_color='lightsalmon'))

                    fig.update_layout(title=state+"'s Deaths and Confirmed From
    "+start+" To "+end,

          #https://plotly.com/python/discrete-color/



                    xaxis_tickfont_size=14,

                    xaxis=dict(

                    title='days',

                    titlefont_size=16,

                    tickfont_size=14,

                    ),

                    yaxis=dict(
    		title="Deaths and Confirmed",
    		titlefont_size=16,
    		tickfont_size=14,
    		),
    		legend=dict(
    		x=0,
    		y=1.0,
    		bgcolor='rgba(255, 255, 255, 0)',
    		bordercolor='rgba(255, 255, 255, 0)'
    		),
    		barmode='group',
    		bargap=0.15, # gap between bars of adjacent location coordinates.
    		bargroupgap=0.1 # gap between bars of the same location
    coordinate.
    		)
    		html = fig.to_html(include_plotlyjs="require", full_html=False)
    		put_html(html)
    		i=0
    #https://pywebio.readthedocs.io/en/latest/
    #https://pywebio-charts.pywebio.online/?app=plotly
```

```
          elif tybar == "day's VS Deaths":

              put_markdown("## day's VS Deaths")

              from pygame import mixer

              mixer.init()

              mixer.music.load("death.mp3")

              mixer.music.play()

              from pywebio.output import put_html

              import plotly.express as px

              col=[]

              for i in x:

                  col.append('indianred')

  #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

              fig = px.bar(x=x,y=y,color_discrete_sequence=col)

              fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

              fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

              fig.update_layout(title=state+"'s Deaths From "+start+" To "+end,

  #https://plotly.com/python/discrete-color/



              xaxis_tickfont_size=14,
              xaxis=dict(

              title='days',

              titlefont_size=16,

              tickfont_size=14,

              ),

              yaxis=dict(

              title="Deaths",

              titlefont_size=16,

              tickfont_size=14,

              ),

              legend=dict(

              x=0,

              y=1.0,

              bgcolor='rgba(255, 255, 255, 0)',

              bordercolor='rgba(255, 255, 255, 0)'

              ),

              barmode='group',

              bargap=0.15, # gap between bars of adjacent location coordinates.

              bargroupgap=0.1 # gap between bars of the same location
coordinate.
```

```
                )

                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS Confirmed":

                put_markdown("## day's VS Confirmed")

                from pygame import mixer

                mixer.init()

                mixer.music.load("con.mp3")

                mixer.music.play()

                col=[]

                for i in x:

                    col.append('lightsalmon')

        #df = px.data.gapminder().query("continent == 'Europe' and year == 2007
and pop > 2.e6")

                fig = px.bar(x=x,y=z,color_discrete_sequence=col)

                fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')

                fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

                fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

                xaxis_tickfont_size=14,
                xaxis=dict(
                title='days',
                titlefont_size=16,
                tickfont_size=14,
                ),
                yaxis=dict(
                title="Confirmed",
                titlefont_size=16,
                tickfont_size=14,
                ),
                legend=dict(
                x=0,
                y=1.0,
                bgcolor='rgba(255, 255, 255, 0)',
                bordercolor='rgba(255, 255, 255, 0)'
                ),
                barmode='group',
                bargap=0.15, # gap between bars of adjacent location coordinates.
                bargroupgap=0.1 # gap between bars of the same location coordinate.
```

```
            )
            html = fig.to_html(include_plotlyjs="require", full_html=False)
            put_html(html)
            i=0
        elif tybar == "end":
            i=1
elif graphtype == 'Line':
    from pygame import mixer
    mixer.init()
    mixer.music.load("line.mp3")
    mixer.music.play()
    put_markdown('## Line Graph')
    time.sleep(2)
    i=0
    while i==0:
        time.sleep(3)
        from pygame import mixer
        mixer.init()
        mixer.music.load("choose.mp3")
```

```
                    mixer.music.play()

                    tybar=select("Choose : ",["day's VS Deaths and Confirmed","day's VS
Deaths","day's VS Confirmed","end"])

                    if tybar == "day's VS Deaths and Confirmed":

                        put_markdown("## day's VS Deaths and Confirmed")

                        from pygame import mixer

                        mixer.init()

                        mixer.music.load("deco.mp3")

                        mixer.music.play()

                        from pywebio.output import put_html

                        import plotly.express as px

                        import plotly.graph_objects as go

        #df = px.data.gapminder().query("continent=='Oceania'")

        #fig = px.line(df,x=y, y=z,color_discrete_sequence=col )

                        fig = go.Figure()


fig.add_trace(go.Line(x=x,y=y,name='Deaths',marker_color='indianred'))


fig.add_trace(go.Line(x=x,y=z,name='Confirmed',marker_color='lightsalmon'))

                        fig.update_layout(title=state+"'s Deaths and Confirmed From
"+start+" To "+end,
                        xaxis_tickfont_size=14,

                        xaxis=dict(

                        title='days',

                        titlefont_size=16,

                        tickfont_size=14,

                        ),

                        yaxis=dict(

                        title='Deaths and Confirmed',

                        titlefont_size=16,

                        tickfont_size=14,

                        ),

                        legend=dict(x=0,y=1.0,bgcolor='rgba(255, 255, 255, 0)',

                        bordercolor='rgba(255, 255, 255, 0)'),

                        barmode='group',

                        bargap=0.15, # gap between bars of adjacent location coordinates.

                        bargroupgap=0.1 # gap between bars of the same location
coordinate.

                        )

                        html = fig.to_html(include_plotlyjs="require", full_html=False)

                        put_html(html)
```

```
            i=0
        elif tybar == "day's VS Deaths":
            put_markdown("## day's VS Deaths")
            from pygame import mixer
            mixer.init()
            mixer.music.load("death.mp3")
            mixer.music.play()
            from pywebio.output import put_html
            import plotly.express as px
            col=[]
            for i in x:
                col.append('indianred')
#df = px.data.gapminder().query("continent=='Oceania'")
            fig = px.line(x=x, y=y,color_discrete_sequence=col )
            fig.update_layout(title=state+"'s Deaths From "+start+" To "+end,
#https://plotly.com/python/discrete-color/


            xaxis_tickfont_size=14,
            xaxis=dict(
title='days',

titlefont_size=16,

tickfont_size=14,

),

yaxis=dict(

title='Deaths',

titlefont_size=16,

tickfont_size=14,

),

legend=dict(

x=0,

y=1.0,

bgcolor='rgba(255, 255, 255, 0)',

bordercolor='rgba(255, 255, 255, 0)'

),

barmode='group',

bargap=0.15, # gap between bars of adjacent location coordinates.

bargroupgap=0.1 # gap between bars of the same location
coordinate.

)
```

```
            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)

            i=0

        elif tybar == "day's VS Confirmed":

            put_markdown("## day's VS Confirmed")

            from pygame import mixer

            mixer.init()

            mixer.music.load("con.mp3")

            mixer.music.play()

            from pywebio.output import put_html

            import plotly.express as px

            col=[]

            for i in x:

                col.append('lightsalmon')

#df = px.data.gapminder().query("continent=='Oceania'")

            fig = px.line(x=x, y=z,color_discrete_sequence=col )

            fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

#https://plotly.com/python/discrete-color/
```

```
xaxis_tickfont_size=14,

xaxis=dict(

title='days',

titlefont_size=16,

tickfont_size=14,

),

yaxis=dict(

title='Confirmed',

titlefont_size=16,

tickfont_size=14,

),

legend=dict(

x=0,

y=1.0,

bgcolor='rgba(255, 255, 255, 0)',

bordercolor='rgba(255, 255, 255, 0)'

),

barmode='group',

bargap=0.15, # gap between bars of adjacent location coordinates.
```

```
                    bargroupgap=0.1 # gap between bars of the same location
coordinate.
                    )
                    html = fig.to_html(include_plotlyjs="require", full_html=False)
                    put_html(html)
                    i=0
                elif tybar == 'end':
                    i=1
        elif graphtype == 'Pie':
            from pygame import mixer
            mixer.init()
            mixer.music.load("pie.mp3")
            mixer.music.play()
            put_markdown('## Pie Graph')
            time.sleep(2)
            i=0
            while i==0:
                time.sleep(3)
                from pygame import mixer
                mixer.init()
                mixer.music.load("choose.mp3")

                mixer.music.play()

                tybar=select("Choose : ",["day's VS Deaths and Confirmed","day's VS
Deaths","day's VS Confirmed","end"])

                if tybar == "day's VS Deaths and Confirmed":

                    put_markdown("## day's VS Deaths and Confirmed")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("deco.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.graph_objects as go

                    from plotly.subplots import make_subplots

                    fig = make_subplots(rows=1, cols=2, specs=[[{'type':'domain'},
{'type':'domain'}]])

                    fig.add_trace(go.Pie(labels=x, values=y, name="Deaths"),1, 1)

                    fig.add_trace(go.Pie(labels=x, values=z, name="Confirmed"),1, 2)

                    fig.update_traces(textposition='inside')

                    fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')
```

```
                fig.update_layout(title_text=state+"'s Deaths and Confirmed From
"+start+" To "+end)


    # Add annotations in the center of the donut pies.


                html = fig.to_html(include_plotlyjs="require", full_html=False)

                put_html(html)

                i=0

            elif tybar == "day's VS Deaths":

                put_markdown("## day's VS Deaths")

                from pygame import mixer

                mixer.init()

                mixer.music.load("death.mp3")

                mixer.music.play()

                from pywebio.output import put_html

                import plotly.express as px

    #df = px.data.gapminder().query("continent == 'Asia'")

                fig = px.pie(values=y, names=x)

                fig.update_traces(textposition='inside')
```

```
                  fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                  fig.update_layout(title=state+"'s Deaths From "+start+" To "+end)

                  html = fig.to_html(include_plotlyjs="require", full_html=False)

                  put_html(html)

                  i=0

              elif tybar == "day's VS Confirmed":

                  put_markdown("## day's VS Confirmed")

                  from pygame import mixer

                  mixer.init()

                  mixer.music.load("con.mp3")

                  mixer.music.play()

                  from pywebio.output import put_html

                  import plotly.express as px

       #df = px.data.gapminder().query("continent == 'Asia'")

                  fig = px.pie(values=z, names=x)

                  fig.update_traces(textposition='inside')

                  fig.update_layout(uniformtext_minsize=12,
uniformtext_mode='hide')

                  fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end)
```

```
            html = fig.to_html(include_plotlyjs="require", full_html=False)

            put_html(html)

            i=0

        elif tybar == "end":

            i=1



elif graphtype == 'Scatter':

    from pygame import mixer

    mixer.init()

    mixer.music.load("scatter.mp3")

    mixer.music.play()

    put_markdown('## Scatter Graph')

    time.sleep(2)

    i=0

    while i==0:

        time.sleep(3)

        from pygame import mixer

        mixer.init()

        mixer.music.load("choose.mp3")
```

```
                mixer.music.play()

                tybar=select("Choose : ",["day's VS Deaths and Confirmed","day's VS
Deaths","day's VS Confirmed","end"])

                if tybar == "day's VS Deaths and Confirmed":

                    put_markdown("## day's VS Deaths and Confirmed")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("deco.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.graph_objects as go

                    fig = go.Figure()

                    fig.add_trace(go.Scatter(x=y,y=x,name='Deaths',

                    marker=dict(

                    color='rgba(156, 165, 196, 0.95)',

                    line_color='rgba(156, 165, 196, 1.0)',

                    )

                    ))

                    fig.add_trace(go.Scatter(x=z, y=x,name='Confirmed',

                    marker=dict(
                    color='rgba(204, 204, 204, 0.95)',

                    line_color='rgba(217, 217, 217, 1.0)'

                    )

                    ))



                    fig.update_traces(mode='markers', marker=dict(line_width=1,
symbol='circle', size=10))



                    fig.update_layout(

                    title=state+"'s Deaths and Confirmed From "+start+" To "+end,

                    margin=dict(l=140, r=40, b=50, t=80),

                    legend=dict(

                    font_size=10,

                    yanchor='middle',

                    xanchor='right',

                    ),

                    width=800,

                    height=600,

                    paper_bgcolor='white',

                    plot_bgcolor='white',
                    hovermode='closest',

                    )

                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "day's VS Deaths":

                    put_markdown("## day's VS Deaths")

                    from pygame import mixer

                    mixer.init()

                    mixer.music.load("death.mp3")

                    mixer.music.play()

                    from pywebio.output import put_html

                    import plotly.express as px



        #df = pd.DataFrame(dict(school=x, salary=y,gender=x))

                    col=[]

                    for i in x:

                        col.append('indianred')

        # Use column names of df for the different parameters x, y, color, ...
```

```
            fig = px.scatter(x=x, y=y,color_discrete_sequence=col,
             title=state+"'s Deaths From "+start+" To "+end,
             labels={"x":"day's","y":"Deaths"} # customize axis label
            )

            html = fig.to_html(include_plotlyjs="require", full_html=False)
            put_html(html)
            i=0
        elif tybar == "day's VS Confirmed":
            put_markdown("## day's VS Confirmed")
            from pygame import mixer
            mixer.init()
            mixer.music.load("con.mp3")
            from pywebio.output import put_html
            import plotly.express as px

    #df = pd.DataFrame(dict(school=x, salary=y,gender=x))
            col=[]
            for i in x:
                col.append('lightsalmon')

# Use column names of df for the different parameters x, y, color, ...

            fig = px.scatter(x=x, y=z,color_discrete_sequence=col,

             title=state+"'s Confirmed From "+start+" To "+end)

            fig.update_layout(title=state+"'s Confirmed From "+start+" To "+end,

#https://plotly.com/python/discrete-color/



            xaxis_tickfont_size=14,

            xaxis=dict(

            title='days',

            titlefont_size=16,

            tickfont_size=14,

            ),

            yaxis=dict(

            title='Confirmed',

            titlefont_size=16,

            tickfont_size=14,

            ),

            legend=dict(
                    x=0,

                    y=1.0,

                    bgcolor='rgba(255, 255, 255, 0)',

                    bordercolor='rgba(255, 255, 255, 0)'

                    ),

                    barmode='group',

                    bargap=0.15, # gap between bars of adjacent location coordinates.

                    bargroupgap=0.1 # gap between bars of the same location
coordinate.

                    )



                    html = fig.to_html(include_plotlyjs="require", full_html=False)

                    put_html(html)

                    i=0

                elif tybar == "end":

                    i=1


# Use column names of df for the different parameters x, y, color, ...
```

```
elif graphtype == 'Table':

    from pygame import mixer

    mixer.init()

    mixer.music.load("table.mp3")

    mixer.music.play()

    put_markdown('## Table')

    time.sleep(2)

    from pywebio.output import put_html

    import plotly.graph_objects as go

    from plotly.colors import n_colors

    np.random.seed(1)



    colors = n_colors('rgb(255, 200, 200)', 'rgb(200, 0, 0)', 9, colortype='rgb')

    a = x

    b = y

    c = z

    headerColor = 'grey'

    col=[]
```

```
for i in x:
    col.append('white')
fig = go.Figure(data=[go.Table(
header=dict(
values=['<b>days</b>', '<b>Deaths</b>', '<b>Confirmed</b>'],
line_color='darkslategray',fill_color='white',
align='center',font=dict(color='black', size=12)
),
cells=dict(
values=[a, b, c],
line_color='darkslategray',
fill_color =col,
align='center', font=dict(color='black', size=11)
))
])


html = fig.to_html(include_plotlyjs="require", full_html=False)
put_html(html)
```

```
def mltom():

    df = pd.read_csv('https://covid19.who.int/WHO-COVID-19-global-data.csv')

    #put_text(df.head(10))



    country = df['Country'].unique()

    from pygame import mixer

    mixer.init()

    mixer.music.load("country.mp3")

    mixer.music.play()

    user_info=select("Choose The Country",country)#radio

    date = df["Date_reported"].unique()

    from pygame import mixer

    mixer.init()

    mixer.music.load("date.mp3")

    mixer.music.play()

    start = select("Start Date：",date)

    put_info('Start Date: ',start)

    from pygame import mixer

    mixer.init()
    mixer.music.load("date1.mp3")

    mixer.music.play()

    end = select("End Date : ",date)

    put_info('End Date: ',end)

    df_c = df[df.Country == user_info ]

    sep_c =  df_c.loc[(df_c["Date_reported"] >= start) & (df_c["Date_reported"] <=
end)]

    x = sep_c["Date_reported"]

    y = sep_c["New_cases"]

    z = sep_c["New_deaths"]



#preparing data

    data = sep_c["New_cases"]

    scaler = MinMaxScaler(feature_range=(0,1))

    scaled_data = scaler.fit_transform(data.values.reshape(-1,1))



#training the data

    from pygame import mixer

    mixer.init()

    mixer.music.load("test.mp3")
    mixer.music.play()

    put_success('Testing the data....................')

    predicion_days = 60

    x_train= []

    y_train = []

    for x in range(predicion_days,len(scaled_data)):

        x_train.append(scaled_data[x-predicion_days:x,0])

        y_train.append(scaled_data[x,0])

    x_train,y_train=np.array(x_train),np.array(y_train)

    x_train = np.reshape(x_train,(x_train.shape[0],x_train.shape[1],1))



#creating prediction network

    model = Sequential()


model.add(LSTM(units=50,return_sequences=True,input_shape=(x_train[1].sha
pe[1],1)))

    model.add(Dropout(0.2))

    model.add(LSTM(units=50,return_sequences=True))

    model.add(Dropout(0.2))
    model.add(LSTM(units=50))

    model.add(Dropout(0.2))

    model.add(Dense(units=1))




    model.compile(optimizer='adam', loss='mean_squared_error')

    model.fit(x_train,y_train,epochs=25,batch_size=32)


#testing the model


    test_start = start

    test_end = end

    test_data =data

    actual_prices = data

    total_dataset = pd.concat((data,test_data),axis=0)



    model_inputs = total_dataset[len(total_dataset) - len(test_data) -
predicion_days:].values

    model_inputs = model_inputs.reshape(-1,1)

    model_inputs = scaler.fit_transform(model_inputs)
    x_test = []

    for x in range(predicion_days, len(model_inputs)):

        x_test.append(model_inputs[x-predicion_days:x,0])

    x_test = np.array(x_test)

    x_test = np.reshape(x_test,(x_test.shape[0],x_test.shape[1],1))


#predicting the data

    prediction_prices = model.predict(x_test)

    prediction_prices = scaler.inverse_transform(prediction_prices)




#predict next day

    from pygame import mixer

    mixer.init()

    mixer.music.load("pre.mp3")

    mixer.music.play()

    put_success('Predicting Next Day cases')

    time.sleep(2)
    real_data = [model_inputs[len(model_inputs) + 1 -
predicion_days:len(model_inputs) + 1,0]]

    real_data = np.array(real_data)

    real_data = np.reshape(real_data,(real_data.shape[0],real_data.shape[1],1))



    prediction = model.predict(real_data)

    predict = scaler.inverse_transform(prediction)



    prediction_prices_len =len(prediction_prices)

    comparing = prediction_prices[prediction_prices_len - 1]

    put_info('Tomorrow cases will be : ',int(predict))

    combine = 'covid - 19'

    if comparing >= predict:

        from pygame import mixer

        mixer.init()

        mixer.music.load("tom.mp3")

        mixer.music.play()

        popup(combine , 'Tomorrow"S cases will be less than today')

        time.sleep(4)

    else:
        from pygame import mixer

        mixer.init()

        mixer.music.load("tomm.mp3")

        mixer.music.play()

        popup(combine , 'Tomorrow"S cases will be greater than today')

        time.sleep(4)
```

```
if __name__ == '__main__':

#https://towardsdatascience.com/pywebio-write-interactive-web-app-in-script-
way-using-python-14f50155af4e

#https://pywebio.readthedocs.io/en/latest/output.html#pywebio.output.put_image

#https://www.tutorialspoint.com/python/tk_fonts.htm

    put_image(open('logo.jpg', 'rb').read())

    put_image(open('logo6.png', 'rb').read())
```

```
put_text('\n')

put_image(open('cc.jpg', 'rb').read())



def run(string):

    regex = re.compile('[@_!#$%^&*()<>?/\|}{~:0-9]')

    if(regex.search(string) == None):

        return True

    else:

        return False

i=0

put_text('\n')

from pygame import mixer

mixer.init()

mixer.music.load("what.mp3")

mixer.music.play()

while i==0:

    name=input("What Is Your Name?")

    if(not name):

        toast('Should Not Be Empty',color='red')
```

```
            from pygame import mixer
            mixer.init()
            mixer.music.load("name.mp3")
            mixer.music.play()
        else:
            if run(name)== False:
                toast("Name Should Be In Alphabet's Can Not Contain Any Special
Character's Or Numerical's")
                from pygame import mixer
                mixer.init()
                mixer.music.load("whatis.mp3")
                mixer.music.play()
            else:
                i=1
    i=0
    from pygame import mixer
    mixer.init()
    mixer.music.load("ageis.mp3")
    mixer.music.play()
    while i==0:
        age=input_group('What Is Your age?',[input(name='age',type=FLOAT)])

        if(not age['age']):

            toast('Should Not Be Empty',color='red')

            from pygame import mixer

            mixer.init()

            mixer.music.load("name.mp3")

            mixer.music.play()

        elif not type(age['age']) == int:

            toast("Age Should Be In Numerical's Can Not Contain Any Special
Character's or Alphabet's")

            from pygame import mixer

            mixer.init()

            mixer.music.load("age.mp3")

            mixer.music.play()

        elif age['age'] <= 0:

            toast("Age Should Be In Numerical's Can Not Contain Any Special
Character's or Alphabet's")

            from pygame import mixer

            mixer.init()

            mixer.music.load("age.mp3")
            mixer.music.play()

        else:

            i=1

    if age['age']<=18:

        from pygame import mixer

        mixer.init()

        mixer.music.load("sorry.mp3")

        mixer.music.play()

        toast("Sorry,"+name+" You Can't Access Our Website",color='red')

        time.sleep(3)


    else:

        from pygame import mixer

        mixer.init()

        mixer.music.load("congrats.mp3")

        mixer.music.play()

        toast("Congrats,"+name+" You Can Use Our Website!")#put_success

        time.sleep(3)

        #put_text("\n\n")
```

```
#put_info('Hi, Welcome To Our Website '+ name)

#put_image(open('data2.jpg', 'rb').read(),width='180px', height='180px')

put_text('\n')

put_image(open('om.jpeg', 'rb').read(),width='150px', height='150px')

from pygame import mixer

mixer.init()

mixer.music.load("my.mp3")

mixer.music.play()

put_markdown('## myself:kolla.om vivek,BE-CSE,3rd year')

time.sleep(5)

i=0

while i==0:

    from pygame import mixer

    mixer.init()

    mixer.music.load("which.mp3")

    mixer.music.play()

    app = select("Which One Do You Want To Go For : ",["All
Countries","India (state's)","Predicting Next Day Covid Case's"])

    if app == "All Countries":

        covid_app()
```

```
          elif app == "India (state's)":
             india_app()
          elif app == "Predicting Next Day Covid Case's":
             mltom()
          #time.sleep(1)
          from pygame import mixer
          mixer.init()
          mixer.music.load("do.mp3")
          mixer.music.play()
          rad=radio("Do You Want To Continue",options=['yes','no'])
          if rad=='yes':
             i=0
          else:
             i=1
from pygame import mixer
mixer.init()
mixer.music.load("stay.mp3")
mixer.music.play()
toast('Stay Home , Stay Safe')
```

```
time.sleep(2)

from pygame import mixer

mixer.init()

mixer.music.load("thanks.mp3")

mixer.music.play()

toast('Thanks, For visiting our web-site!')#put_success

time.sleep(3)

#pywebio.output.clear()

clear(scope=None)

from pygame import mixer

mixer.init()

mixer.music.load("fur.mp3")

mixer.music.play()

put_markdown("# Further Covid-19 Related information is below :")

put_link("MOHFW Main Website", url="https://www.mohfw.gov.in/", app=None,
new_window=True, scope=None)

put_text("\n")

put_link("Covid-19 Live Dashboard",
url="https://news.google.com/covid19/map?hl=en-
IN&mid=%2Fm%2F03rk0&gl=IN&ceid=IN%3Aen", app=None,
new_window=True, scope=None)
```

```
put_text("\n")

put_link("covid-19 contact tracing app for India",
url="https://play.google.com/store/apps/details?id=nic.goi.aarogyasetu",
app=None, new_window=True, scope=None)

put_text("\n")

put_link("Andhra Pradesh sachivalayam center for vaccination ",
url="https://www.google.com/search?client=ms-android-samsung-
ss&tbs=lf:1,lf_ui:2&tbm=lcl&sxsrf=AOaemvKwTWIbtpoT477Nv6bq0hhz4nGLNA:
1636384621085&q=ap+sachivalayam+center&rflfq=1&num=10&ved=2ahUKEwj
Pybjgh4n0AhVX7nMBHVeNACEQtgN6BAgREAc#rlfi=hd:;si:;mv:[[17.9456331,8
3.5551527],[14.2120362,77.1918017]];tbs:lrf:!1m4!1u3!2m2!3m1!1e1!1m4!1u2!2
m2!2m1!1e1!2m1!1e2!2m1!1e3!3sIAE,lf:1,lf_ui:2", app=None,
new_window=True, scope=None)

put_text("\n")

put_info("""To prevent the spread of COVID-19:

 -> Maintain a safe distance from others (at least 1 metre), even if they don’t
appear to be sick.

 -> Wear a mask in public, especially indoors or when physical distancing is not
possible.

 -> Choose open, well-ventilated spaces over closed ones. Open a window if
indoors.

 -> Clean your hands often. Use soap and water, or an alcohol-based hand rub.

 -> Get vaccinated when it’s your turn. Follow local guidance about vaccination.
```

-> Cover your nose and mouth with your bent elbow or a tissue when you cough or sneeze.

-> Stay home if you feel unwell.""")