A Project report on

TRACKING CORONA VIRUS

A Dissertation submitted to JNTU Hyderabad in partial

fulfillment of the academic requirements for the award of the degree.

Bachelor of Technology
In
Computer Science and Engineering

Submitted by

N. THOMAS (19H51A0550)
N. SAHITHI (19H51A0551)
R. RAKSHITH (19H51A0521)

Under the esteemed guidance of

Dr. D Murali

(HOD Dept of Cyber security)

Department of Computer Science and Engineering

CMR COLLEGE OF ENGINEERING AND TECHNOLOGY
(An Autonomous Institution under UGC & JNTUH, Approved by AICTE, Permanently Affiliated to JNTUH, Accredited by NBA.)
KANDLAKOYA, MEDCHAL ROAD, HYDERABAD - 501401.

2019- 2023

CMR COLLEGE OF ENGINEERING & TECHNOLOGY

KANDLAKOYA, MEDCHAL ROAD, HYDERABAD – 501401

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled **"TRACKING CORONA VIRUS"** is a bonafied work done by **Rakshith (19H51A0521), Thomas (19H51A0550), Sahithi (19H51A0551),** in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Information Technology, submitted to the Department of Information Technology, CMR College of Engineering & Technology, Hyderabad during the academic year 2019-2023. The Results embodies in this project report have not been submitted to any other university or institute for the award of any degree.

Dr. D. Murali
Professor and HOD
Dept. of Cyber Security

Dr. K. Vijaya Kumar
Professor and HOD
Dept. of CSE

Submitted for viva voice Examination held on _____

External Examiner

ABSTRACT

Corona virus Disease (COVID-19) is an infectious disease caused by a newly discovered virus called Corona virus. This COVID-19 is declared as a pandemic by WHO. We have seen many disasters and hazards which effects several lives and areas, but this COVID-19 makes a drastic change to the world by effecting many lives. It has been changed the normal life citizens to a new wave form. It affects different people in different ways, its impact has been deleterious from both health perspective and an economic one which cannot be expected by us. So, let us have some idea on how it effects on the world, what are the death rates, recoveries in each place by the present technology. Now we are giving analysis on the data of COVID-19 for the awareness by statistical reports on its exponential growth and to take preventive measures to be safe from COVID-19 and to know what happening around us and how it effects. So, we are using data driven technologies to make a case study on COVID-19. We are going to visualize the data which we collected using Python packages like plotly, folium, seaborn etc. The Data Set we collected is country wise and it is taken from the 2019 Novel Corona virus Visual Dashboard operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE).

TABLE OF CONTENTS

CHAPTERS			DESCRIPTION	PAGE NUMBER
			List of figures	
			Abstract	
1			**Introduction**	
	1.1		Description	1
	1.2		Scope	1
	1.3		Objectives	2
2			**Background Work**	
	2.1		Existing solutions	3
	2.2		Disadvantages	4
3			**Proposed System**	
	3.1		Description	5
		3.1.1	Advantages of Proposed solutions	5
	3.2		Requirements Specifications	
		3.2.1	Requirements for software	5
	3.3		Modules Description	6
	3.4		Libraries	6-7
	3.5		Covid19 Dataset	7
	3.6		Checking NULL values	7
	3.7		Use case Diagram	8
4			**Designing**	
	4.1		UML Diagram	9
	4.2		ER Diagram	9
5			**Results and Conclusion**	
	5.1		Implementation	10-16
	5.2		Execution Screens	16-20
6			**Conclusion And Future work**	21
7			**References**	22

INTRODUCTION

1.1 DESCRIPTION

- A novel corona virus (COVID-19) was identified in 2019 in Wuhan, China. This is a new corona virus that has not been previously identified in humans. A novel corona virus (COVID-19) was identified in 2019 in Wuhan, China. This is a new corona virus that has not been previously identified in humans.
- Corona viruses are a large family of viruses that are known to cause illness ranging from the common cold to more severe diseases such as Middle East Respiratory Syndrome (MERS) and Severe Acute Respiratory Syndrome (SARS).
- This course provides a general introduction to COVID-19 and emerging respiratory viruses and is intended for public health professionals, incident managers and personnel working for the United Nations, international organizations and NGOs.
- As the official disease name was established after material creation, any mention of nCoV refers to COVID-19, the infectious disease caused by the most recently discovered corona virus.
- In this project, you will learn how to preprocess and merge datasets to calculate needed measures and prepare them for an Analysis. we are going to work with the COVID19 dataset, which consists of the data related to the cumulative number of confirmed cases, per day, in each Country.

1.2 SCOPE

- This application can be used by any people to know the analysis of covid data by maintaining the records, where people can check the available current details in the application.

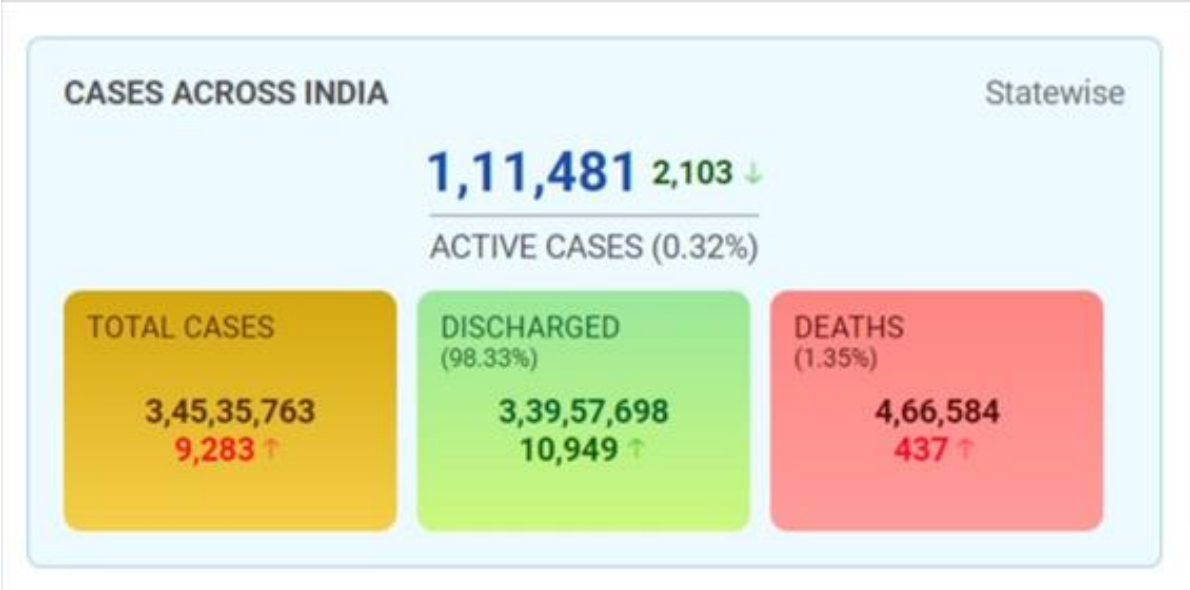
1.3 OBJECTIVES

- Confirmed tested cases of Corona virus infection
- We going to compare with other diseases like Ebola, H1N1 etc...
- The number of people who have reportedly died while sick with Corona virus.
- We are going to predict the number of cases in feature too if this condition continues further
- The number of people who have reportedly recovered from it.

BACKGROUND WORK

2.1 EXSISTING SOLUTION

GOVERNMENT COVID TRACKER



Government of India is taking all necessary steps to ensure that we are prepared well to face the challenge and threat posed by the growing pandemic of COVID-19 the Corona Virus. The most important factor in preventing the spread of the Virus locally is to empower the citizens with the right information and taking precautions as per the advisories being issued by Ministry of Health & Family Welfare. Governments are taking a wide range of measures in response to the COVID-19 outbreak. This tool aims to track and compare policy responses around the world, rigorously and consistently. These policies are recorded on a scale to reflect the extent of government action, and scores are aggregated into a suite of policy indices. The data can help decision-makers and citizens understand governmental responses in a consistent way, aiding efforts to fight the pandemic

2.2 DISADVANTAGES

- Thirty-five percent said they would be comfortable with the sharing of their location data for this purpose, 10 percentage points more than the share who said they were OK with the sharing of this data just to allow the government to better track.
- over $200 million total for failing to protect consumers' geolocation data after third-party groups reportedly abused the aggregated data shared with them by the carriers.
- If tracing data anonymity is compromised, people could be traced longitudinally. the current digital tracing efforts as a handover of confidentiality to the government similar to when the government increased its supervisory authority

PROPOSED SYSTEM

3.1 DESCRIPTION

In our proposed solution i.e., Tracking corona virus is the data analysis of covid-19 through maps, graphs by collecting the data from various sources. Here we give day to day information like increase in covid cases, deaths, recovery etc. Here we have some idea on how it effects on the world, what are the death rates, recoveries in each place by the present technology. Now we are giving analysis on the data of COVID-19 for the awareness by statistical reports on its exponential growth and to take preventive measures to be safe from COVID-19 and to know what happening around us and how it effects. The maps represent the worldwide covid cases analysis and cases over a specific time in a specific country. The graphs which are presented represents the No. of deaths, recovered and deaths/recovered per 100 cases. It also shows the number of new cases per day over various countries.

3.1.1. ADVANTAGES OF PROPOSED SYSTEM

- Simple and easy to operate.
- It is very fast and clear.
- Not easy for data loss.
- we can predict the future conditions.
- Need short time to find any update or data related to covid.
- Find covid news, cases, analysis.
- It helps the people all over the world to take certain precautions and safety measures.
- Saves time and reduces work.
- We can predict number of deaths, recoveries.
- Customized reports for better analysis.

3.2 REQUIREMENT SPECIFICATIONS

3.2.1 SOFTWARE INTERFACE

- Operating system : Windows 11
- DBMS : MongoDB.
- Compiler : Visual studio code && Jupiter
- Front-end tools :React

3.3 MODULES DESCRIPTION

• Covid News:

It gives the day-to-day covid news updates all over the world through the news UPI.

• Covid Cases:

It gives Number of deaths, covid recoveries and covid cases in each country and all over the world.

• Covid Analysis:

1.It gives the detailed data of worldwide covid cases through plotlygraphs, area maps, scatter graphs.

2.It compares covid with other diseases.

3.The graphs show the number of death, cases, recover per hundred.

3.4 Libraries

- **Plotly Python** = The Plotly Python library is an **interactive open-source library.** It can plot various types of graphs and charts like scatter plots, line charts, bar charts, box plots, histograms, pie charts, etc.
- **Pandas** = Pandas has been one of the most popular and favorite data science tools used in Python programming language **for analysis**. And Pandas is

seriously a game changer when it comes to cleaning, transforming, manipulating and analyzing data. In simple terms, Pandas helps to clean the mess.

- **NumPy** = Using NumPy, mathematical and logical operations on arrays can be performed.

3.5 Importing COVID19 dataset and preparing it for the analysis by dropping columns and aggregating rows

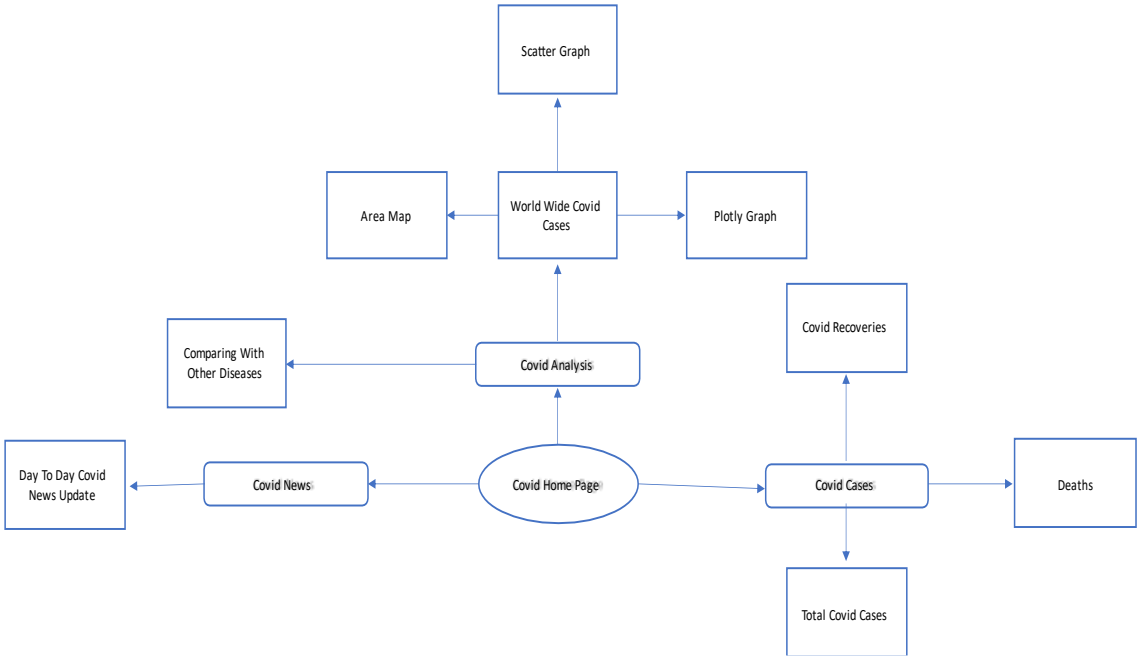
- The dataset used here are collected from Novel Coronavirus Visual Dashboard operated by the Johns Hopkins University Center for Systems Science and Engineering (JHU CSSE).
- The dataset are: **country_daywise, country_wise, daywise, c19**.
- We also created dataset **confirmed, recovered, deaths** from the dataset **c19** and grouped them together by Date.

3.6 Checking for null values and filling the dummy values.

- Checking our data contains any null values.
- If our data contains null value, it will not give the desired output so we are going to replace the null values with any value or we can delete the column or the row.
- To check our data contains null values we can use “isnull ()” which will output the number of null values in each column.

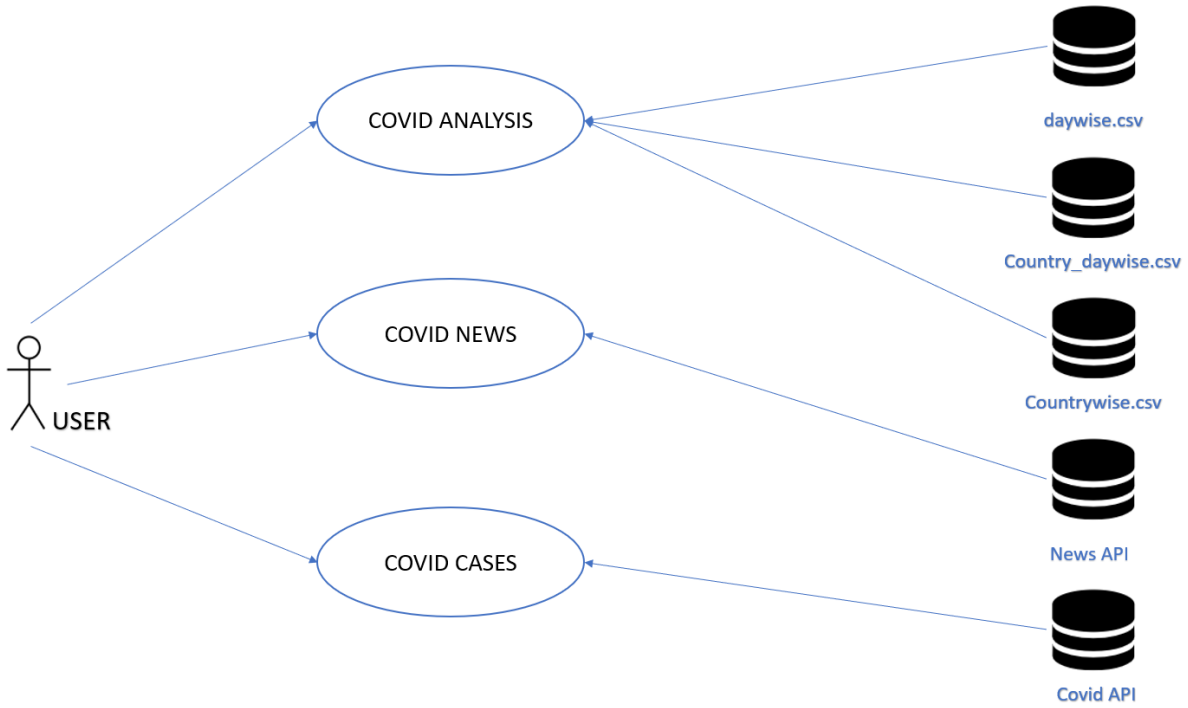
In our dataset c19 we are having null value to the column “Province/State” so we are filling it with empty string.

3.7 SYSTEM ARCHITECTURE/ USECASE DIAGRAM

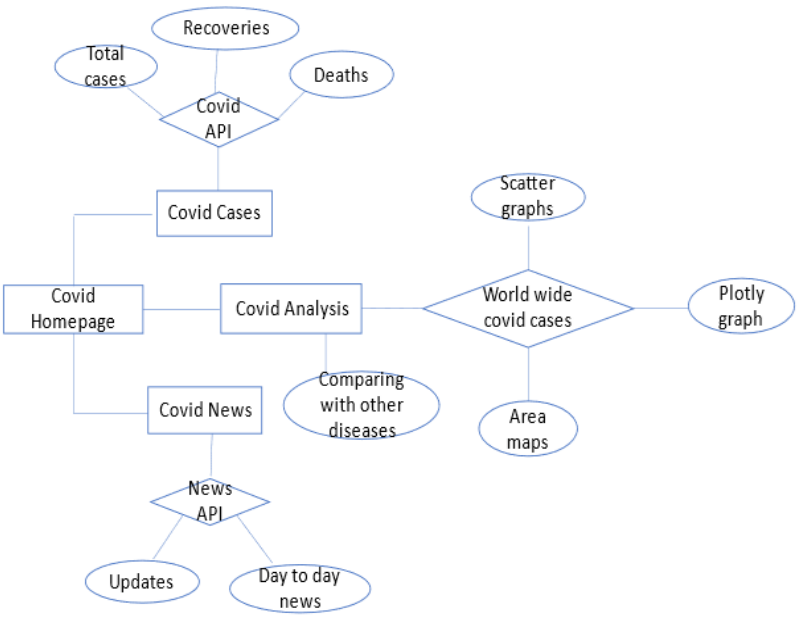


DESIGNING

4.1 UML DIAGRAM



4.2 ER DIAGRAM



RESULT AND DISSCUSION

5.1 IMPLEMENTATION

```
#installing lybrabries
#folium is use to vizualise the covid cases.
#ploty use to plot dynamic plots or we can say to create interactive plots.
```

In [2]:

```
import plotly.express as px
import plotly.graph_objects as go
import plotly.figure_factory as ff
from plotly.subplots import make_subplots

import folium
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline

import math
import random
from datetime import timedelta          # used in coustam date format
import warnings
warnings.filterwarnings('ignore')              #to ignore the warnings

#color pallette
cnf = '#393e46'
dth = '#ff2e63'
rec = '#21bf73'
act = '#fe9801'
```

In [ ]:

In [3]:

```
#Data set Preparation
```

In [4]:

```
df = pd.read_csv("c19.csv",parse_dates=['Date'])
```

In [5]:

```
df['Province/State'] = df['Province/State'].fillna("")
df
country_daywise = pd.read_csv("country_daywise.csv")
countrywise = pd.read_csv("countrywise.csv")
daywise = pd.read_csv("daywise.csv")
```

In [7]:

```
country_daywise.head()
```

In [8]:

```
countrywise.head()
```

In [9]:

```
daywise.head()
```

In [10]:

```
 #confirmed || recovered || deaths per day
```

In [11]:

```
confirmed = df.groupby('Date').sum()['Confirmed'].reset_index()
recovered = df.groupby('Date').sum()['Recovered'].reset_index()
deaths = df.groupby('Date').sum()['Deaths'].reset_index()
```

In [12]:

```
confirmed.head()
```

In [13]:

```
recovered.head()
```

In [14]:

```
deaths.head()
```

In [15]:

```
# checking null valuse in "df"
```

In [16]:

```
df.isnull().sum()
```

In [17]:

```
#SCATTER GRAPH --- imported from plotly.graph_objects
```

In [18]:

```
fig = go.Figure()

fig.add_trace(go.Scatter(x = confirmed['Date'], y=confirmed['Confirmed'],
mode='lines+markers', name = 'Confirmed', line = dict(color = "Orange", width=2)))
fig.add_trace(go.Scatter(x = recovered['Date'], y=recovered['Recovered'],
mode='lines+markers', name = 'Recovered', line = dict(color = "Green", width=2)))
fig.add_trace(go.Scatter(x = deaths['Date'], y=deaths['Deaths'],
mode='lines+markers', name = 'Deaths', line = dict(color = "Red", width=2)))

fig.update_layout(title='WORLDWIDE COVID CASES', xaxis_tickfont_size = 14, yaxis =
dict(title = "Number Of Cases"))

fig.show()
```

In [19]:

```
#CASE DENSITY ANIMATION ON WORLD MAP
```

In [20]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'> RangeIndex: 191620 entries, 0 to 191619
Data columns (total 9 columns): # Column Non-Null Count Dtype --- ------ -----
--------- ----- 0 Date 191620 non-null datetime64[ns] 1 Province/State 191620
non-null object 2 Country 191620 non-null object 3 Lat 191620 non-null float64
4 Long 191620 non-null float64 5 Confirmed 191620 non-null int64 6 Recovered
191620 non-null int64 7 Deaths 191620 non-null int64 8 Active 191620 non-null
int64 dtypes: datetime64[ns](1), float64(2), int64(4), object(2) memory usage:
13.2+ MB
```

In [21]:

```
#as u see date is in datetime format so convert into string format
```

In [22]:

```
df['Date'] = df['Date'].astype(str)
```

In [23]:

```
#so date is converted into string format
```

In [24]:

```
# plotting the density map using -->  import plotly.express as px
```

In [25]:
```
fid = px.density_mapbox(df, lat = 'Lat', lon='Long', hover_name='Country',
                        hover_data=['Confirmed', 'Recovered', 'Deaths'],
animation_frame='Date',
                        color_continuous_scale='Portland', radius=10, zoom=0,
height=700)

fid.update_layout(title = "WorldWide Covid Cases Analysics")
fid.update_layout(mapbox_style = 'open-street-map', mapbox_center_lon=0)

fid.show()
```
In [26]:
```
#convert date to its original format
```
In [27]:
```
#CASES OVER THE TIME WITH AREA PLOT
```
In [28]:
```
df['Date'] = pd.to_datetime(df['Date'])
df.info()
```
```
<class 'pandas.core.frame.DataFrame'> RangeIndex: 191620 entries, 0 to 191619
Data columns (total 9 columns): # Column Non-Null Count Dtype --- ------ -----
--------- ----- 0 Date 191620 non-null datetime64[ns] 1 Province/State 191620
non-null object 2 Country 191620 non-null object 3 Lat 191620 non-null float64
4 Long 191620 non-null float64 5 Confirmed 191620 non-null int64 6 Recovered
191620 non-null int64 7 Deaths 191620 non-null int64 8 Active 191620 non-null
int64 dtypes: datetime64[ns](1), float64(2), int64(4), object(2) memory usage:
13.2+ MB
```
In [29]:
```
#TREE MAP using --> import plotly.express as px
```
In [30]:
```

t = df.groupby('Date')['Confirmed',  'Deaths', 'Recovered',
'Active'].sum().reset_index()
t = t[t['Date']==max(t['Date'])].reset_index(drop = True)
# getting latest data only

t=t.melt(id_vars = 'Date', value_vars = ['Active', 'Deaths', 'Recovered'])
# melt plots

fig=px.treemap(t, path=['variable'], values = 'value', height=250, width=800,
color_discrete_sequence=[act, rec, dth])
fig.data[0].textinfo = 'label+text+value'

fig.show()
```
In [31]:
```
t = df.groupby('Date')['Recovered', 'Deaths', 'Active'].sum().reset_index();
t = t.melt(id_vars = 'Date', value_vars = ['Recovered', 'Deaths', 'Active'],
var_name = 'Case', value_name = 'Count')
t
```
In [32]:
```
#confirmed cases with choropleth map
```
In [33]:
```
country_daywise.head():
```
In [34]:

```
f = px.choropleth(country_daywise, locations='Country', locationmode='country
names', color = np.log(country_daywise['Confirmed']),
                  hover_name = 'Country', animation_frame=country_daywise['Date'],
                  title='Cases over time',
color_continuous_scale=px.colors.sequential.Inferno)

f.update(layout_coloraxis_showscale=True)

f.show()
```

In [35]:

```
#DEATHS AND RECOVERIES PER 100 CASES useing --> import plotly.express as px
```

In [36]:

```
daywise.tail()
```

In [37]:

```
fc = px.bar(daywise, x='Date', y='Confirmed', color_discrete_sequence=[act])
fd = px.bar(daywise, x='Date', y='Deaths', color_discrete_sequence=[dth])

f = make_subplots(rows=1, cols=2, shared_xaxes=False,
horizontal_spacing=0.1,subplot_titles=('Confired cases', 'Deaths cases'))

# adding "fc" and "fd" into figure canvas "f"
f.add_trace(fc['data'][0], row=1, col=1)
f.add_trace(fd['data'][0], row=1, col=2)

f.update_layout(height=400)

f.show()
```

In [38]:

```
#conecting to the jovian
```

In [39]:

```
# conformed and death cases
```

In [40]:

```
daywise
```

In [41]:

```
# per 100 cases using --> import plotly.express as px
```

In [42]:

```
f1 = px.line(daywise, x='Date', y = 'Deaths / 100 Cases',
color_discrete_sequence=[dth])

f2 = px.line(daywise, x='Date', y = 'Recovered / 100 Cases',
color_discrete_sequence=[rec])

f3 = px.line(daywise, x='Date', y = 'Deaths / 100 Recovered',
color_discrete_sequence=[rec])

f = make_subplots(rows = 1, cols = 3, shared_xaxes=False,
horizontal_spacing=0.2,subplot_titles=('Deaths / 100 cases', 'Recovered / 100
cases', 'Deaths / 100 Recovered'))

f.add_trace(f1['data'][0], row=1, col=1)
f.add_trace(f2['data'][0], row=1, col=2)
f.add_trace(f3['data'][0], row=1, col=3)

f.update_layout(height = 400)
f.show()
```

In [43]:
```
#New cases and No. of countries
```
In [44]:
```
fc = px.line(daywise, x='Date', y = 'Confirmed', color_discrete_sequence=[act])
fd = px.line(daywise, x='Date', y = 'No. of Countries',
color_discrete_sequence=[dth])

f = make_subplots(rows = 1, cols = 2, shared_xaxes=False, horizontal_spacing=0.1,
                  subplot_titles=('No. of New Cases Per Day', 'No. of Countries'))

f.add_trace(fc['data'][0], row = 1, col = 1)
f.add_trace(fd['data'][0], row = 1, col = 2)

f.show()
```
In [45]:
```
#top 15 countries case analysis
```
In [46]:
```
top = 15
fc = px.bar(countrywise.sort_values('Confirmed').tail(top), x = 'Confirmed',
y='Country', text = 'Confirmed', orientation='h', color_discrete_sequence=[cnf])

fd = px.bar(countrywise.sort_values('Deaths').tail(top), x = 'Deaths',
y='Country', text = 'Deaths', orientation='h', color_discrete_sequence=[dth])

fa = px.bar(countrywise.sort_values('Active').tail(top), x = 'Active',
y='Country', text = 'Active', orientation='h', color_discrete_sequence=[act])

fnc = px.bar(countrywise.sort_values('New Cases').tail(top), x = 'New Cases',
y='Country', text = 'New Cases', orientation='h', color_discrete_sequence=[act])



f = make_subplots(rows = 5 , cols = 2, shared_xaxes=False, horizontal_spacing=
0.14, vertical_spacing=0.1, subplot_titles=('Confirmed Cases', 'Deaths Reported',
'Active Cases', 'New Cases'))

f.add_trace(fc['data'][0], row = 1, col = 1)
f.add_trace(fd['data'][0], row = 1, col = 2)

f.add_trace(fa['data'][0], row = 2, col = 1)

f.add_trace(fnc['data'][0], row = 2, col = 2)


f.update_layout(height = 4000)
f.show()
```
In [47]:
```
# bar plot
```
In [48]:
```
#top confiremd cases
```
In [49]:
```
#f = px.bar(country_daywise, x = 'Date', y ='Confirmed', color = 'Country',
height=600, title = 'Confired', color_discrete_sequence=px.colors.cyclical.mygbm)
#f.show()
```

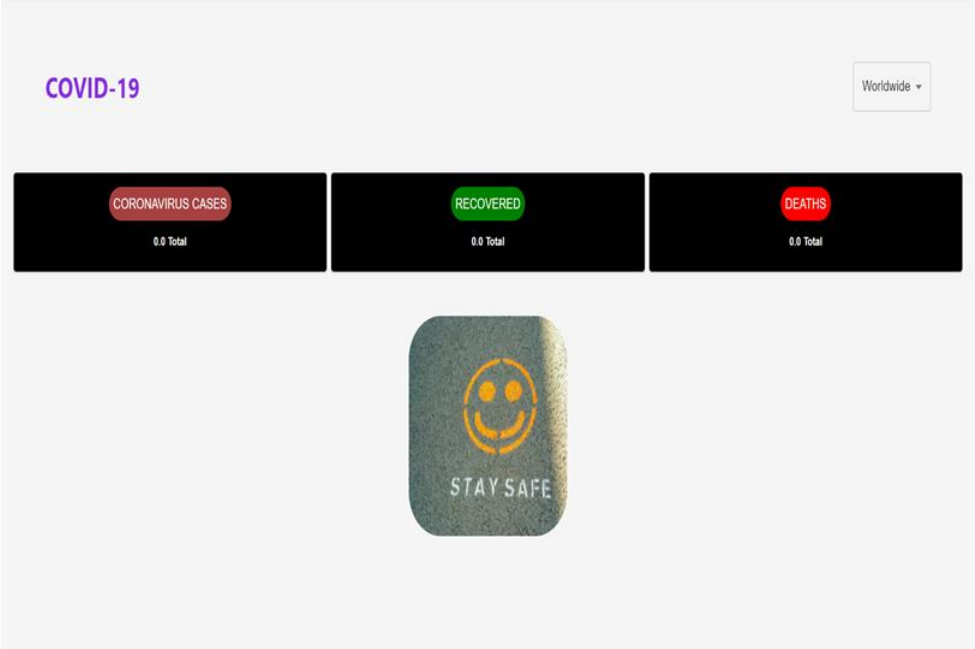
In [50]:
```
#top death cases
```
In [51]:
```
#f = px.bar(country_daywise, x = 'Date', y ='Deaths', color = 'Country',
height=600, title = 'Deaths', color_discrete_sequence=px.colors.cyclical.mygbm)
#f.show()
```
In [52]:
```
#top recovery cases
```
In [53]:
```
#f = px.bar(country_daywise, x = 'Date', y ='Recovered', color = 'Country',
height=600, title = 'Recovered', color_discrete_sequence=px.colors.cyclical.mygbm)
#f.show()
```
In [54]:
```
# covid - 19 vs other similar epidemics
```
In [55]:
```
country_daywise
```
In [56]:
```
epidemics = pd.DataFrame({
    'epidemic' : ['COVID19', 'SARS', 'EBOLA', 'MERS', 'H1N1'],
    'start_year' : [2019, 2002, 2013, 2012, 2009],
    'end_year' : [2021, 2004, 2016, 2020, 2010],
    'confirmed' : [country_daywise['Confirmed'].sum(), 8422, 28446, 2519,
6724149],
    'deaths' : [country_daywise['Deaths'].sum(), 813, 11323, 866, 19654]
})

epidemics['mortality'] = round((epidemics['deaths']/epidemics['confirmed'])*100,2)
epidemics.head()
```
In [57]:
```
temp = epidemics.melt(id_vars='epidemic', value_vars=['confirmed', 'deaths',
'mortality'], var_name='Case', value_name='Value')
temp
```
In [58]:
```
fig = px.bar(temp, x = 'epidemic', y='Value', color='epidemic', text='Value',
facet_col='Case', color_discrete_sequence=px.colors.qualitative.Bold)
fig.update_traces(textposition = 'outside')
fig.update_layout(uniformtext_minsize = 8, uniformtext_mode = 'hide')
fig.update_yaxes(showticklabels = False)
fig.layout.yaxis2.update(matches = None)
fig.layout.yaxis3.update(matches = None)
fig.show()
```
In [59]:
```
pip install jovian --upgrade
Requirement already up-to-date: jovian in
c:\users\thimothy\anaconda3\lib\site-packages (0.2.41)Note: you may need to
restart the kernel to use updated packages. Requirement already satisfied,
skipping upgrade: click in c:\users\thimothy\anaconda3\lib\site-packages (from
jovian) (7.1.2) Requirement already satisfied, skipping upgrade: requests in
c:\users\thimothy\anaconda3\lib\site-packages (from jovian) (2.24.0)
Requirement already satisfied, skipping upgrade: pyyaml in
c:\users\thimothy\anaconda3\lib\site-packages (from jovian) (5.3.1)
Requirement already satisfied, skipping upgrade: uuid in
c:\users\thimothy\anaconda3\lib\site-packages (from jovian) (1.30) Requirement
already satisfied, skipping upgrade: urllib3!=1.25.0,!=1.25.1,<1.26,>=1.21.1
```

```
in c:\users\thimothy\anaconda3\lib\site-packages (from requests->jovian)
(1.25.9) Requirement already satisfied, skipping upgrade: idna<3,>=2.5 in
c:\users\thimothy\anaconda3\lib\site-packages (from requests->jovian) (2.10)
Requirement already satisfied, skipping upgrade: certifi>=2017.4.17 in
c:\users\thimothy\anaconda3\lib\site-packages (from requests->jovian)
(2020.6.20) Requirement already satisfied, skipping upgrade: chardet<4,>=3.0.2
in c:\users\thimothy\anaconda3\lib\site-packages (from requests->jovian)
(3.0.4)
In [60]:
import jovian
In [61]:
jovian.commit()
[jovian] Updating notebook "thomas02032001/minifinal" on https://jovian.ai/
[jovian] Committed successfully! https://jovian.ai/thomas02032001/minifinal
'https://jovian.ai/thomas02032001/minifinal'
```

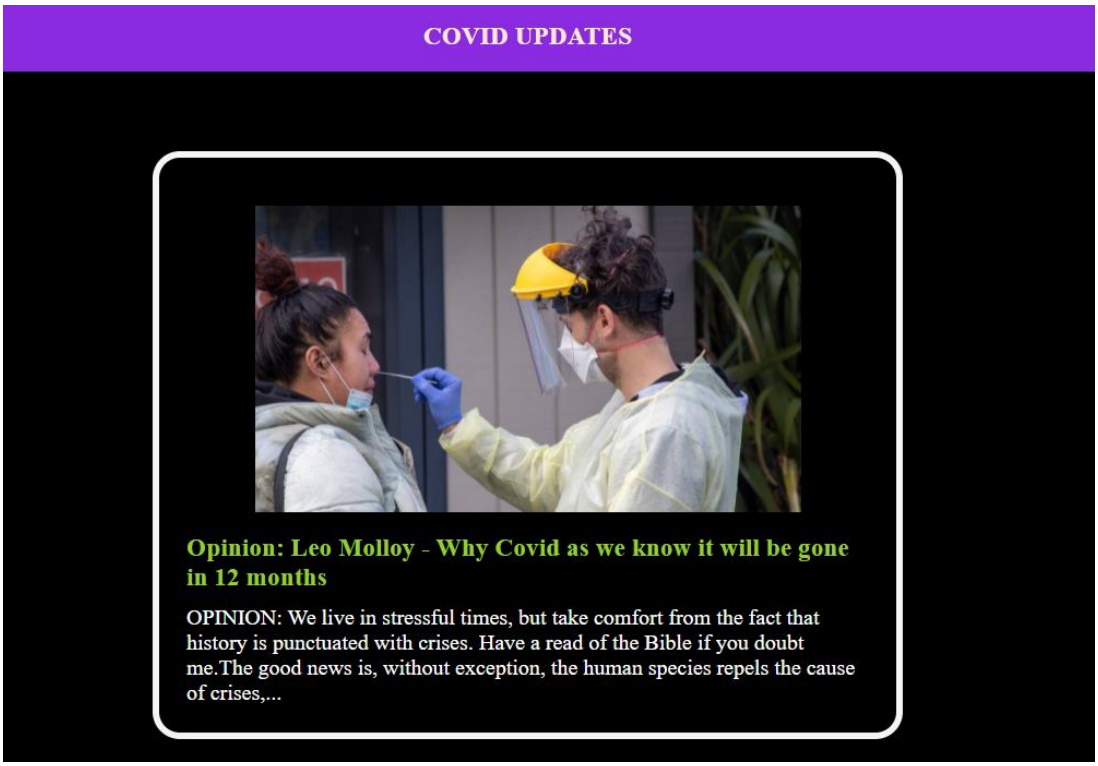
5.2 EXECUTION SCREENS:

HOME PAGE

COVID COUNT WORLDWIDE

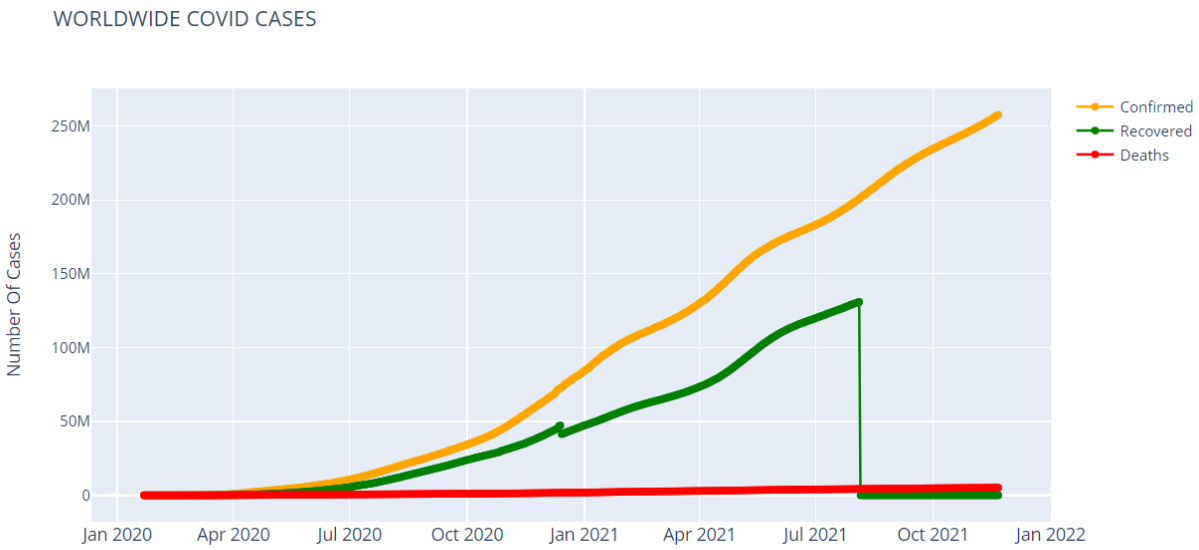


COVID NEWS UPDATES

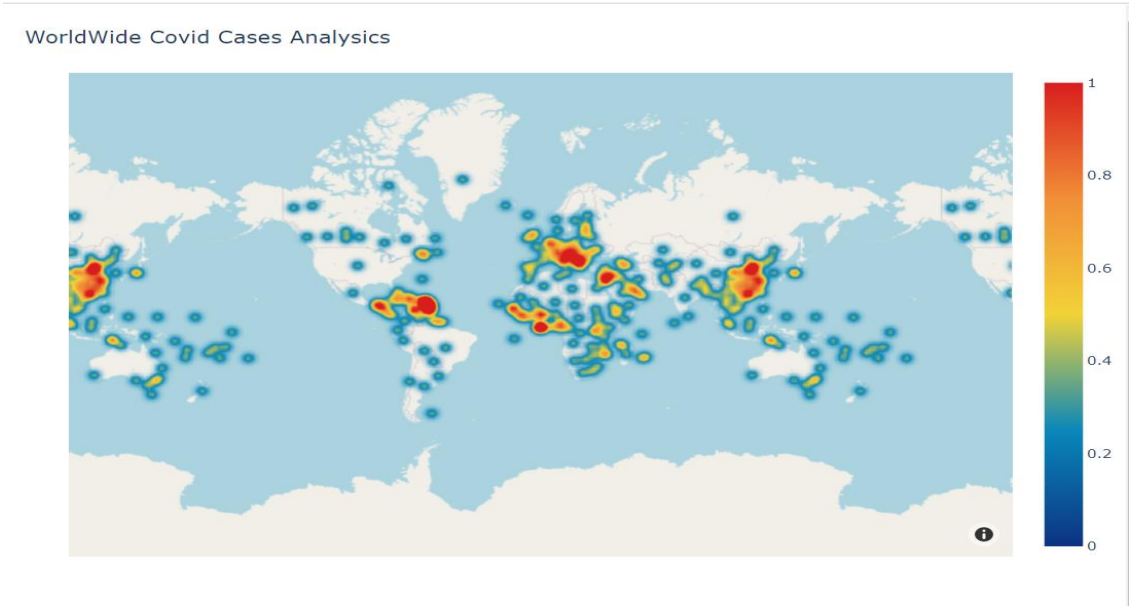


COVID ANALYSIS

Scatter Graph



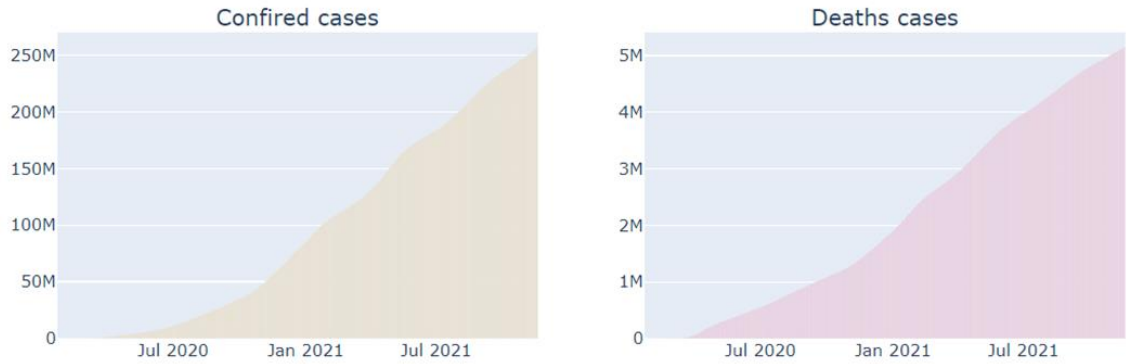
Density Map Box



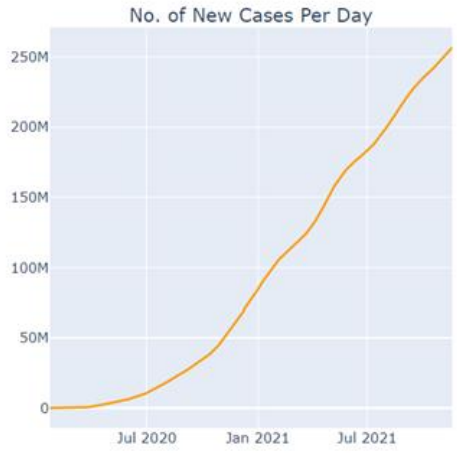
Tree Map

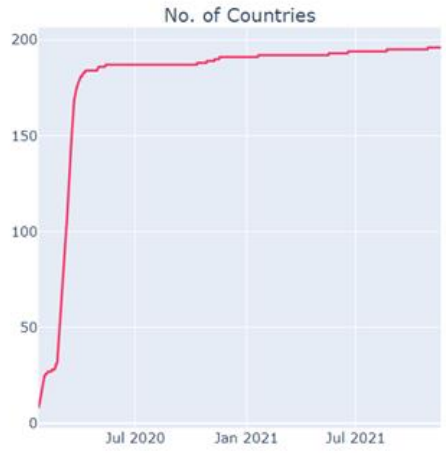
Choropleth map

Analysis of Confirmed and Death Cases.

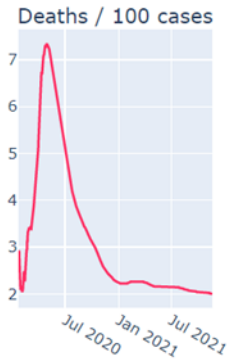


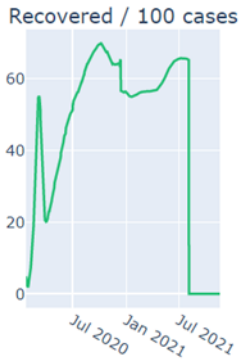
Analysis of Number of New Cases and Number of Countries Per Day.

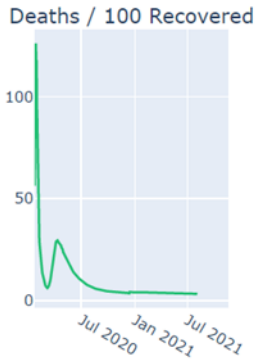




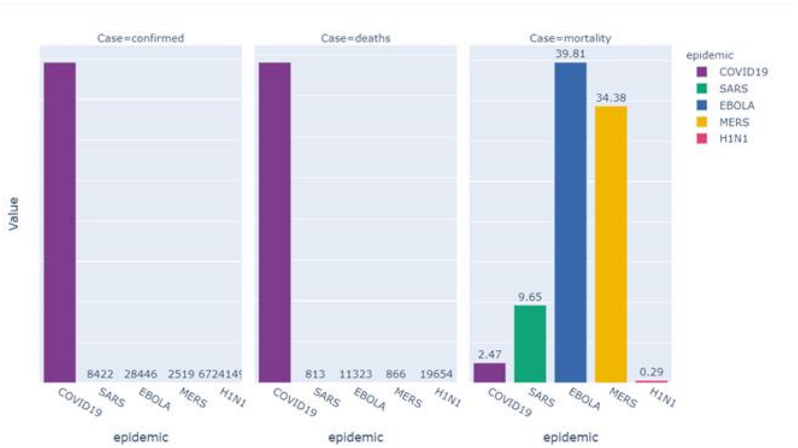
Comparing Deaths, Recovered, Deaths Per 100 Cases







Comparing With Other Diseases



SUMMARY AND CONCLUSION

- The idea we made helps us to predict the worldwide covid cases in various countries.
- We can predict number of deaths, recoveries and information about covid data.
- We are linked with a news page which provides day to day updates about covid cases.
- we can predict the future conditions.
- Finds covid cases, news, analysis.

REFERENCES

- https://www.analyticsvidhya.com/blog/2021/02/an-intuitive-guide-to-visualization-in-python/
- https://www.lighthouselabs.ca/en/blog/the-five-stages-of-data-analysis
- https://www.lighthouselabs.ca/en/blog/the-five-stages-of-data-analysis
- https://www.jovian.ai/