Novel Corona Virus (COVID19)

A deadly virus that has hit the world in 2019 and the crisis still continues.. Sadly India in 2021 is one of the worst hit countries by COVID19

The purpose of this study is to take a look at current data and understand how Corona has impacted India, how each state suffered, what could have be done better, how we can still save lives, etc.

Overview of our data in hand.

Data Source: https://www.kaggle.com/sudalairajkumar/covid19-in-india (https://www.kaggle.com/sudalairajkumar/covid19-in-india)

Table 1. covid_19_india.csv: Contains Total number of cases, Total recoveries etc.

This table contains total 9 columns:

```
1. Sno                      :   Serial Number
2. Date                     :   Date of the data collection
3. Time                     :   Time of the data collection
4. State/UnionTerritory     :   Which State or UT data the data belongs to.
5. ConfirmedIndianNational  :   If the data belongs to Indian
6. ConfirmedForeignNational :   If the data belongs to the foriegn national
7. Cured                    :   How many reported recovered for that date
8. Deaths                   :   How many deaths reported for that date
9. Confirmed                :   How many new confirmed cases reported for that d
ate
```

Table 2. covid_vaccine_statewise.csv: How many are vaccinated, how many remianing etc.

This table contains total 18 columns:

```
1. Updated On                              :     Data update date
2. State                                   :     State for which data is collect
ed
3. Total Individuals Vaccinated            :     So far how many individuals are
vaccinated
4. Total Sessions Conducted                :     So far how many vaccination ses
sions held
5. Total Sites                             :     How many areas targeted
6. First Dose Administered                 :     How many first doses are given
7. Second Dose Administered                :     How many second doses are given
8. Male(Individuals Vaccinated)            :     How many males vaccinated
9. Female(Individuals Vaccinated)          :     How many females vaccinated
10. Transgender(Individuals Vaccinated)    :     How many transgender vaccinated
11. Total Covaxin Administered             :     Total COVAXIN vaccines given
12. Total CoviShield Administered          :     Total CoviShield vaccines given
13. AEFI                                   :     Adverse event following immuniz
ation (After effects)
14. 18-30 years (Age)                      :     How many vaccinated in 18-30 ag
e bracket
15. 30-45 years (Age)                      :     How many vaccinated in 30-45 ag
e bracket
16. 45-60 years (Age)                      :     How many vaccinated in 45-60 ag
e bracket
17. 60+ years (Age)                        :     How many vaccinated above 60 ag
e bracket
18. Total Doses Administered               :     Total Vaccines given till date
```

In [353]:
```
# Importing important libraries
import pandas as pd
import numpy as np
%matplotlib inline
from matplotlib import pyplot as plt
import matplotlib
from matplotlib import cm
import geopandas as gpd
import plotly.express as px
import seaborn as sns
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, ipl
ot

init_notebook_mode(connected = True)  ## THIS LINE IS VERY IMPORTANT AS THI
S WILL PLOT THE CHART WHILE KERNAL IS RUNNINg

import plotly.graph_objects as go

from IPython.display  import HTML,display
import warnings
warnings.filterwarnings('ignore')


# Reading data in pandas dataframes
testing_data= pd.read_csv('StatewiseTestingDetails.csv')
stats_data = pd.read_csv('covid_19_india.csv')
vaccine_data = pd.read_csv('covid_vaccine_statewise.csv')
```

Overall Covid Situation

In [354]:
```
stats_data.head()
```

Out[354]:

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

In [355]:
```
## Indian_States.shp is a file containing Indian State boundary geometry.
# This file will only open when all four extensions are downloaded i.e. Ind
ian_States.shp  Indian_States.dbf, Indian_States.prj,Indian_States.shx
gdf = gpd.read_file('./Indian_States.shp') ## Geopandas helps reading geome
try files.gpd here means geopandas
```

In [356]:
```
# Now the state names in Indian_States table and our stats_data table shoul
d match, hence creating a list of state names

state_names = {'Andaman & Nicobar Island':'An&Ni', 'Arunanchal Pradesh':'Ar
unachal','Dadara & Nagar Havelli':'Dadara & Nagar','Daman & Diu':'D&D','Him
achal Pradesh':'Himachal','Jammu & Kashmir':'J&K', 'Madhya Pradesh':'MP','N
CT of Delhi':'Delhi','Uttar Pradesh':'UP','Andhra Pradesh':'AP'}
```

In [357]:
```
gdf['st_nm'].replace(state_names, inplace=True)
```

In [358]:
```
# Same dictionary with slight modifications
stats_state_names = {'Andaman & Nicobar Islands':'An&Ni', 'Arunanchal Prade
sh':'Arunachal','Dadra and Nagar Haveli and Daman and Diu':'Dadara & Nagar'
,'Daman & Diu':'D&D','Himachal Pradesh':'Himachal','Jammu and Kashmir':'J&
K', 'Madhya Pradesh':'MP','NCT of Delhi':'Delhi','Uttar Pradesh':'UP','Andh
ra Pradesh':'AP', 'Telengana':'Telangana'}
```

In [359]:
```
stats_data['State/UnionTerritory'].replace(stats_state_names,inplace = True
)
```

Observation: We can notice that states data in stats_data table have "Unassigned" and "Cases being reassigned to states" values, these might need our attention going forward.

Since the Corona data we have is Cumulative data so to analyze current situation fetching records with just most recent date in dataset should be more than enough and we can drop rest of the data (Only for visualization purposes).

Most Recent data:

In [360]:
```
latest_date = stats_data.sort_values(by = 'Date', ascending = False)
latest_date.head()
```

Out[360]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational
14653	14654	2021-05-07	8:00 AM	West Bengal	-	-
14635	14636	2021-05-07	8:00 AM	Ladakh	-	-
14633	14634	2021-05-07	8:00 AM	Karnataka	-	-
14632	14633	2021-05-07	8:00 AM	Jharkhand	-	-
14631	14632	2021-05-07	8:00 AM	J&K	-	-

In [361]:
```
latest_stats = stats_data[stats_data.Date == '2021-05-07']
```

In [362]:
```
# Sno, Date,Time and nationality are of no use to us as of now, and also St
ate/UnionTerritory is to big too type, so lets have this sorted

latest_stats.drop(['Sno','Date','Time','ConfirmedIndianNational','Confirmed
ForeignNational'],axis = 1, inplace = True)
latest_stats.rename(columns={'State/UnionTerritory':'state'},inplace=True)
```

In [363]:
```
## Lets check if the latest the data have the "Unassigned" or "Cases being
 reassigned to states" values
latest_stats['state'].unique()

### Not there !! we are good to go
```

Out[363]:
```
array(['Andaman and Nicobar Islands', 'AP', 'Arunachal Pradesh', 'Assam',
       'Bihar', 'Chandigarh', 'Chhattisgarh', 'Dadara & Nagar', 'Delhi',
       'Goa', 'Gujarat', 'Haryana', 'Himachal', 'J&K', 'Jharkhand',
       'Karnataka', 'Kerala', 'Ladakh', 'Lakshadweep', 'MP',
       'Maharashtra', 'Manipur', 'Meghalaya', 'Mizoram', 'Nagaland',
       'Odisha', 'Puducherry', 'Punjab', 'Rajasthan', 'Sikkim',
       'Tamil Nadu', 'Telangana', 'Tripura', 'Uttarakhand', 'UP',
       'West Bengal'], dtype=object)
```

Now, I will merge this table with Indian_State table so that with the help of geometry in Indian_State table, we can see the map view of cases.

In [364]:
```
## latest_stats_final = latest_stats.merge(gdf, left_on='state',right_on='st_nm')
## converting geo dataframe to pd dataframes throws error while plotting. hence convert pd dataframe to geo ## df as below
latest_stats_final = gdf.merge(latest_stats, left_on='st_nm',right_on='state')
latest_stats_final.head()
```

Out[364]:

	st_nm	geometry	state	Cured	Deaths	Confirmed
0	Assam	MULTIPOLYGON (((89.74323 26.30362, 89.74290 26...	Assam	242980	1531	277687
1	Bihar	MULTIPOLYGON (((84.50720 24.26323, 84.50355 24...	Bihar	435574	3077	553803
2	Chandigarh	POLYGON ((76.84147 30.75996, 76.83599 30.73623...	Chandigarh	38591	541	47552
3	Chhattisgarh	POLYGON ((83.33532 24.09885, 83.35346 24.09627...	Chhattisgarh	675294	9950	816489
4	Dadara & Nagar	POLYGON ((73.20657 20.12216, 73.20797 20.10650...	Dadara & Nagar	6917	4	8502

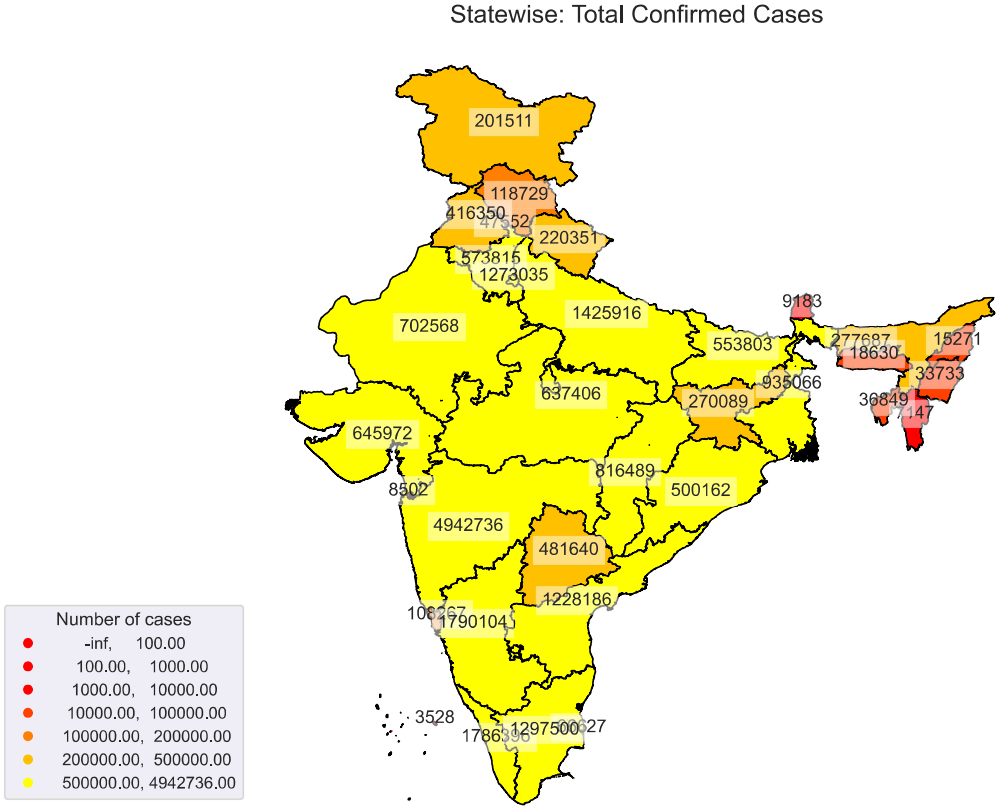
In [365]:
```
## representative_point().coords[:] is a function that helps getting the x-y cordinate from geometry data, ## this is inbuilt function we are just consuming this.
latest_stats_final['coord'] = latest_stats_final['geometry'].apply(lambda x: x.representative_point().coords[:])
latest_stats_final['coord'] = [coords[0] for coords in latest_stats_final['coord']]
latest_stats_final.head()
```

Out[365]:

	st_nm	geometry	state	Cured	Deaths	Confirmed	coord
0	Assam	MULTIPOLYGON (((89.74323 26.30362, 89.74290 26...	Assam	242980	1531	277687	(90.76265036100918, 26.052225148671738)
1	Bihar	MULTIPOLYGON (((84.50720 24.26323, 84.50355 24...	Bihar	435574	3077	553803	(86.12533485112044, 25.902658098342613)
2	Chandigarh	POLYGON ((76.84147 30.75996, 76.83599 30.73623...	Chandigarh	38591	541	47552	(76.78051175496091, 30.730184003289168)
3	Chhattisgarh	POLYGON ((83.33532 24.09885, 83.35346 24.09627...	Chhattisgarh	675294	9950	816489	(81.50899708552288, 20.94914121899211)
4	Dadara & Nagar	POLYGON ((73.20657 20.12216, 73.20797 20.10650...	Dadara & Nagar	6917	4	8502	(73.00811326266309, 20.206507820529538)

I will be plotting counts on the basis of confirmed cases, deaths reported, recovery reported and the co-ordinates will give us the state. (so no use of state column explicitly)

In [366]:
```
sns.set_context('talk')
sns.set_style('dark')
cmap = 'autumn'  ## Map color
figsize = (20,15)

ax = latest_stats_final.plot(column = 'Confirmed', cmap = cmap,figsize = fi
gsize, scheme ='User_Defined', classification_kwds = dict(bins = [100,1000,
10000,100000,200000,500000]),edgecolor = 'black',legend = True)


for idx, row in latest_stats_final.iterrows():
    ax.text(row.coord[0] , row.coord[1] , s = row['Confirmed'], horizontala
lignment = 'center', bbox ={'facecolor': 'white', 'alpha':0.5, 'pad': 5, 'e
dgecolor':'white'})

ax.get_legend().set_bbox_to_anchor((0.15,0.3))
ax.get_legend().set_title('Number of cases')
ax.set_title("Statewise: Total Confirmed Cases ", size = 25)
leg = ax.get_legend()
for lbl in leg.get_texts():
    label_text = lbl.get_text()
    lower = label_text.split()[0]
    upper = label_text.split()[1]
    #new_text = f'{float(lower):,.0f} - {float(upper):,.0f}'
    #lbl.set_text(new_text)

ax.set_axis_off()
plt.axis('equal')
plt.show()
```



Observation: Maximum number of cases so far are reported in Maharashtra (4942736), followed by Kerala with 1790104 and Uttar Pradesh with 1425916 cases as on the latest date in dataset.

Next lets plot and see Deaths due to corona Vs Recovery in each state.

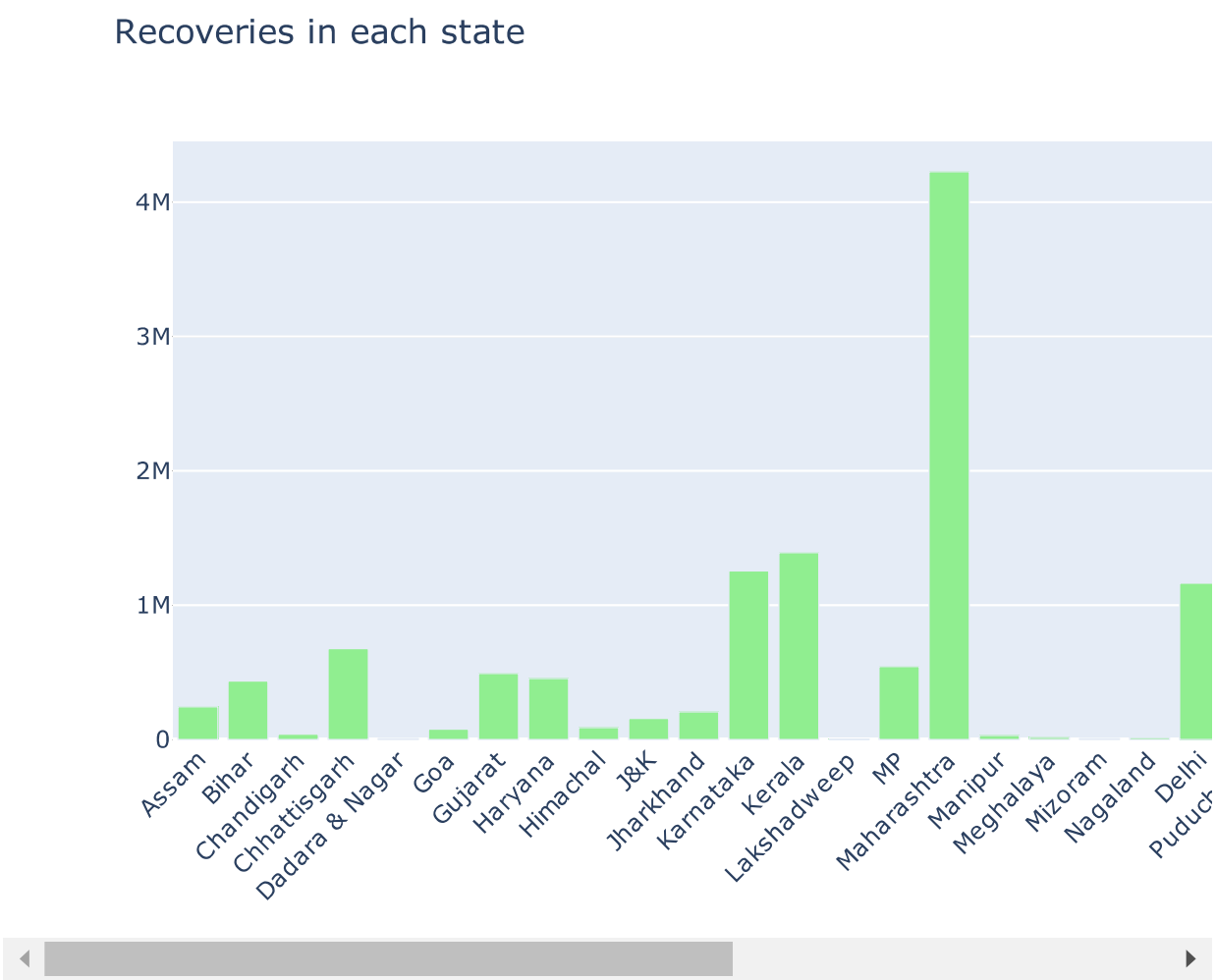
Broad Overview

In [367]:
```
latest_stats_final.head()
```

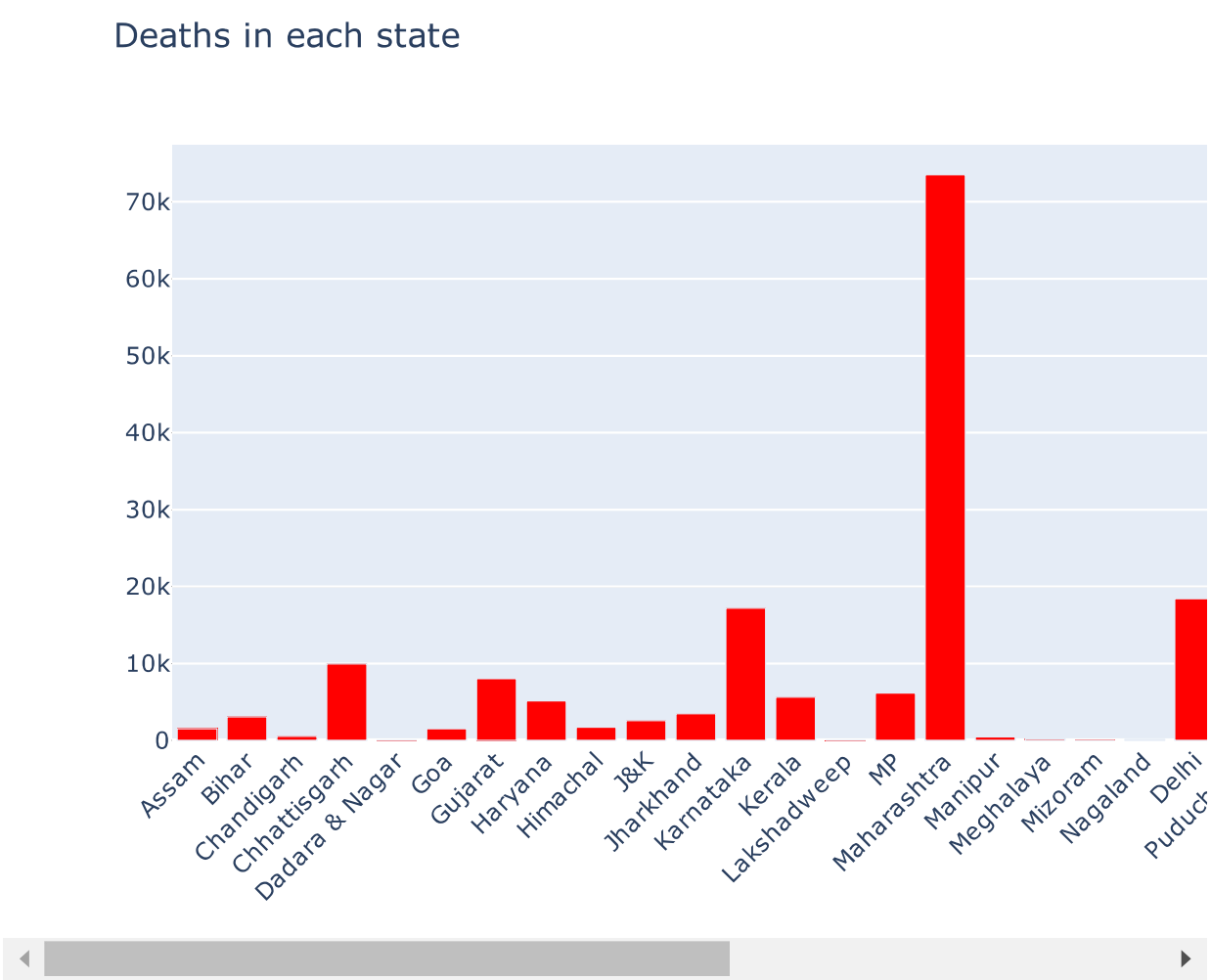
Out[367]:

	st_nm	geometry	state	Cured	Deaths	Confirmed	coord
0	Assam	MULTIPOLYGON (((89.74323 26.30362, 89.74290 26...	Assam	242980	1531	277687	(90.76265036100918, 26.052225148671738)
1	Bihar	MULTIPOLYGON (((84.50720 24.26323, 84.50355 24...	Bihar	435574	3077	553803	(86.12533485112044, 25.902658098342613)
2	Chandigarh	POLYGON ((76.84147 30.75996, 76.83599 30.73623...	Chandigarh	38591	541	47552	(76.78051175496091, 30.730184003289168)
3	Chhattisgarh	POLYGON ((83.33532 24.09885, 83.35346 24.09627...	Chhattisgarh	675294	9950	816489	(81.50899708552288, 20.94914121899211)
4	Dadara & Nagar	POLYGON ((73.20657 20.12216, 73.20797 20.10650...	Dadara & Nagar	6917	4	8502	(73.00811326266309, 20.206507820529538)

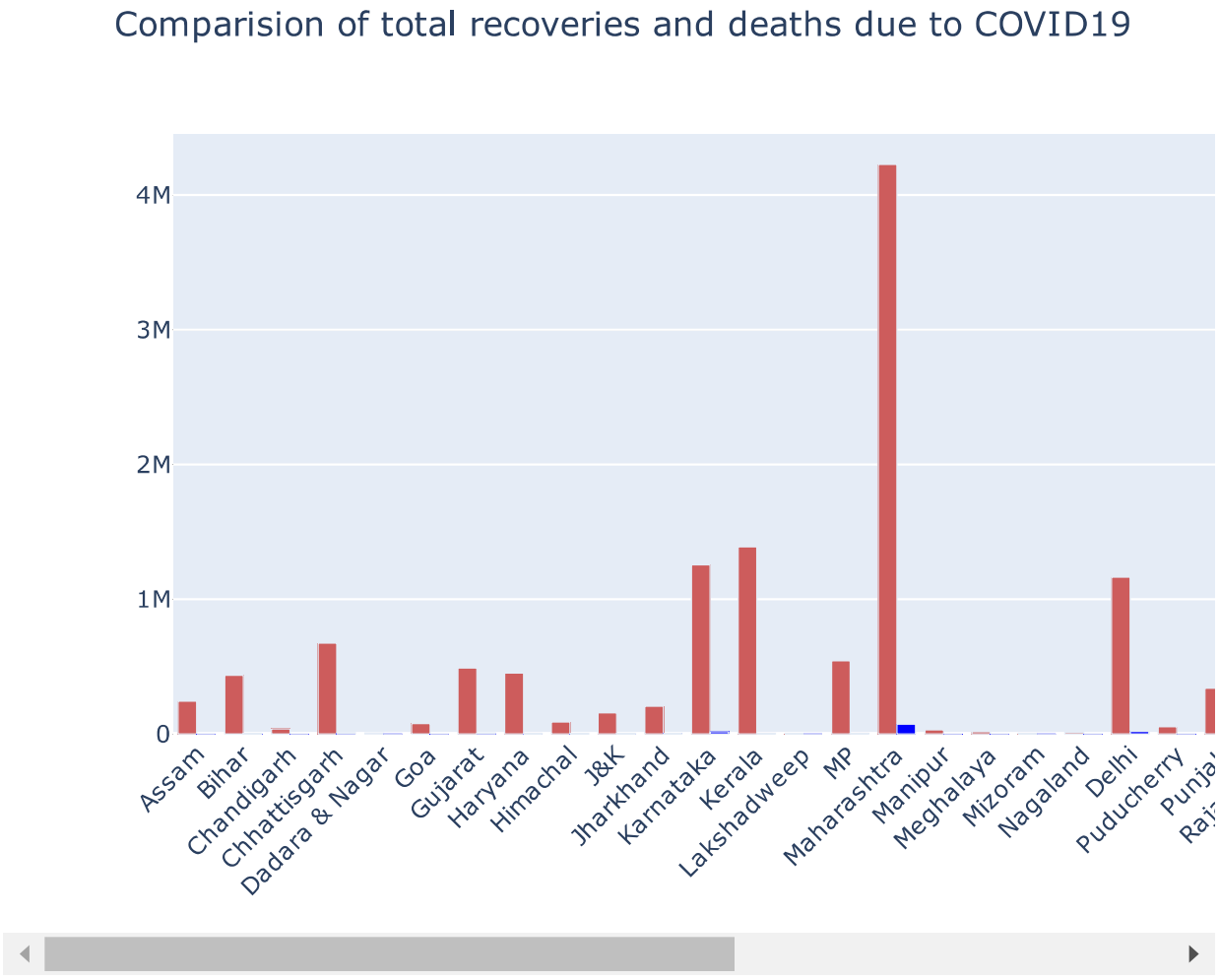
In [368]:
```
## Total recoveries statewise
fig = go.Figure()
fig.add_trace(go.Bar(
    x=latest_stats_final['state'],
    y=latest_stats_final['Cured'],
    name='Recovered',
    marker_color='lightgreen'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Recoveries in each state',
    barmode='group', xaxis_tickangle=-45)
fig.show()
```



In [369]:
```
## Total deaths statewise
fig = go.Figure()
fig.add_trace(go.Bar(
    x=latest_stats_final['state'],
    y=latest_stats_final['Deaths'],
    name='Deaths',
    marker_color='red'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Deaths in each state',
    barmode='group', xaxis_tickangle=-45)
fig.show()
```



In [370]:
```
fig = go.Figure()
fig.add_trace(go.Bar(
    x=latest_stats_final['state'],
    y=latest_stats_final['Cured'],
    name='Recovered',
    marker_color='indianred'
))
fig.add_trace(go.Bar(
    x=latest_stats_final['state'],
    y=latest_stats_final['Deaths'],
    name='Deaths',
    marker_color='blue'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Comparision of total recoveries and deaths due to COVID19',
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



Observation: Although the life loss can never be taken granted but the sliver lining of the cloud is the high Recovery rate.

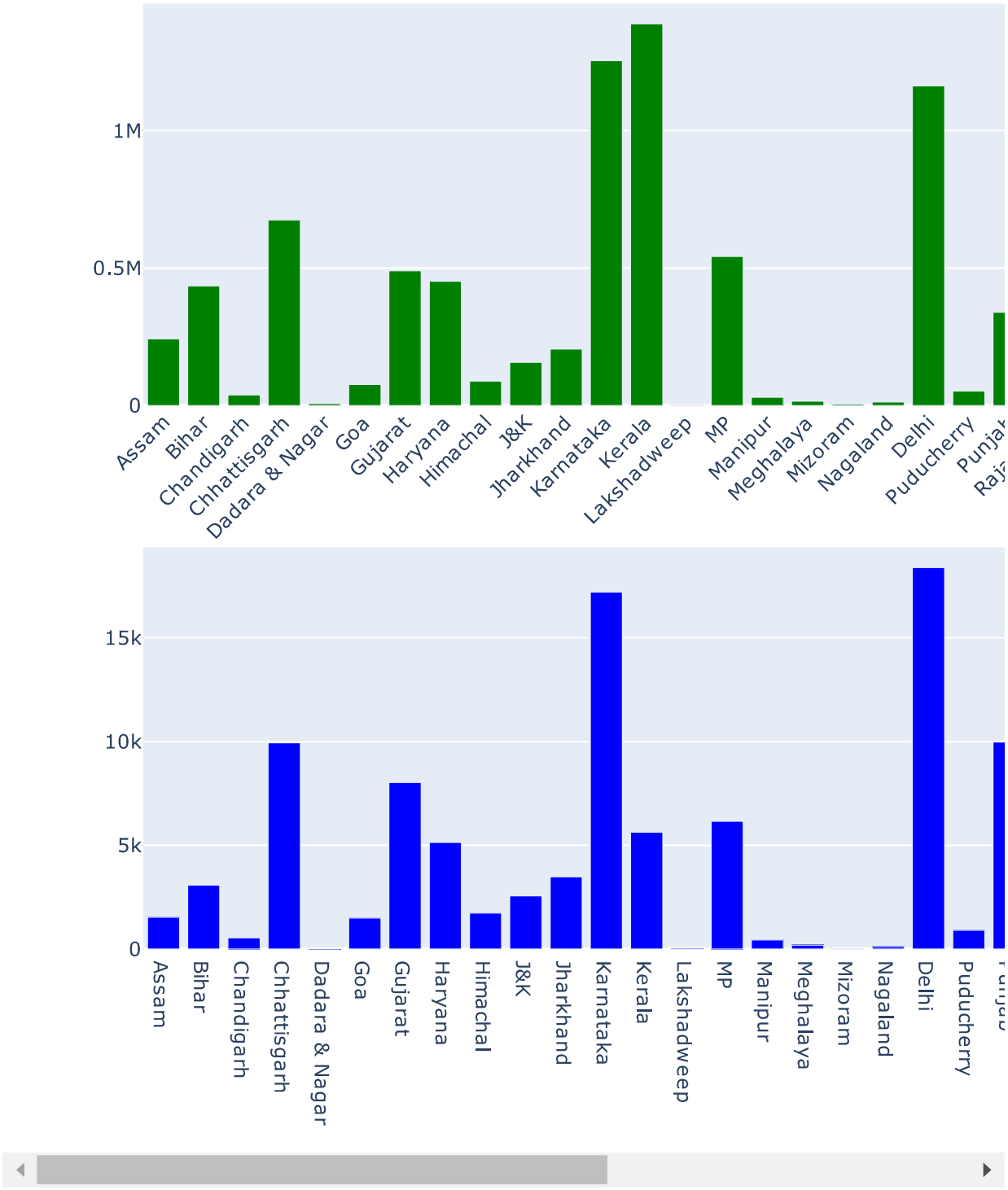
Because the data of Maharashtra is very high, its hard to read the visuals. Since we now know that Maharshatra is most impacted in Recovery and Deaths. Lets put this learning aside and take a closer look at data without Maharashtra.

In [371]:
```
all_but_maharashtra = latest_stats_final.copy()
index_names = all_but_maharashtra[all_but_maharashtra['state'] =='Maharasht
ra'].index
all_but_maharashtra.drop(index_names,inplace=True)
print(latest_stats_final.shape)
print(all_but_maharashtra.shape)
```

```
(33, 7)
(32, 7)
```

In [372]:
```
from plotly.subplots import make_subplots
fig = make_subplots(rows=2, cols=1)

fig.add_trace(go.Bar(
    x=all_but_maharashtra['state'],
    y=all_but_maharashtra['Cured'],
    name='Recovery',
    marker_color='green'

), row =1,col =1)

fig.add_trace(go.Bar(
    x=all_but_maharashtra['state'],
    y=all_but_maharashtra['Deaths'],
    name='Deaths',
    marker_color='blue'

), row =2,col =1)
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=800,
    title_text='All but Maharashtra - Comparision of total recoveries and d
eaths due to COVID19 ',
    barmode='group', xaxis_tickangle=-45)

fig.show()
```

All but Maharashtra - Comparision of total recoveries and deaths



Observation: Although the number of cases are second highest in Kerala, but the Fatalities in Delhi and Karnataka are way higher than Kerala.

We can plot a graph of Fatality rates of statewise.

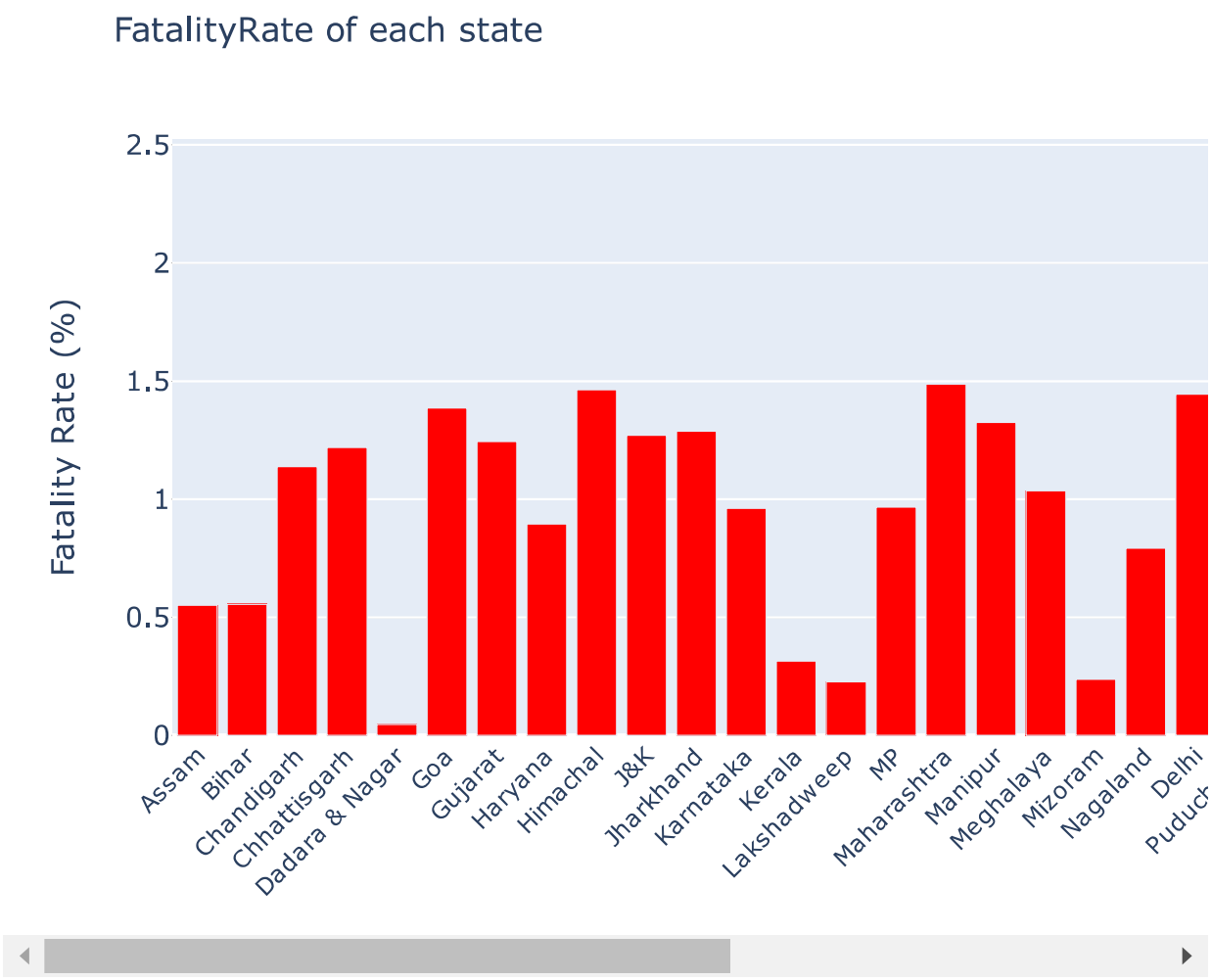
How Fatality rate is Calculated : == > FatalityRate = (Number of Deaths/Number of Cases)*100

In [373]:
```
latest_stats_final['FatalityRate'] = (latest_stats_final['Deaths']/latest_stats_final['Confirmed'])*100
latest_stats_final.head()
```

Out[373]:

	st_nm	geometry	state	Cured	Deaths	Confirmed	coord
0	Assam	MULTIPOLYGON (((89.74323 26.30362, 89.74290 26...	Assam	242980	1531	277687	(90.76265036100918, 26.052225148671738)
1	Bihar	MULTIPOLYGON (((84.50720 24.26323, 84.50355 24...	Bihar	435574	3077	553803	(86.12533485112044, 25.902658098342613)
2	Chandigarh	POLYGON ((76.84147 30.75996, 76.83599 30.73623...	Chandigarh	38591	541	47552	(76.78051175496091, 30.730184003289168)
3	Chhattisgarh	POLYGON ((83.33532 24.09885, 83.35346 24.09627...	Chhattisgarh	675294	9950	816489	(81.50899708552288, 20.94914121899211)
4	Dadara & Nagar	POLYGON ((73.20657 20.12216, 73.20797 20.10650...	Dadara & Nagar	6917	4	8502	(73.00811326266309, 20.206507820529538)

In [374]:
```
## Fatality Rate statewise
fig = go.Figure()
fig.add_trace(go.Bar(
    x=latest_stats_final['state'],
    y=latest_stats_final['FatalityRate'],
    name='FatalityRate',
    marker_color='red'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='FatalityRate of each state',
    yaxis=dict(
        title='Fatality Rate (%)',
        titlefont_size=16,
        tickfont_size=14,
    ),
    barmode='group', xaxis_tickangle=-45)
fig.show()
```



Observation : Although more cases are reported in Maharashtra, Kerala, Karanataka & Delhi, Punjab and J&K are having highest Fatility Rates.

Vaccination Data

In [375]:
```
vaccine_data.head()
```

Out[375]:

	Updated On	State	Total Individuals Vaccinated	Total Sessions Conducted	Total Sites	First Dose Administered	Second Dose Administered	Male(Individuals Vaccinated
0	16/01/2021	India	48276.0	3455.0	2957.0	48276.0	0.0	23757.(
1	17/01/2021	India	58604.0	8532.0	4954.0	58604.0	0.0	27348.(
2	18/01/2021	India	99449.0	13611.0	6583.0	99449.0	0.0	41361.(
3	19/01/2021	India	195525.0	17855.0	7951.0	195525.0	0.0	81901.(
4	20/01/2021	India	251280.0	25472.0	10504.0	251280.0	0.0	98111.(

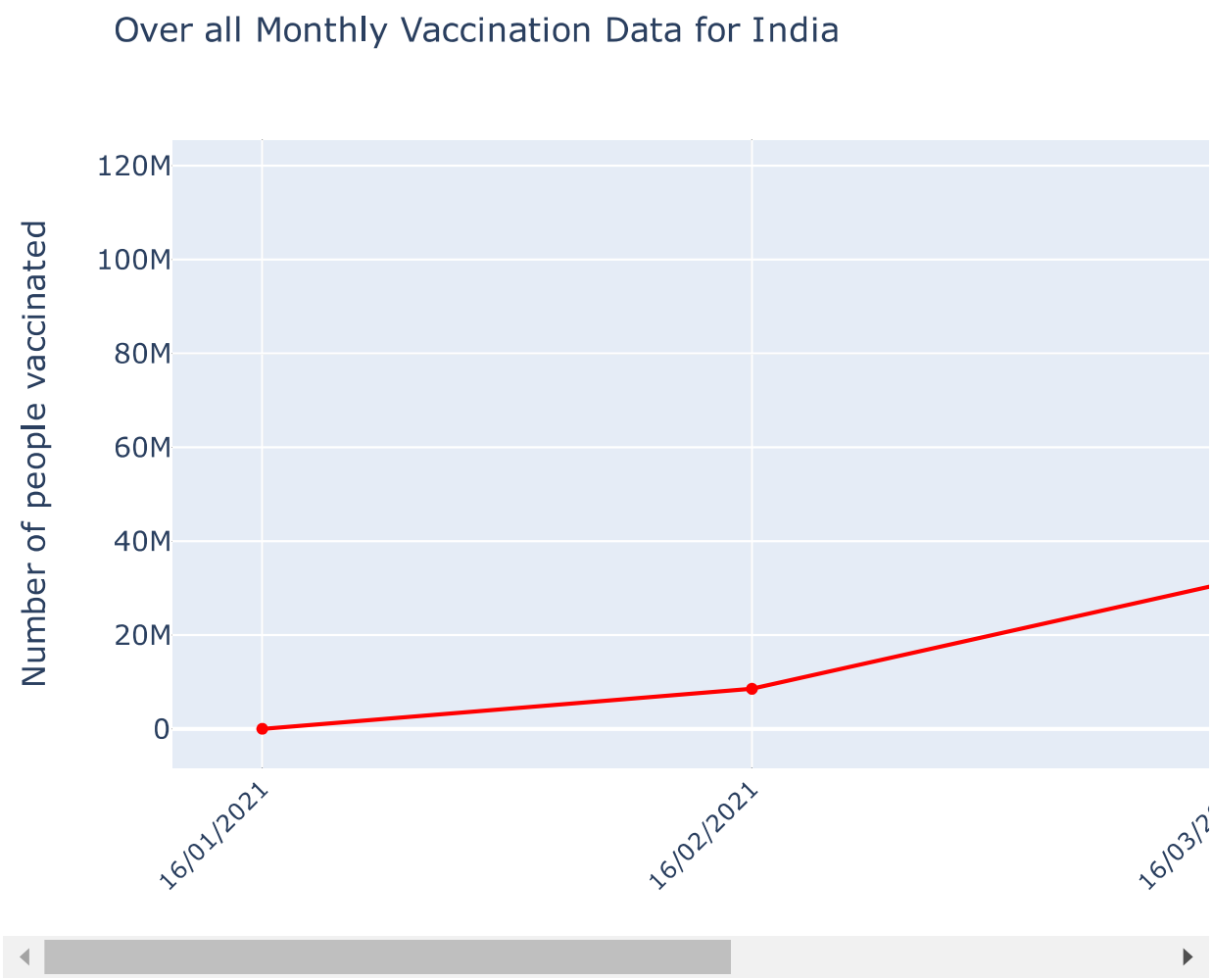
Vaccination Overview:

The data we have is cumulative. And as per the recorded data the Vaccination started from 16th Jan 2021. So for monthly trend we can pick the data of 16th of every month and see how we are doing on Vaccination.

In [376]:
```
monthly_vacc = vaccine_data[vaccine_data['Updated On'].isin(['16/01/2021',
'16/02/2021','16/03/2021','16/04/2021'])]
```

In [377]:
```
monthly_vacc_india  = monthly_vacc[monthly_vacc.State == 'India']
```

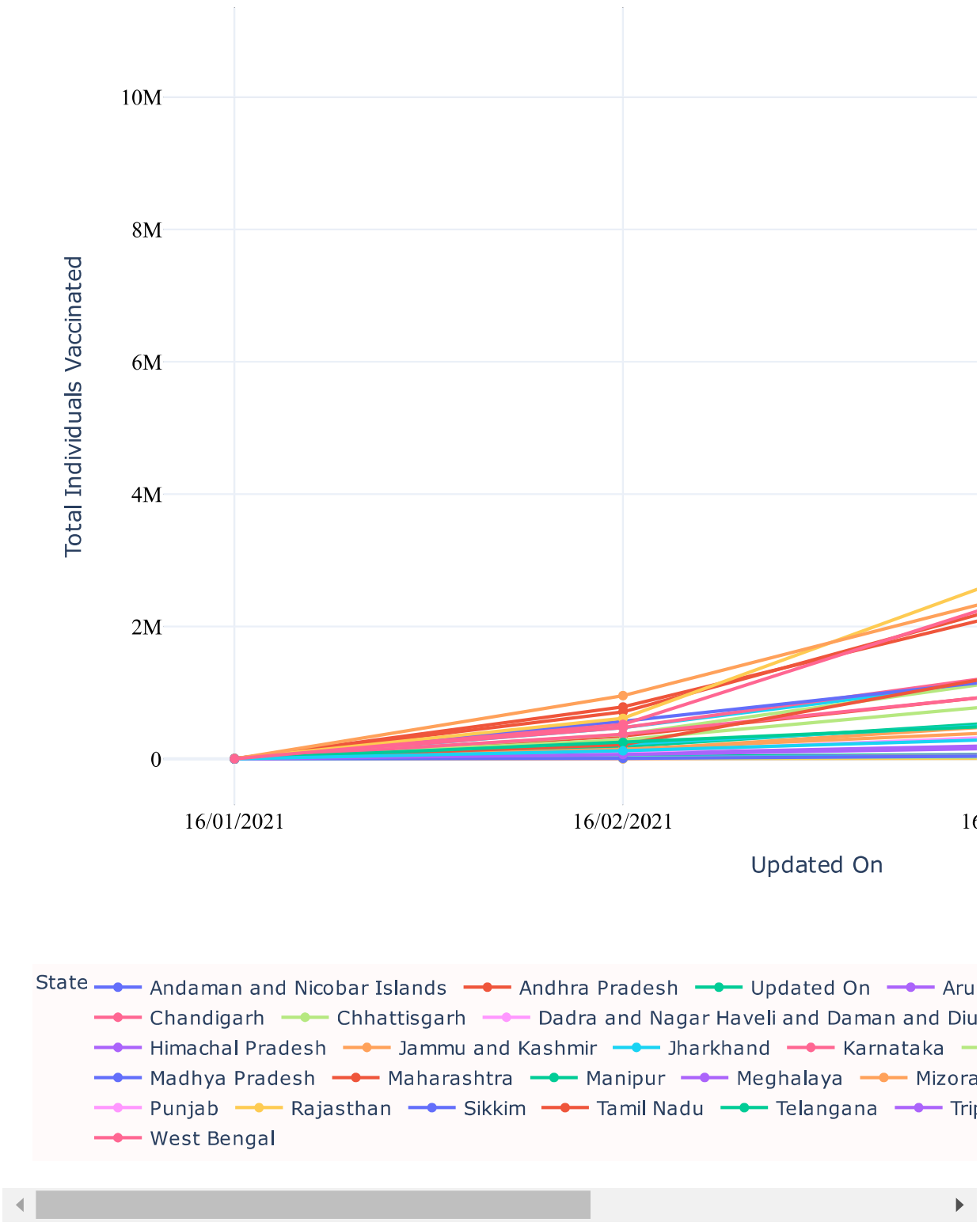
In [405]:
```
fig = go.Figure()
fig.add_trace(go.Line(
    x=monthly_vacc_india['Updated On'],
    y=monthly_vacc_india['Total Doses Administered'],
    name='Number of People vaccinated',
    marker_color='red'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Over all Monthly Vaccination Data for India',
    yaxis=dict(
        title='Number of people vaccinated',
        titlefont_size=16,
        tickfont_size=14,
    ),
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



In [379]:
```
## Most Vaccinated state.
index_names = monthly_vacc[monthly_vacc.State == 'India'].index
monthly_vacc.drop(index_names,inplace=True)
```

In [380]:
```
df = monthly_vacc[['Updated On', 'State','Total Individuals Vaccinated']].c
opy()
```

In [381]:
```
# monthly statewise progress of Vaccines

fig = px.line(df, x="Updated On", y="Total Individuals Vaccinated", color=
'State',template= "plotly_white")
fig.update_xaxes(tickfont=dict(family='Rockwell', color='black', size=14))
fig.update_yaxes(tickfont=dict(family='Rockwell', color='black', size=14))
fig.update_traces(mode='lines + markers')
fig.update_layout(legend_orientation="h",legend=dict(x= -.1, y=-.2),
                  autosize=False,
                  width= 1000,
                  height= 800,
                  title_text='Monthly Vaccination Data for each state',
                  title_x=0.5,
                 paper_bgcolor='snow',
                 plot_bgcolor = "snow")
fig.show()
```



Observation: Vaccination is significantly increased across all parts of india since April 2021, that is after the second wave of Covid. Maybe more Vaccination in early months could have prevented this stage of pandemic we are currently in.

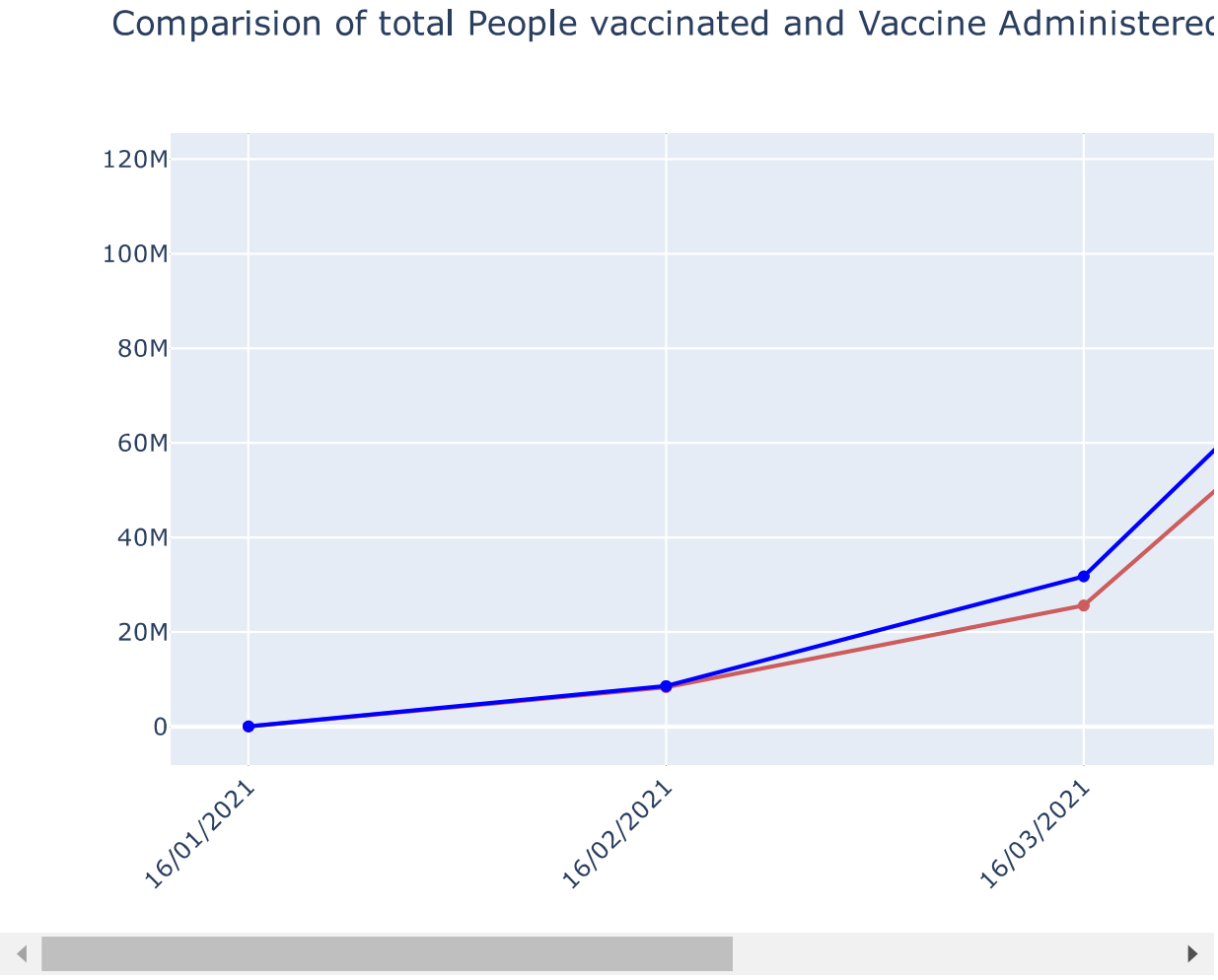
In [382]:
```
# Lets take a look since begining how many vaccines were available and how
 many people used it to get vaccinated
df = monthly_vacc[['Updated On', 'Total Doses Administered','Total Individu
als Vaccinated']].copy()
```

In [383]:
```
df1 = pd.DataFrame()
df2 = pd.DataFrame()
df1['People _vaccinated'] = df.groupby(['Updated On'])['Total Individuals V
accinated'].sum()
df1['Vaccine_administered'] = df.groupby(['Updated On'])['Total Doses Admin
istered'].sum()
df2 = df1.reset_index()
df2.head()
```

Out[383]:

	Updated On	People _vaccinated	Vaccine_administered
0	16/01/2021	48276.0	48276.0
1	16/02/2021	8337345.0	8576951.0
2	16/03/2021	25615590.0	31776993.0
3	16/04/2021	102775417.0	117153438.0

In [406]:
```
fig = go.Figure()
fig.add_trace(go.Line(
    x=df2['Updated On'],
    y=df2['People _vaccinated'],
    name='People _vaccinated',
    marker_color='indianred'
))
fig.add_trace(go.Line(
    x=df2['Updated On'],
    y=df2['Vaccine_administered'],
    name='Vaccine_administered',
    marker_color='blue'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Comparision of total People vaccinated and Vaccine Administ
ered',
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



Observation: Larger sub-population has got first dose of the vaccine and only fewer people got fully vaccinated.

Deeper look into last weeks new cases in each state to understand if state is seeing rise/decline in daily cases.

In [385]:
```
stats_data.rename(columns={'State/UnionTerritory':'state'}, inplace =True)
```

In [386]:
```
delhi = stats_data[stats_data['state'] == 'Delhi'].copy()
delhi['new_cases'] = delhi['Confirmed'].diff()
delhi = delhi.iloc[-10:]
```

In [387]:
```
maharashtra = stats_data[stats_data['state'] == 'Maharashtra'].copy()
maharashtra['new_cases'] = maharashtra['Confirmed'].diff()
maharashtra = maharashtra.iloc[-10:]
```

In [388]:
```
kerala = stats_data[stats_data['state'] == 'Kerala'].copy()
kerala['new_cases'] = kerala['Confirmed'].diff()
kerala = kerala.iloc[-10:]
```

In [389]:
```
telangana = stats_data[stats_data['state'] == 'Telangana'].copy()
telangana['new_cases'] = telangana['Confirmed'].diff()
telangana = telangana.iloc[-10:]
```

In [390]:
```
# Total population of Maharashtra, Delhi, Kerala, Telangana

my_dict  = {'Delhi':19814000 ,'Maharashtra':122153000 ,'Kerala':35125000 ,
'Telangana':37220000,
'UP':199812341,
'Bihar':103804637,
'West Bengal':91347736,
'MP':72597565,
'TN':72138958,
'Rajasthan':68621012,
'Karnataka':61130704,
'Gujarat':60383628,
'AP':49386799,
'Odisha': 41947358,
'Jharkhand':32966238,
'Assam':31169272,
'Punjab':27704236,
'Chattisgarh':25540196,
'Haryana':25353081,
'J&K':12548926,
'Uttarakhand': 10116752,
'Himachal': 6864602,
'Tripura': 3671032,
'Meghalaya': 2964007,
'Manipur': 2721756,
'Nagaland': 1980602,
'Goa': 1457723,
'Arunachal':1382611,
'Mizoram': 1091014,
'Sikkim': 607688,
'Puducherry':1247953,
'D&D': 585764,
'A&N': 380581,
'Lakshadweep':64473,
'Chandigarh':1055450,
'Ladakh': 274000   }
df5 = pd.DataFrame(list(my_dict.items()),columns=['State','Total_Populatio
n'])
```

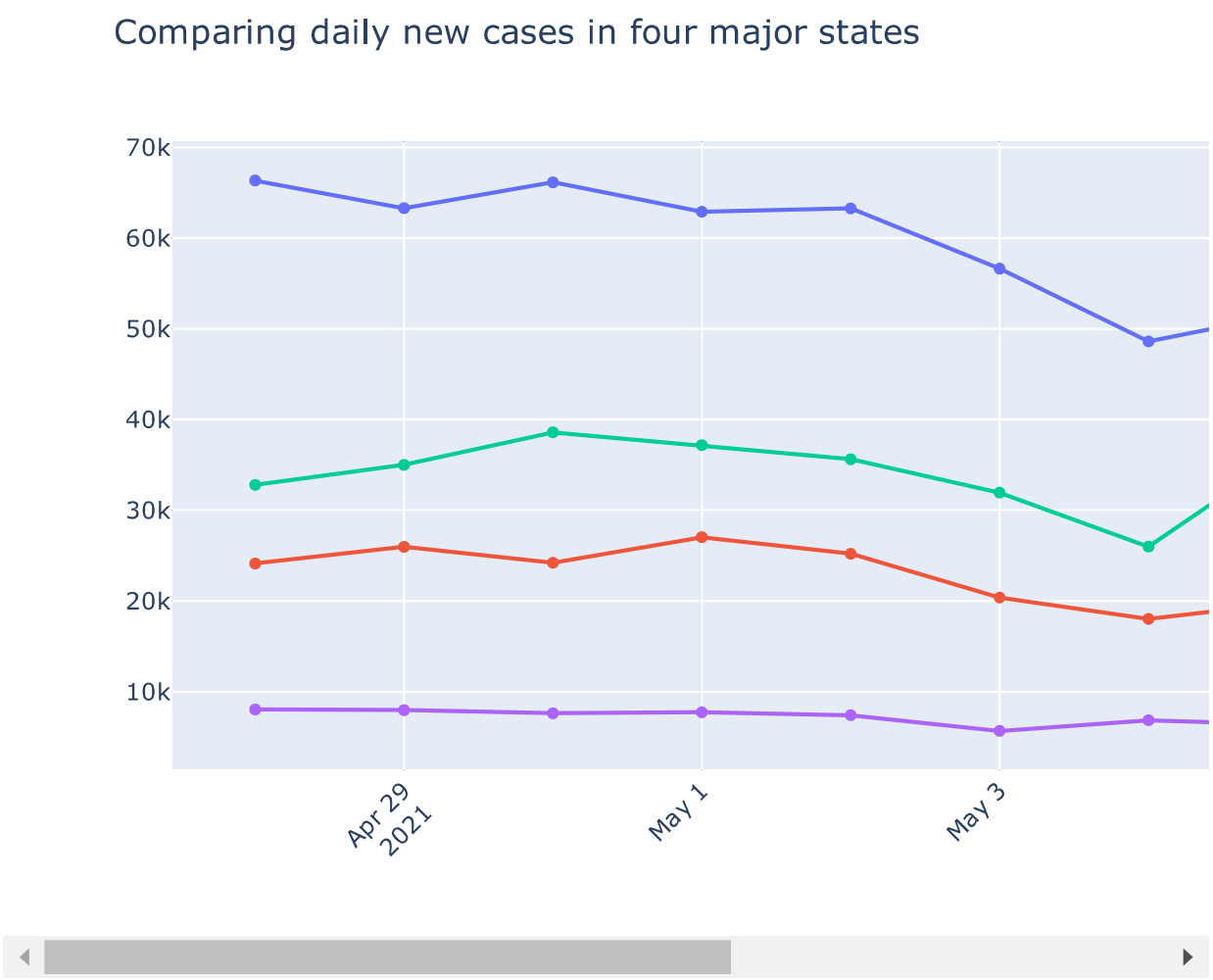
In [391]:
```
df5.shape
```

Out[391]: (36, 2)

In [398]:
```
latest_vacc = vaccine_data[vaccine_data['Updated On'] == '06/05/2021']
#state_latest_vacc = latest_vacc[latest_vacc['State'].isin(['Delhi','Mahara
shtra','Kerala','Telangana'])]
pop_vs_vacc = latest_vacc[['State','Total Individuals Vaccinated', 'Total D
oses Administered']].copy()
```

In [399]:
```
pop_vs_vacc = pop_vs_vacc.merge(df5, left_on='State',right_on='State')
pop_vs_vacc['Percentage_popuation_vaccinated'] = (pop_vs_vacc['Total Indivi
duals Vaccinated']/pop_vs_vacc['Total_Population'])*100
```

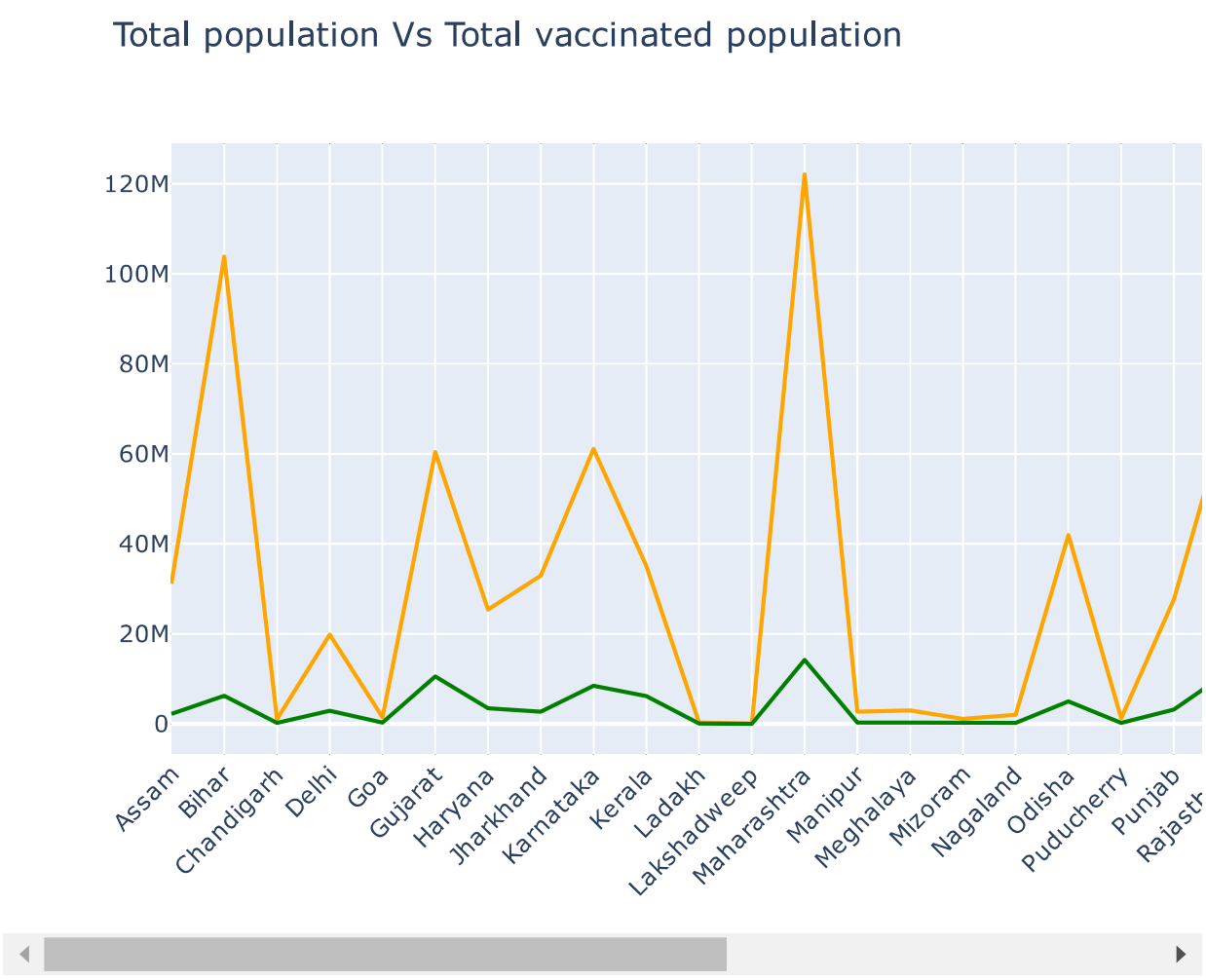
In [400]:
```
fig = go.Figure()
fig.add_trace(go.Line(
    x=maharashtra['Date'],
    y=maharashtra['new_cases'],
    name='maharashtra',
))
fig.add_trace(go.Line(
    x=delhi['Date'],
    y=delhi['new_cases'],
    name='delhi',
))
fig.add_trace(go.Line(
    x=kerala['Date'],
    y=kerala['new_cases'],
    name='kerala',
))
fig.add_trace(go.Line(
    x=telangana['Date'],
    y=telangana['new_cases'],
    name='telangana',
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Comparing daily new cases in four major states',
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



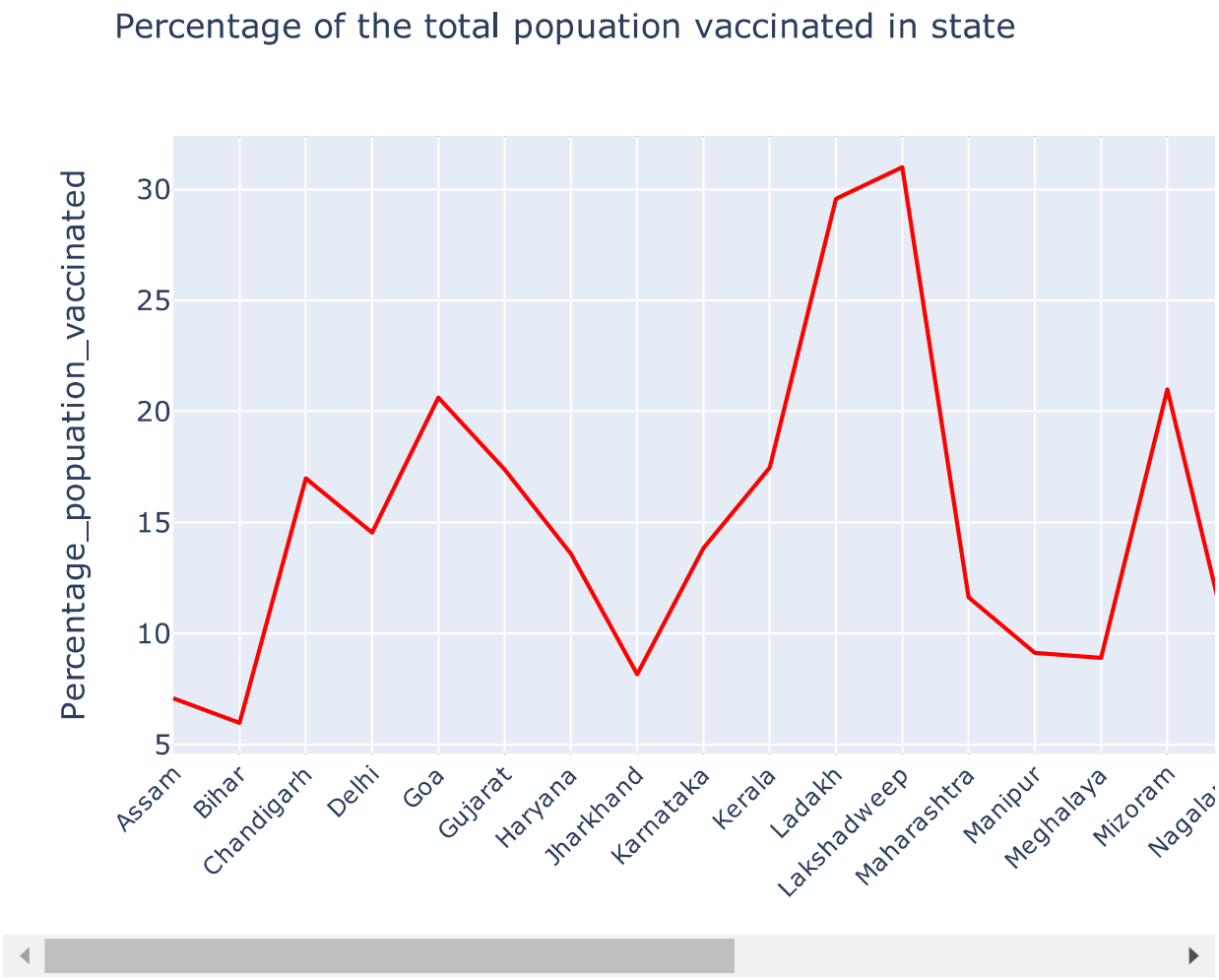
Observation: Apart from Maharashtra other three states are starting to see decline in daily cases. Lets see the vaccination details of these states to understand this decline further.

Note: from google.com I have taken the total population data for these four states.

In [403]:
```
fig = go.Figure()
fig.add_trace(go.Line(
    x=pop_vs_vacc['State'],
    y=pop_vs_vacc['Total_Population'],
    name='Total Population',
    marker_color='orange'
))
fig.add_trace(go.Line(
    x=pop_vs_vacc['State'],
    y=pop_vs_vacc['Total Individuals Vaccinated'],
    name='Total Individuals Vaccinated',
    marker_color='green'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Total population Vs Total vaccinated population',
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



In [404]:
```
fig = go.Figure()
fig.add_trace(go.Line(
    x=pop_vs_vacc['State'],
    y=pop_vs_vacc['Percentage_popuation_vaccinated'],
    name='Percentage_popuation_vaccinated',
    marker_color='red'
))
# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(autosize=False,
    width=1000,
    height=500,
    title_text='Percentage of the total popuation vaccinated in state',
    yaxis=dict(
        title='Percentage_popuation_vaccinated',
        titlefont_size=16,
        tickfont_size=14,
    ),
    barmode='group', xaxis_tickangle=-45)

fig.show()
```



Observation: We saw that Delhi and Kerala are now seeing a decline in cases and here also we see that amongst these four states Delhi and Kerala are having more % of their population vaccinated, which could be a possible reason to see the decline in daily cases now.

Also, lesser cases are reported(as per population density) in Ladakh, Lakshadweep, Sikkim, Tripura as the higher "percentage" of the populatiom is vaccinated.

Conclusion: Focusing on vaccinating our overall population more aggressively can help largely control the pandemic.

Further I am trying to collect the data of how many vaccines are available with all the states and how many they need, so that the states in more need can get vaccines from the states that have most of the population already vaccinated.

In [ ]: