In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly
import plotly.express as px
import plotly.graph_objects as go
import folium
from folium import plugins
plt.rcParams['figure.figsize'] = 10, 12
import warnings
from sklearn.metrics import mean_squared_error
warnings.filterwarnings('ignore')
%matplotlib inline
```

In [2]:

```
df_India= pd.read_csv('covid_19_India.csv')
India_coord = pd.read_excel('Indian Coordinates.xlsx')
#guys i will send you these files and these are
#the file you have to use for your project
```

In [3]:

```
print(df_India.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2450 entries, 0 to 2449
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Sno                       2450 non-null   int64
 1   Date                      2450 non-null   object
 2   Time                      2450 non-null   object
 3   State/UnionTerritory      2450 non-null   object
 4   ConfirmedIndianNational   2450 non-null   object
 5   ConfirmedForeignNational  2450 non-null   object
 6   Cured                     2450 non-null   int64
 7   Deaths                    2450 non-null   int64
 8   Confirmed                 2450 non-null   int64
dtypes: int64(4), object(5)
memory usage: 172.4+ KB
None
```

In [4]:
```
df_India.head()
```

Out[4]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cu
0	1	30/01/20	6:00 PM	Kerala	1	0	
1	2	31/01/20	6:00 PM	Kerala	1	0	
2	3	01/02/20	6:00 PM	Kerala	2	0	
3	4	02/02/20	6:00 PM	Kerala	3	0	
4	5	03/02/20	6:00 PM	Kerala	3	0	

In [5]:
```
df_India.tail()
```

Out[5]:

	Sno	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational
2445	2446	29/05/20	8:00 AM	Tripura	-	-
2446	2447	29/05/20	8:00 AM	Uttarakhand	-	-
2447	2448	29/05/20	8:00 AM	Uttar Pradesh	-	-
2448	2449	29/05/20	8:00 AM	West Bengal	-	-
2449	2450	29/05/20	8:00 AM	Cases being reassigned to states	-	-

In [6]:
```
df_India.dtypes
```

Out[6]:
```
Sno                          int64
Date                        object
Time                        object
State/UnionTerritory        object
ConfirmedIndianNational     object
ConfirmedForeignNational    object
Cured                        int64
Deaths                       int64
Confirmed                    int64
dtype: object
```

In [7]:
```
print(India_coord.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 3 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Name of State / UT  35 non-null     object 
 1   Latitude            35 non-null     float64
 2   Longitude           35 non-null     float64
dtypes: float64(2), object(1)
memory usage: 968.0+ bytes
None
```

In [8]:
```
India_coord.head()
```

Out[8]:

	Name of State / UT	Latitude	Longitude
0	Andaman And Nicobar	11.667026	92.735983
1	Andhra Pradesh	14.750429	78.570026
2	Arunachal Pradesh	27.100399	93.616601
3	Assam	26.749981	94.216667
4	Bihar	25.785414	87.479973

In [9]:
```
def replace_dash_with_zeros(inp):
    return int(inp.replace("-","0"))

df_India.drop(['Sno'],axis=1,inplace=True)
df_India['Date'] = pd.to_datetime(df_India['Date'], format = "%d/%m/%y")
# https://www.stat.berkeley.edu/~s133/dates.html
df_India['ConfirmedIndianNational'] = df_India['ConfirmedIndianNational'].apply(r
df_India['ConfirmedForeignNational'] = df_India['ConfirmedForeignNational'].apply
df_India.sort_values("Confirmed", ascending = False, inplace = True)
df_India
```

Out[9]:

	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured
2397	2020-05-28	8:00 AM	Maharashtra	0	0	17918
2433	2020-05-29	8:00 AM	Maharashtra	0	0	17918
2361	2020-05-27	8:00 AM	Maharashtra	0	0	16954
2325	2020-05-26	8:00 AM	Maharashtra	0	0	15786
2290	2020-05-25	8:00 AM	Maharashtra	0	0	14600
...	...	...	...	...	...	...
1047	2020-04-17	5:00 PM	Nagaland	0	0	0
1179	2020-04-21	5:00 PM	Nagaland	0	0	0
981	2020-04-15	5:00 PM	Nagaland	0	0	0
1113	2020-04-19	5:00 PM	Nagaland	0	0	0
1014	2020-04-16	5:00 PM	Nagaland	0	0	0

2450 rows × 8 columns

In [10]:
```
df_India.loc[df_India["ConfirmedForeignNational"] == "-",:]
```

Out[10]:

Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deatl

In [11]:
```
list(zip(df_India.columns,df_India.dtypes,df_India.isna().sum()))
```

Out[11]:
```
[('Date', dtype('<M8[ns]'), 0),
 ('Time', dtype('O'), 0),
 ('State/UnionTerritory', dtype('O'), 0),
 ('ConfirmedIndianNational', dtype('int64'), 0),
 ('ConfirmedForeignNational', dtype('int64'), 0),
 ('Cured', dtype('int64'), 0),
 ('Deaths', dtype('int64'), 0),
 ('Confirmed', dtype('int64'), 0)]
```

In [12]:
```
print(f'We have data available from : {df_India.Date.min()} to {df_India.Date.ma
```

```
We have data available from : 2020-01-30 00:00:00 to 2020-05-29 00:00:00
```

In [13]:
```
df_India.groupby(["State/UnionTerritory", "Date"]).sum()
```

Out[13]:

		ConfirmedIndianNational	ConfirmedForeignNational	Cured	Deaths	Con
State/UnionTerritory	**Date**					
Andaman and Nicobar Islands	**2020-03-26**	1	0	0	0	
	2020-03-27	1	0	0	0	
	2020-03-28	6	0	0	0	
	2020-03-29	0	0	0	0	
	2020-03-30	0	0	0	0	
...	**...**	...	...	...	...	
West Bengal	**2020-05-25**	0	0	1339	272	
	2020-05-26	0	0	1414	278	
	2020-05-27	0	0	1486	283	
	2020-05-28	0	0	1578	289	
	2020-05-29	0	0	1578	289	

2450 rows × 5 columns

In [14]:

```
States = df_India['State/UnionTerritory'].unique().tolist()
States
```

Out[14]:

```
['Maharashtra',
 'Tamil Nadu',
 'Delhi',
 'Gujarat',
 'Rajasthan',
 'Madhya Pradesh',
 'Uttar Pradesh',
 'Cases being reassigned to states',
 'West Bengal',
 'Andhra Pradesh',
 'Bihar',
 'Karnataka',
 'Punjab',
 'Telengana',
 'Jammu and Kashmir',
 'Odisha',
 'Haryana',
 'Kerala',
 'Assam',
 'Uttarakhand',
 'Jharkhand',
 'Chhattisgarh',
 'Chandigarh',
 'Himachal Pradesh',
 'Tripura',
 'Unassigned',
 'Goa',
 'Ladakh',
 'Puducherry',
 'Manipur',
 'Andaman and Nicobar Islands',
 'Meghalaya',
 'Nagaland',
 'Arunachal Pradesh',
 'Dadar Nagar Haveli',
 'Sikkim',
 'Mizoram']
```

In [15]:

```
States.remove("Cases being reassigned to states")
States.remove("Unassigned")
States
```

Out[15]:

```
['Maharashtra',
 'Tamil Nadu',
 'Delhi',
 'Gujarat',
 'Rajasthan',
 'Madhya Pradesh',
 'Uttar Pradesh',
 'West Bengal',
 'Andhra Pradesh',
 'Bihar',
 'Karnataka',
 'Punjab',
 'Telengana',
 'Jammu and Kashmir',
 'Odisha',
 'Haryana',
 'Kerala',
 'Assam',
 'Uttarakhand',
 'Jharkhand',
 'Chhattisgarh',
 'Chandigarh',
 'Himachal Pradesh',
 'Tripura',
 'Goa',
 'Ladakh',
 'Puducherry',
 'Manipur',
 'Andaman and Nicobar Islands',
 'Meghalaya',
 'Nagaland',
 'Arunachal Pradesh',
 'Dadar Nagar Haveli',
 'Sikkim',
 'Mizoram']
```

In [16]:

```
len(States)
```

Out[16]:

```
35
```

Merging Data Frames

In [17]:

```
df_final_India = pd.DataFrame()
dates = pd.DataFrame({"Date": pd.date_range(df_India.Date.min(),df_India.Date.ma
for state in States:
    all_dates_df = pd.merge(dates,
                            df_India.loc[df_India['State/UnionTerritory'] == stat
                           how = "left")
    all_dates_df['State/UnionTerritory'] = state
    all_dates_df = all_dates_df.fillna(0)
    all_dates_df['New Cases'] = all_dates_df['Confirmed'] - all_dates_df['Confirm
#     print(state)
#     display(all_dates_df.loc[all_dates_df['New Cases'] <  0,:])
    df_final_India = pd.concat([df_final_India, all_dates_df],axis = 0)
print("Finally we have a data of Size: ",df_final_India.shape)
df_final_India.head()
```

```
Finally we have a data of Size:  (4235, 9)
```

Out[17]:

	Date	Time	State/UnionTerritory	ConfirmedIndianNational	ConfirmedForeignNational	Cured	De
0	2020-01-30	0	Maharashtra	0.0	0.0	0.0	
1	2020-01-31	0	Maharashtra	0.0	0.0	0.0	
2	2020-02-01	0	Maharashtra	0.0	0.0	0.0	
3	2020-02-02	0	Maharashtra	0.0	0.0	0.0	
4	2020-02-03	0	Maharashtra	0.0	0.0	0.0	

In [18]:

```
df_final_India.dropna(inplace = True)
df_final_India.shape
```

Out[18]:

```
(4200, 9)
```

In [19]:

```
del df_final_India['Time']
del df_final_India['ConfirmedIndianNational']
del df_final_India['ConfirmedForeignNational']
```

In [20]:

```
df_final_India
```

Out[20]:

	Date	State/UnionTerritory	Cured	Deaths	Confirmed	New Cases
1	2020-01-31	Maharashtra	0.0	0.0	0.0	0.0
2	2020-02-01	Maharashtra	0.0	0.0	0.0	0.0
3	2020-02-02	Maharashtra	0.0	0.0	0.0	0.0
4	2020-02-03	Maharashtra	0.0	0.0	0.0	0.0
5	2020-02-04	Maharashtra	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...
116	2020-05-25	Mizoram	1.0	0.0	1.0	0.0
117	2020-05-26	Mizoram	1.0	0.0	1.0	0.0
118	2020-05-27	Mizoram	1.0	0.0	1.0	0.0
119	2020-05-28	Mizoram	1.0	0.0	1.0	0.0
120	2020-05-29	Mizoram	1.0	0.0	1.0	0.0

4200 rows × 6 columns

In [21]:

```
df_final_India.groupby(["State/UnionTerritory", "Date"]).sum()
```

Out[21]:

		Cured	Deaths	Confirmed	New Cases
State/UnionTerritory	**Date**				
Andaman and Nicobar Islands	**2020-01-31**	0.0	0.0	0.0	0.0
	2020-02-01	0.0	0.0	0.0	0.0
	2020-02-02	0.0	0.0	0.0	0.0
	2020-02-03	0.0	0.0	0.0	0.0
	2020-02-04	0.0	0.0	0.0	0.0
...	**...**	...	...	...	...
West Bengal	**2020-05-25**	1339.0	272.0	3667.0	208.0
	2020-05-26	1414.0	278.0	3816.0	149.0
	2020-05-27	1486.0	283.0	4009.0	193.0
	2020-05-28	1578.0	289.0	4192.0	183.0
	2020-05-29	1578.0	289.0	4192.0	0.0

4200 rows × 4 columns

In [22]:
```
df_final_India = df_final_India.groupby(["State/UnionTerritory", "Date"]).sum().r
df_final_India
```

Out[22]:

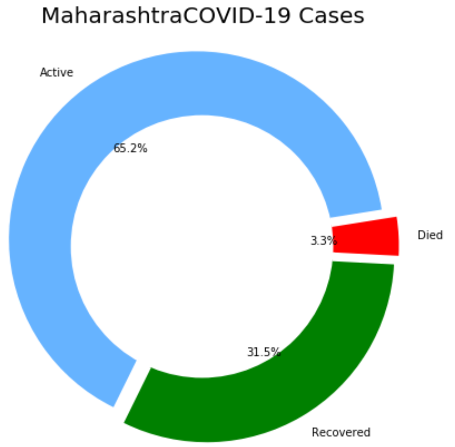
	State/UnionTerritory	Date	Cured	Deaths	Confirmed	New Cases
0	Andaman and Nicobar Islands	2020-01-31	0.0	0.0	0.0	0.0
1	Andaman and Nicobar Islands	2020-02-01	0.0	0.0	0.0	0.0
2	Andaman and Nicobar Islands	2020-02-02	0.0	0.0	0.0	0.0
3	Andaman and Nicobar Islands	2020-02-03	0.0	0.0	0.0	0.0
4	Andaman and Nicobar Islands	2020-02-04	0.0	0.0	0.0	0.0
...	...	...	...	...	...	...
4195	West Bengal	2020-05-25	1339.0	272.0	3667.0	208.0
4196	West Bengal	2020-05-26	1414.0	278.0	3816.0	149.0
4197	West Bengal	2020-05-27	1486.0	283.0	4009.0	193.0
4198	West Bengal	2020-05-28	1578.0	289.0	4192.0	183.0
4199	West Bengal	2020-05-29	1578.0	289.0	4192.0	0.0

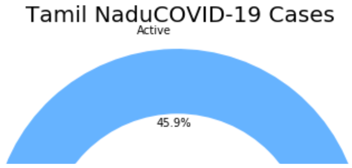
4200 rows × 6 columns

Statewise Covid19 Status in India

In [23]:
```
def plot_pie(active,cured,death,title):
    labels = ['Active','Recovered','Died']
    sizes = [active,cured,death]
    color= ['#66b3ff','green','red']
    explode = []

    for i in labels:
        explode.append(0.05)

    plt.figure(figsize= (15,6))
    plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=9, explode =explo
    centre_circle = plt.Circle((0,0),0.70,fc='white')

    fig = plt.gcf()
    fig.gca().add_artist(centre_circle)
    plt.title(title +  'COVID-19 Cases',fontsize = 20)
    plt.axis('equal')
    plt.tight_layout()
```

In [24]:
```
total_cases_india = 0
cured_cases_india = 0
death_cases_india = 0
active_cases_india = 0
state_df = pd.DataFrame()

for state in States:
    one_state_df = df_final_India.loc[df_final_India['State/UnionTerritory'] == s
    state_df = pd.concat([state_df,pd.DataFrame(one_state_df.iloc[-1,:]).T],axis
    total_cases = one_state_df['Confirmed'].values[-1]
    cured = one_state_df['Cured'].values[-1]
    deaths = one_state_df['Deaths'].values[-1]
    active = total_cases - cured - deaths
    plot_pie(active, cured, deaths,state)
    total_cases_india += total_cases
    cured_cases_india += cured
    death_cases_india += deaths
    active_cases_india += active
```





In [25]:

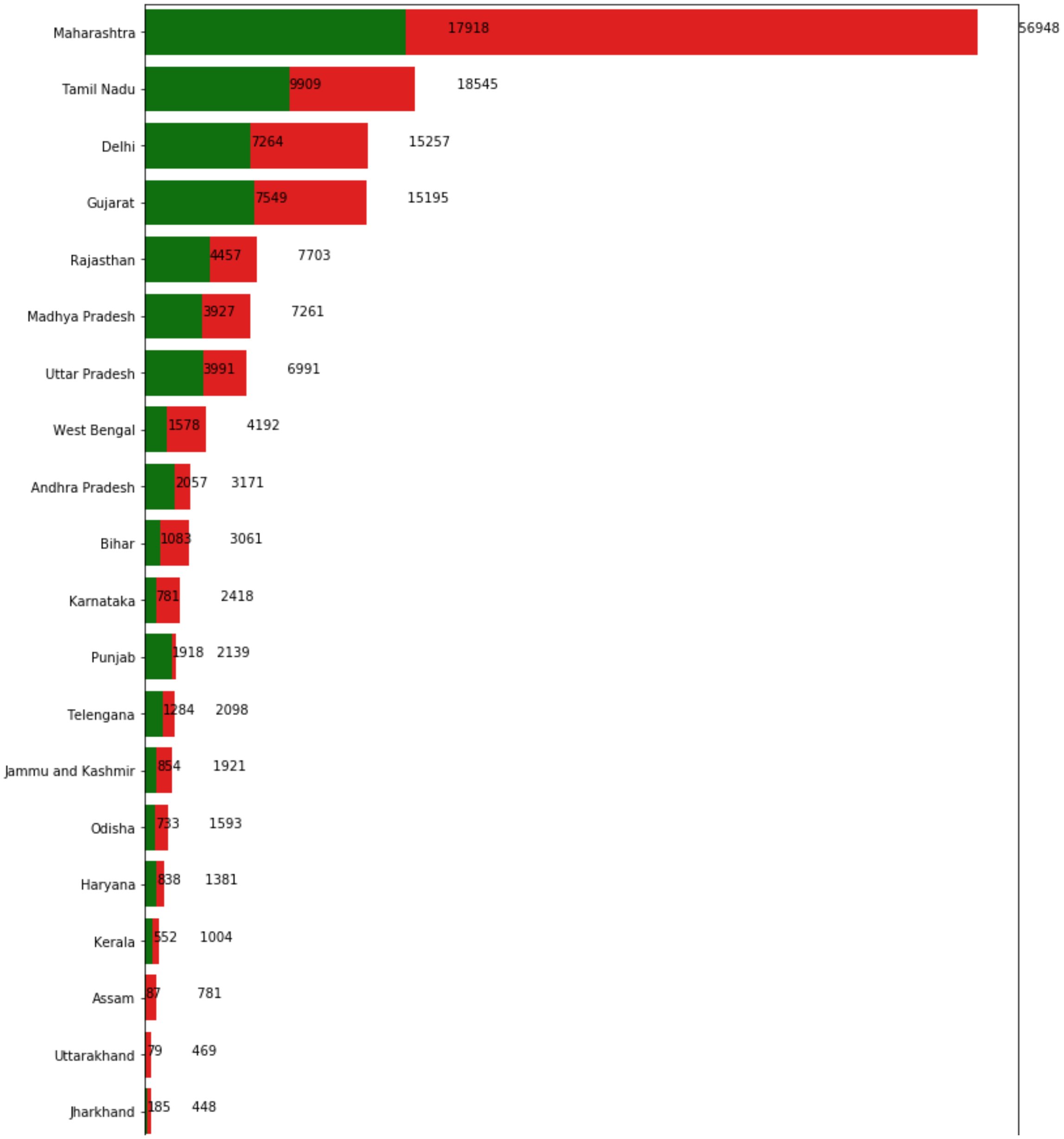
```python
state_df.reset_index(inplace = True,drop = True)
state_df
```

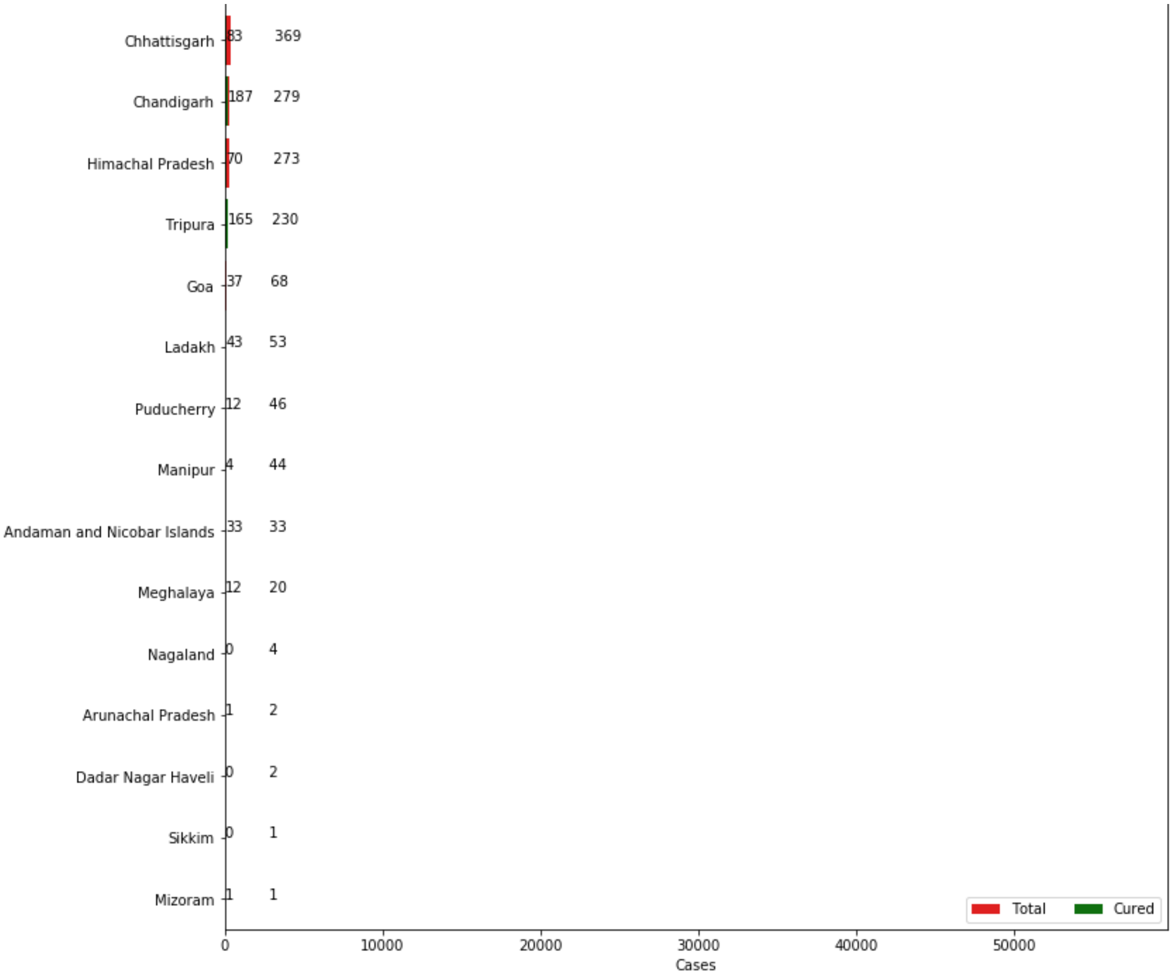
Out[25]:

	State/UnionTerritory	Date	Cured	Deaths	Confirmed	New Cases
0	Maharashtra	2020-05-29	17918	1897	56948	0
1	Tamil Nadu	2020-05-29	9909	133	18545	0
2	Delhi	2020-05-29	7264	303	15257	0
3	Gujarat	2020-05-29	7549	938	15195	0
4	Rajasthan	2020-05-29	4457	173	7703	0
5	Madhya Pradesh	2020-05-29	3927	313	7261	0
6	Uttar Pradesh	2020-05-29	3991	182	6991	0
7	West Bengal	2020-05-29	1578	289	4192	0
8	Andhra Pradesh	2020-05-29	2057	58	3171	0
9	Bihar	2020-05-29	1083	15	3061	0
10	Karnataka	2020-05-29	781	47	2418	0
11	Punjab	2020-05-29	1918	40	2139	0
12	Telengana	2020-05-29	1284	63	2098	0
13	Jammu and Kashmir	2020-05-29	854	26	1921	0
14	Odisha	2020-05-29	733	7	1593	0
15	Haryana	2020-05-29	838	18	1381	0
16	Kerala	2020-05-29	552	7	1004	0
17	Assam	2020-05-29	87	4	781	0
18	Uttarakhand	2020-05-29	79	4	469	0
19	Jharkhand	2020-05-29	185	4	448	0
20	Chhattisgarh	2020-05-29	83	0	369	0
21	Chandigarh	2020-05-29	187	4	279	0
22	Himachal Pradesh	2020-05-29	70	5	273	0
23	Tripura	2020-05-29	165	0	230	0
24	Goa	2020-05-29	37	0	68	0
25	Ladakh	2020-05-29	43	0	53	0
26	Puducherry	2020-05-29	12	0	46	0
27	Manipur	2020-05-29	4	0	44	0
28	Andaman and Nicobar Islands	2020-05-29	33	0	33	0
29	Meghalaya	2020-05-29	12	1	20	0
30	Nagaland	2020-05-29	0	0	4	0
31	Arunachal Pradesh	2020-05-29	1	0	2	0
32	Dadar Nagar Haveli	2020-05-29	0	0	2	0

	State/UnionTerritory	Date	Cured	Deaths	Confirmed	New Cases
33	Sikkim	2020-05-29	0	0	1	0
34	Mizoram	2020-05-29	1	0	1	0

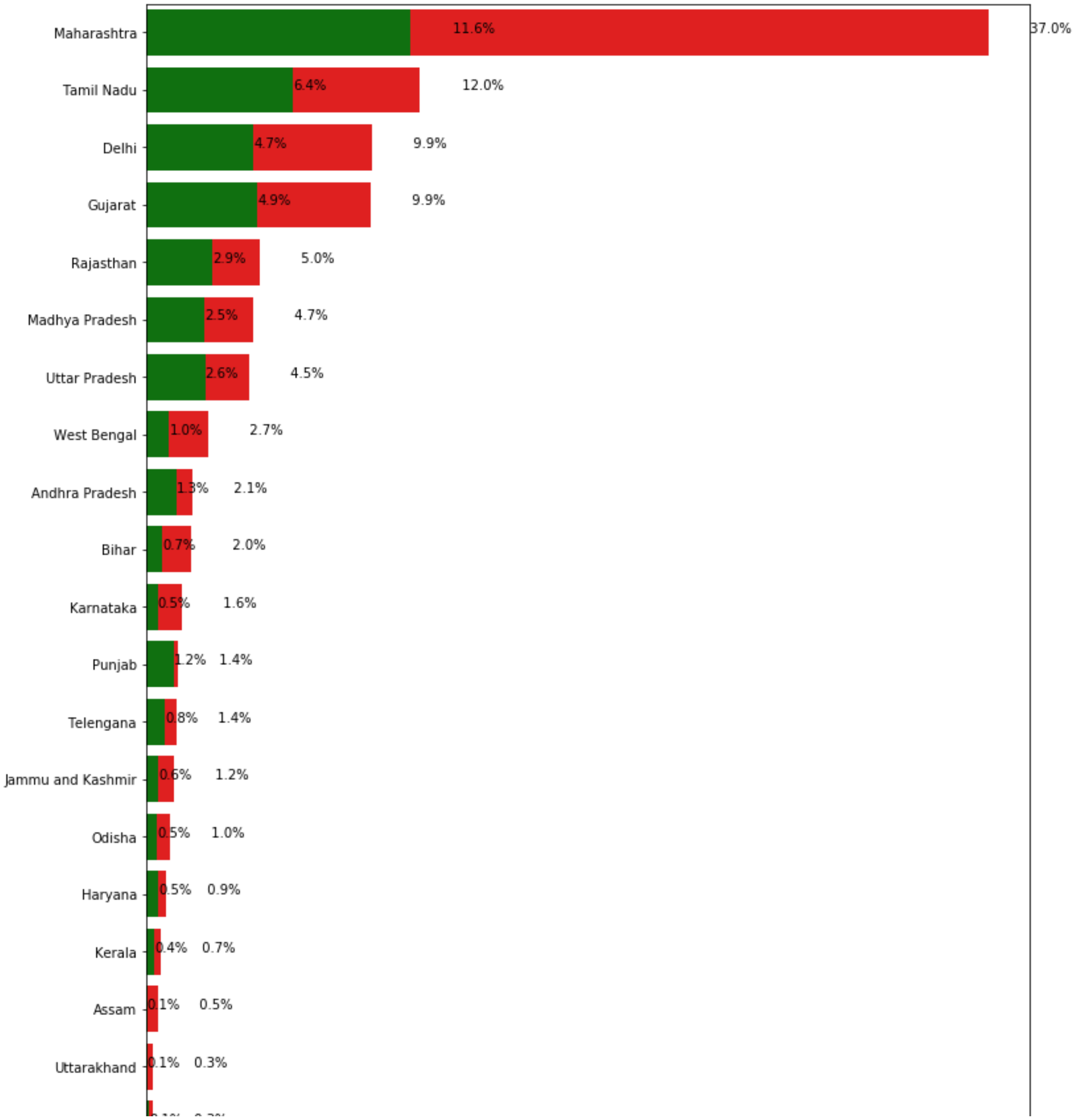
```
In [26]: f, ax = plt.subplots(figsize=(12, 28))
         data = state_df[['State/UnionTerritory','Confirmed','Cured','Deaths']]
         data.sort_values('Confirmed',ascending=False,inplace=True)
         sns.set_color_codes("pastel")
         sns.barplot(x="Confirmed", y="State/UnionTerritory", data=data,label="Total", col
         sns.set_color_codes("muted")
         sns.barplot(x="Cured", y="State/UnionTerritory", data=data, label="Cured", color
         ax.legend(ncol=5, loc="lower right", frameon=True)
         ax.set(ylabel="",xlabel="Cases")
         i = 0
         for p in ax.patches:
             x = p.get_x() + p.get_width() + 3
             y = p.get_y() + p.get_height()/2
             if i <= len(States):
                 ax.annotate(" "*10 + str(int(p.get_width())), (x, y))
             else:
                 ax.annotate(int(p.get_width()), (x, y))

             i += 1
```

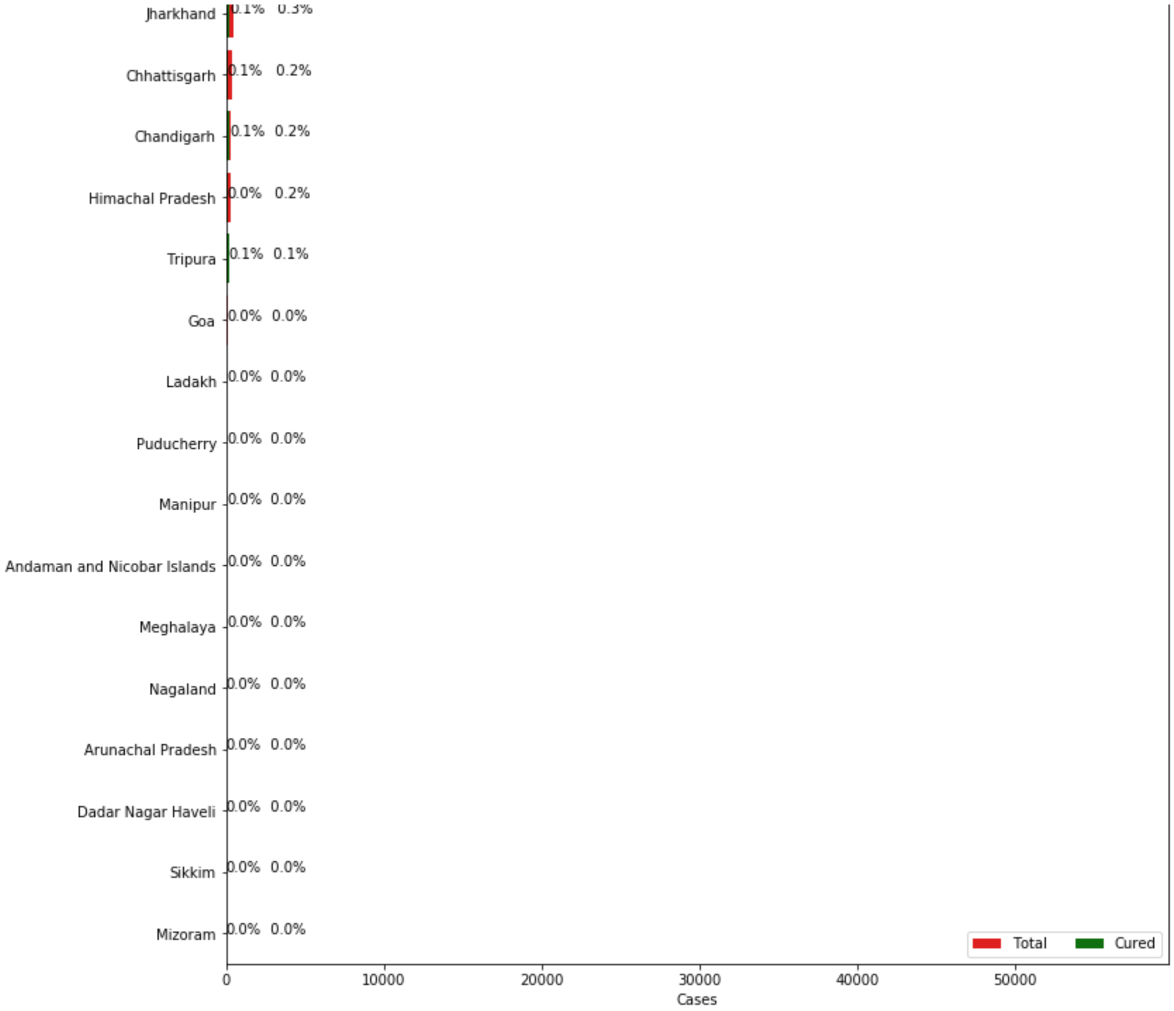



Chhattisgarh
83
369
Chandigarh
187
279
Himachal Pradesh
70
273
Tripura
165
230
Goa
37
68
Ladakh
43
53
Puducherry
12
46
Manipur
4
44
Andaman and Nicobar Islands
33
33
Meghalaya
12
20
Nagaland
0
4
Arunachal Pradesh
1
2
Dadar Nagar Haveli
0
2
Sikkim
0
1
Mizoram
1
1
🟥 Total
🟩 Cured
0
10000
20000
30000
40000
50000
Cases

In [27]:

```python
f, ax = plt.subplots(figsize=(12, 28))
data = state_df[['State/UnionTerritory','Confirmed','Cured','Deaths']]
data.sort_values('Confirmed',ascending=False,inplace=True)
sns.set_color_codes("pastel")
sns.barplot(x="Confirmed", y="State/UnionTerritory", data=data,label="Total", col
sns.set_color_codes("muted")
sns.barplot(x="Cured", y="State/UnionTerritory", data=data, label="Cured", color=
ax.legend(ncol=5, loc="lower right", frameon=True)
ax.set(ylabel="",xlabel="Cases")
total = total_cases_india
i = 0
for p in ax.patches:
    percentage = '{:.1f}%'.format(100 * p.get_width()/total)
    x = p.get_x() + p.get_width() + 3
    y = p.get_y() + p.get_height()/2
    if i <= len(States):
        ax.annotate(" "*10 + str(percentage), (x, y))
    else:
        ax.annotate(percentage, (x, y))

    i += 1
```



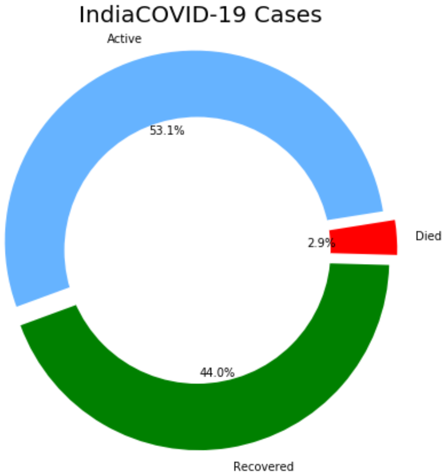


Overall Covid19 Status in India

In [28]:

```
print("Total infected cases in India: ", total_cases_india)
print("Total cured cases in India: ", cured_cases_india)
print("Total active cases in India: ", active_cases_india)
print("Total death cases in India: ", death_cases_india)
plot_pie(active_cases_india, cured_cases_india, death_cases_india, "India")
```

```
Total infected cases in India:  154001.0
Total cured cases in India:  67692.0
Total active cases in India:  81778.0
Total death cases in India:  4531.0
```



VISUALISING THE SPREADS GEOGRAPHICALLY

In [29]:

```
India_coord.rename(columns = {"Name of State / UT" : "State/UnionTerritory"},inpl
```

In [30]:
```python
set(India_coord['State/UnionTerritory'].values).symmetric_difference(set(state_d
```

Out[30]:
```
{'Andaman And Nicobar ',
 'Andaman and Nicobar Islands',
 'Arunachal Pradesh',
 'Arunachal Pradesh ',
 'Assam',
 'Assam ',
 'Bihar',
 'Bihar ',
 'Chandigarh',
 'Chandigarh ',
 'Chhattisgarh',
 'Chhattisgarh ',
 'Dadar Nagar Haveli',
 'Dadra And Nagar Haveli ',
 'Goa',
 'Goa ',
 'Gujarat',
 'Himachal Pradesh',
 'Himachal Pradesh ',
 'Jammu and Kashmir',
 'Jharkhand',
 'Jharkhand ',
 'Ladakh',
 'Lakshadweep ',
 'Madhya Pradesh',
 'Madhya Pradesh ',
 'Manipur',
 'Manipur ',
 'Meghalaya',
 'Meghalaya ',
 'Mizoram',
 'Mizoram ',
 'Nagaland',
 'Nagaland ',
 'Odisha',
 'Orissa ',
 'Puducherry',
 'Puducherry ',
 'Sikkim',
 'Sikkim ',
 'Tripura',
 'Tripura ',
 'Union Territory of Jammu and Kashmir',
 'Union Territory of Ladakh',
 'West Bengal',
 'West Bengal '}
```

In [31]:
```python
India_coord['State/UnionTerritory'] = India_coord['State/UnionTerritory'].str.str
state_df['State/UnionTerritory'] = state_df['State/UnionTerritory'].str.strip()
```

In [32]:
```
set(India_coord['State/UnionTerritory'].values).symmetric_difference(set(state_df
```

Out[32]:
```
{'Andaman And Nicobar',
 'Andaman and Nicobar Islands',
 'Dadar Nagar Haveli',
 'Dadra And Nagar Haveli',
 'Gujarat',
 'Jammu and Kashmir',
 'Ladakh',
 'Lakshadweep',
 'Odisha',
 'Orissa',
 'Union Territory of Jammu and Kashmir',
 'Union Territory of Ladakh'}
```

In [33]:

```
India_coord.loc[India_coord.shape[0]] = ['Gujarat','22.2587','71.1924']
India_coord
```

Out[33]:

	State/UnionTerritory	Latitude	Longitude
0	Andaman And Nicobar	11.667	92.736
1	Andhra Pradesh	14.7504	78.57
2	Arunachal Pradesh	27.1004	93.6166
3	Assam	26.75	94.2167
4	Bihar	25.7854	87.48
5	Chandigarh	30.72	76.78
6	Chhattisgarh	22.0904	82.16
7	Dadra And Nagar Haveli	20.2666	73.0166
8	Delhi	28.67	77.23
9	Goa	15.492	73.818
10	Haryana	28.45	77.02
11	Himachal Pradesh	31.1	77.1666
12	Union Territory of Jammu and Kashmir	33.45	76.24
13	Jharkhand	23.8004	86.42
14	Karnataka	12.5704	76.92
15	Kerala	8.90037	76.57
16	Lakshadweep	10.5626	72.6369
17	Madhya Pradesh	21.3004	76.13
18	Maharashtra	19.2502	73.1602
19	Manipur	24.8	93.95
20	Meghalaya	25.5705	91.88
21	Mizoram	23.7104	92.72
22	Nagaland	25.667	94.1166
23	Orissa	19.8204	85.9
24	Puducherry	11.935	79.83
25	Punjab	31.52	75.98
26	Rajasthan	26.45	74.64
27	Sikkim	27.3333	88.6166
28	Telengana	18.1124	79.0193
29	Tamil Nadu	12.9204	79.15
30	Tripura	23.8354	91.28
31	Uttar Pradesh	27.6	78.05
32	Uttarakhand	30.3204	78.05

	State/UnionTerritory	Latitude	Longitude
33	West Bengal	22.5804	88.3299
34	Union Territory of Ladakh	34.1	77.34
35	Gujarat	22.2587	71.1924

```
In [34]: set(India_coord['State/UnionTerritory'].values).symmetric_difference(set(state_df
```

```
Out[34]: {'Andaman And Nicobar',
          'Andaman and Nicobar Islands',
          'Dadar Nagar Haveli',
          'Dadra And Nagar Haveli',
          'Jammu and Kashmir',
          'Ladakh',
          'Lakshadweep',
          'Odisha',
          'Orissa',
          'Union Territory of Jammu and Kashmir',
          'Union Territory of Ladakh'}
```

```
In [35]: India_coord['State/UnionTerritory'] = np.where(India_coord['State/UnionTerritory'
                                                 "Andaman and Nicobar Islands",India_coord
         India_coord['State/UnionTerritory'] = np.where(India_coord['State/UnionTerritory'
                                                 "Jammu and Kashmir",India_coord['State/Un
         India_coord['State/UnionTerritory'] = np.where(India_coord['State/UnionTerritory'
                                                 "Ladakh",India_coord['State/UnionTerritor
         India_coord['State/UnionTerritory'] = np.where(India_coord['State/UnionTerritory'
                                                 "Odisha",India_coord['State/UnionTerritor
         India_coord['State/UnionTerritory'] = np.where(India_coord['State/UnionTerritory'
                                                 "Dadar Nagar Haveli",India_coord['State/U
```

```
In [36]: set(India_coord['State/UnionTerritory'].values).symmetric_difference(set(state_df
```

```
Out[36]: {'Lakshadweep'}
```

In [37]:

```
df_full = pd.merge(India_coord,state_df,on='State/UnionTerritory').reset_index(dr
df_full
```

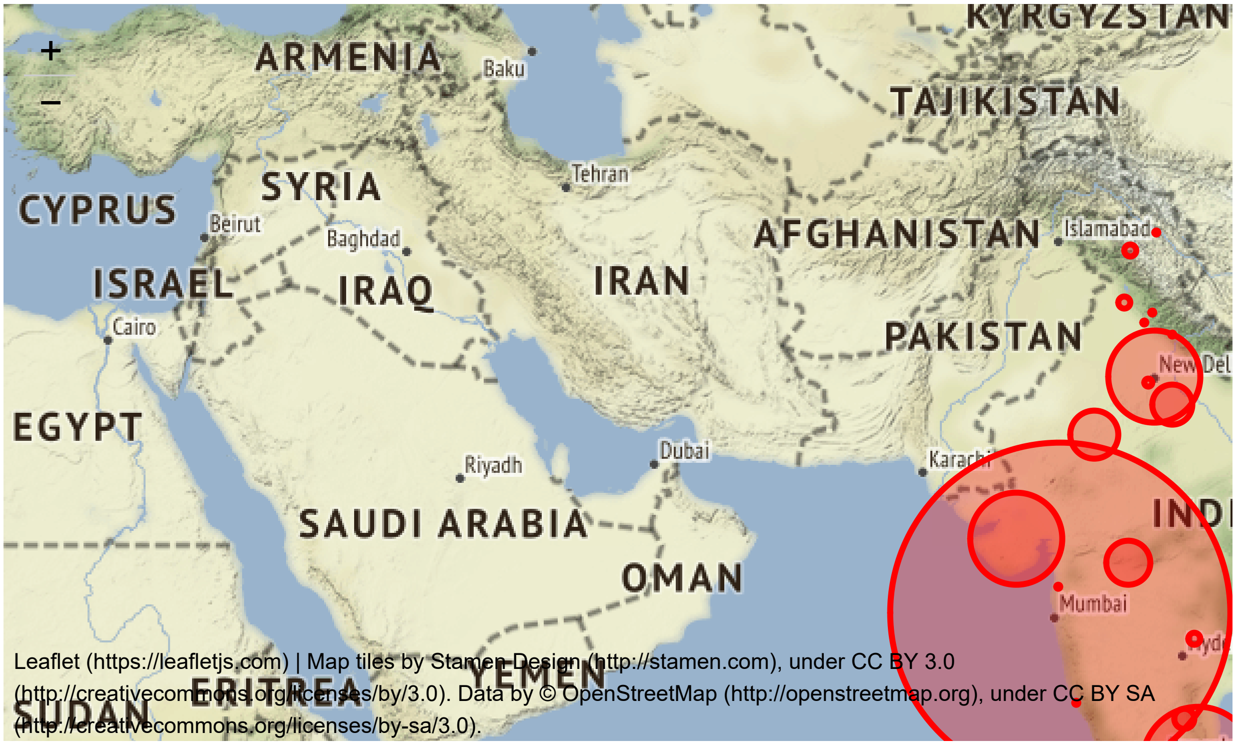
Out[37]:

	State/UnionTerritory	Latitude	Longitude	Date	Cured	Deaths	Confirmed	New Cases
0	Andaman and Nicobar Islands	11.667	92.736	2020-05-29	33	0	33	0
1	Andhra Pradesh	14.7504	78.57	2020-05-29	2057	58	3171	0
2	Arunachal Pradesh	27.1004	93.6166	2020-05-29	1	0	2	0
3	Assam	26.75	94.2167	2020-05-29	87	4	781	0
4	Bihar	25.7854	87.48	2020-05-29	1083	15	3061	0
5	Chandigarh	30.72	76.78	2020-05-29	187	4	279	0
6	Chhattisgarh	22.0904	82.16	2020-05-29	83	0	369	0
7	Dadar Nagar Haveli	20.2666	73.0166	2020-05-29	0	0	2	0
8	Delhi	28.67	77.23	2020-05-29	7264	303	15257	0
9	Goa	15.492	73.818	2020-05-29	37	0	68	0
10	Haryana	28.45	77.02	2020-05-29	838	18	1381	0
11	Himachal Pradesh	31.1	77.1666	2020-05-29	70	5	273	0
12	Jammu and Kashmir	33.45	76.24	2020-05-29	854	26	1921	0
13	Jharkhand	23.8004	86.42	2020-05-29	185	4	448	0
14	Karnataka	12.5704	76.92	2020-05-29	781	47	2418	0
15	Kerala	8.90037	76.57	2020-05-29	552	7	1004	0
16	Madhya Pradesh	21.3004	76.13	2020-05-29	3927	313	7261	0
17	Maharashtra	19.2502	73.1602	2020-05-29	17918	1897	56948	0
18	Manipur	24.8	93.95	2020-05-29	4	0	44	0
19	Meghalaya	25.5705	91.88	2020-05-29	12	1	20	0
20	Mizoram	23.7104	92.72	2020-05-29	1	0	1	0

	State/UnionTerritory	Latitude	Longitude	Date	Cured	Deaths	Confirmed	New Cases
21	Nagaland	25.667	94.1166	2020-05-29	0	0	4	0
22	Odisha	19.8204	85.9	2020-05-29	733	7	1593	0
23	Puducherry	11.935	79.83	2020-05-29	12	0	46	0
24	Punjab	31.52	75.98	2020-05-29	1918	40	2139	0
25	Rajasthan	26.45	74.64	2020-05-29	4457	173	7703	0
26	Sikkim	27.3333	88.6166	2020-05-29	0	0	1	0
27	Telengana	18.1124	79.0193	2020-05-29	1284	63	2098	0
28	Tamil Nadu	12.9204	79.15	2020-05-29	9909	133	18545	0
29	Tripura	23.8354	91.28	2020-05-29	165	0	230	0
30	Uttar Pradesh	27.6	78.05	2020-05-29	3991	182	6991	0
31	Uttarakhand	30.3204	78.05	2020-05-29	79	4	469	0
32	West Bengal	22.5804	88.3299	2020-05-29	1578	289	4192	0
33	Ladakh	34.1	77.34	2020-05-29	43	0	53	0
34	Gujarat	22.2587	71.1924	2020-05-29	7549	938	15195	0

In [69]:
```
map = folium.Map(location=[20, 70], zoom_start=4,tiles='Stamenterrain')

for lat, lon, value, name in zip(df_full['Latitude'], df_full['Longitude'], df_fu
    folium.CircleMarker([lat, lon], radius=value*0.0015, popup = ('<strong>State<
map
```

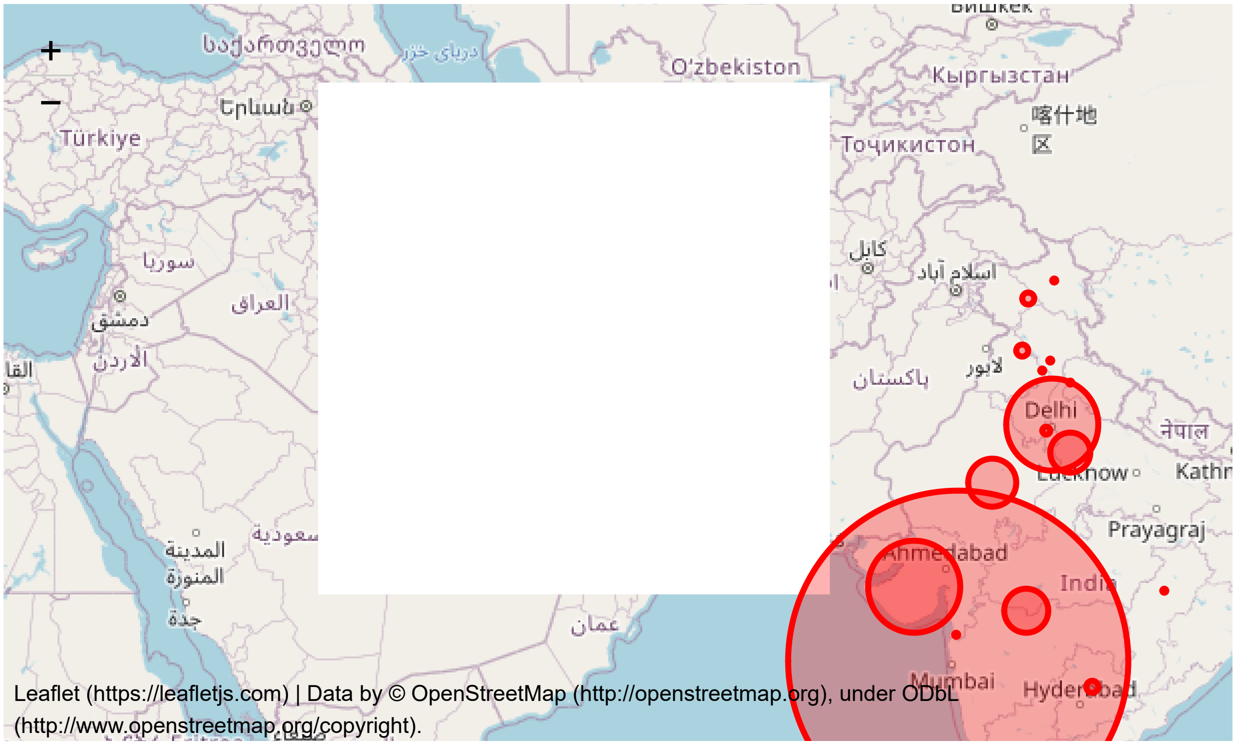
Out[69]:



Leaflet (https://leafletjs.com) | Map tiles by Stamen Design (http://stamen.com), under CC BY 3.0 (http://creativecommons.org/licenses/by/3.0). Data by © OpenStreetMap (http://openstreetmap.org), under CC BY SA (http://creativecommons.org/licenses/by-sa/3.0).

In [39]:
```
map = folium.Map(location=[20, 70], zoom_start=4,tiles='OpenStreetMap')

for lat, lon, value, name in zip(df_full['Latitude'], df_full['Longitude'], df_fu
    folium.CircleMarker([lat, lon], radius=value*0.0015, popup = ('<strong>State<
map
```

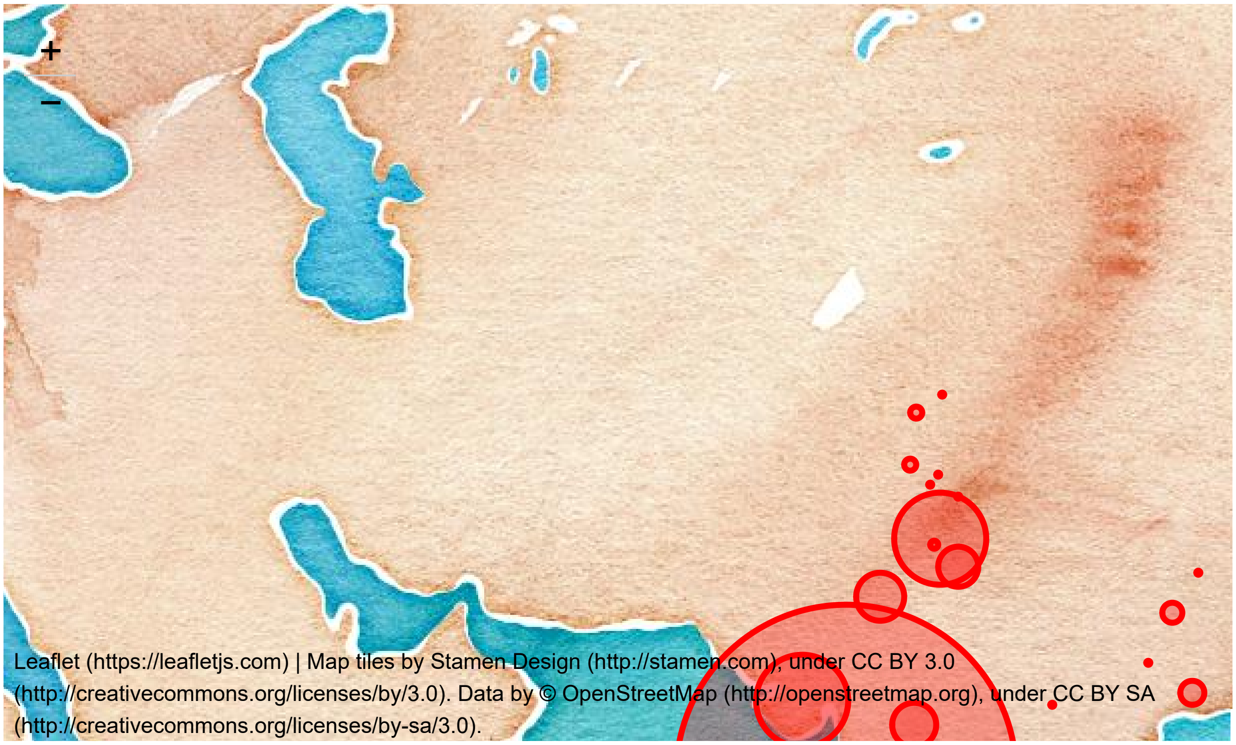
Out[39]:



Leaflet (https://leafletjs.com) | Data by © OpenStreetMap (http://openstreetmap.org), under ODbL (http://www.openstreetmap.org/copyright).

```
In [70]: map = folium.Map(location=[20, 70], zoom_start=4,tiles='Stamenwatercolor')

         for lat, lon, value, name in zip(df_full['Latitude'], df_full['Longitude'], df_fu
             folium.CircleMarker([lat, lon], radius=value*0.0015, popup = ('<strong>State<
         map
```

Out[70]:



Leaflet (https://leafletjs.com) | Map tiles by Stamen Design (http://stamen.com), under CC BY 3.0 (http://creativecommons.org/licenses/by/3.0). Data by © OpenStreetMap (http://openstreetmap.org), under CC BY SA (http://creativecommons.org/licenses/by-sa/3.0).

Lets check the trend of the virus

In [41]:
```
df_daywise_India = df_final_India.groupby("Date")['Confirmed','Cured','Deaths',"N
df_daywise_India
```

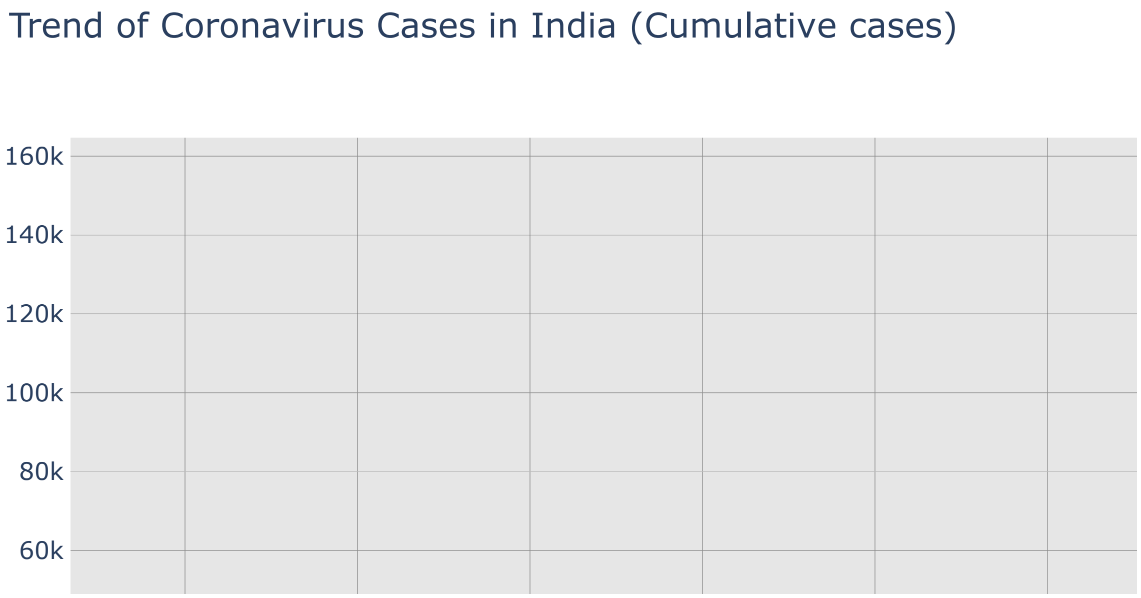
Out[41]:

	Date	Confirmed	Cured	Deaths	New Cases
0	2020-01-31	1.0	0.0	0.0	0.0
1	2020-02-01	2.0	0.0	0.0	1.0
2	2020-02-02	3.0	0.0	0.0	1.0
3	2020-02-03	3.0	0.0	0.0	0.0
4	2020-02-04	3.0	0.0	0.0	0.0
...	...	...	...	...	...
115	2020-05-25	136203.0	57721.0	4021.0	6673.0
116	2020-05-26	142410.0	60491.0	4167.0	6207.0
117	2020-05-27	147754.0	64426.0	4337.0	5344.0
118	2020-05-28	154001.0	67692.0	4531.0	6247.0
119	2020-05-29	154001.0	67692.0	4531.0	0.0

120 rows × 5 columns

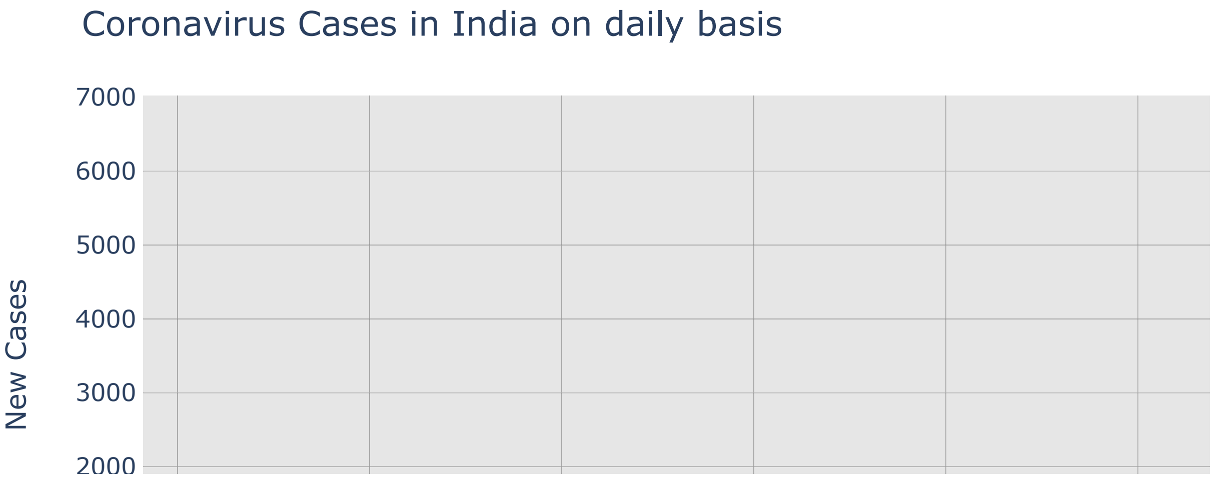
In [42]:

```
fig = go.Figure()
fig.add_trace(go.Scatter(x=df_daywise_India['Date'], y = df_daywise_India['Confir
fig.update_layout(title_text='Trend of Coronavirus Cases in India (Cumulative cas
fig.show()
```



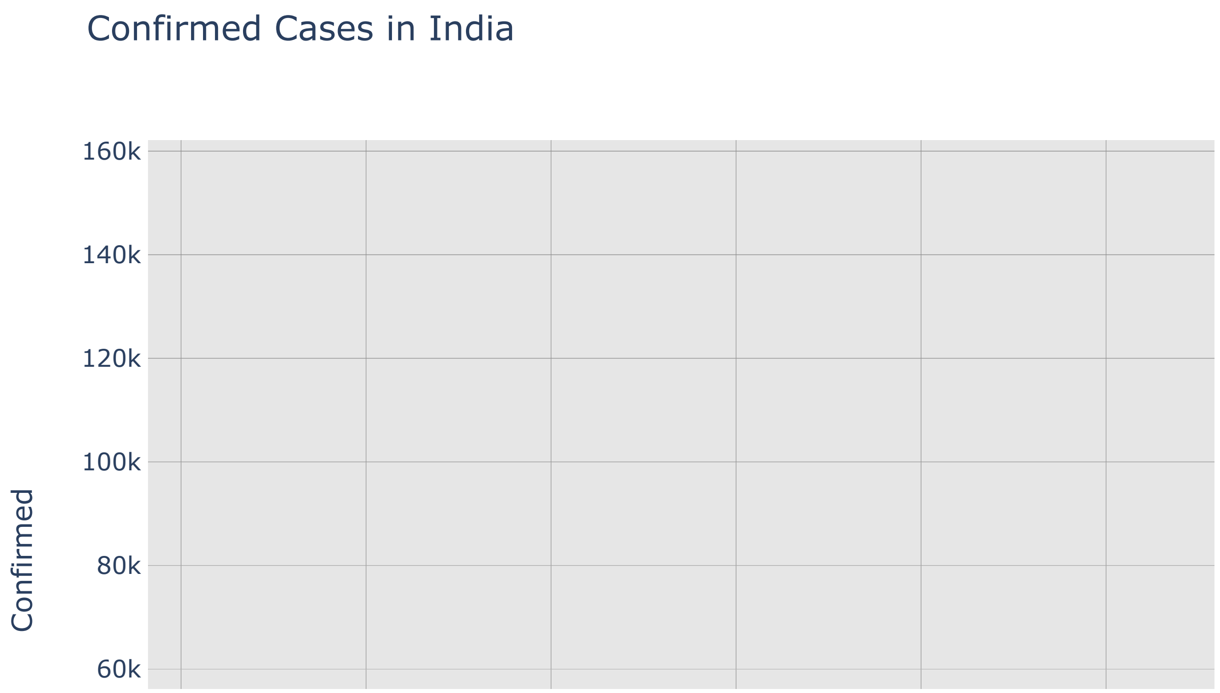
In [43]:

```
fig = px.bar(df_daywise_India, x="Date", y="New Cases", barmode='group', height=4
fig.update_layout(title_text='Coronavirus Cases in India on daily basis',plot_bgc
fig.show()
```



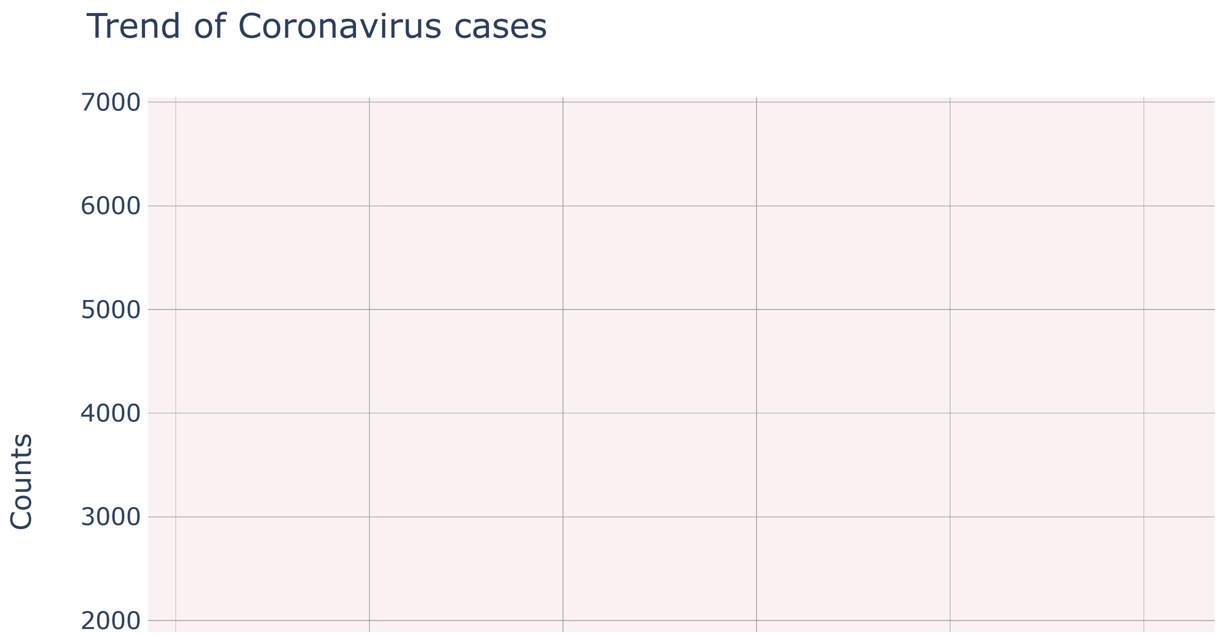
In [44]:

```
fig = px.bar(df_daywise_India, x="Date", y="Confirmed", color='Confirmed', orient
             title='Confirmed Cases in India', color_discrete_sequence = px.color

'''Colour Scale for plotly
https://plot.ly/python/builtin-colorscales/
'''

fig.update_layout(plot_bgcolor='rgb(230, 230, 230)')
fig.show()
```



In [45]:

```
fig = px.line(x=df_daywise_India['Date'], y=df_daywise_India['New Cases'], labels
fig.update_layout( showlegend=False,title_text="Trend of Coronavirus cases")
fig.update_layout(plot_bgcolor='rgb(250, 242, 242)')
fig.show()
```



Forecasting Using fbprophet

In [46]:

```
from fbprophet import Prophet
```

In [47]:

```
df = df_daywise_India.iloc[:-1,]
df_train = df.loc[df['Date']<= "2020-05-23",:]
df_test = df.loc[df['Date'] > "2020-05-23",:]
```

In [48]:

```
confirmed_train = df_train[['Date','Confirmed']]
confirmed_test = df_test[['Date','Confirmed']]

deaths_train = df_train[['Date','Deaths']]
deaths_test = df_test[['Date','Deaths']]

recovered_train = df_train[['Date','Cured']]
recovered_test = df_test[['Date','Cured']]
```

In [49]:

```
confirmed_train.columns = ['ds','y']
confirmed_train.tail()
```

Out[49]:

	ds	y
109	2020-05-19	100325.0
110	2020-05-20	105654.0
111	2020-05-21	110956.0
112	2020-05-22	116827.0
113	2020-05-23	123202.0

In [50]:

```
m = Prophet()
m.fit(confirmed_train)
future = m.make_future_dataframe(periods=5,freq = "D")
future.tail(5)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonalit
y=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=
True to override this.
```

Out[50]:

	ds
114	2020-05-24
115	2020-05-25
116	2020-05-26
117	2020-05-27
118	2020-05-28

In [51]:

```
forecast = m.predict(future)
forecast
```

Out[51]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_te
0	2020-01-31	-97.838797	-2063.265139	2060.041995	-97.838797	-97.838797	140.54
1	2020-02-01	-93.950037	-1854.705145	1956.186312	-93.950037	-93.950037	201.23
2	2020-02-02	-90.061276	-2516.290095	1499.427936	-90.061276	-90.061276	-264.95
3	2020-02-03	-86.172515	-2226.815130	1706.090993	-86.172515	-86.172515	-170.78
4	2020-02-04	-82.283755	-2209.011716	1954.771607	-82.283755	-82.283755	-59.30
...	...	...	...	...	...	...	
114	2020-05-24	118334.300220	116155.386606	120096.421449	118334.300220	118334.300220	-264.95
115	2020-05-25	122043.020651	119834.184008	123874.099628	121974.588189	122138.383157	-170.78
116	2020-05-26	125751.741081	123487.989786	127778.470630	125531.973980	125964.166385	-59.30
117	2020-05-27	129460.461512	127471.181392	131526.620504	129058.746455	129818.022559	39.86
118	2020-05-28	133169.181943	131109.804190	135341.344783	132587.605752	133749.860389	113.39

119 rows × 16 columns

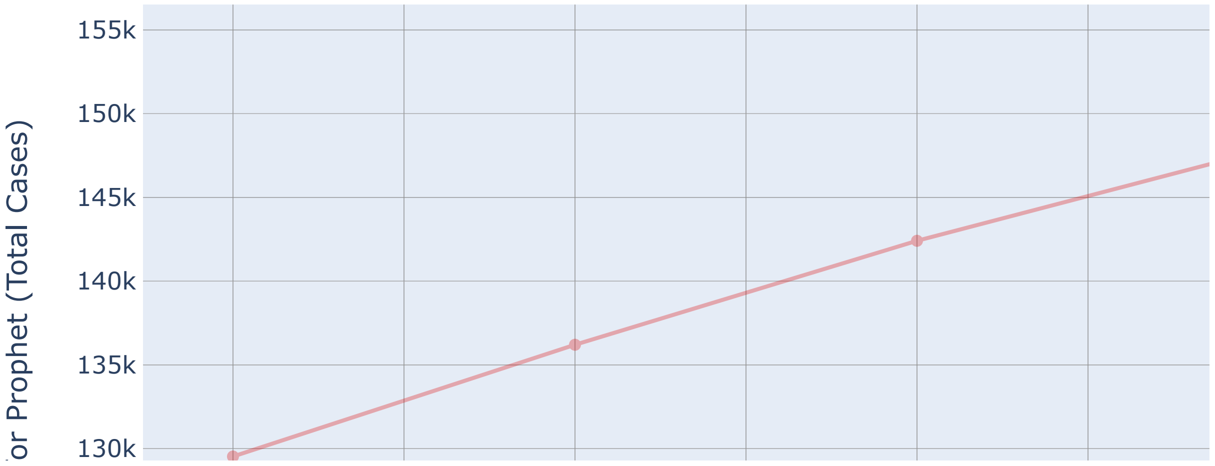
In [52]:

```
result_df = forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail(5)
result_df['Actual'] = confirmed_test['Confirmed']
result_df
```

Out[52]:

	ds	yhat	yhat_lower	yhat_upper	Actual
114	2020-05-24	118069.348165	116155.386606	120096.421449	129530.0
115	2020-05-25	121872.236271	119834.184008	123874.099628	136203.0
116	2020-05-26	125692.436800	123487.989786	127778.470630	142410.0
117	2020-05-27	129500.325211	127471.181392	131526.620504	147754.0
118	2020-05-28	133282.572397	131109.804190	135341.344783	154001.0

In [53]:
```
trace0 = go.Scatter(
        x = result_df['ds'],
        y = result_df['Actual'],
        mode = 'lines+markers',
        name='Actuals',
        line = dict(color = '#dd0000', shape = 'linear'),
        opacity = 0.3,
        connectgaps=True
)
trace1 = go.Scatter(
        x = result_df['ds'],
        y = result_df['yhat'],
        name='Predicted',
        mode = 'lines+markers',
        marker = dict(
            size = 10,
            color = '#44dd00'),
        opacity = 0.3
)
data = [trace0, trace1]
layout = go.Layout(
    yaxis=dict(
        title="Results for Prophet (Total Cases)"
    )
)
fig = go.Figure(data=data, layout=layout)
fig.show()
```



In [54]:

```
recovered_train.columns = ['ds','y']
recovered_train.tail()
```

Out[54]:

	ds	y
109	2020-05-19	39174.0
110	2020-05-20	42298.0
111	2020-05-21	45300.0
112	2020-05-22	48534.0
113	2020-05-23	51784.0

In [55]:

```
m = Prophet()
m.fit(recovered_train)
future = m.make_future_dataframe(periods=5,freq = "D")
future.tail(5)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonalit
y=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=
True to override this.
```

Out[55]:

	ds
114	2020-05-24
115	2020-05-25
116	2020-05-26
117	2020-05-27
118	2020-05-28

In [56]:

```
forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail(5)
result_df = forecast.tail(5)
result_df['Actual'] = recovered_test['Cured']
result_df
```

Out[56]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms
114	2020-05-24	45515.407078	43452.200005	47314.186745	45515.407078	45515.407078	-161.358909
115	2020-05-25	47132.639318	45098.524667	49112.153561	47104.754825	47168.913193	-137.227500
116	2020-05-26	48749.871559	46648.179302	50497.094906	48670.189832	48852.027903	-97.221104
117	2020-05-27	50367.103799	48363.114846	52379.714859	50209.351017	50546.133872	13.160212
118	2020-05-28	51984.336039	50116.075477	54052.251528	51738.441544	52254.787610	77.499145

In [57]:

```
trace0 = go.Scatter(
        x = result_df['ds'],
        y = result_df['Actual'],
        mode = 'lines+markers',
        name='Actuals',
        line = dict(color = '#dd0000', shape = 'linear'),
        opacity = 0.3,
        connectgaps=True
)
trace1 = go.Scatter(
        x = result_df['ds'],
        y = result_df['yhat'],
        name='Predicted',
        mode = 'lines+markers',
        marker = dict(
            size = 10,
            color = '#44dd00'),
        opacity = 0.3
)
data = [trace0, trace1]
layout = go.Layout(
    yaxis=dict(
        title="Results for Prophet (Recovered)"
    )
)
fig = go.Figure(data=data, layout=layout)
fig.show()
```

or Prophet (Recove
60k
55k

In [58]:

```
deaths_train.columns = ['ds','y']
deaths_train.tail()
```

Out[58]:

	ds	y
109	2020-05-19	3163.0
110	2020-05-20	3303.0
111	2020-05-21	3435.0
112	2020-05-22	3583.0
113	2020-05-23	3720.0

In [59]:

```
m = Prophet(seasonality_mode= 'multiplicative')
m.fit(deaths_train)
future = m.make_future_dataframe(periods=5,freq = "D")
future.tail(5)
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seasonalit
y=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_seasonality=
True to override this.
```

Out[59]:

	ds
114	2020-05-24
115	2020-05-25
116	2020-05-26
117	2020-05-27
118	2020-05-28

In [60]:

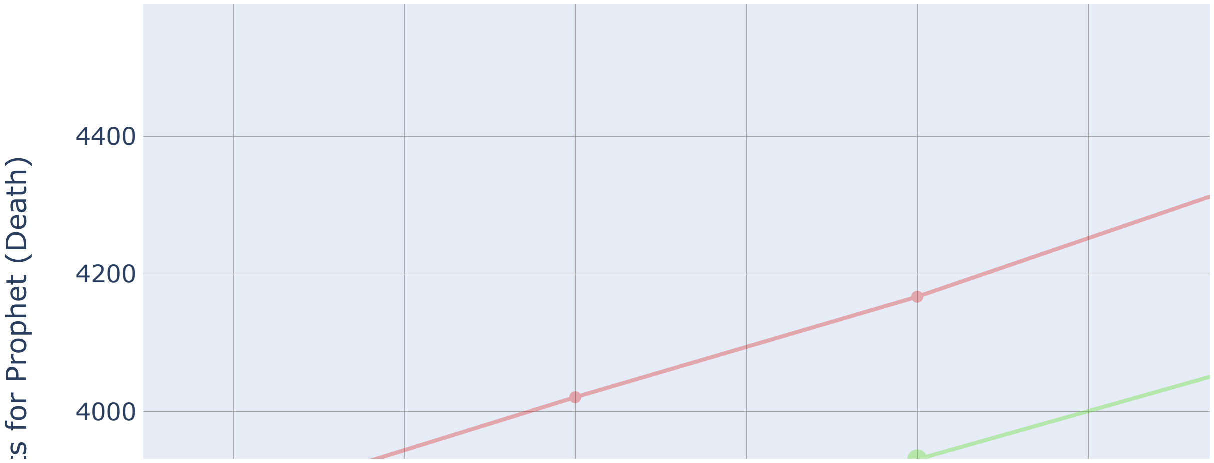
```
forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail(5)
result_df = forecast.tail(5)
result_df['Actual'] = deaths_test['Deaths']
result_df
```

Out[60]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	multiplicative_terms
114	2020-05-24	3714.156291	3631.804529	3689.724787	3714.156291	3714.275148	-0.014758
115	2020-05-25	3827.401179	3759.792711	3817.221796	3824.909336	3829.625698	-0.009904
116	2020-05-26	3940.646067	3901.011863	3961.501562	3933.959268	3946.885779	-0.002530
117	2020-05-27	4053.890955	4035.998157	4102.258673	4040.444815	4064.937653	0.004178
118	2020-05-28	4167.135842	4164.734149	4233.868782	4148.149507	4183.998175	0.007728

In [61]:

```
trace0 = go.Scatter(
        x = result_df['ds'],
        y = result_df['Actual'],
        mode = 'lines+markers',
        name='Actuals',
        line = dict(color = '#dd0000', shape = 'linear'),
        opacity = 0.3,
        connectgaps=True
)
trace1 = go.Scatter(
        x = result_df['ds'],
        y = result_df['yhat'],
        name='Predicted',
        mode = 'lines+markers',
        marker = dict(
            size = 10,
            color = '#44dd00'),
        opacity = 0.3
)
data = [trace0, trace1]
layout = go.Layout(
    yaxis=dict(
        title="Results for Prophet (Death)"
    )
)
fig = go.Figure(data=data, layout=layout)
fig.show()
```



In [62]:

```
from sklearn.linear_model import LinearRegression
confirmed['day'] = confirmed['ds'].dt.day
confirmed['month'] = confirmed['ds'].dt.month
confirmed['year'] = confirmed['ds'].dt.year
# del confirmed['ds']
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-62-ec75aec9acf3> in <module>
      1 from sklearn.linear_model import LinearRegression
----> 2 confirmed['day'] = confirmed['ds'].dt.day
      3 confirmed['month'] = confirmed['ds'].dt.month
      4 confirmed['year'] = confirmed['ds'].dt.year
      5 # del confirmed['ds']

NameError: name 'confirmed' is not defined
```

In [ ]:

```
pwd
```

In [ ]:

```
ls
```

In [ ]:

```

```