In [2]:

```
#Case Study: Analysing the Outbreak of COVID 19 using Machine Learning
#Problem Statement
#We need a strong model that predicts how the virus could spread across d
ifferent countries and regions.
#The goal of this task is to build a model that predicts the spread of th
e virus till 10th of June

#NOTE: The model was built on a test dataset updated till May 25th. But y
ou can access the
#source to these datasets at the ‘John Hopkins University Coronavirus Res
ource Centre’ which gets updated on a daily basis, so you can run this mo
del for the date you prefer.

#Tasks to be performed:
#Analysing the present condition in India
#Exploring the world wide data
#Forecasting the worldwide COVID-19 cases using Prophet for world and Ind
ia

#importinglibraries

import pandas as pd

#visualization libraries
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import folium
from folium import plugins

#plotsize manipulation
plt.rcParams['figure.figsize']=10,12

#disablewarnings
import warnings
warnings.filterwarnings('ignore')
```

In [3]:

```
#Reading the Datasets

India_coord=pd.read_excel('Indian Coordinates.xlsx')
df=pd.read_excel('state_wise1.xlsx')
df_india= df.copy()
df
```

Out[3]:

	Name of State / UT	Confirmed	Recovered	Deaths	Active
0	Maharashtra	52667	15786	1695	35186
1	Tamil Nadu	17082	8731	119	8232
2	Gujarat	14468	6636	888	6944
3	Delhi	14053	6771	276	7006
4	Rajasthan	7376	4072	167	3137
5	Madhya Pradesh	6859	3571	300	2988
6	Uttar Pradesh	6497	3660	169	2668
7	West Bengal	3816	1414	278	2124
8	Andhra Pradesh	2983	1947	57	979
9	Bihar	2737	733	13	1991
10	Unassigned	2970	0	0	2970
11	Karnataka	2182	705	44	1431
12	Punjab	2081	1913	40	128
13	Telangana	1920	1164	56	700
14	Jammu and Kashmir	1668	809	23	836
15	Odisha	1438	649	7	782
16	Haryana	1213	802	16	395
17	Kerala	897	532	6	359
18	Assam	549	63	4	479
19	Jharkhand	405	148	4	253
20	Uttarakhand	349	58	4	284
21	Chhattisgarh	292	67	0	225
22	Chandigarh	266	187	4	75
23	Himachal Pradesh	223	63	4	153
24	Tripura	198	165	0	33
25	Goa	67	19	0	48
26	Ladakh	53	43	0	10
27	Puducherry	49	17	0	32
28	Manipur	36	4	0	32
29	Andaman and Nicobar Islands	33	33	0	0
30	Meghalaya	15	12	1	2
31	Nagaland	3	0	0	3
32	Dadra and Nagar Haveli and Daman and Diu	2	1	0	1
33	Arunachal Pradesh	2	1	0	1
34	Mizoram	1	1	0	0
35	Sikkim	1	0	0	1

	Name of State / UT	Confirmed	Recovered	Deaths	Active
36	Lakshadweep	0	0	0	0

In [4]:

```
#Analysing COVID19 Cases in India

df=pd.read_excel('state_wise1.xlsx')
df_india= df.copy()
df['Total cases']=df['Confirmed']
total_cases= df['Total cases'].sum()
print('Total number of covid confirmed case till 25th may in India:',tota
l_cases)
```

```
Total number of covid confirmed case till 25th may in India: 145451
```

In [5]:

```
#Number of Active COVID-19 cases in affected State/Union Territories

df=pd.read_excel('state_wise1.xlsx')
df_india= df.copy()
df
df.style.background_gradient(cmap='Reds')
```

Out[5]:

	Name of State / UT	Confirmed	Recovered	Deaths	Active
0	Maharashtra	52667	15786	1695	35186
1	Tamil Nadu	17082	8731	119	8232
2	Gujarat	14468	6636	888	6944
3	Delhi	14053	6771	276	7006
4	Rajasthan	7376	4072	167	3137
5	Madhya Pradesh	6859	3571	300	2988
6	Uttar Pradesh	6497	3660	169	2668
7	West Bengal	3816	1414	278	2124
8	Andhra Pradesh	2983	1947	57	979
9	Bihar	2737	733	13	1991
10	Unassigned	2970	0	0	2970
11	Karnataka	2182	705	44	1431
12	Punjab	2081	1913	40	128
13	Telangana	1920	1164	56	700
14	Jammu and Kashmir	1668	809	23	836
15	Odisha	1438	649	7	782
16	Haryana	1213	802	16	395
17	Kerala	897	532	6	359
18	Assam	549	63	4	479
19	Jharkhand	405	148	4	253
20	Uttarakhand	349	58	4	284
21	Chhattisgarh	292	67	0	225
22	Chandigarh	266	187	4	75
23	Himachal Pradesh	223	63	4	153
24	Tripura	198	165	0	33
25	Goa	67	19	0	48
26	Ladakh	53	43	0	10
27	Puducherry	49	17	0	32
28	Manipur	36	4	0	32
29	Andaman and Nicobar Islands	33	33	0	0
30	Meghalaya	15	12	1	2
31	Nagaland	3	0	0	3
32	Dadra and Nagar Haveli and Daman and Diu	2	1	0	1
33	Arunachal Pradesh	2	1	0	1
34	Mizoram	1	1	0	0
35	Sikkim	1	0	0	1

	Name of State / UT	Confirmed	Recovered	Deaths	Active
36	Lakshadweep	0	0	0	0

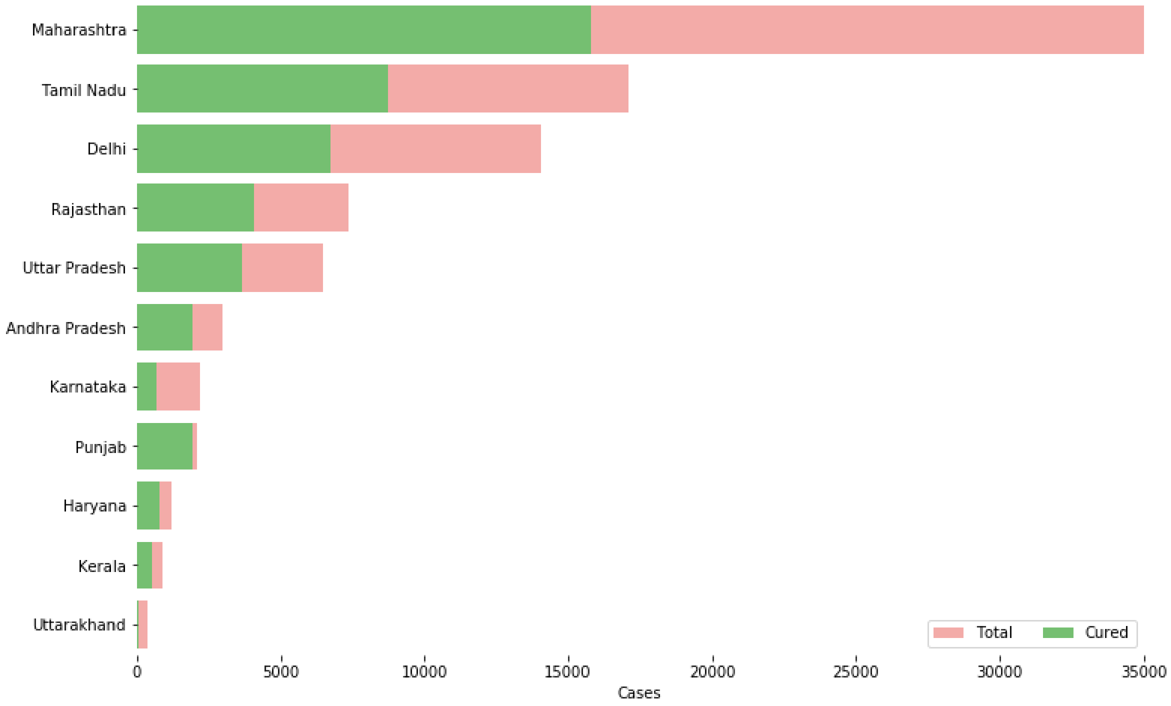
In [6]:

```
#Visualising the spread geographically

df_full = pd.merge(India_coord,df,on='Name of State / UT')
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_full['Confirmed'], df_full['Name of State / UT']):
    folium.CircleMarker([lat, lon], radius=value*0.003, popup = ('<strong>State</strong>: ' + str(name).capitalize() + ''),color='red',fill_color='red',fill_opacity=0.3 ).add_to(map)

map
```

Out[6]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [7]:

```
#Confirmed vs Recovered figures

f, ax = plt.subplots(figsize=(12,8))
data= df_full[['Name of State / UT','Confirmed','Recovered','Deaths']]
data.sort_values('Confirmed',ascending=False,inplace=True)
sns.set_color_codes("pastel")
sns.barplot(x="Confirmed",y="Name of State / UT",data=data,label="Total",
color="r")

sns.set_color_codes("muted")
sns.barplot(x="Recovered",y="Name of State / UT",data=data,label="Cured",
color="g")

ax.legend(ncol=2, loc= "lower right", frameon=True)
ax.set(xlim=(0,35000), ylabel="", xlabel="Cases")
sns.despine(left=True,bottom=True)
```



In [8]:

```
#Exploring Worldwide Data

df = pd.read_csv('covid_19_clean_complete.csv',parse_dates=['Date'])
df.rename(columns={'ObservationDate':'Date', 'Country/Region':'Country'},
inplace=True)
df_confirmed = pd.read_csv("time_series_covid19_confirmed_global.csv")
df_recovered = pd.read_csv("time_series_covid19_recovered_global.csv")
df_deaths = pd.read_csv("time_series_covid19_deaths_global.csv")
df_confirmed.rename(columns={'Country/Region':'Country'}, inplace=True)
df_recovered.rename(columns={'Country/Region':'Country'}, inplace=True)
df_deaths.rename(columns={'Country/Region':'Country'}, inplace=True)
df_deaths.head()
```

Out[8]:

	Province/State	Country	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20
0	NaN	Afghanistan	33.0000	65.0000	0	0	0	0
1	NaN	Albania	41.1533	20.1683	0	0	0	0
2	NaN	Algeria	28.0339	1.6596	0	0	0	0
3	NaN	Andorra	42.5063	1.5218	0	0	0	0
4	NaN	Angola	-11.2027	17.8739	0	0	0	0

5 rows × 130 columns

In [9]:

```
df2 = df.groupby(["Date", "Country", "Province/State"])[['Date', 'Provinc
e/State', 'Country', 'Confirmed', 'Deaths', 'Recovered']].sum().reset_ind
ex()
df2.head()
```

Out[9]:

	Date	Country	Province/State	Confirmed	Deaths	Recovered
0	2020-01-22	Australia	Australian Capital Territory	0	0	0
1	2020-01-22	Australia	New South Wales	0	0	0
2	2020-01-22	Australia	Northern Territory	0	0	0
3	2020-01-22	Australia	Queensland	0	0	0
4	2020-01-22	Australia	South Australia	0	0	0

In [10]:

```
df.query('Country=="India"').groupby("Date")[['Confirmed','Deaths','Recov
ered']].sum().reset_index()
```

Out[10]:

	Date	Confirmed	Deaths	Recovered
0	2020-01-22	0	0	0
1	2020-01-23	0	0	0
2	2020-01-24	0	0	0
3	2020-01-25	0	0	0
4	2020-01-26	0	0	0
...	...	...	...	...
120	2020-05-21	118226	3584	48553
121	2020-05-22	124794	3726	51824
122	2020-05-23	131423	3868	54385
123	2020-05-24	138536	4024	57692
124	2020-05-25	144950	4172	60706

125 rows × 4 columns

In [11]:

```
df.groupby("Date").sum().head()
```

Out[11]:

	Lat	Long	Confirmed	Deaths	Recovered
Date					
2020-01-22	5613.201163	6063.516762	555	17	28
2020-01-23	5613.201163	6063.516762	654	18	30
2020-01-24	5613.201163	6063.516762	941	26	35
2020-01-25	5613.201163	6063.516762	1434	42	38
2020-01-26	5613.201163	6063.516762	2118	56	51

In [12]:

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

In [13]:

```
#Forecasting Total Number of Cases Worldwide
#In this segment, we’re going to generate a week ahead forecast of
#confirmed cases of COVID-19 using Prophet, with specific prediction
#intervals by creating a base model both with and without tweaking of
#seasonality-related parameters and additional regressors.

from fbprophet import Prophet
```

In [14]:

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

In [15]:

```
confirmed.columns=['ds','y']
confirmed['ds']=pd.to_datetime(confirmed['ds'])
```

In [16]:

```
confirmed.tail()
```

Out[16]:

	ds	y
120	2020-05-21	5102418
121	2020-05-22	5210811
122	2020-05-23	5310356
123	2020-05-24	5407607
124	2020-05-25	5495055

In [17]:

```
#Forecasting Confirmed COVID-19 Cases Worldwide with Prophet (Base model)

m = Prophet(interval_width=0.95)
m.fit(confirmed)
future = m.make_future_dataframe(periods=16)
future.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[17]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

In [18]:

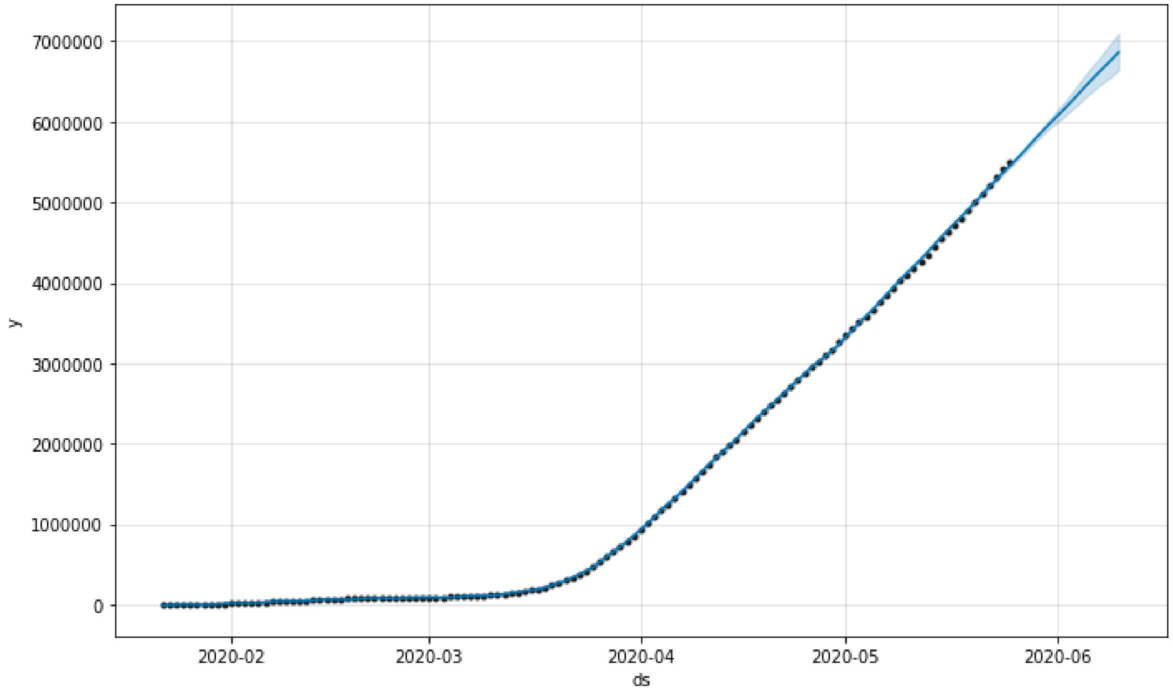
```
#predicting the future with date, and upper and lower limit of y value

forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[18]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	6.516870e+06	6.366168e+06	6.675386e+06
137	2020-06-07	6.603948e+06	6.440255e+06	6.769829e+06
138	2020-06-08	6.688484e+06	6.503585e+06	6.883020e+06
139	2020-06-09	6.772695e+06	6.566569e+06	6.986527e+06
140	2020-06-10	6.862291e+06	6.644579e+06	7.097276e+06

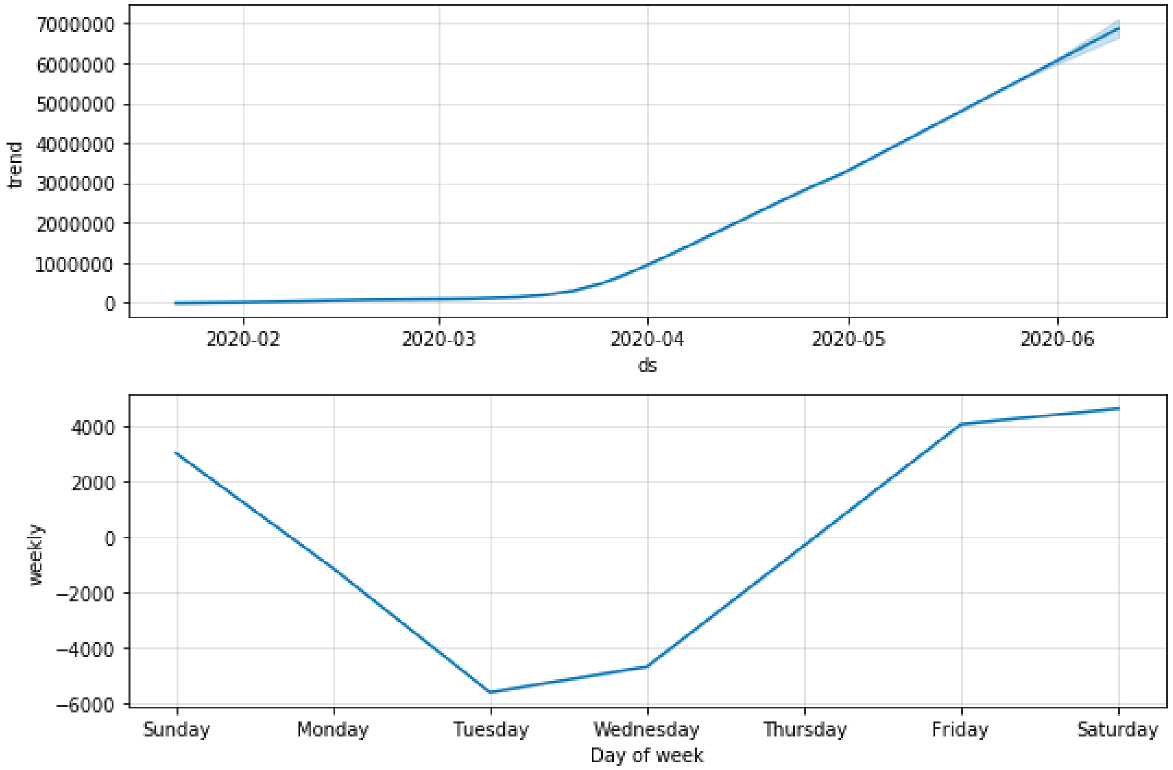
In [19]:

```
confirmed_forecast_plot=m.plot(forecast)
```



In [20]:

```
confirmed_forecast_plot =m.plot_components(forecast)
```



In [21]:

```
#Forecasting Worldwide Recovered using Prophet (Base model)

recovered.columns=['ds','y']
recovered['ds']=pd.to_datetime(confirmed['ds'])
```

In [22]:

```
recovered.tail()
```

Out[22]:

	ds	y
120	2020-05-21	1895640
121	2020-05-22	2001920
122	2020-05-23	2056599
123	2020-05-24	2112135
124	2020-05-25	2174434

In [23]:

```
m = Prophet(interval_width=0.95)
m.fit(recovered)
futurerecovered = m.make_future_dataframe(periods=16)
futurerecovered.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[23]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

In [24]:

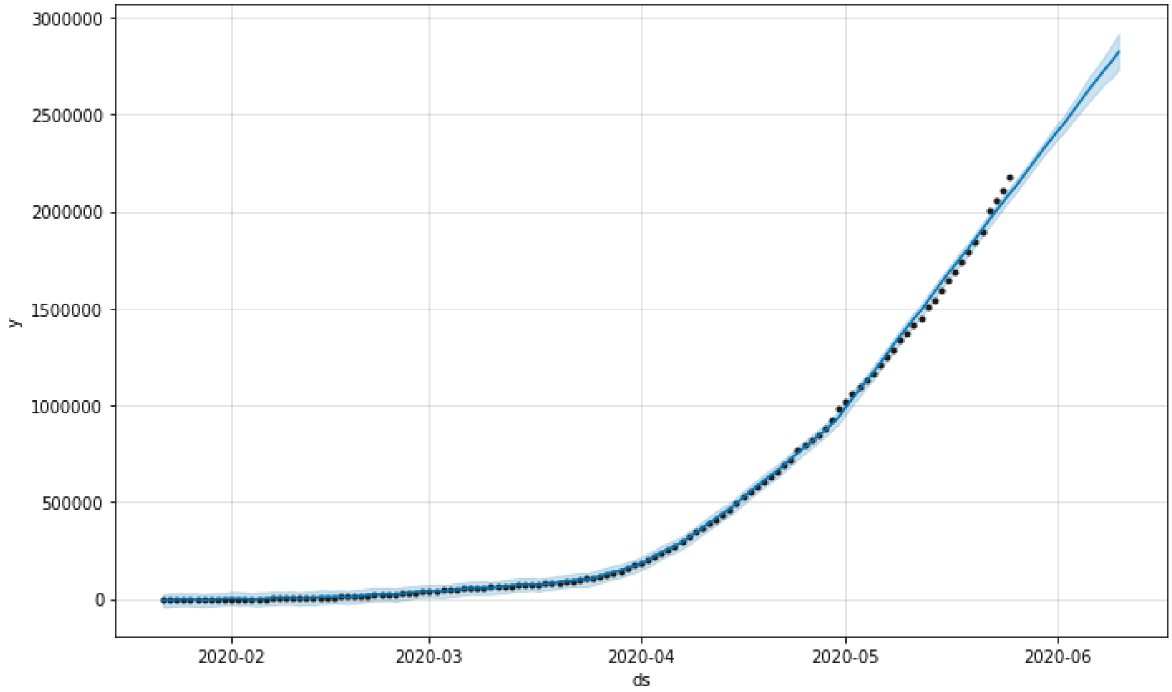
```
##predicting the future with date, and upper and lower limit of y value

forecastrecovered = m.predict(future)
forecastrecovered[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[24]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	2.644054e+06	2.580487e+06	2.710608e+06
137	2020-06-07	2.687980e+06	2.620449e+06	2.753389e+06
138	2020-06-08	2.733485e+06	2.661983e+06	2.806000e+06
139	2020-06-09	2.773868e+06	2.688409e+06	2.859681e+06
140	2020-06-10	2.822697e+06	2.734970e+06	2.918154e+06

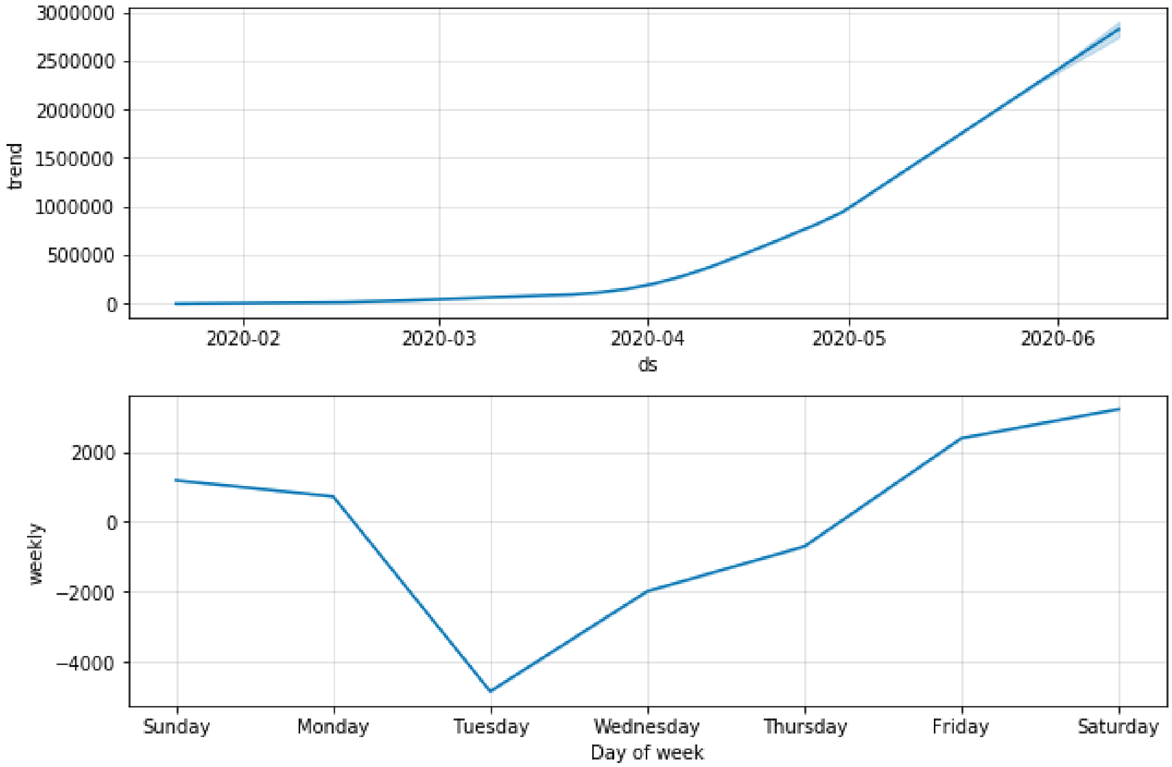
In [25]:

```
confirmed_forcast_plot=m.plot(forecastrecovered)
```



In [26]:

```
confirmed_forecastrecovered_plot =m.plot_components(forecastrecovered)
```



In [27]:

```
#Forecasting Worldwide Deaths using Prophet (Base model)

deaths.columns=['ds','y']
deaths['ds']=pd.to_datetime(confirmed['ds'])
```

In [28]:

```
deaths.tail()
```

Out[28]:

	ds	y
120	2020-05-21	332924
121	2020-05-22	338160
122	2020-05-23	342097
123	2020-05-24	345059
124	2020-05-25	346232

In [29]:

```
m = Prophet(interval_width=0.95)
m.fit(deaths)
futuredeaths = m.make_future_dataframe(periods=16)
futuredeaths.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[29]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

In [30]:

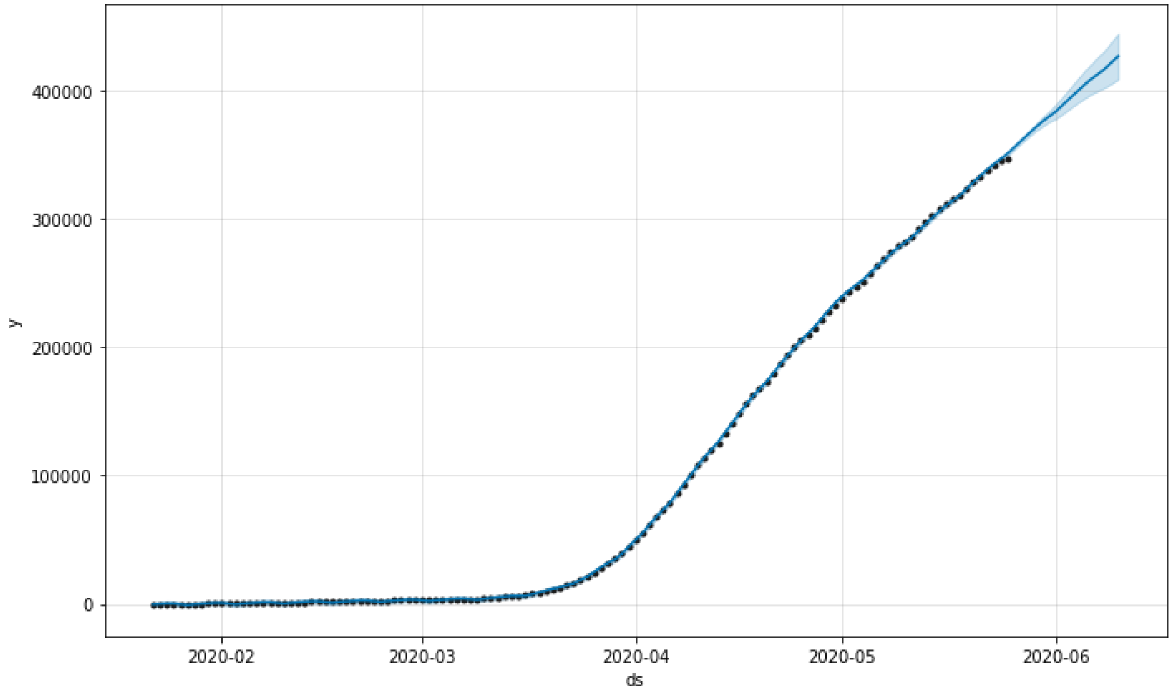
```
#predicting the future with date, and upper and lower limit of y value

forecastdeaths = m.predict(future)
forecastdeaths[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[30]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	408484.289728	396901.214814	420231.225444
137	2020-06-07	412489.007795	399610.796830	426091.961390
138	2020-06-08	416518.692694	402170.255199	430999.667249
139	2020-06-09	421742.826735	405352.614765	437889.754407
140	2020-06-10	426786.548363	408933.883872	444530.505509

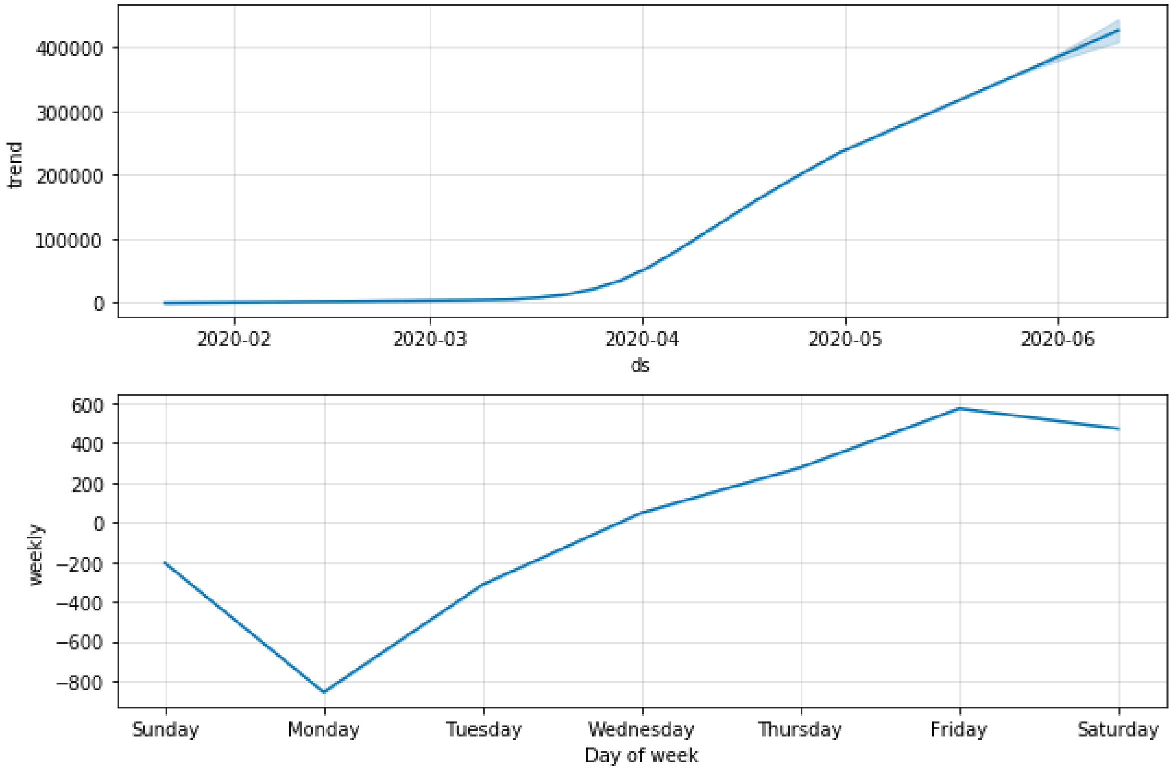
In [31]:

```
death_forcast_plot=m.plot(forecastdeaths)
```



In [32]:

```
confirmed_forecastdeath_plot =m.plot_components(forecastdeaths)
```



In [75]:

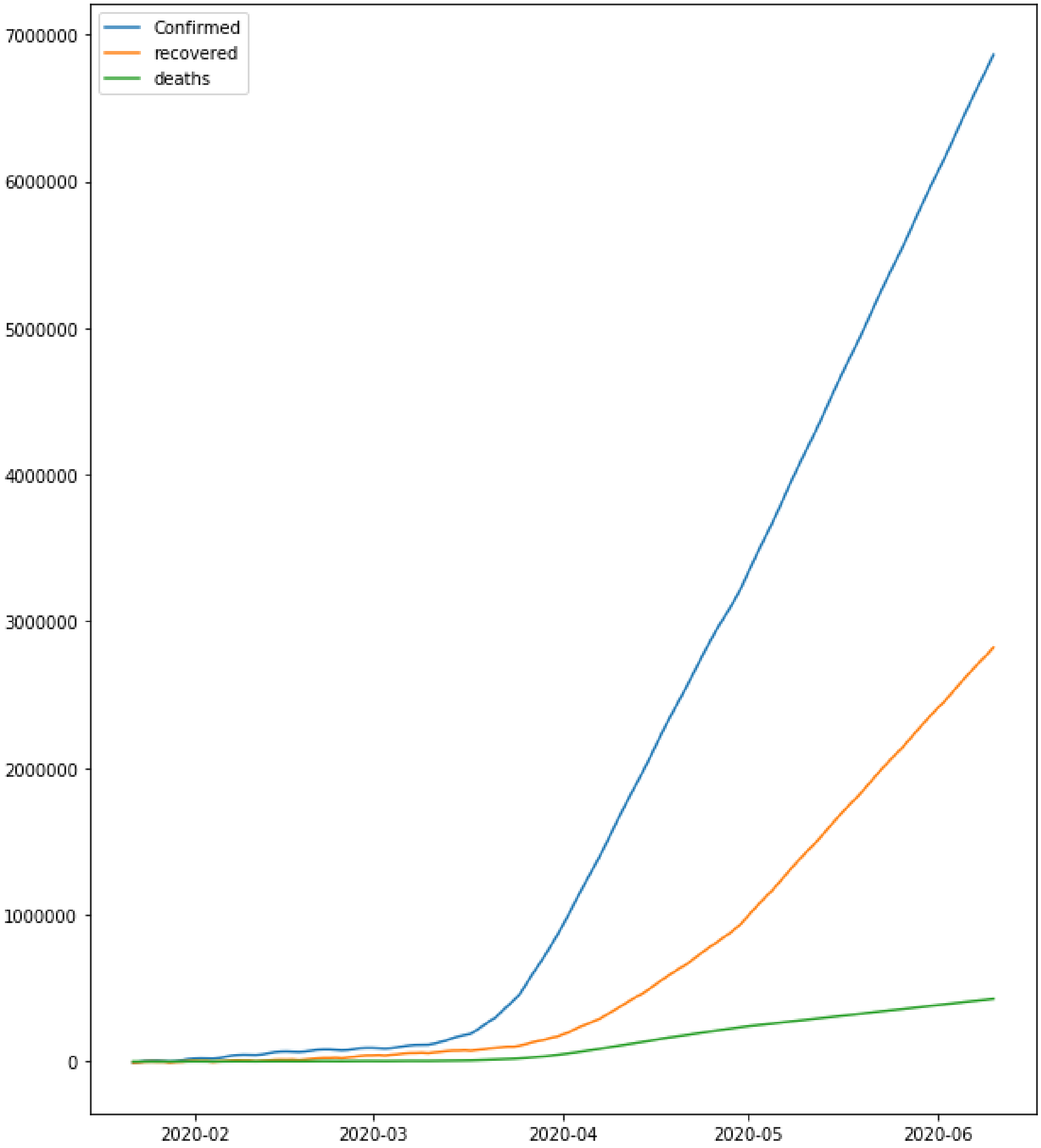
```
dfdateso=forecast['ds']
dfconfirmedo=forecast['yhat']
dfrecoveredo=forecastrecovered['yhat']
dfdeathso=forecastdeaths['yhat']
```

In [76]:

```
import numpy as np
plt.plot(dfdateso, dfconfirmedo,label='Confirmed')
plt.plot(dfdateso, dfrecoveredo,label='recovered')
plt.plot(dfdateso, dfdeathso,label='deaths')
plt.legend()
```

Out[76]:

```
<matplotlib.legend.Legend at 0x2b39b51aac8>
```



In [33]:

```
#Forecasting Indian Corona numbers using Prophet model

df2=df.query('Country=="India"').groupby("Date")[['Confirmed','Deaths','R
ecovered']].sum().reset_index()
```

In [34]:

```
df2.groupby('Date').sum().tail()
```

Out[34]:

Date	Confirmed	Deaths	Recovered
2020-05-21	118226	3584	48553
2020-05-22	124794	3726	51824
2020-05-23	131423	3868	54385
2020-05-24	138536	4024	57692
2020-05-25	144950	4172	60706

In [35]:

```
confirmedindia = df2.groupby('Date').sum()['Confirmed'].reset_index()
deathsindia = df2.groupby('Date').sum()['Deaths'].reset_index()
recoveredindia = df2.groupby('Date').sum()['Recovered'].reset_index()
```

In [36]:

```
from fbprophet import Prophet
```

In [37]:

```
confirmedindia = df2.groupby('Date').sum()['Confirmed'].reset_index()
deathsindia = df2.groupby('Date').sum()['Deaths'].reset_index()
recoveredindia = df2.groupby('Date').sum()['Recovered'].reset_index()
```

In [38]:

```
#Forecasting Confirmed COVID-19 Cases in India with Prophet (Base model)

confirmedindia.columns=['ds','y']
confirmedindia['ds']=pd.to_datetime(confirmedindia['ds'])
```

In [39]:

```
confirmedindia.tail()
```

Out[39]:

	ds	y
120	2020-05-21	118226
121	2020-05-22	124794
122	2020-05-23	131423
123	2020-05-24	138536
124	2020-05-25	144950

In [40]:

```
m = Prophet(interval_width=0.95)
m.fit(confirmedindia)
future = m.make_future_dataframe(periods=16)
future.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[40]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

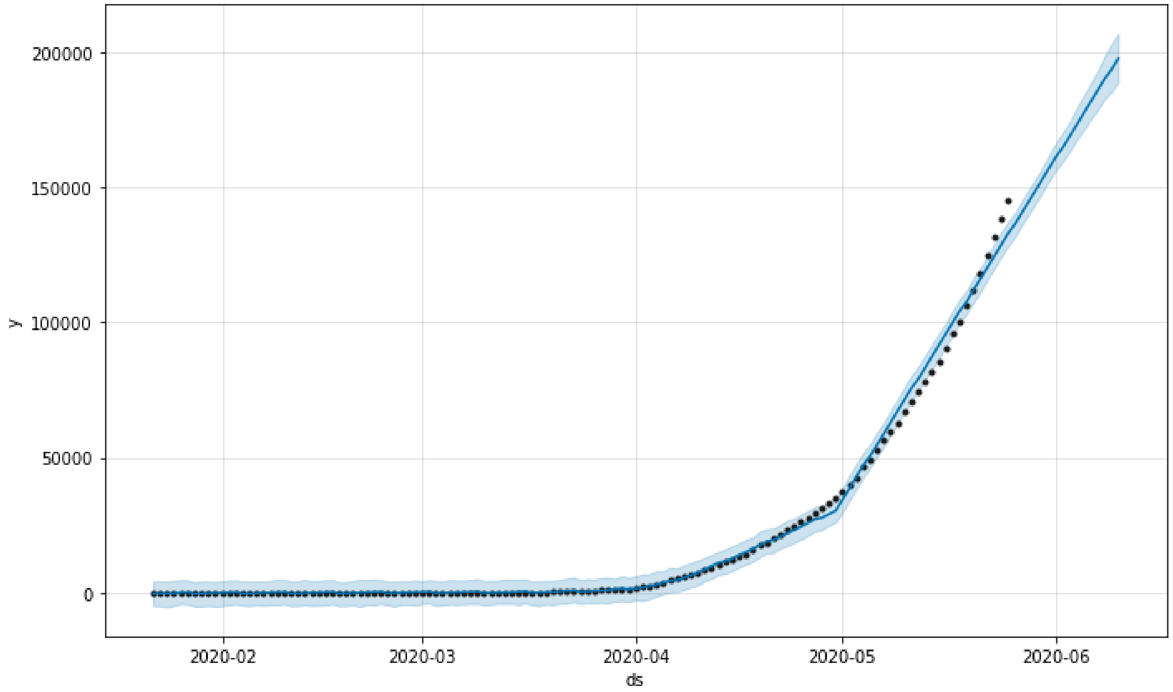
In [45]:

```
##predicting the future with date, and upper and lower limit of y value

forecastconfirmedindia = m.predict(future)
forecastconfirmedindia[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

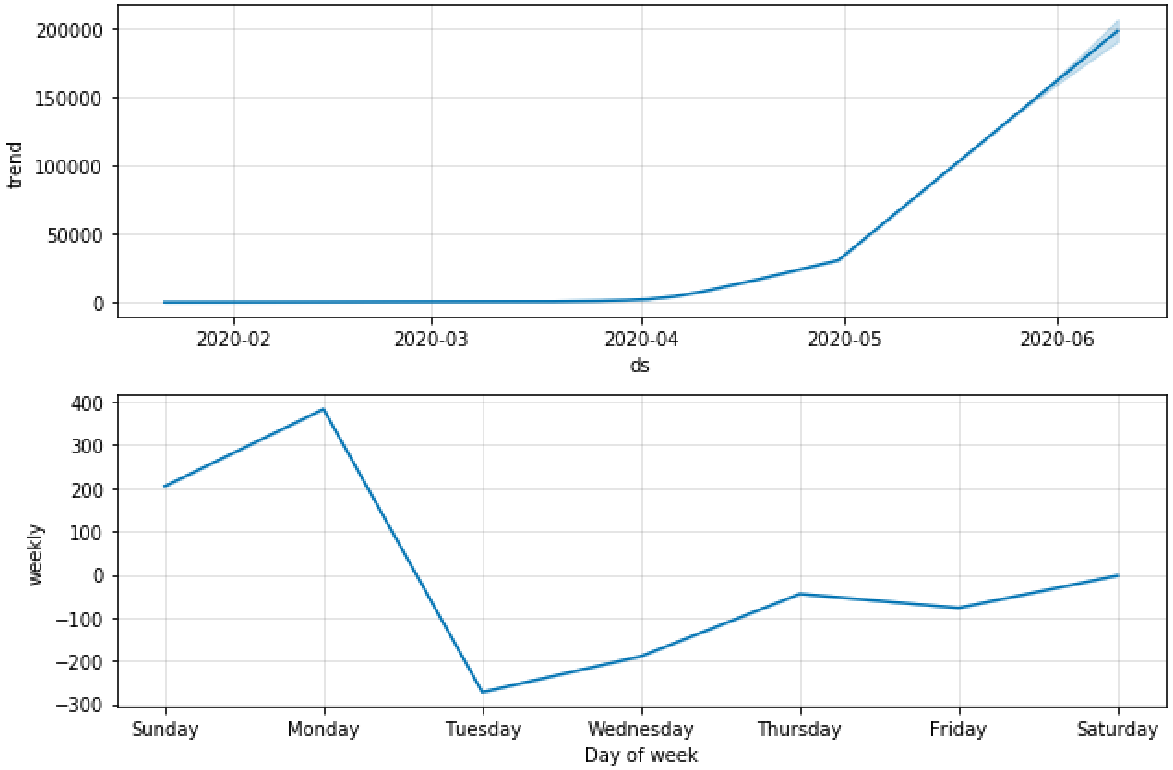
Out[45]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	181503.092433	174882.401801	188973.826133
137	2020-06-07	185794.325700	179214.043222	193816.264400
138	2020-06-08	190057.525876	182374.412962	198442.630029
139	2020-06-09	193486.807298	185732.893413	201860.730637
140	2020-06-10	197653.979586	189123.252785	207843.396201

In [44]:
```
confirmedindia_forcast_plot=m.plot(forecastconfirmedindia)
```



In [47]:
```
confirmed_forecastindia_plot =m.plot_components(forecastconfirmedindia)
```



In [48]:
```
#Forecasting India Recovered Cases with Prophet (Base model)

recoveredindia.columns=['ds','y']
recoveredindia['ds']=pd.to_datetime(recoveredindia['ds'])
```

In [49]:

```
recoveredindia.tail()
```

Out[49]:

	ds	y
120	2020-05-21	48553
121	2020-05-22	51824
122	2020-05-23	54385
123	2020-05-24	57692
124	2020-05-25	60706

In [50]:

```
m = Prophet(interval_width=0.95)
m.fit(recoveredindia)
future = m.make_future_dataframe(periods=16)
future.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[50]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

In [51]:

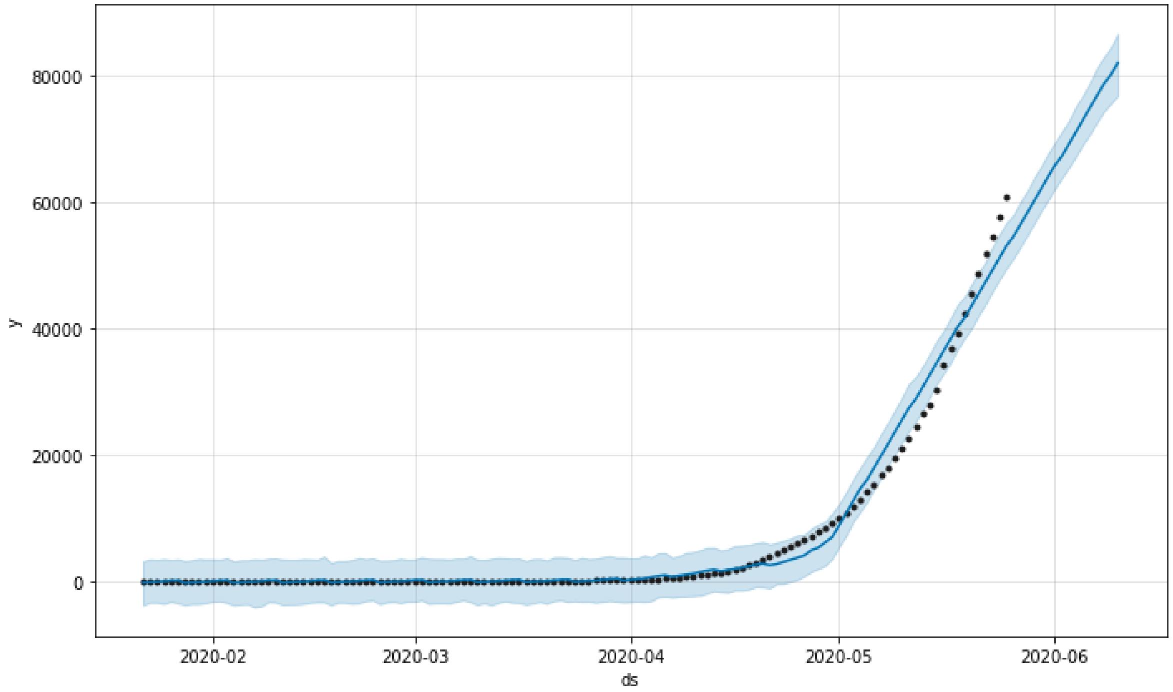
```
##predicting the future with date, and upper and lower limit of y value

forecastrecoveredindia = m.predict(future)
forecastrecoveredindia[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[51]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	74807.689360	70638.360446	78849.823778
137	2020-06-07	76728.181421	72517.915154	81182.348597
138	2020-06-08	78606.138343	74249.420734	82924.274169
139	2020-06-09	80045.108980	75503.768258	84422.369769
140	2020-06-10	81935.291070	76817.319734	86629.674909

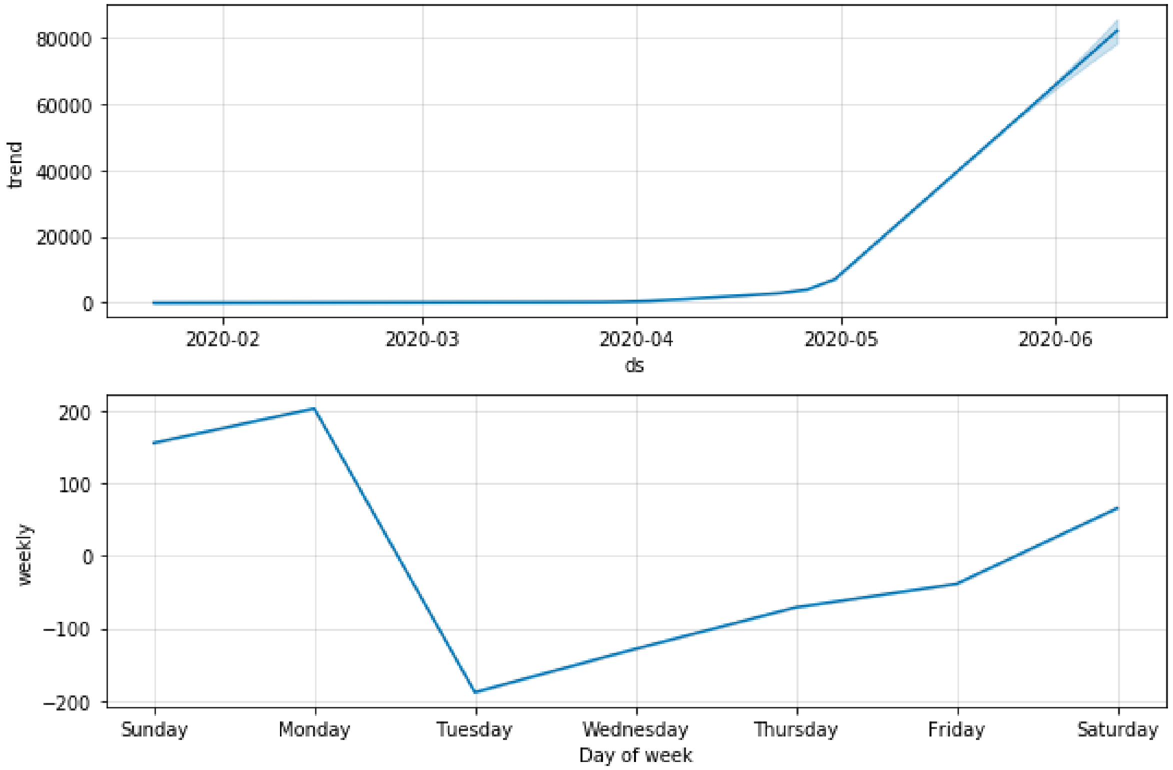
In [52]:

```
recoveredindia_forcast_plot=m.plot(forecastrecoveredindia)
```



In [53]:

```
recovered_forecastindia_plot =m.plot_components(forecastrecoveredindia)
```



In [54]:

```
#Forecasting Deaths in India using Prophet (Base model)

deathsindia.columns=['ds','y']
deathsindia['ds']=pd.to_datetime(deathsindia['ds'])
```

In [55]:

```
deathsindia.tail()
```

Out[55]:

	ds	y
120	2020-05-21	3584
121	2020-05-22	3726
122	2020-05-23	3868
123	2020-05-24	4024
124	2020-05-25	4172

In [56]:

```
m = Prophet(interval_width=0.95)
m.fit(deathsindia)
future = m.make_future_dataframe(periods=16)
future.tail()
```

```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_seas
onality=True to override this.
INFO:fbprophet:Disabling daily seasonality. Run prophet with daily_season
ality=True to override this.
```

Out[56]:

	ds
136	2020-06-06
137	2020-06-07
138	2020-06-08
139	2020-06-09
140	2020-06-10

In [57]:

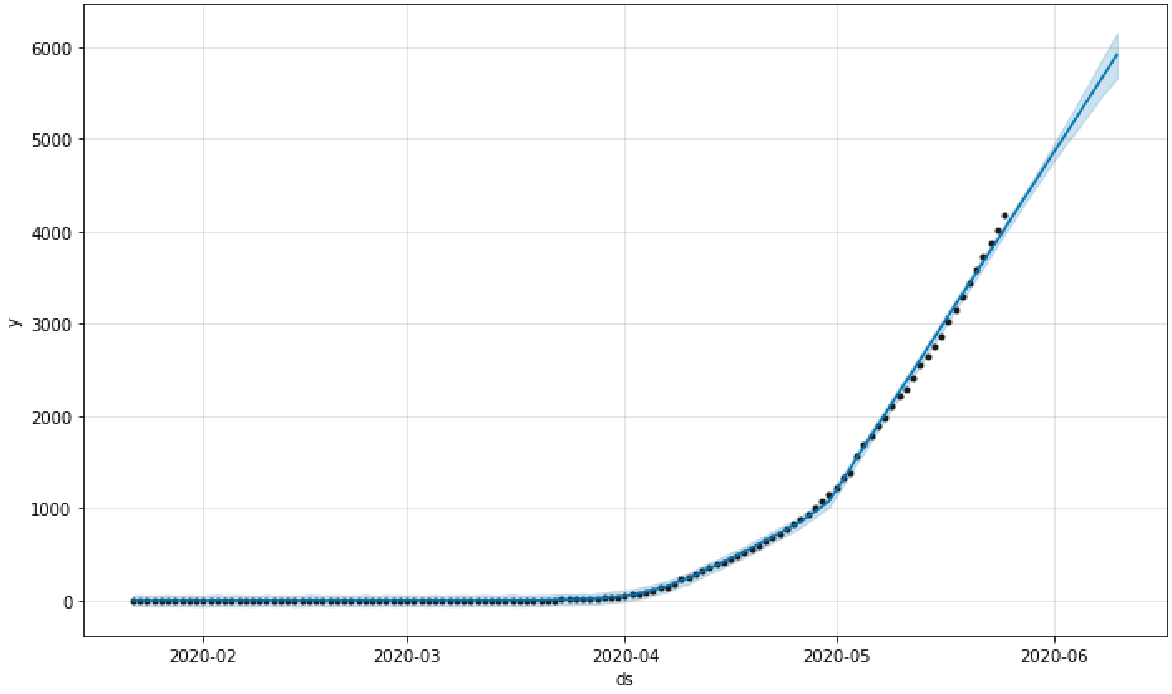
```
#predicting the future with date, and upper and lower limit of y value

forecastdeathsindia = m.predict(future)
forecastdeathsindia[['ds', 'yhat', 'yhat_lower', 'yhat_upper']].tail()
```

Out[57]:

	ds	yhat	yhat_lower	yhat_upper
136	2020-06-06	5442.095366	5272.568325	5603.048278
137	2020-06-07	5562.118480	5375.240867	5747.195074
138	2020-06-08	5684.020650	5487.383256	5884.410626
139	2020-06-09	5796.449597	5568.700307	6012.393305
140	2020-06-10	5915.074229	5663.101980	6149.033747

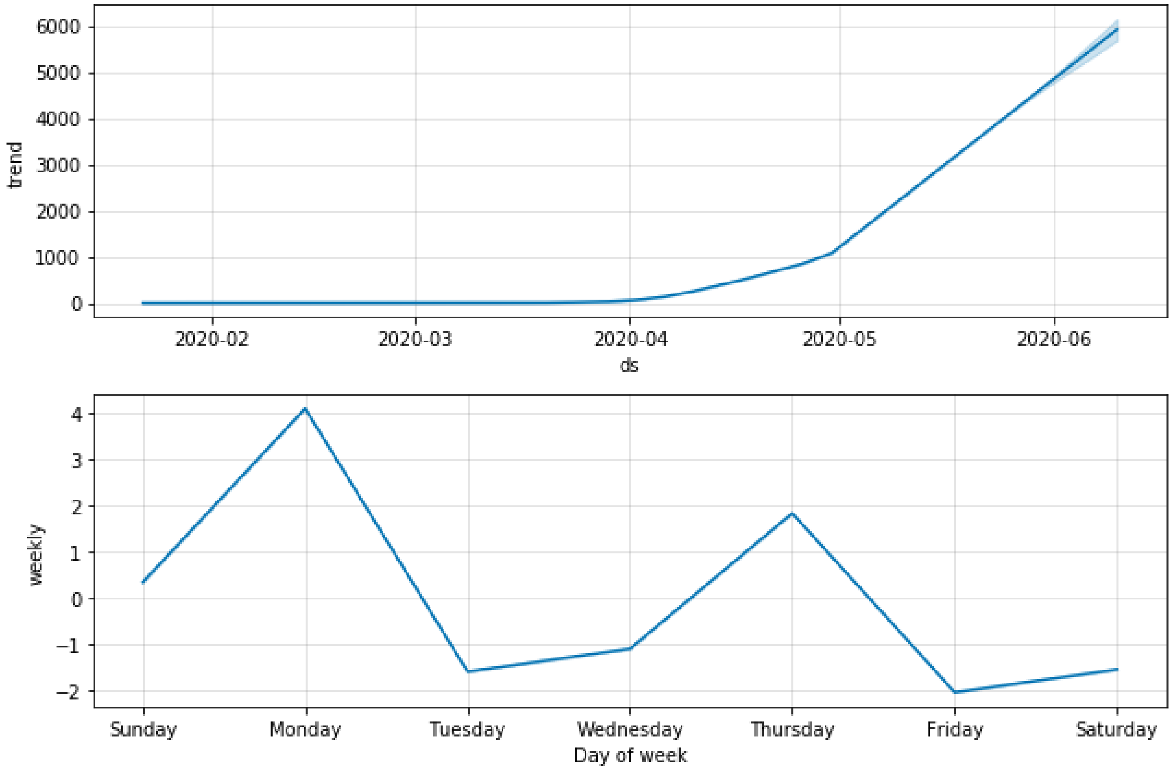
In [58]:

```
deathsindia_forcast_plot=m.plot(forecastdeathsindia)
```



In [59]:

```
deaths_forecastindia_plot =m.plot_components(forecastdeathsindia)
```

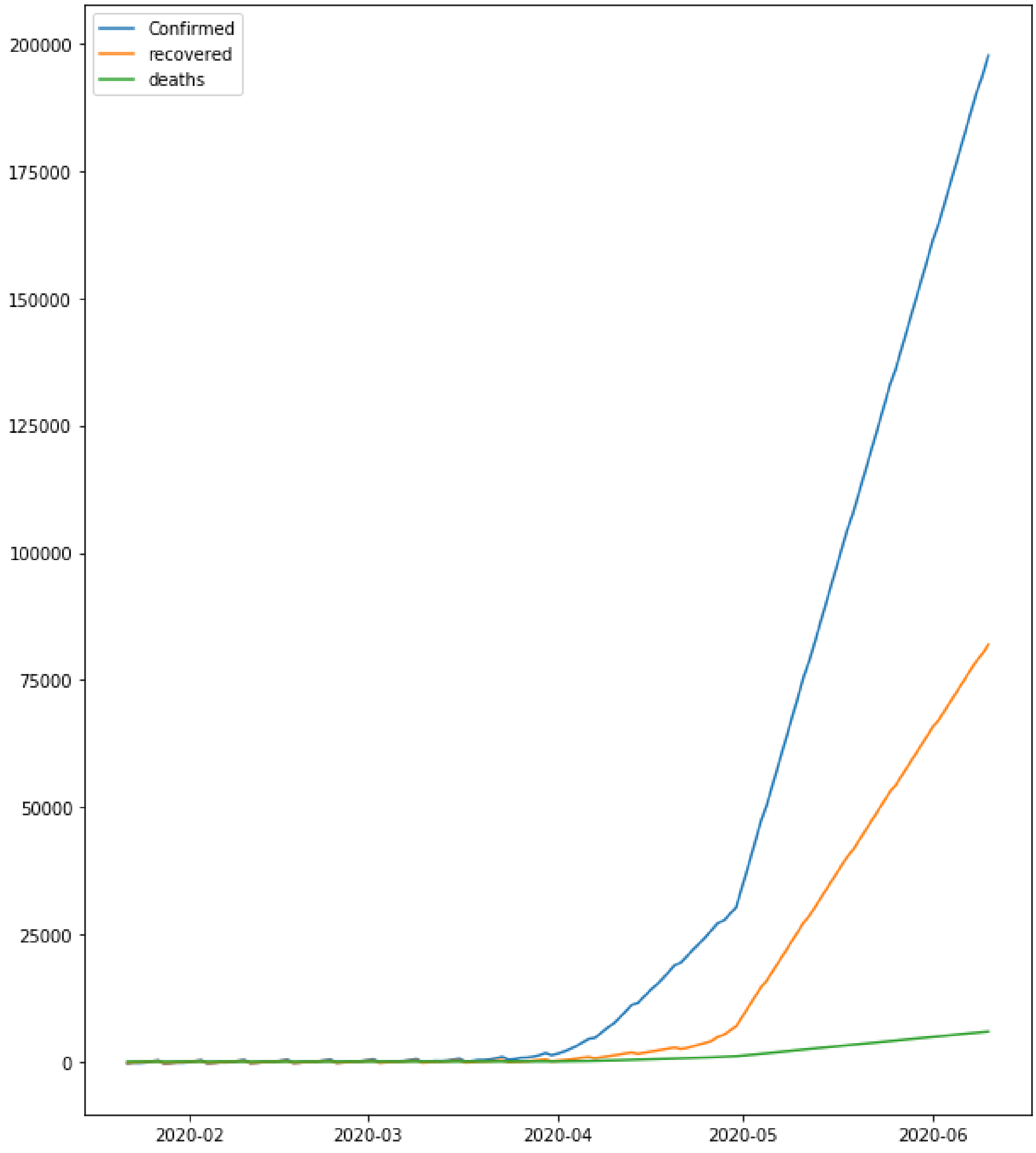


In [68]:

```
dfdates=forecastconfirmedindia['ds']
dfconfirmedindia1=forecastconfirmedindia['yhat']
dfrecoveredindia1=forecastrecoveredindia['yhat']
dfdeathsindia1=forecastdeathsindia['yhat']
```

```
In [73]: import numpy as np
         plt.plot(dfdates, dfconfirmedindia1,label='Confirmed')
         plt.plot(dfdates, dfrecoveredindia1,label='recovered')
         plt.plot(dfdates, dfdeathsindia1,label='deaths')
         plt.legend()
```

```
Out[73]: <matplotlib.legend.Legend at 0x2b39b710708>
```



In [ ]:

```
#Conclusion
#This is a humble request to all our learners.
#Don’t take your cough and cold lightly as you would. If you look at the
 data, the number of cases in India is rising just like in USA. We will r
each  mark of 200,000 cases by 10th June. Don’t let lower awareness and f
ewer test numbers ruin the health of our world.
#But the Best part here is Recovery rate is rising at the better pace as
 compared to confirmed cases it can reach mark of 82000 by 10th of June.
#It shows us that if there are 100 patients getting admitted on 1st day o
n the 14th day around 97 patients will receive discharge ie.2.8% fatality
rate
#This Data shows there will be a time in net few months when government w
ill declare covid-19 as a normal flu in comparison to no. of recoveries a
nd give permissions to business to re-open
#New rules of living will be made where they will give us list of precaut
ions we should take while we interact with people outside our homes.
#Let’s give a hand in fighting this pandemic at least by quarantining our
selves by staying indoors and protecting ourselves and others around us.
#Take precautions and stay indoors.
```