In [6]:

```
import pandas as pd
from matplotlib import pyplot
from pandas import read_csv
from pandas import to_datetime
from pandas import DataFrame
from pandas import date_range
from pandas.tseries.offsets import MonthEnd
from fbprophet import Prophet
```

In [7]:

```
df = read_csv('GHI.csv', header=0)
# summarize shape
print(df.shape)
# show first few rows
print(df.head())
```

```
(691, 2)
       Date  Volumes
0  1/2/2020        6
1  1/3/2020        6
2  1/6/2020        6
3  1/7/2020       15
4  1/8/2020       10
```

In [3]:

```
# plot the time series
#df.plot()
#pyplot.show()
```

In [8]:

```
df.columns = ['ds', 'y']
df['ds']= to_datetime(df['ds'])
print(df.head())
```

```
          ds   y
0 2020-01-02   6
1 2020-01-03   6
2 2020-01-06   6
3 2020-01-07  15
4 2020-01-08  10
```

In [25]:

```
df['cap'] = 38
df['floor'] = 1
m = Prophet(growth = 'logistic',
            changepoint_prior_scale=0.100,
            daily_seasonality=False,
            weekly_seasonality=False,
            seasonality_mode='multiplicative',
           seasonality_prior_scale=0.1,)
m.add_seasonality(name='daily', period=1, prior_scale=0.01, fourier_order=10)
m.add_seasonality(name='weekly', period=7, prior_scale=0.05, fourier_order=10)
m.add_seasonality(name='monthly', period=30.5, prior_scale=0.07, fourier_order=15)
m.fit(df)
```

Out[25]:

```
<fbprophet.forecaster.Prophet at 0x15afb88cf40>
```

In [ ]:

```

```

In [31]:

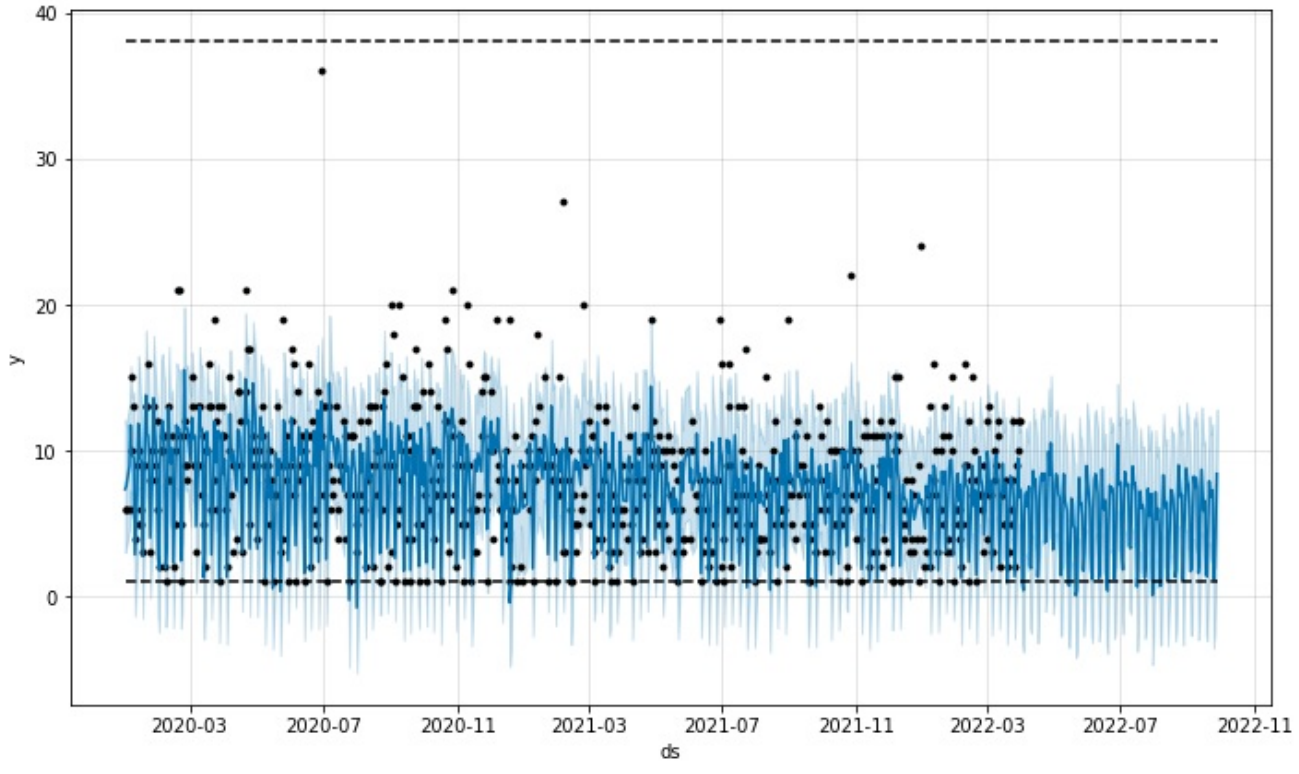
```
## Hyperparameter tuning
#import itertools
#import numpy as np
#from fbprophet.diagnostics import cross_validation
#from fbprophet.diagnostics import performance_metrics

#param_grid = {
#    'changepoint_prior_scale': [0.001, 0.01, 0.1, 0.5],
#    'seasonality_prior_scale': [0.01, 0.1, 1.0, 5.0, 10.0, 15],
#}

# Generate all combinations of parameters
#all_params = [dict(zip(param_grid.keys(), v)) for v in itertools.product(*param_grid.val
ues())]
#maes = []  # Store the RMSEs for each params here

# Use cross validation to evaluate all parameters
#for params in all_params:
#    m2 = Prophet(**params).fit(df)  # Fit model with given params
#    df_cv = cross_validation(m2, initial='365.25 days', period='30 days', horizon = '30
0 days', parallel="processes")
#    df_p = performance_metrics(df_cv, rolling_window=1)
#    maes.append(df_p['mae'].values[0])

# Find the best parameters
#tuning_results = pd.DataFrame(all_params)
#tuning_results['mae'] = maes
#print(tuning_results)
```

In [ ]:

```

```

In [26]:

```
future = m.make_future_dataframe(periods=180)
future['cap'] = 38
future['floor'] = 1


forecast = m.predict(future)
forecast[['ds', 'yhat', 'yhat_lower', 'yhat_upper']]
fig1 = m.plot(forecast)
```



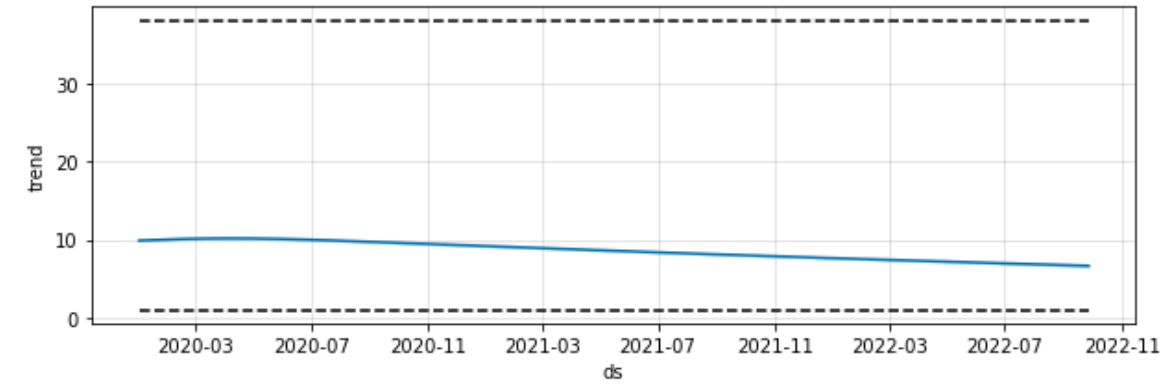
In [ ]:

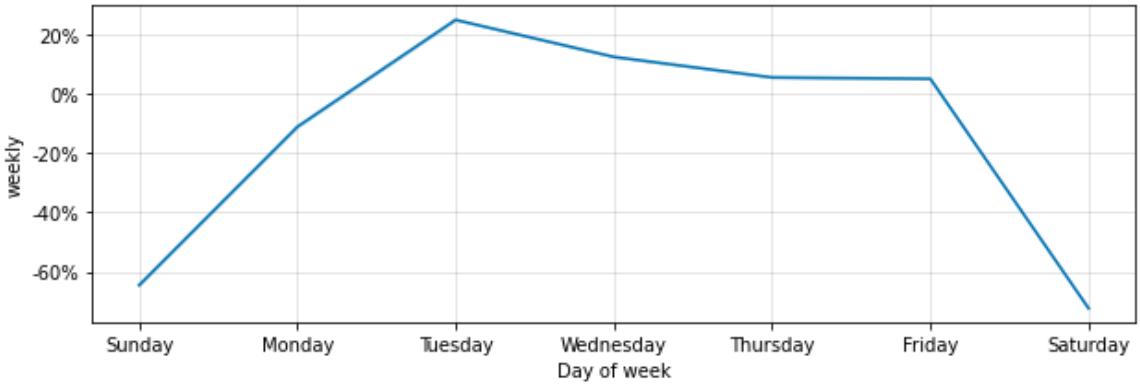
```
```

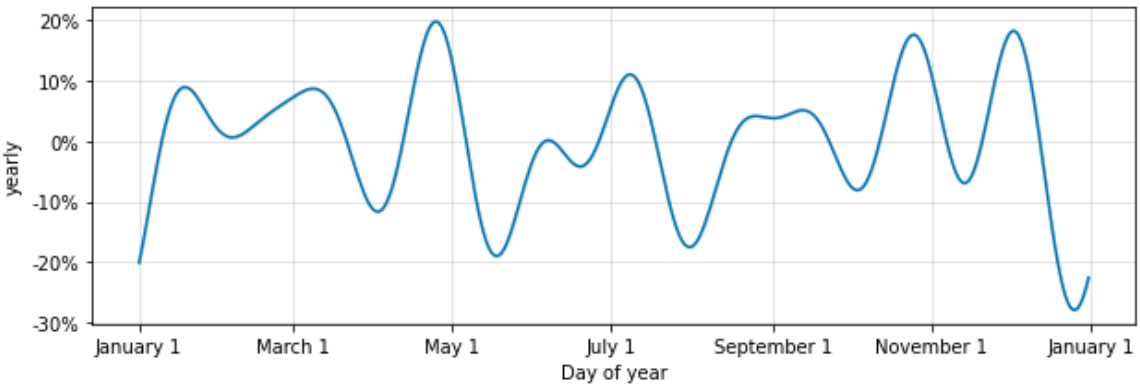
In [27]:

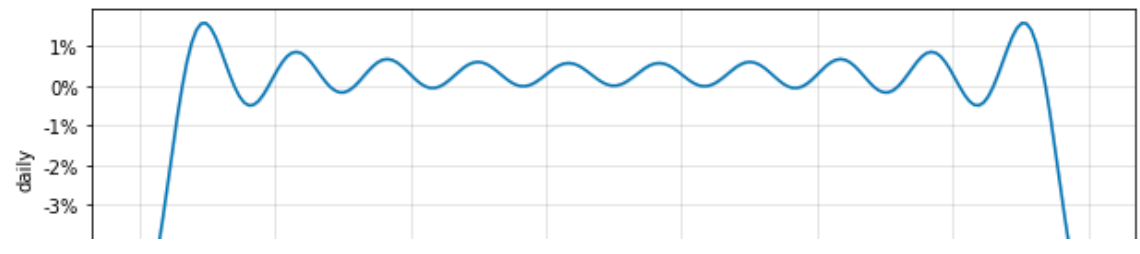
```
fig2 = m.plot_components(forecast)
```

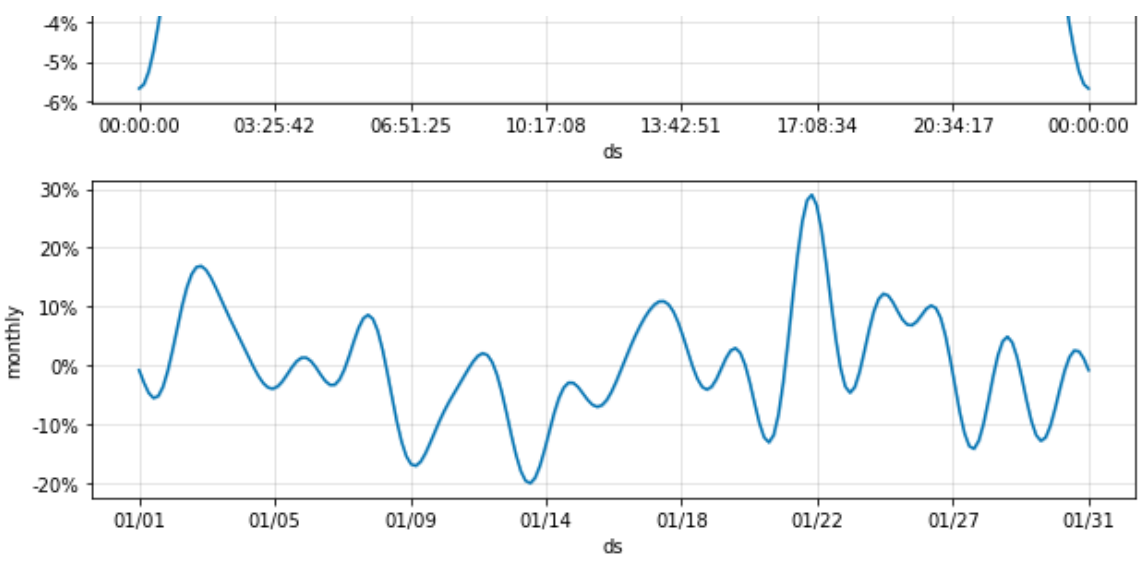
```
C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:422: UserWarning:

FixedFormatter should only be used together with FixedLocator
```











In [28]:

```
from fbprophet.plot import plot_plotly, plot_components_plotly
plot_plotly(m, forecast)
```

In [29]:

```
from fbprophet.diagnostics import cross_validation
df_cv = cross_validation(m, initial='365 days', period='30 days', horizon = '9 days')
```

```
INFO:fbprophet:Making 15 forecasts with cutoffs between 2021-01-26 00:00:00 and 2022-03-2
2 00:00:00
WARNING:fbprophet:Seasonality has period of 365.25 days which is larger than initial wind
ow. Consider increasing initial.
```

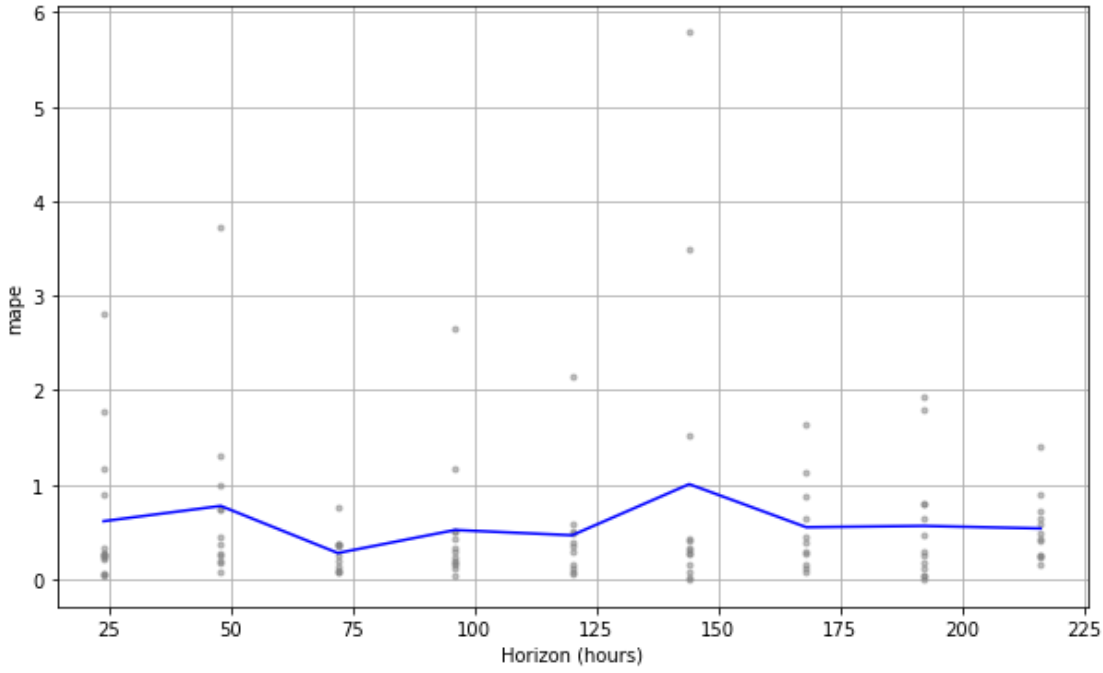
In [30]:

```
from fbprophet.diagnostics import performance_metrics
from fbprophet.plot import plot_cross_validation_metric
df_p = performance_metrics(df_cv)
fig = plot_cross_validation_metric(df_cv, metric='mape')
df_p.head()
```

```
C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:526: FutureWarning:

casting timedelta64[ns] values to int64 with .astype(...) is deprecated and will raise in
a future version. Use .view(...) instead.

C:\Users\User\anaconda3\lib\site-packages\fbprophet\plot.py:527: FutureWarning:

casting timedelta64[ns] values to int64 with .astype(...) is deprecated and will raise in
a future version. Use .view(...) instead.
```

Out[30]:

	horizon	mse	rmse	mae	mape	mdape	coverage
0	**1 days**	**9.132294**	**3.021969**	**2.431790**	**0.616635**	**0.264826**	**0.785714**
1	**2 days**	**11.947654**	**3.456538**	**2.904134**	**0.777588**	**0.411394**	**0.750000**
2	**3 days**	**6.771066**	**2.602127**	**2.224381**	**0.279655**	**0.253880**	**0.909091**
3	**4 days**	**20.755230**	**4.555791**	**3.115989**	**0.523109**	**0.289536**	**0.769231**
4	**5 days**	**13.136910**	**3.624488**	**2.766823**	**0.467488**	**0.346505**	**0.818182**



In [ ]:

```
```

In [20]:

```
##To save predicted volumes to another csv file for validation
import pandas as pd
import os
df2 = pd.read_csv('GHI.csv', header=0)

df3 = pd.DataFrame()
df3['ds'] = forecast['ds'].copy()
df3['Actual'] = df2['Volumes'].copy()
df3['Predicted'] = forecast['yhat'].copy()
df3['Predicted'] = df3['Predicted'].astype(float).round(0)
```

```
df3.head()


df3.set_index('ds', inplace=True)
df3.to_csv('GHI-val.csv')
```

In [ ]:

```

```

In [ ]:

```

```

In [7]:

```
from sklearn.metrics import mean_squared_error, r2_score,mean_absolute_error, mean_absolu
te_percentage_error
df_val = read_csv('GHI-val.csv', header=0)
#df_val
df_val["Actual"].astype(float)
df_val["Predicted"].astype(float)
#df_val.dtypes
r2_score(df_val['Actual'],df_val['Predicted'])
```

Out[7]:

```
0.48828436537313025
```

In [8]:

```
mean_squared_error(df_val['Actual'],df_val['Predicted'])
```

Out[8]:

```
25.988505747126435
```

In [9]:

```
mean_absolute_error(df_val['Actual'],df_val['Predicted'])
```

Out[9]:

```
3.9655172413793105
```

In [10]:

```
mean_absolute_percentage_error(df_val['Actual'],df_val['Predicted'])
```

Out[10]:

```
996486124446920.5
```

In [27]:

```
##To Save the Model
import json
from fbprophet.serialize import model_to_json, model_from_json
with open('serialized_model.json', 'w') as fout:
    json.dump(model_to_json(m), fout)
```

In [23]:

```
import pandas as pd
cutoffs = pd.date_range(start='2020-12-01', end='2021-03-01', freq='M')
print(cutoffs)
```

```
DatetimeIndex(['2020-12-31', '2021-01-31', '2021-02-28'], dtype='datetime64[ns]', freq='M
')
```