In [94]:
```
import gzip
import json
with gzip.open("w4w22vvpgiymhgr475umvp256a.json.gz", "r") as f:
   data = f.read().decode('utf-8')
```

In [95]:
```
import pandas as pd
```

In [96]:
```
s = '[' + ','.join(data.rstrip().split('\n')) + ']'
df = pd.read_json(s)
```

In [97]:
```
df.Item[12334]
```

Out[97]:
```
{'ticker': {'S': 'AAPL'},
 'time': {'N': '1638522900'},
 'low': {'N': '162.8'},
 'volume_weighted_price': {'N': '162.8102'},
 'time_str': {'S': '2021-12-03 04:15'},
 'open': {'N': '162.8'},
 'volume': {'N': '1802'},
 'high': {'N': '162.85'},
 'close': {'N': '162.85'}}
```

In [98]:
```
import datetime
```

In [99]:
```
ticker = []
time = []
low = []
time_str = []
open_ = []
volume = []
high = []
close = []
for dictionary in df.Item:
    ticker.append(dictionary['ticker']['S'])
    time.append(int(dictionary['time']['N']))
    open_.append(float(dictionary['open']['N']))
    close.append(float(dictionary['close']['N']))
    high.append(float(dictionary['high']['N']))
    low.append(float(dictionary['low']['N']))
    volume.append(float(dictionary['volume']['N']))
df_cleaned = pd.DataFrame({"ticker": ticker, "time":time, "open":open_, "clos
```

In [100]:
```
df_cleaned.head()
```

Out[100]:

	ticker	time	open	close	high	low	volume
0	AAPL	1616188260	120.0100	120.0200	120.02	120.01	2047.0
1	AAPL	1627305960	148.2700	148.2699	148.28	148.22	6915.0
2	AAPL	1614764940	126.1500	126.1400	126.15	126.14	642.0
3	AAPL	1607612340	122.4618	122.4450	122.47	122.33	261503.0
4	AAPL	1637312460	158.5000	158.3800	158.56	158.38	5370.0

In [101]:
```
df_cleaned['time'] = pd.to_datetime(df_cleaned['time'], unit='s')
```

In [102]:
```
df_cleaned.head()
```

Out[102]:

	ticker	time	open	close	high	low	volume
0	AAPL	2021-03-19 21:11:00	120.0100	120.0200	120.02	120.01	2047.0
1	AAPL	2021-07-26 13:26:00	148.2700	148.2699	148.28	148.22	6915.0
2	AAPL	2021-03-03 09:49:00	126.1500	126.1400	126.15	126.14	642.0
3	AAPL	2020-12-10 14:59:00	122.4618	122.4450	122.47	122.33	261503.0
4	AAPL	2021-11-19 09:01:00	158.5000	158.3800	158.56	158.38	5370.0

In [103]:
```
df_cleaned.sort_values(by='time', ascending=True, inplace=True)
df_cleaned.reset_index(drop=True, inplace=True)
```

In [104]:
```
df_cleaned = df_cleaned.set_index('time')
```

In [105]:
```
df_cleaned = df_cleaned.groupby(pd.Grouper(freq="H")).mean()
df_cleaned.reset_index(inplace=True)
df_cleaned.dropna(inplace=True)
```

In [106]:
```
from fbprophet import Prophet
from fbprophet.plot import plot_plotly
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import plotly.offline as py
py.init_notebook_mode()
%matplotlib inline
```

In [107]:
```
df_select=df_cleaned[['time', "close", "volume"]]
df_select = df_select.rename(columns = {'time':'ds', 'close':'y','volume':'vo
df_select.head()
```

Out[107]:

	ds	y	volume
0	2020-11-19 09:00:00	117.535200	906.360000
1	2020-11-19 10:00:00	117.636250	587.250000
2	2020-11-19 11:00:00	117.664286	960.750000
3	2020-11-19 12:00:00	117.912222	3101.333333
4	2020-11-19 13:00:00	117.973858	6229.183333

In [108]:
```
len(df_select)
```

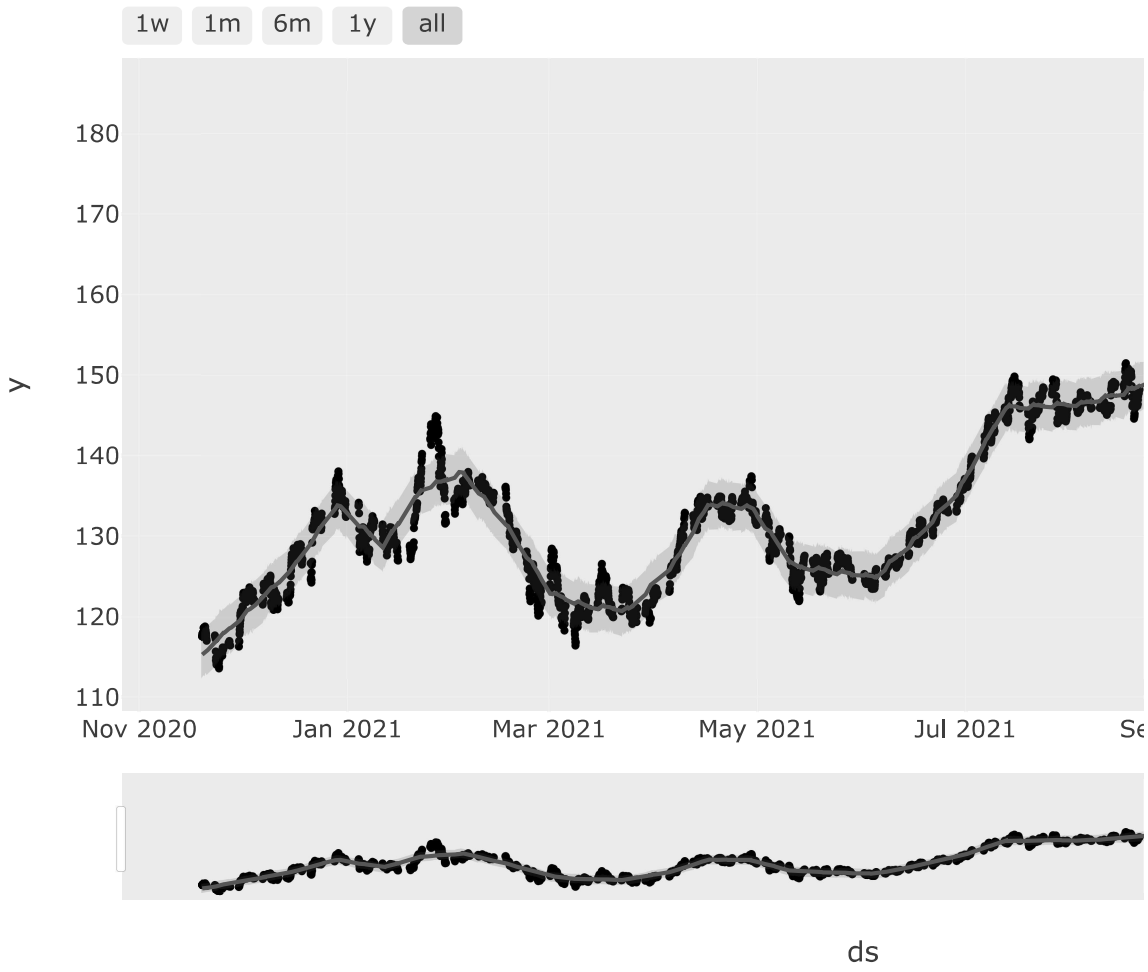
Out[108]: 4177

In [113]:
```
prophet_model = Prophet()
prophet_model.fit(df_select)
#future 1 year prediction
forecast = prophet_model.predict(prophet_model.make_future_dataframe(periods=
```

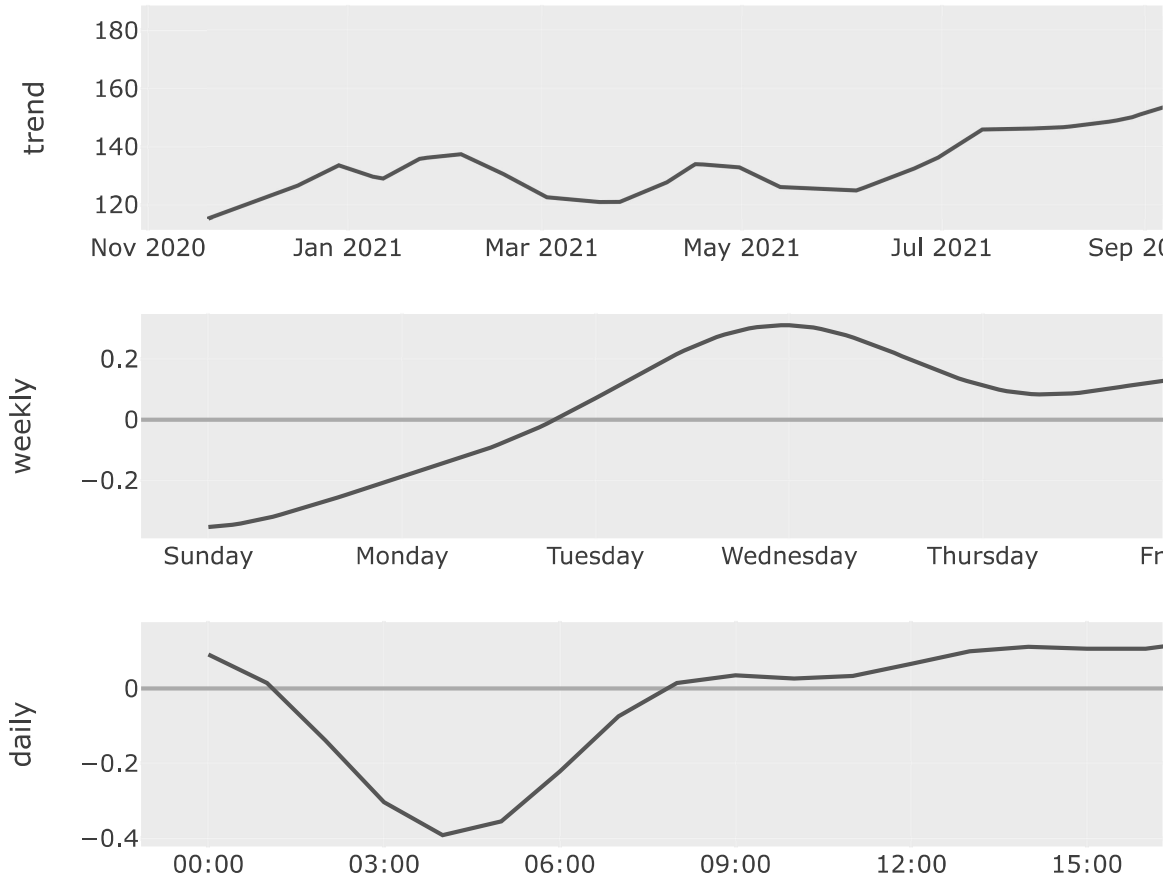
```
INFO:fbprophet:Disabling yearly seasonality. Run prophet with yearly_season
ality=True to override this.
```

In [114]:
```
#过去两千行，跑一个model，output 一个table，未来一个小时like
```

In [115]:
```
from fbprophet.plot import plot_plotly, plot_components_plotly
plot_plotly(prophet_model, forecast)
```



In [116]:
```
plot_components_plotly(prophet_model, forecast)
```



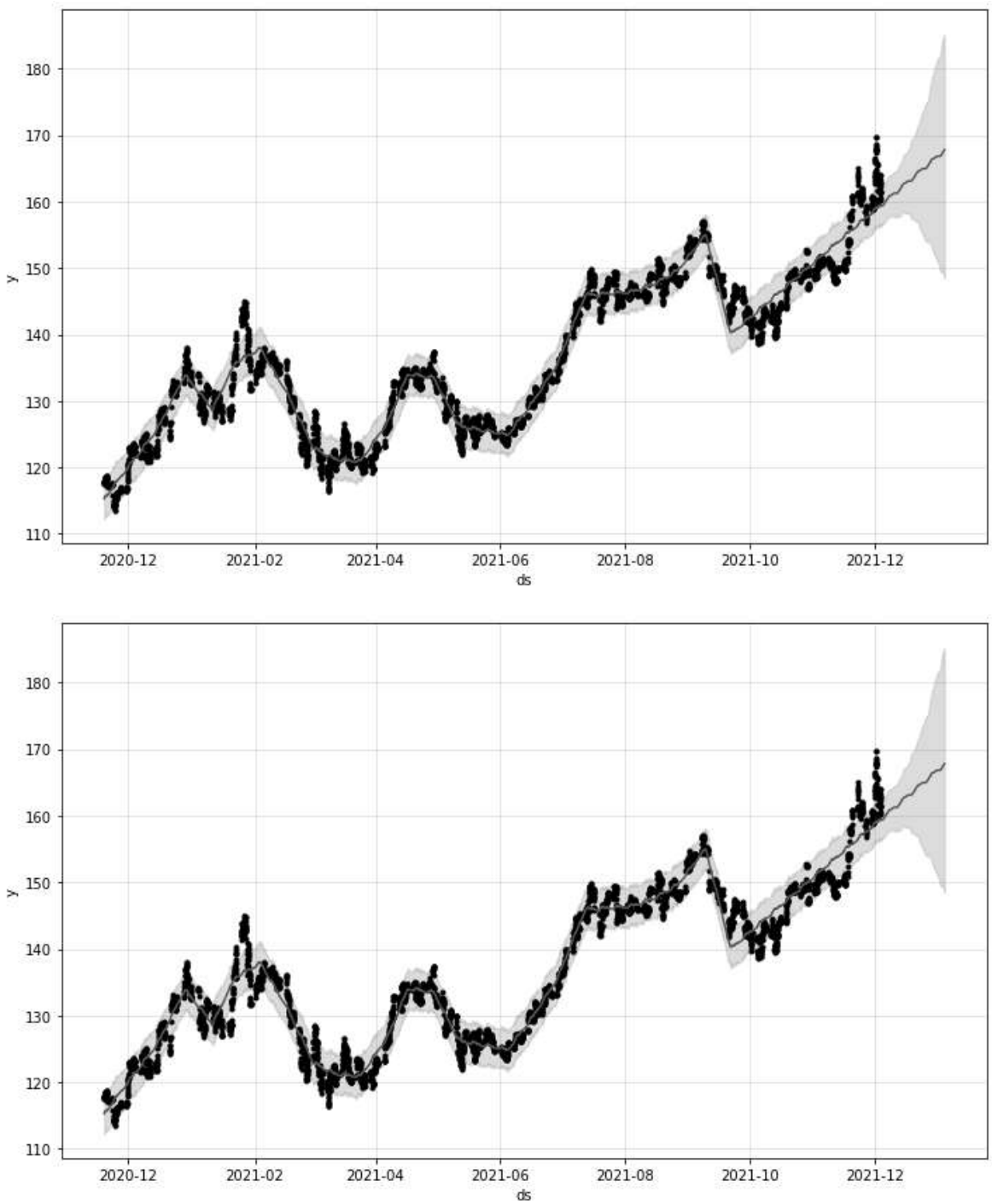
In [118]:
```
forecast.head()
```

Out[118]:

	ds	trend	yhat_lower	yhat_upper	trend_lower	trend_upper	additive_terms	adc
0	2020-11-19 09:00:00	115.194933	112.557767	118.548600	115.194933	115.194933	0.118527	
1	2020-11-19 10:00:00	115.211668	112.417815	118.405048	115.211668	115.211668	0.110813	
2	2020-11-19 11:00:00	115.228402	112.310141	118.264847	115.228402	115.228402	0.119643	
3	2020-11-19 12:00:00	115.245137	112.401827	118.377822	115.245137	115.245137	0.154445	
4	2020-11-19 13:00:00	115.261872	112.437094	118.167970	115.261872	115.261872	0.191199	

In [119]:
```
prophet_model.plot(forecast)
```

Out[119]:



In [ ]:
```

```