In [1]:
```python
import warnings
warnings.filterwarnings("ignore")

import pandas as pd
import numpy as np
import seaborn as sns
import datetime
import matplotlib.pyplot as plt

from statsmodels.tsa.seasonal import seasonal_decompose
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

from pmdarima.arima import auto_arima

from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error
```

In [2]:
```python
project_data = pd.read_csv("HomeC.csv", low_memory=False)
```

Data Preprocessing

In [3]:
```python
project_data.head()
```

Out[3]:

	time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	c
0	1451624400	0.932833	0.003483	0.932833	0.000033	0.020700	0.061917	0.442633	0.124150	0.00
1	1451624401	0.934333	0.003467	0.934333	0.000000	0.020717	0.063817	0.444067	0.124000	0.00
2	1451624402	0.931817	0.003467	0.931817	0.000017	0.020700	0.062317	0.446067	0.123533	0.00
3	1451624403	1.022050	0.003483	1.022050	0.000017	0.106900	0.068517	0.446583	0.123133	0.00
4	1451624404	1.139400	0.003467	1.139400	0.000133	0.236933	0.063983	0.446533	0.122850	0.00

5 rows × 32 columns

In [4]:
```python
project_data.tail()
```

Out[4]:

	time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]
503906	1452128306	1.599333	0.003233	1.599333	0.000050	0.104017	0.625033	0.041750	0.005233
503907	1452128307	1.924267	0.003217	1.924267	0.000033	0.422383	0.637733	0.042033	0.004983
503908	1452128308	1.978200	0.003217	1.978200	0.000050	0.495667	0.620367	0.042100	0.005333
503909	1452128309	1.990950	0.003233	1.990950	0.000050	0.494700	0.634133	0.042100	0.004917
503910	\	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

5 rows × 32 columns

In [5]:
```python
project_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503911 entries, 0 to 503910
Data columns (total 32 columns):
 #   Column                Non-Null Count   Dtype
```

```
---  ------               --------------   -----
 0   time                 503911 non-null  object
 1   use [kW]             503910 non-null  float64
 2   gen [kW]             503910 non-null  float64
 3   House overall [kW]   503910 non-null  float64
 4   Dishwasher [kW]      503910 non-null  float64
 5   Furnace 1 [kW]       503910 non-null  float64
 6   Furnace 2 [kW]       503910 non-null  float64
 7   Home office [kW]     503910 non-null  float64
 8   Fridge [kW]          503910 non-null  float64
 9   Wine cellar [kW]     503910 non-null  float64
 10  Garage door [kW]     503910 non-null  float64
 11  Kitchen 12 [kW]      503910 non-null  float64
 12  Kitchen 14 [kW]      503910 non-null  float64
 13  Kitchen 38 [kW]      503910 non-null  float64
 14  Barn [kW]            503910 non-null  float64
 15  Well [kW]            503910 non-null  float64
 16  Microwave [kW]       503910 non-null  float64
 17  Living room [kW]     503910 non-null  float64
 18  Solar [kW]           503910 non-null  float64
 19  temperature          503910 non-null  float64
 20  icon                 503910 non-null  object
 21  humidity             503910 non-null  float64
 22  visibility           503910 non-null  float64
 23  summary              503910 non-null  object
 24  apparentTemperature  503910 non-null  float64
 25  pressure             503910 non-null  float64
 26  windSpeed            503910 non-null  float64
 27  cloudCover           503910 non-null  object
 28  windBearing          503910 non-null  float64
 29  precipIntensity      503910 non-null  float64
 30  dewPoint             503910 non-null  float64
 31  precipProbability    503910 non-null  float64
dtypes: float64(28), object(4)
memory usage: 123.0+ MB
```

In [6]:
```
project_data.isna().sum()
```

Out[6]:
```
time                   0
use [kW]               1
gen [kW]               1
House overall [kW]     1
Dishwasher [kW]        1
Furnace 1 [kW]         1
Furnace 2 [kW]         1
Home office [kW]       1
Fridge [kW]            1
Wine cellar [kW]       1
Garage door [kW]       1
Kitchen 12 [kW]        1
Kitchen 14 [kW]        1
Kitchen 38 [kW]        1
Barn [kW]              1
Well [kW]              1
Microwave [kW]         1
Living room [kW]       1
Solar [kW]             1
temperature            1
icon                   1
humidity               1
visibility             1
summary                1
apparentTemperature    1
pressure               1
windSpeed              1
cloudCover             1
```

```
windBearing            1
precipIntensity        1
dewPoint               1
precipProbability      1
dtype: int64
```

In [7]:
```
project_data = project_data[0:-1]
```

In [8]:
```
project_data.isnull().sum()
```

Out[8]:
```
time                   0
use [kW]               0
gen [kW]               0
House overall [kW]     0
Dishwasher [kW]        0
Furnace 1 [kW]         0
Furnace 2 [kW]         0
Home office [kW]       0
Fridge [kW]            0
Wine cellar [kW]       0
Garage door [kW]       0
Kitchen 12 [kW]        0
Kitchen 14 [kW]        0
Kitchen 38 [kW]        0
Barn [kW]              0
Well [kW]              0
Microwave [kW]         0
Living room [kW]       0
Solar [kW]             0
temperature            0
icon                   0
humidity               0
visibility             0
summary                0
apparentTemperature    0
pressure               0
windSpeed              0
cloudCover             0
windBearing            0
precipIntensity        0
dewPoint               0
precipProbability      0
dtype: int64
```

In [9]:
```
project_data[project_data.isnull().any(axis=1)]
```

Out[9]:

time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	Wine cellar [kW]	...	visibility	summary

0 rows × 32 columns

In [10]:
```
pd.to_datetime(project_data['time'], unit='s').head(3)
```

Out[10]:
```
0   2016-01-01 05:00:00
1   2016-01-01 05:00:01
2   2016-01-01 05:00:02
Name: time, dtype: datetime64[ns]
```

In [11]:
```
project_data['time'] = pd.DatetimeIndex(pd.date_range('2016-01-01 05:00', periods=len(pr
project_data = project_data.set_index('time')
project_data.head()
```

Out[11]:

	use [kW]	gen [kW]	House	Dishwasher	Furnace	Furnace	Home	Fridge	Wine

time			overall [kW]	[kW]	1 [kW]	2 [kW]	office [kW]	[kW]	cellar [kW]
2016-01-01 05:00:00	0.932833	0.003483	0.932833	0.000033	0.020700	0.061917	0.442633	0.124150	0.006983
2016-01-01 05:01:00	0.934333	0.003467	0.934333	0.000000	0.020717	0.063817	0.444067	0.124000	0.006983
2016-01-01 05:02:00	0.931817	0.003467	0.931817	0.000017	0.020700	0.062317	0.446067	0.123533	0.006983
2016-01-01 05:03:00	1.022050	0.003483	1.022050	0.000017	0.106900	0.068517	0.446583	0.123133	0.006983
2016-01-01 05:04:00	1.139400	0.003467	1.139400	0.000133	0.236933	0.063983	0.446533	0.122850	0.006850

5 rows × 31 columns

In [12]:
```
project_data['year'] = project_data.index.year
project_data['month'] = project_data.index.month
project_data['day'] = project_data.index.day
project_data['weekday'] = project_data.index.day_name()
project_data['weekofyear'] = project_data.index.weekofyear
project_data['hour'] = project_data.index.hour
project_data['minute'] = project_data.index.minute
project_data.head()
```

Out[12]:

time	use [kW]	gen [kW]	House overall [kW]	Dishwasher [kW]	Furnace 1 [kW]	Furnace 2 [kW]	Home office [kW]	Fridge [kW]	Wine cellar [kW]
2016-01-01 05:00:00	0.932833	0.003483	0.932833	0.000033	0.020700	0.061917	0.442633	0.124150	0.006983
2016-01-01 05:01:00	0.934333	0.003467	0.934333	0.000000	0.020717	0.063817	0.444067	0.124000	0.006983
2016-01-01 05:02:00	0.931817	0.003467	0.931817	0.000017	0.020700	0.062317	0.446067	0.123533	0.006983
2016-01-01 05:03:00	1.022050	0.003483	1.022050	0.000017	0.106900	0.068517	0.446583	0.123133	0.006983
2016-01-01 05:04:00	1.139400	0.003467	1.139400	0.000133	0.236933	0.063983	0.446533	0.122850	0.006850

5 rows × 38 columns

In [13]:
```
project_data.columns = [i.replace(' [kW]', '') for i in project_data.columns]

project_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 503910 entries, 2016-01-01 05:00:00 to 2016-12-16 03:29:00
Data columns (total 38 columns):
 #   Column               Non-Null Count   Dtype  
---  ------               --------------   -----  
 0   use                  503910 non-null  float64
 1   gen                  503910 non-null  float64
 2   House overall        503910 non-null  float64
 3   Dishwasher           503910 non-null  float64
 4   Furnace 1            503910 non-null  float64
 5   Furnace 2            503910 non-null  float64
 6   Home office          503910 non-null  float64
 7   Fridge               503910 non-null  float64
 8   Wine cellar          503910 non-null  float64
 9   Garage door          503910 non-null  float64
 10  Kitchen 12           503910 non-null  float64
 11  Kitchen 14           503910 non-null  float64
 12  Kitchen 38           503910 non-null  float64
 13  Barn                 503910 non-null  float64
 14  Well                 503910 non-null  float64
 15  Microwave            503910 non-null  float64
 16  Living room          503910 non-null  float64
 17  Solar                503910 non-null  float64
 18  temperature          503910 non-null  float64
 19  icon                 503910 non-null  object 
 20  humidity             503910 non-null  float64
 21  visibility           503910 non-null  float64
 22  summary              503910 non-null  object 
 23  apparentTemperature  503910 non-null  float64
 24  pressure             503910 non-null  float64
 25  windSpeed            503910 non-null  float64
 26  cloudCover           503910 non-null  object 
 27  windBearing          503910 non-null  float64
 28  precipIntensity      503910 non-null  float64
 29  dewPoint             503910 non-null  float64
 30  precipProbability    503910 non-null  float64
 31  year                 503910 non-null  int64  
 32  month                503910 non-null  int64  
 33  day                  503910 non-null  int64  
 34  weekday              503910 non-null  object 
 35  weekofyear           503910 non-null  int64  
 36  hour                 503910 non-null  int64  
 37  minute               503910 non-null  int64  
dtypes: float64(28), int64(6), object(4)
memory usage: 149.9+ MB
```

Correlation and Seasonality Detection

There are 3 Kitchen Items, 2 Furnace Items Stitching them will reduce the dimentions of the data set

In [14]:
```python
kitchen_columns = ['Kitchen 12','Kitchen 14','Kitchen 38']
furnace_columns = ['Furnace 1','Furnace 2']

project_data['kitchen'] = project_data[kitchen_columns].sum(axis=1)
project_data['furnace'] = project_data[furnace_columns].sum(axis=1)
```

In [15]:
```python
project_data = project_data.drop(columns=kitchen_columns+furnace_columns, axis=1)
```

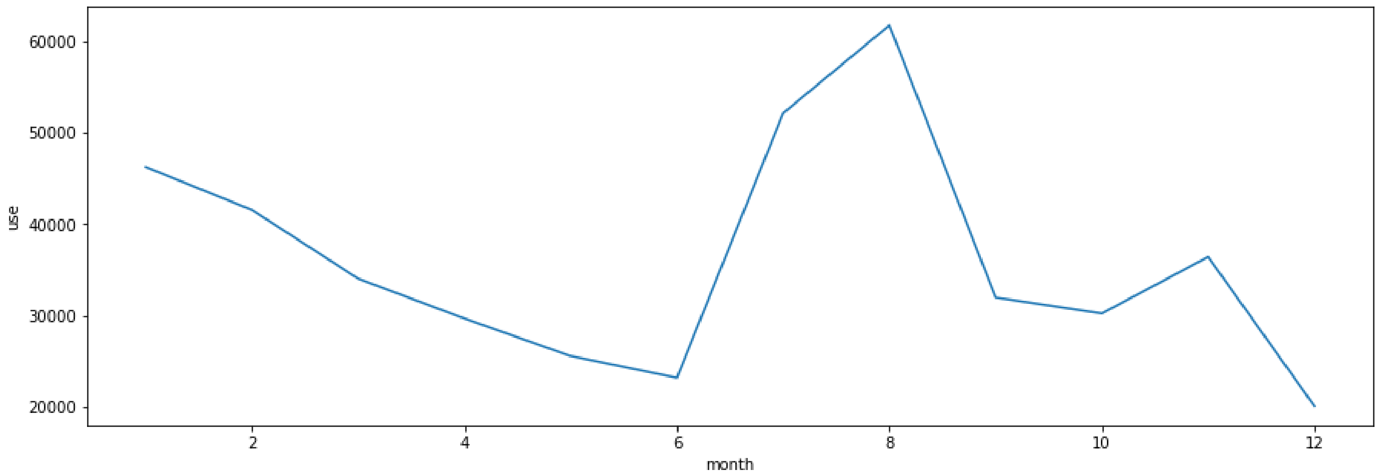
Correlation Analysis

In [16]:
```python
transformed_train_df = project_data.groupby(['month']).agg({'use':'sum'})
transformed_train_df['month'] = transformed_train_df.index
transformed_train_df.rename(columns = {'use':'use'}, inplace = True)
print(transformed_train_df.shape, transformed_train_df.columns)
```

```
(12, 2) Index(['use', 'month'], dtype='object')
```

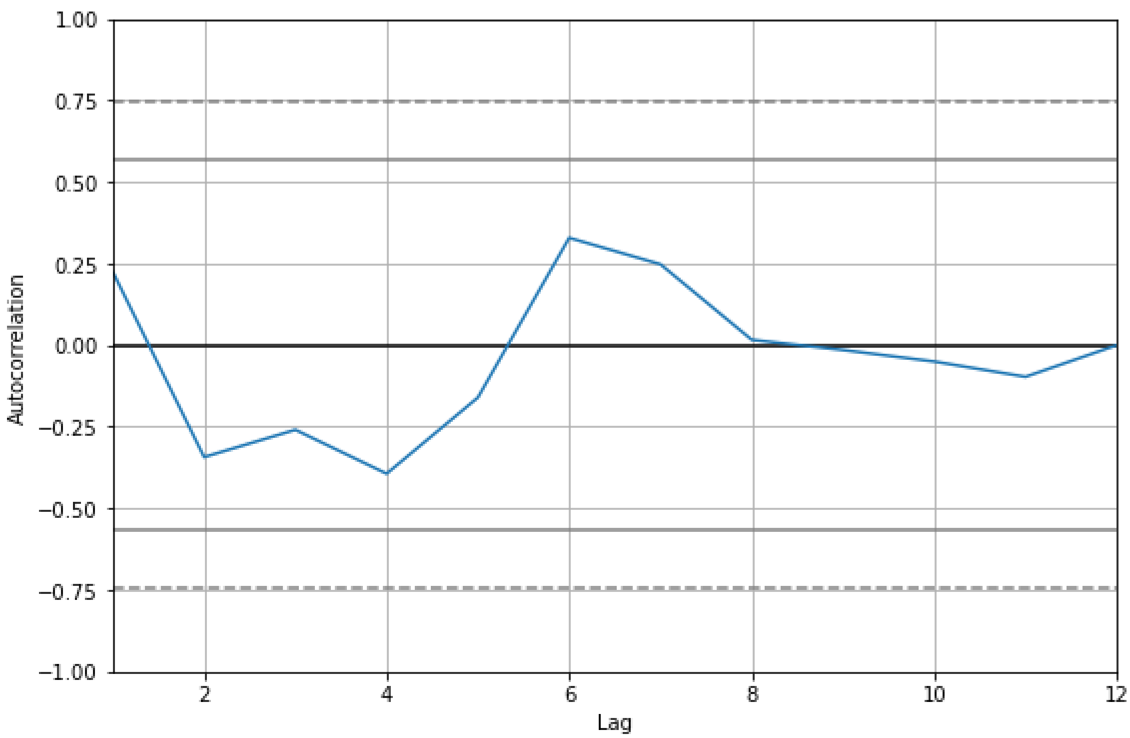
In [17]:
```python
plt.figure(figsize=(15,5))
sns.lineplot(data=transformed_train_df, x=transformed_train_df['month'], y=transformed_t
```

Out[17]:
```
<AxesSubplot:xlabel='month', ylabel='use'>
```



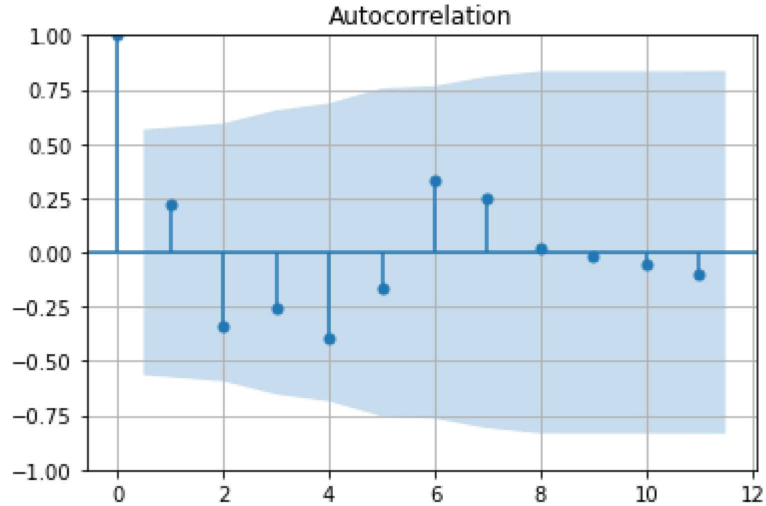
In [18]:
```python
plt.figure(figsize = (9, 6))
pd.plotting.autocorrelation_plot(transformed_train_df['use'])
print('Autocorrelation =', round(transformed_train_df['use'].autocorr(), 4))
```

```
Autocorrelation = 0.2644
```

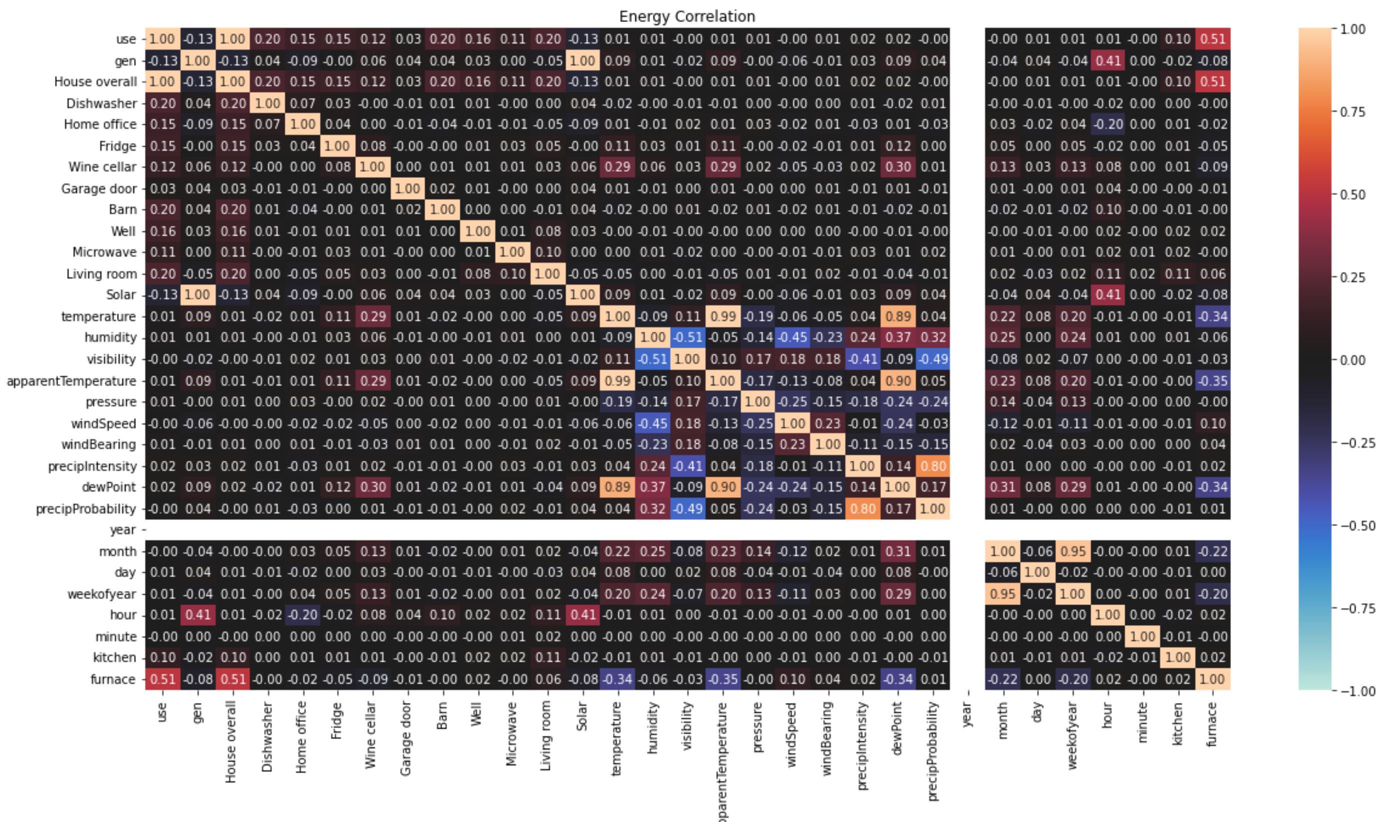


In [19]:
```python
plt.figure(figsize = (15, 16))
plot_acf(transformed_train_df['use'])
plt.grid()
plt.show()
```

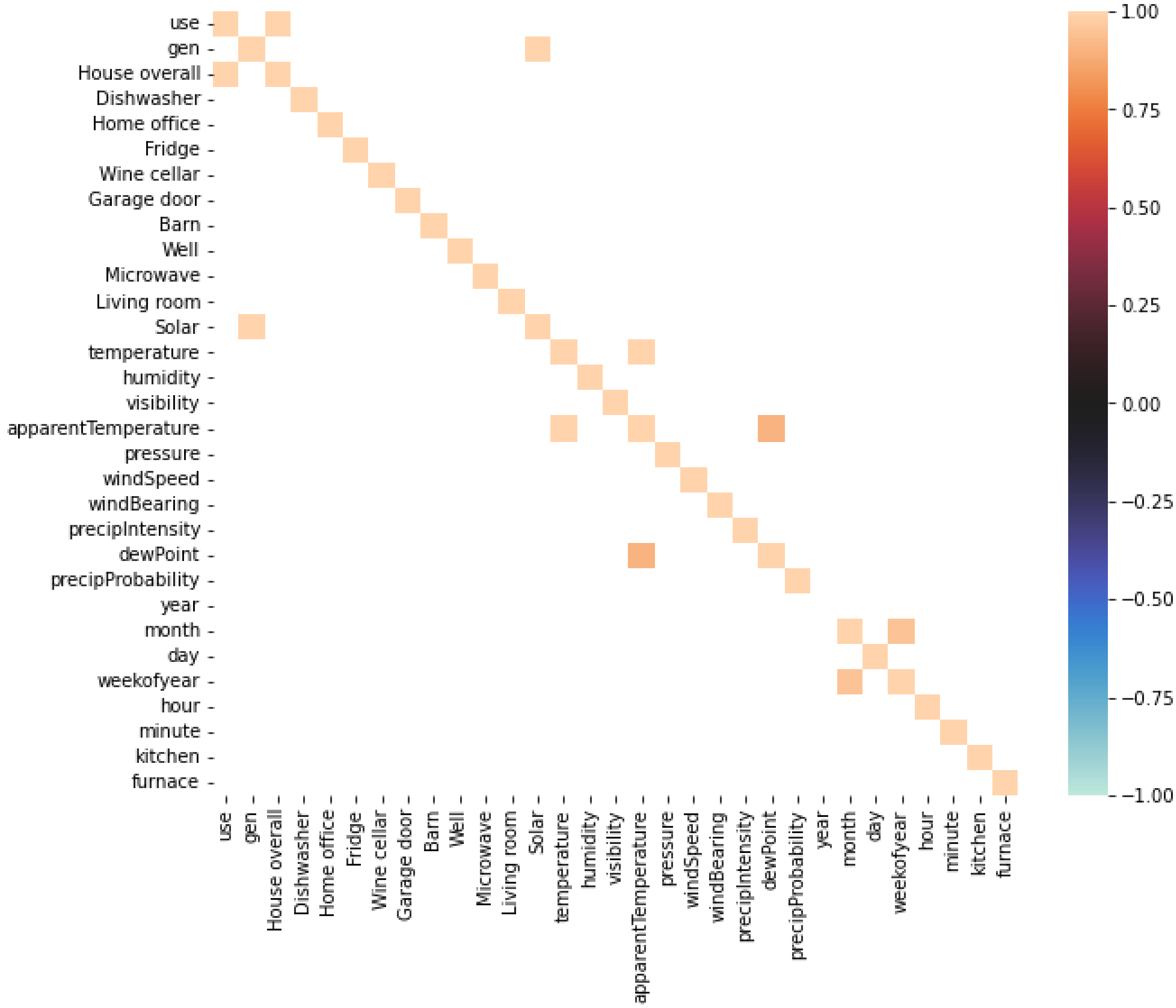
```
<Figure size 1080x1152 with 0 Axes>
```



In [20]:
```
#Checking Correlations (all Energy features)
fig = plt.subplots(figsize=(20, 10))
sns.heatmap(project_data.corr(), annot=True, fmt='.2f', vmin=-1.0, vmax=1.0, center=0)
plt.title('Energy Correlation', fontsize=12);
```



In [21]:
```
fig = plt.subplots(figsize=(10, 8))
corr = project_data.corr()
sns.heatmap(corr[corr>0.9], vmax=1, vmin=-1, center=0)
plt.show()
```



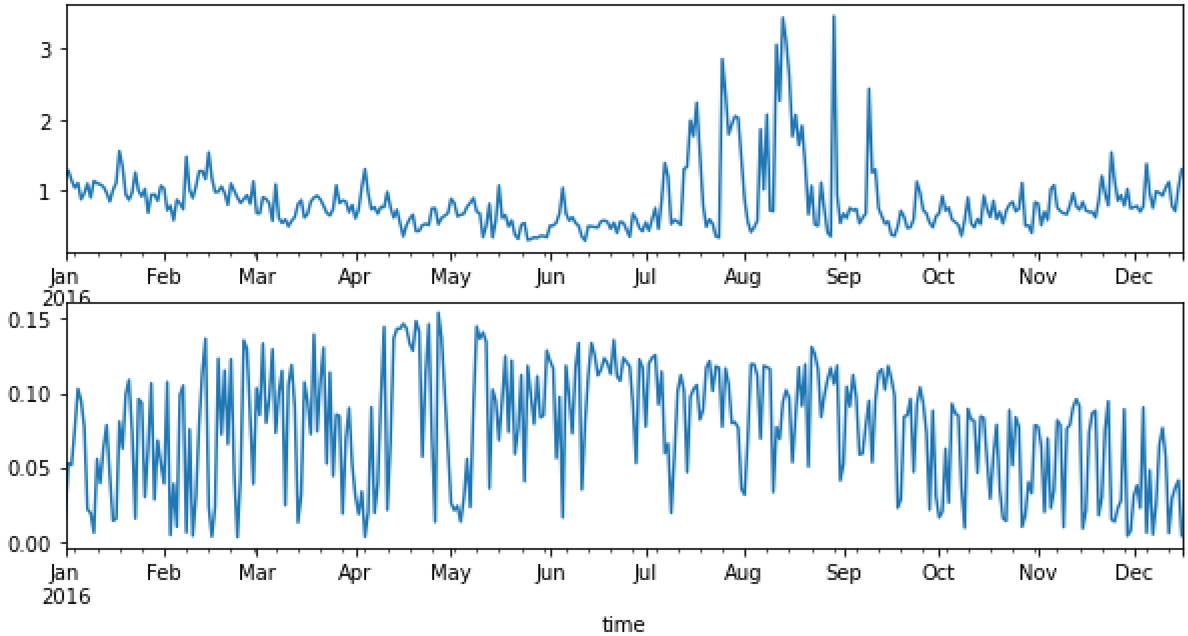
Correlation Summary

```
- use and House overall has highest correlation >90%
- gen & solar has highest correlation >90%
```

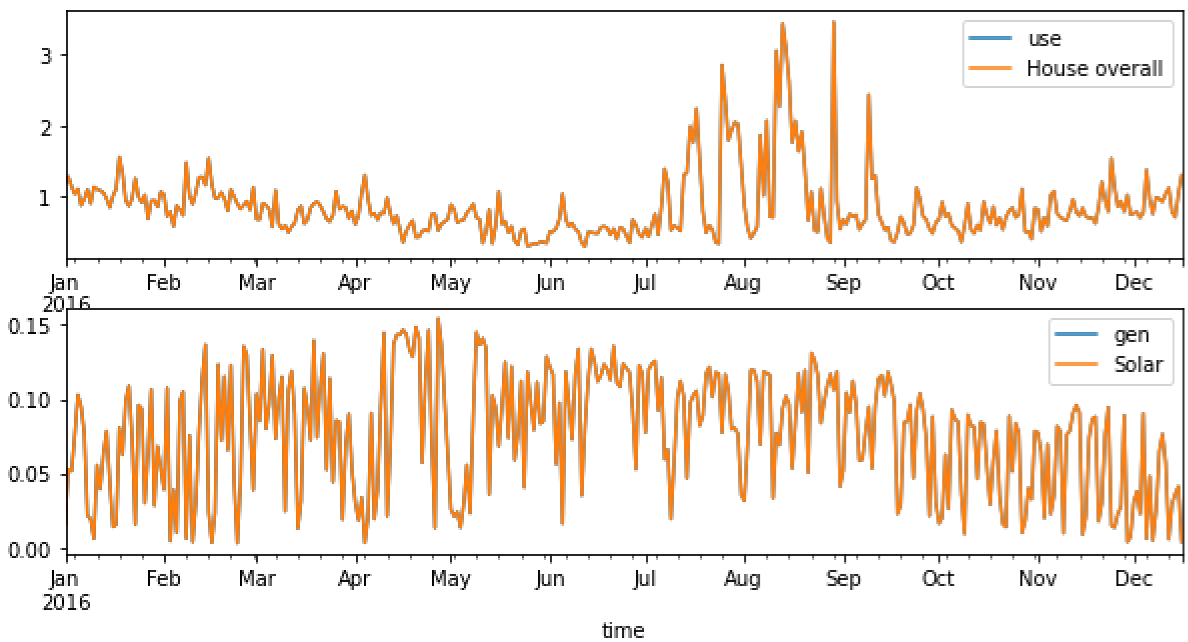
Seasonality & In Depth Exploratory Data Analysis

In [22]:
```
fig, axes = plt.subplots(2,1, figsize=(10,5))
project_data['use'].resample('D').mean().plot(ax=axes[0])
project_data['gen'].resample('D').mean().plot(ax=axes[1])
```

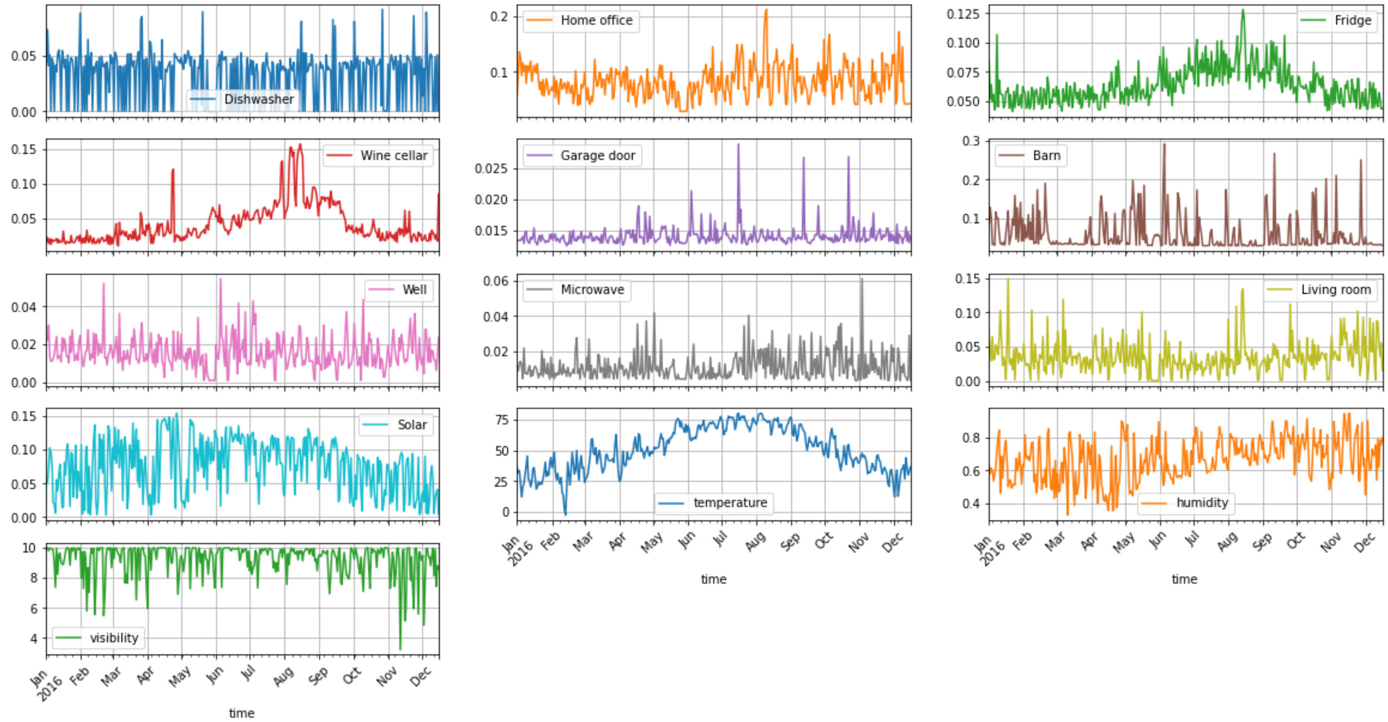
Out[22]: <AxesSubplot:xlabel='time'>



In [23]:
```
## As per correlation analysis summary

fig, axes = plt.subplots(2,1, figsize=(10,5))
project_data[['use','House overall']].resample('D').mean().plot(ax=axes[0])
project_data[['gen','Solar']].resample('D').mean().plot(ax=axes[1]);
```



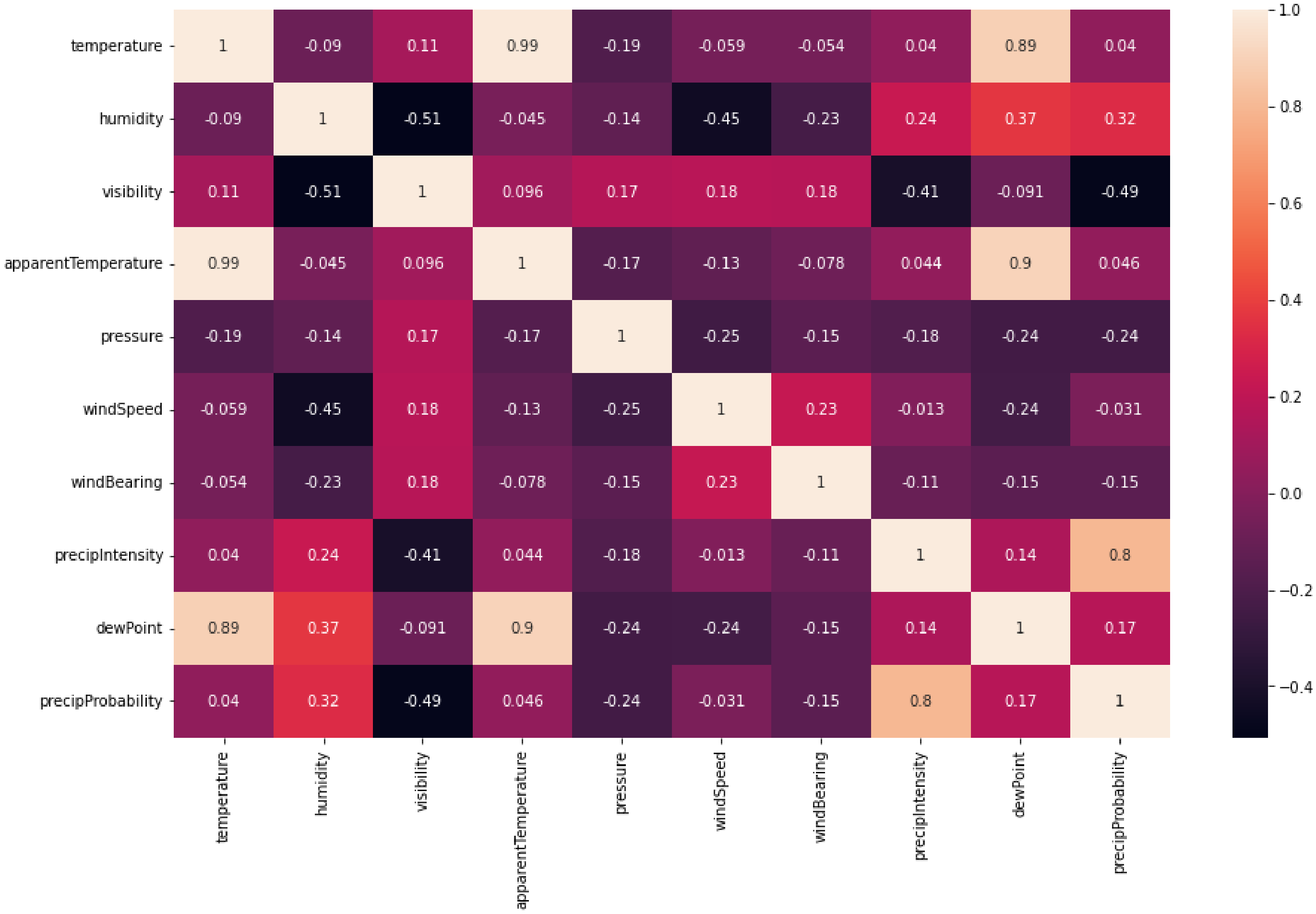
In [24]:
```
#Energy features (resample by day)
project_data[project_data.columns[3:18].tolist()].resample('D').mean().plot(subplots=Tru
                                                                 grid=True, rot=45);
```



In [25]:

```
weather_columns = ['temperature',
'humidity',
'visibility',
'apparentTemperature',
'pressure',
'windSpeed',
'cloudCover',
'windBearing',
'precipIntensity',
'dewPoint',
'precipProbability']

plt.figure(figsize=(15,9))
weather_data = project_data.filter(weather_columns, axis=1)
sns.heatmap(weather_data.corr(), annot=True)
```

Out[25]: <AxesSubplot:>

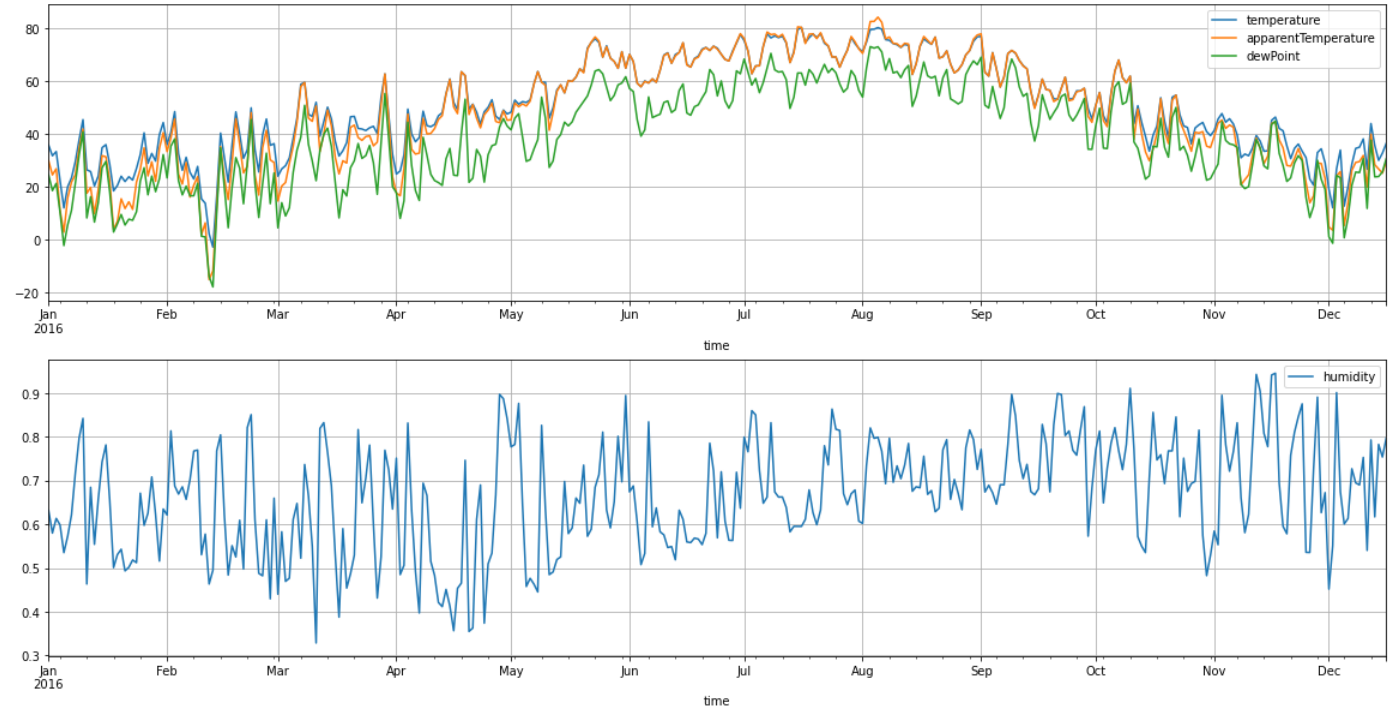


Weather Data Correlation Summary

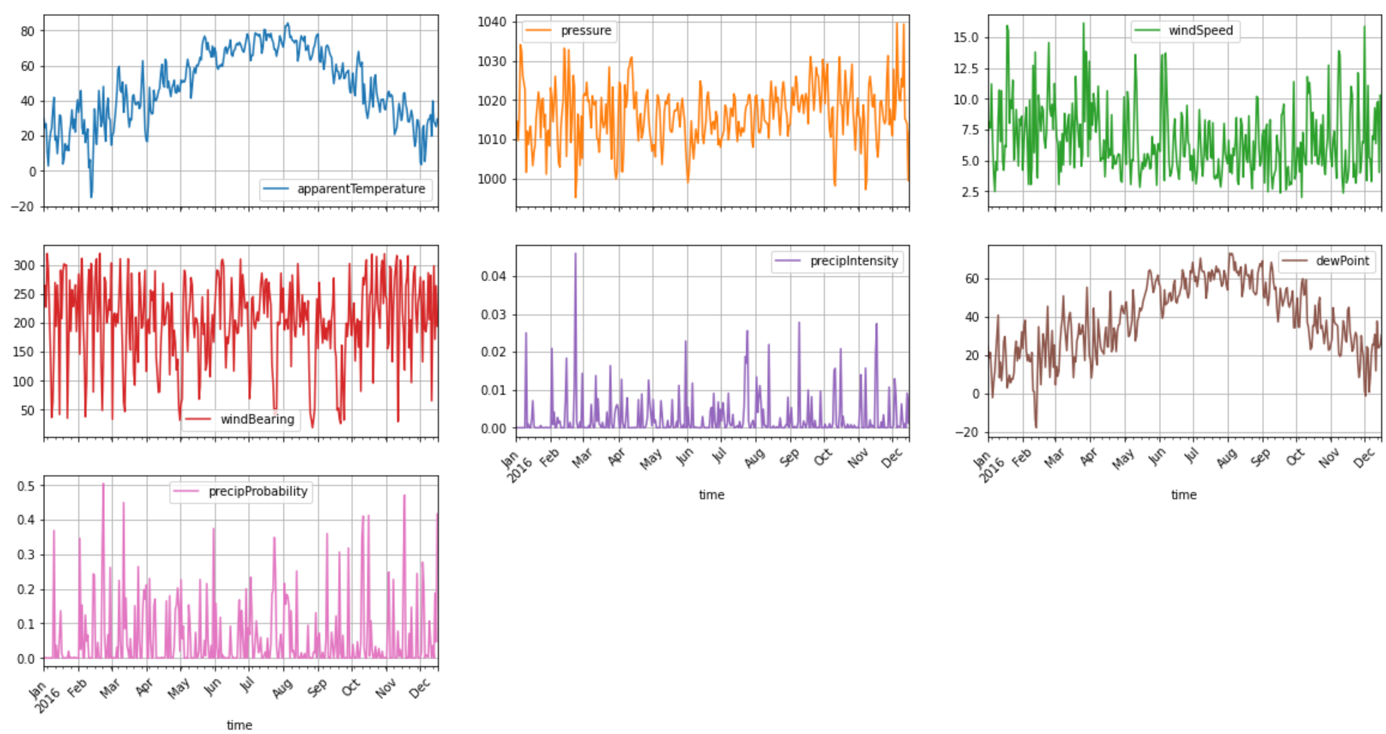
    - temperature, apparentTemperature, dewPoint has correlation > 90%

In [26]:
```
#Let's check a few correlations about the weather data
fig, axes = plt.subplots(2,1, figsize=(20,10))
project_data[['temperature','apparentTemperature', 'dewPoint']].resample('D').mean().plo
project_data[['humidity']].resample('D').mean().plot(ax=axes[1], grid=True)
```

Out[26]: <AxesSubplot:xlabel='time'>



In [27]:
```
#Rest of Weather features (resample by day)
project_data[project_data.columns[17:26].tolist()].resample('D').mean().plot(subplots=Tr
                                                               grid=True, rot=45);
```

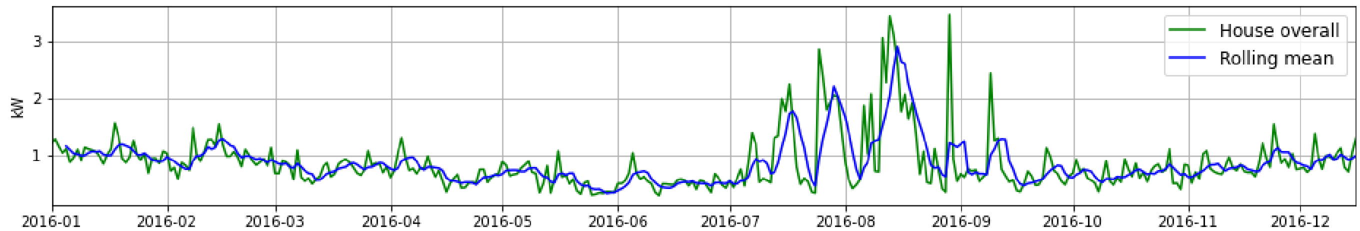


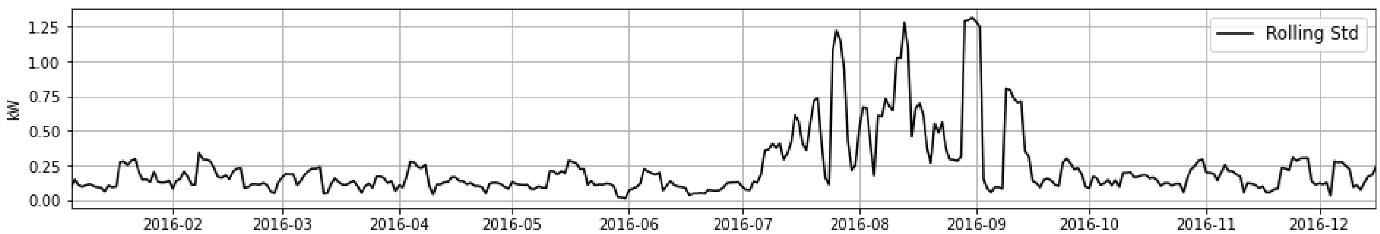
ARIMA

In [28]:
```
# Data resampling by day
data_daily = project_data['House overall'].resample('d').mean()
rollingMEAN = data_daily.rolling(window=5).mean()
rollingSTD = data_daily.rolling(window=5).std()
#Plot
fig, (ax1, ax2) = plt.subplots(2,1,figsize=(16,6))
plt.subplots_adjust(hspace=0.4)
ax1.plot(data_daily, c='green',label='House overall')
ax1.plot(rollingMEAN, c='blue', label='Rolling mean')
ax2.plot(rollingSTD, c='black',label = 'Rolling Std')
```

```
ax1.legend(fontsize=12), ax2.legend(fontsize=12)
ax1.set_ylabel('kW'), ax2.set_ylabel('kW')
ax1.margins(x=0), ax2.margins(x=0)
ax1.grid(), ax2.grid()
```

Out[28]: (None, None)





AUTO-ARIMA

In [29]:

```
result = seasonal_decompose(data_daily, model='additive')
fig = plt.figure()
fig = result.plot()
fig.set_size_inches(10, 10)
```

<Figure size 432x288 with 0 Axes>

In [30]:

```
size = int(len(data_daily)*0.7)
train = data_daily[:size]
test = data_daily[size:]

arima_model = auto_arima(train,
                         start_p=0,
                         d=0,
                         start_q=0,
                         max_p=5,
                         max_d=5,
                         max_q=5,
                         start_P=0,
                         D=1,
                         start_Q=0,
                         max_P=5,
                         max_D=5,
                         max_Q=5,
                         m=12, #if m=1 seasonal is set to False
                         seasonal=True,
                         error_action='warn',
                         trace=True,
                         suppress_warnings=True,
                         stepwise=True,
                         random_state=20,
```

```
                        n_fits=5 # no of fits is taken as 5 we can increatese this hype
                    )
```

```
Performing stepwise search to minimize aic
 ARIMA(0,0,0)(0,1,0)[12] intercept   : AIC=450.335, Time=0.03 sec
 ARIMA(1,0,0)(1,1,0)[12] intercept   : AIC=313.370, Time=0.15 sec
 ARIMA(0,0,1)(0,1,1)[12] intercept   : AIC=331.210, Time=0.21 sec
 ARIMA(0,0,0)(0,1,0)[12]             : AIC=448.376, Time=0.01 sec
 ARIMA(1,0,0)(0,1,0)[12] intercept   : AIC=348.030, Time=0.02 sec
 ARIMA(1,0,0)(2,1,0)[12] intercept   : AIC=310.191, Time=0.65 sec
 ARIMA(1,0,0)(3,1,0)[12] intercept   : AIC=307.438, Time=1.26 sec
 ARIMA(1,0,0)(4,1,0)[12] intercept   : AIC=302.917, Time=1.93 sec
 ARIMA(1,0,0)(5,1,0)[12] intercept   : AIC=302.753, Time=9.59 sec
 ARIMA(1,0,0)(5,1,1)[12] intercept   : AIC=inf, Time=26.45 sec
 ARIMA(1,0,0)(4,1,1)[12] intercept   : AIC=inf, Time=5.70 sec
 ARIMA(0,0,0)(5,1,0)[12] intercept   : AIC=400.160, Time=8.06 sec
 ARIMA(2,0,0)(5,1,0)[12] intercept   : AIC=300.461, Time=10.90 sec
 ARIMA(2,0,0)(4,1,0)[12] intercept   : AIC=300.900, Time=2.74 sec
 ARIMA(2,0,0)(5,1,1)[12] intercept   : AIC=inf, Time=25.64 sec
 ARIMA(2,0,0)(4,1,1)[12] intercept   : AIC=inf, Time=6.16 sec
 ARIMA(3,0,0)(5,1,0)[12] intercept   : AIC=300.484, Time=10.77 sec
 ARIMA(2,0,1)(5,1,0)[12] intercept   : AIC=301.658, Time=13.80 sec
 ARIMA(1,0,1)(5,1,0)[12] intercept   : AIC=299.667, Time=9.50 sec
 ARIMA(1,0,1)(4,1,0)[12] intercept   : AIC=300.406, Time=2.77 sec
 ARIMA(1,0,1)(5,1,1)[12] intercept   : AIC=inf, Time=25.40 sec
 ARIMA(1,0,1)(4,1,1)[12] intercept   : AIC=inf, Time=6.12 sec
 ARIMA(0,0,1)(5,1,0)[12] intercept   : AIC=334.393, Time=9.39 sec
 ARIMA(1,0,2)(5,1,0)[12] intercept   : AIC=301.648, Time=17.11 sec
 ARIMA(0,0,2)(5,1,0)[12] intercept   : AIC=321.785, Time=8.71 sec
 ARIMA(2,0,2)(5,1,0)[12] intercept   : AIC=301.925, Time=18.06 sec
 ARIMA(1,0,1)(5,1,0)[12]             : AIC=297.765, Time=5.49 sec
 ARIMA(1,0,1)(4,1,0)[12]             : AIC=298.489, Time=1.17 sec
 ARIMA(1,0,1)(5,1,1)[12]             : AIC=inf, Time=23.51 sec
 ARIMA(1,0,1)(4,1,1)[12]             : AIC=inf, Time=5.81 sec
 ARIMA(0,0,1)(5,1,0)[12]             : AIC=332.748, Time=4.49 sec
 ARIMA(1,0,0)(5,1,0)[12]             : AIC=300.922, Time=3.60 sec
 ARIMA(2,0,1)(5,1,0)[12]             : AIC=299.755, Time=7.61 sec
 ARIMA(1,0,2)(5,1,0)[12]             : AIC=299.744, Time=5.86 sec
 ARIMA(0,0,0)(5,1,0)[12]             : AIC=398.716, Time=3.38 sec
 ARIMA(0,0,2)(5,1,0)[12]             : AIC=320.155, Time=4.35 sec
 ARIMA(2,0,0)(5,1,0)[12]             : AIC=298.577, Time=4.75 sec
 ARIMA(2,0,2)(5,1,0)[12]             : AIC=300.002, Time=8.19 sec

Best model:  ARIMA(1,0,1)(5,1,0)[12]
Total fit time: 299.352 seconds
```

In [31]:

```
arima_model.summary()
```

Out[31]:

SARIMAX Results

Dep. Variable:	y	**No. Observations:**	245
Model:	SARIMAX(1, 0, 1)x(5, 1, [], 12)	**Log Likelihood**	-140.882
Date:	Wed, 26 Oct 2022	**AIC**	297.765
Time:	22:36:54	**BIC**	325.373
Sample:	0	**HQIC**	308.897
	- 245		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	0.7694	0.048	16.103	0.000	0.676	0.863
ma.L1	-0.2538	0.069	-3.675	0.000	-0.389	-0.118

ar.S.L12	-0.6514	0.057	-11.374	0.000	-0.764	-0.539
ar.S.L24	-0.5026	0.080	-6.260	0.000	-0.660	-0.345
ar.S.L36	-0.4397	0.108	-4.082	0.000	-0.651	-0.229
ar.S.L48	-0.3706	0.145	-2.558	0.011	-0.655	-0.087
ar.S.L60	-0.1733	0.185	-0.936	0.349	-0.536	0.190
sigma2	0.1871	0.007	25.137	0.000	0.173	0.202

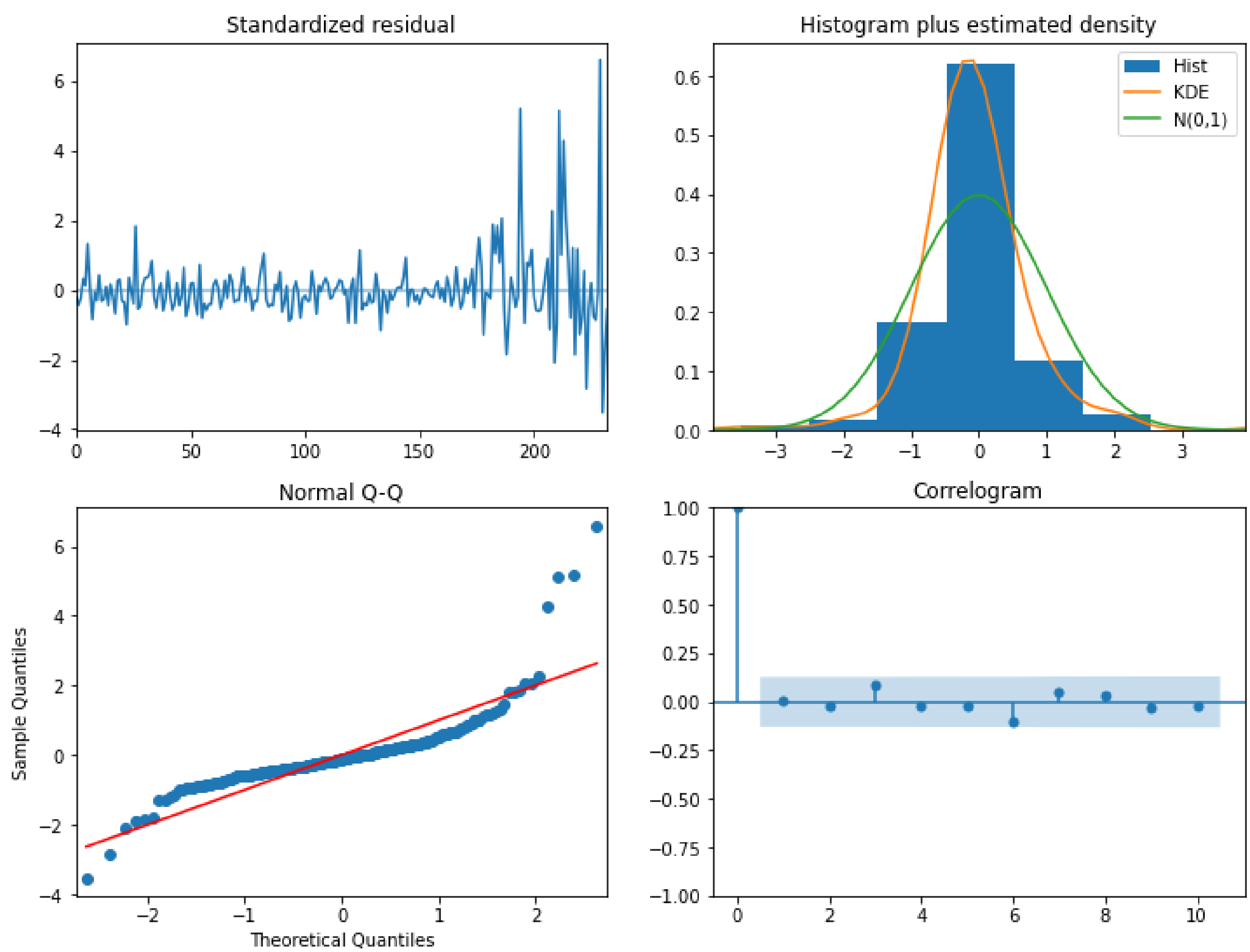
Ljung-Box (L1) (Q):	0.00	**Jarque-Bera (JB):**	2240.36
Prob(Q):	0.96	**Prob(JB):**	0.00
Heteroskedasticity (H):	10.82	**Skew:**	2.51
Prob(H) (two-sided):	0.00	**Kurtosis:**	17.34

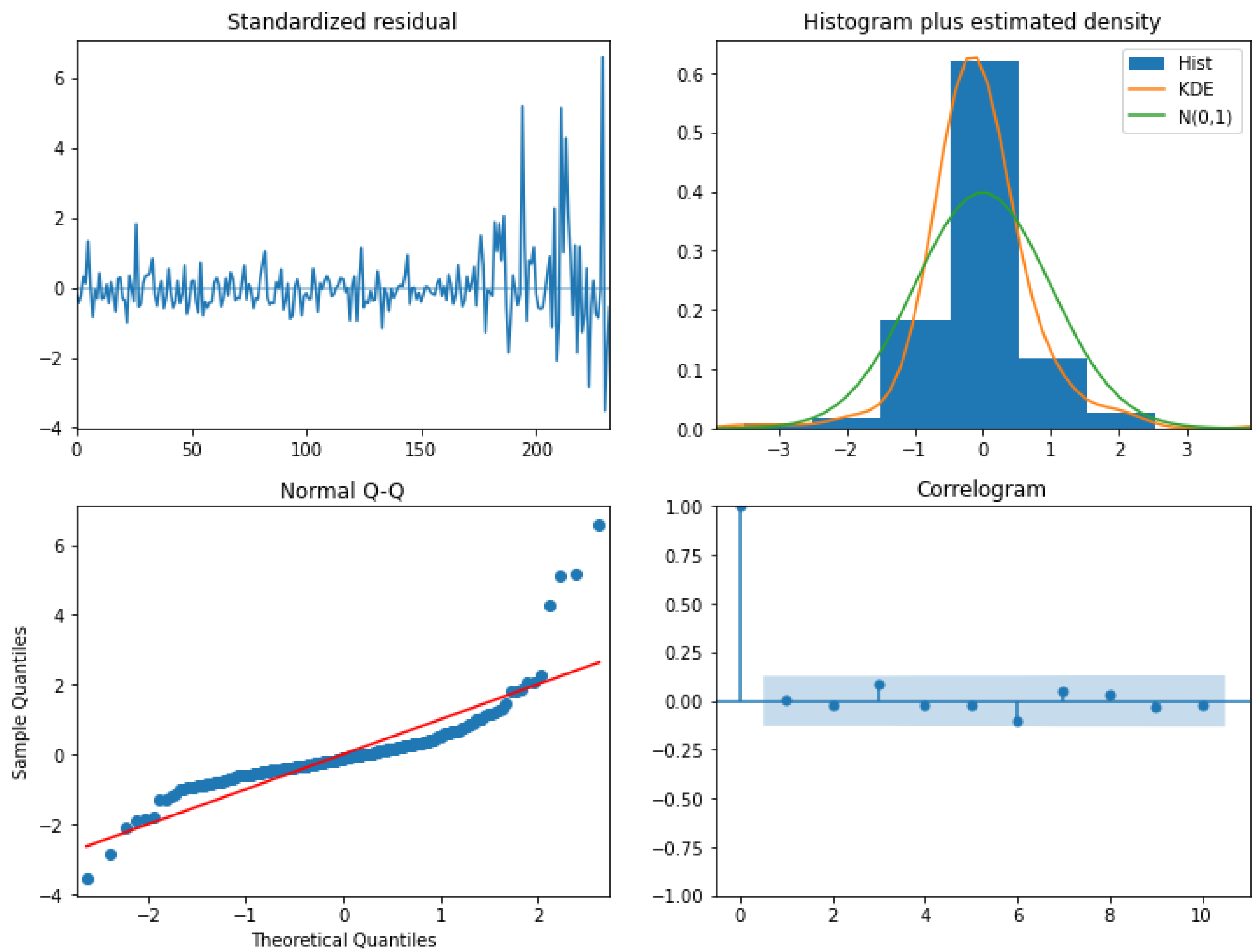
Warnings:
[1] Covariance matrix calculated using the outer product of gradients (complex-step).

Model Residuals

In [32]:
```
arima_model.plot_diagnostics(figsize=(12,9))
```

Out[32]:

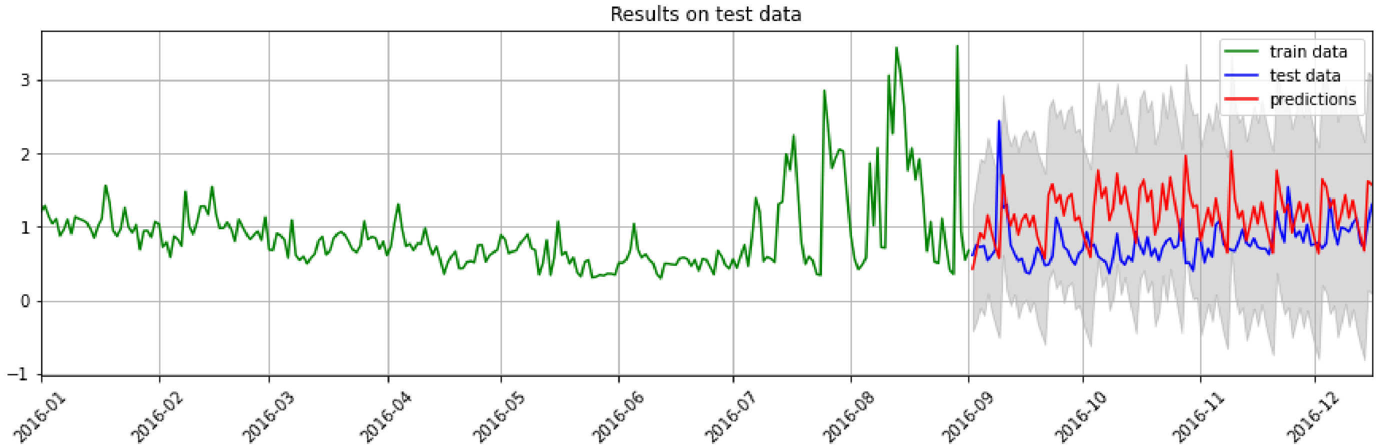




In [33]:
```
y_forec, conf_int  = arima_model.predict(len(test),return_conf_int=True,alpha=0.05)
pred = pd.Series(y_forec, index=test.index)
pred.columns = ['predicted']
confidence = pd.DataFrame(conf_int, columns=['lower', 'upper'])

plt.figure(figsize=(15,4))
plt.plot(train, c='green', label='train data')
plt.plot(test, c='blue', label='test data')
plt.plot(pred, c='red', label='predictions')
plt.legend()
plt.grid(), plt.margins(x=0)
plt.title('Results on test data'), plt.xticks(rotation=45)
plt.fill_between(test.index, confidence['lower'],
                 confidence['upper'], color='k', alpha=.15)


from sklearn.metrics import r2_score
print('Actual values: ', np.around(test[:10].tolist(),3))
print('Predictions:   ', np.around(pred[:10].tolist(),3))
print('RMSE: %.3f' % np.sqrt(mean_squared_error(test, pred)))
MAE = mean_absolute_error(test, pred)
MAPE = np.mean(np.abs(pred - test)/np.abs(test))
MASE = np.mean(np.abs(test - pred ))/(np.abs(np.diff(train)).sum()/(len(train)-1))
print('MAE: %.3f' % MAE)
print('MAPE: %.3f' %MAPE)
print('MASE: %.3f' %MASE)
print('R^2 score: %.3f' % r2_score(test, pred))
```

```
Actual values:  [0.616 0.758 0.724 0.744 0.542 0.617 0.681 2.438 1.256 1.299]
Predictions:    [0.426 0.661 0.912 0.84  1.155 0.917 0.739 0.575 1.705 1.205]
RMSE: 0.581
```

```
MAE: 0.469
MAPE: 0.712
MASE: 1.909
R^2 score: -3.404
```



LSTM

In [34]:

```
from keras.models import Sequential
from keras.layers import LSTM
from keras.layers import Dense, Dropout
from keras.layers import Bidirectional
```

In [35]:

```
n_features = 1
n_steps = 10

model = Sequential()
model.add(LSTM(50, activation='relu', input_shape=(n_steps, n_features)))
model.add(Dense(1))
model.compile(optimizer='adam', loss='mse')
model.summary()
```

```
Metal device set to: Apple M1 Pro
WARNING:tensorflow:Layer lstm will not use cuDNN kernels since it doesn't meet the crite
ria. It will use a generic GPU kernel as fallback when running on GPU.
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 50)                10400

 dense (Dense)               (None, 1)                 51

=================================================================
Total params: 10,451
Trainable params: 10,451
Non-trainable params: 0
_________________________________________________________________
```

```
2022-10-26 22:36:57.049672: I tensorflow/core/common_runtime/pluggable_device/pluggable_
device_factory.cc:305] Could not identify NUMA node of platform GPU ID 0, defaulting to
0. Your kernel may not have been built with NUMA support.
2022-10-26 22:36:57.049982: I tensorflow/core/common_runtime/pluggable_device/pluggable_
device_factory.cc:271] Created TensorFlow device (/job:localhost/replica:0/task:0/devic
e:GPU:0 with 0 MB memory) -> physical PluggableDevice (device: 0, name: METAL, pci bus i
d: <undefined>)
```

In [36]:

```
def train_test_data(seq, steps):
    X, Y = list(), list()
    for i in range(len(seq)):
        sample = i + steps
```

```
        if sample > len(seq)-1:
            break
        x, y = seq[i:sample],seq[sample]
        X.append(x)
        Y.append(y)
    return np.array(X), np.array(Y)
```

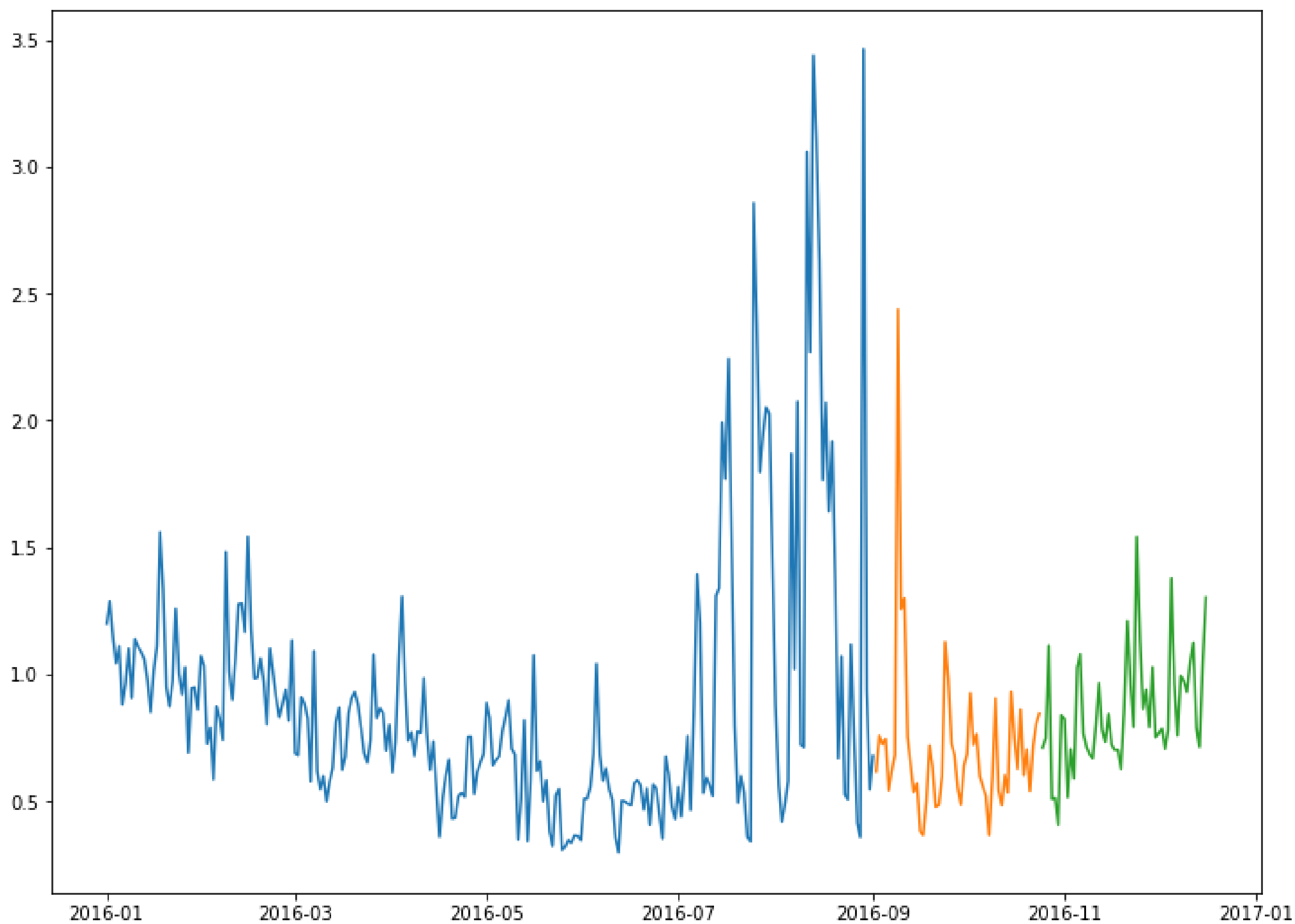
In [37]:

```
def train_test_validation_plot(train_size, test_size):
    plt.figure(figsize=(12,9))
    plt.plot(data_daily[:train_size])
    plt.plot(data_daily[train_size:test_size])
    plt.plot(data_daily[test_size:])
```

In [38]:

```
steps = 10
X, Y = train_test_data(data_daily.tolist(), steps)
X = X.reshape((X.shape[0], X.shape[1], 1))

Training_size = int(len(data_daily)*0.7)
Training_Validation_size = int(((len(data_daily)-size)/2)+size)

X_train, Y_train = X[:Training_size], Y[:Training_size]
X_val, Y_val = X[Training_size:Training_Validation_size], Y[Training_size:Training_Valid
X_test, Y_test = X[Training_Validation_size:], Y[Training_Validation_size:]

print('Training size:', Training_size)
print('Training + Validation size:', Training_Validation_size)

train_test_validation_plot(Training_size, Training_Validation_size)
```

```
Training size: 245
Training + Validation size: 298
```



In [39]:

```
mse_train = list()
mse_val = list()
```

```python
for epoch in range(0,50,5):
    # fit the model with epochs
    model_fit = model.fit(X_train, Y_train, epochs=epoch, verbose=1)

    #model evaluation
    Train_pred = model.predict(X_train, verbose=0)
    Val_pred = model.predict(X_val, verbose=0)

    #computing the training and validation loss
    mse_t = mean_squared_error(Train_pred, Y_train)
    mse_v = mean_squared_error(Val_pred, Y_val)
    mse_train.append(mse_t)
    mse_val.append(mse_v)
```

```
2022-10-26 22:36:57.280615: W tensorflow/core/platform/profile_utils/cpu_utils.cc:128] F
ailed to get CPU frequency: 0 Hz
2022-10-26 22:36:57.353533: I tensorflow/core/grappler/optimizers/custom_graph_optimizer
_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
Epoch 1/5
2022-10-26 22:36:57.834216: I tensorflow/core/grappler/optimizers/custom_graph_optimizer
_registry.cc:113] Plugin optimizer for device_type GPU is enabled.
8/8 [==============================] - 1s 95ms/step - loss: 0.9328
Epoch 2/5
8/8 [==============================] - 1s 86ms/step - loss: 0.5751
Epoch 3/5
8/8 [==============================] - 1s 89ms/step - loss: 0.3108
Epoch 4/5
8/8 [==============================] - 1s 87ms/step - loss: 0.3050
Epoch 5/5
8/8 [==============================] - 1s 86ms/step - loss: 0.2467
Epoch 1/10
8/8 [==============================] - 1s 86ms/step - loss: 0.2514
Epoch 2/10
8/8 [==============================] - 1s 86ms/step - loss: 0.2335
Epoch 3/10
8/8 [==============================] - 1s 83ms/step - loss: 0.2250
Epoch 4/10
8/8 [==============================] - 1s 83ms/step - loss: 0.2248
Epoch 5/10
8/8 [==============================] - 1s 83ms/step - loss: 0.2198
Epoch 6/10
8/8 [==============================] - 1s 83ms/step - loss: 0.2140
Epoch 7/10
8/8 [==============================] - 1s 84ms/step - loss: 0.2090
Epoch 8/10
8/8 [==============================] - 1s 86ms/step - loss: 0.2044
Epoch 9/10
8/8 [==============================] - 1s 84ms/step - loss: 0.1971
Epoch 10/10
8/8 [==============================] - 1s 84ms/step - loss: 0.1925
Epoch 1/15
8/8 [==============================] - 1s 84ms/step - loss: 0.1886
Epoch 2/15
8/8 [==============================] - 1s 86ms/step - loss: 0.1862
Epoch 3/15
8/8 [==============================] - 1s 84ms/step - loss: 0.1855
Epoch 4/15
8/8 [==============================] - 1s 86ms/step - loss: 0.1845
Epoch 5/15
8/8 [==============================] - 1s 86ms/step - loss: 0.1846
Epoch 6/15
8/8 [==============================] - 1s 87ms/step - loss: 0.1854
Epoch 7/15
8/8 [==============================] - 1s 87ms/step - loss: 0.1812
Epoch 8/15
```

```
8/8 [==============================] - 1s 85ms/step - loss: 0.1813
Epoch 9/15
8/8 [==============================] - 1s 88ms/step - loss: 0.1796
Epoch 10/15
8/8 [==============================] - 1s 87ms/step - loss: 0.1793
Epoch 11/15
8/8 [==============================] - 1s 89ms/step - loss: 0.1792
Epoch 12/15
8/8 [==============================] - 1s 88ms/step - loss: 0.1801
Epoch 13/15
8/8 [==============================] - 1s 85ms/step - loss: 0.1771
Epoch 14/15
8/8 [==============================] - 1s 88ms/step - loss: 0.1766
Epoch 15/15
8/8 [==============================] - 1s 86ms/step - loss: 0.1774
Epoch 1/20
8/8 [==============================] - 1s 87ms/step - loss: 0.1737
Epoch 2/20
8/8 [==============================] - 1s 86ms/step - loss: 0.1777
Epoch 3/20
8/8 [==============================] - 1s 87ms/step - loss: 0.1767
Epoch 4/20
8/8 [==============================] - 1s 86ms/step - loss: 0.1751
Epoch 5/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1710
Epoch 6/20
8/8 [==============================] - 1s 86ms/step - loss: 0.1735
Epoch 7/20
8/8 [==============================] - 1s 88ms/step - loss: 0.1739
Epoch 8/20
8/8 [==============================] - 1s 86ms/step - loss: 0.1711
Epoch 9/20
8/8 [==============================] - 1s 88ms/step - loss: 0.1698
Epoch 10/20
8/8 [==============================] - 1s 87ms/step - loss: 0.1760
Epoch 11/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1730
Epoch 12/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1700
Epoch 13/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1685
Epoch 14/20
8/8 [==============================] - 1s 86ms/step - loss: 0.1677
Epoch 15/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1671
Epoch 16/20
8/8 [==============================] - 1s 84ms/step - loss: 0.1678
Epoch 17/20
8/8 [==============================] - 1s 87ms/step - loss: 0.1677
Epoch 18/20
8/8 [==============================] - 1s 87ms/step - loss: 0.1696
Epoch 19/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1790
Epoch 20/20
8/8 [==============================] - 1s 85ms/step - loss: 0.1753
Epoch 1/25
8/8 [==============================] - 1s 84ms/step - loss: 0.1654
Epoch 2/25
8/8 [==============================] - 1s 84ms/step - loss: 0.1733
Epoch 3/25
8/8 [==============================] - 1s 85ms/step - loss: 0.1665
Epoch 4/25
8/8 [==============================] - 1s 85ms/step - loss: 0.1664
Epoch 5/25
8/8 [==============================] - 1s 85ms/step - loss: 0.1682
Epoch 6/25
```

```
8/8 [==============================] - 1s 86ms/step - loss: 0.1670
Epoch 7/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1716
Epoch 8/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1650
Epoch 9/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1676
Epoch 10/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1664
Epoch 11/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1664
Epoch 12/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1658
Epoch 13/25
8/8 [==============================] - 1s 88ms/step - loss: 0.1676
Epoch 14/25
8/8 [==============================] - 1s 88ms/step - loss: 0.1660
Epoch 15/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1654
Epoch 16/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1645
Epoch 17/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1680
Epoch 18/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1707
Epoch 19/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1649
Epoch 20/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1671
Epoch 21/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1654
Epoch 22/25
8/8 [==============================] - 1s 88ms/step - loss: 0.1638
Epoch 23/25
8/8 [==============================] - 1s 88ms/step - loss: 0.1642
Epoch 24/25
8/8 [==============================] - 1s 86ms/step - loss: 0.1643
Epoch 25/25
8/8 [==============================] - 1s 87ms/step - loss: 0.1638
Epoch 1/30
8/8 [==============================] - 1s 86ms/step - loss: 0.1631
Epoch 2/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1632
Epoch 3/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1630
Epoch 4/30
8/8 [==============================] - 1s 83ms/step - loss: 0.1643
Epoch 5/30
8/8 [==============================] - 1s 83ms/step - loss: 0.1696
Epoch 6/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1681
Epoch 7/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1670
Epoch 8/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1628
Epoch 9/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1662
Epoch 10/30
8/8 [==============================] - 1s 87ms/step - loss: 0.1621
Epoch 11/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1626
Epoch 12/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1629
Epoch 13/30
8/8 [==============================] - 1s 87ms/step - loss: 0.1631
Epoch 14/30
```

```
8/8 [==============================] - 1s 87ms/step - loss: 0.1617
Epoch 15/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1624
Epoch 16/30
8/8 [==============================] - 1s 86ms/step - loss: 0.1611
Epoch 17/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1608
Epoch 18/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1609
Epoch 19/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1663
Epoch 20/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1604
Epoch 21/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1629
Epoch 22/30
8/8 [==============================] - 1s 86ms/step - loss: 0.1619
Epoch 23/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1647
Epoch 24/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1610
Epoch 25/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1617
Epoch 26/30
8/8 [==============================] - 1s 86ms/step - loss: 0.1615
Epoch 27/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1631
Epoch 28/30
8/8 [==============================] - 1s 84ms/step - loss: 0.1603
Epoch 29/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1599
Epoch 30/30
8/8 [==============================] - 1s 85ms/step - loss: 0.1601
Epoch 1/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1594
Epoch 2/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1592
Epoch 3/35
8/8 [==============================] - 1s 88ms/step - loss: 0.1592
Epoch 4/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1600
Epoch 5/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1605
Epoch 6/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1608
Epoch 7/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1587
Epoch 8/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1589
Epoch 9/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1598
Epoch 10/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1682
Epoch 11/35
8/8 [==============================] - 1s 93ms/step - loss: 0.1592
Epoch 12/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1592
Epoch 13/35
8/8 [==============================] - 1s 92ms/step - loss: 0.1623
Epoch 14/35
8/8 [==============================] - 1s 87ms/step - loss: 0.1585
Epoch 15/35
8/8 [==============================] - 1s 90ms/step - loss: 0.1583
Epoch 16/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1592
Epoch 17/35
```

```
8/8 [==============================] - 1s 89ms/step - loss: 0.1588
Epoch 18/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1593
Epoch 19/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1587
Epoch 20/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1576
Epoch 21/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1593
Epoch 22/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1573
Epoch 23/35
8/8 [==============================] - 1s 89ms/step - loss: 0.1573
Epoch 24/35
8/8 [==============================] - 1s 86ms/step - loss: 0.1551
Epoch 25/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1623
Epoch 26/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1663
Epoch 27/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1624
Epoch 28/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1623
Epoch 29/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1578
Epoch 30/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1603
Epoch 31/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1652
Epoch 32/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1610
Epoch 33/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1555
Epoch 34/35
8/8 [==============================] - 1s 85ms/step - loss: 0.1570
Epoch 35/35
8/8 [==============================] - 1s 84ms/step - loss: 0.1591
Epoch 1/40
8/8 [==============================] - 1s 86ms/step - loss: 0.1581
Epoch 2/40
8/8 [==============================] - 1s 86ms/step - loss: 0.1565
Epoch 3/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1583
Epoch 4/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1594
Epoch 5/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1560
Epoch 6/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1537
Epoch 7/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1535
Epoch 8/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1540
Epoch 9/40
8/8 [==============================] - 1s 86ms/step - loss: 0.1542
Epoch 10/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1534
Epoch 11/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1532
Epoch 12/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1526
Epoch 13/40
8/8 [==============================] - 1s 86ms/step - loss: 0.1552
Epoch 14/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1516
Epoch 15/40
```

```
8/8 [==============================] - 1s 87ms/step - loss: 0.1529
Epoch 16/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1608
Epoch 17/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1538
Epoch 18/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1509
Epoch 19/40
8/8 [==============================] - 1s 91ms/step - loss: 0.1531
Epoch 20/40
8/8 [==============================] - 1s 91ms/step - loss: 0.1525
Epoch 21/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1522
Epoch 22/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1501
Epoch 23/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1500
Epoch 24/40
8/8 [==============================] - 1s 90ms/step - loss: 0.1482
Epoch 25/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1486
Epoch 26/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1524
Epoch 27/40
8/8 [==============================] - 1s 90ms/step - loss: 0.1551
Epoch 28/40
8/8 [==============================] - 1s 90ms/step - loss: 0.1524
Epoch 29/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1497
Epoch 30/40
8/8 [==============================] - 1s 90ms/step - loss: 0.1507
Epoch 31/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1589
Epoch 32/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1487
Epoch 33/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1477
Epoch 34/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1540
Epoch 35/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1488
Epoch 36/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1474
Epoch 37/40
8/8 [==============================] - 1s 87ms/step - loss: 0.1501
Epoch 38/40
8/8 [==============================] - 1s 90ms/step - loss: 0.1529
Epoch 39/40
8/8 [==============================] - 1s 89ms/step - loss: 0.1527
Epoch 40/40
8/8 [==============================] - 1s 88ms/step - loss: 0.1521
Epoch 1/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1517
Epoch 2/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1547
Epoch 3/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1535
Epoch 4/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1484
Epoch 5/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1433
Epoch 6/45
8/8 [==============================] - 1s 87ms/step - loss: 0.1431
Epoch 7/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1451
Epoch 8/45
```

```
8/8 [==============================] - 1s 85ms/step - loss: 0.1428
Epoch 9/45
8/8 [==============================] - 1s 87ms/step - loss: 0.1429
Epoch 10/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1416
Epoch 11/45
8/8 [==============================] - 1s 88ms/step - loss: 0.1447
Epoch 12/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1534
Epoch 13/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1567
Epoch 14/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1534
Epoch 15/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1522
Epoch 16/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1471
Epoch 17/45
8/8 [==============================] - 1s 87ms/step - loss: 0.1483
Epoch 18/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1441
Epoch 19/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1410
Epoch 20/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1415
Epoch 21/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1499
Epoch 22/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1457
Epoch 23/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1445
Epoch 24/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1430
Epoch 25/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1414
Epoch 26/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1452
Epoch 27/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1509
Epoch 28/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1512
Epoch 29/45
8/8 [==============================] - 1s 83ms/step - loss: 0.1445
Epoch 30/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1433
Epoch 31/45
8/8 [==============================] - 1s 83ms/step - loss: 0.1419
Epoch 32/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1364
Epoch 33/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1375
Epoch 34/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1392
Epoch 35/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1414
Epoch 36/45
8/8 [==============================] - 1s 86ms/step - loss: 0.1465
Epoch 37/45
8/8 [==============================] - 1s 85ms/step - loss: 0.1427
Epoch 38/45
8/8 [==============================] - 1s 82ms/step - loss: 0.1385
Epoch 39/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1351
Epoch 40/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1354
Epoch 41/45
```

```
8/8 [==============================] - 1s 84ms/step - loss: 0.1352
Epoch 42/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1317
Epoch 43/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1349
Epoch 44/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1303
Epoch 45/45
8/8 [==============================] - 1s 84ms/step - loss: 0.1377
```

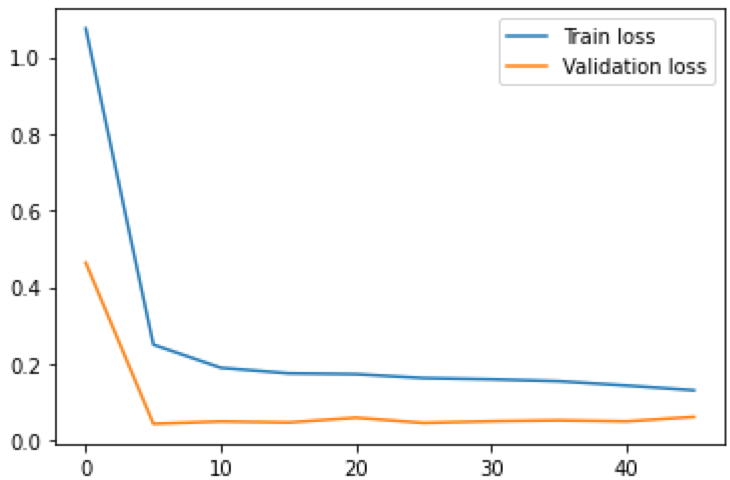
In [40]:

```
# Plot the loss results
def plot_loss_results(mse_train, mse_val):
    plt.plot(range(0,50,5), mse_train, label='Train loss')
    plt.plot(range(0,50,5), mse_val,  label='Validation loss')
    plt.legend()
    print('Train MSE minimum:', min(mse_train))
    print('Validation MSE minimum:', min(mse_val))
```

In [41]:

```
plot_loss_results(mse_train, mse_val)
```

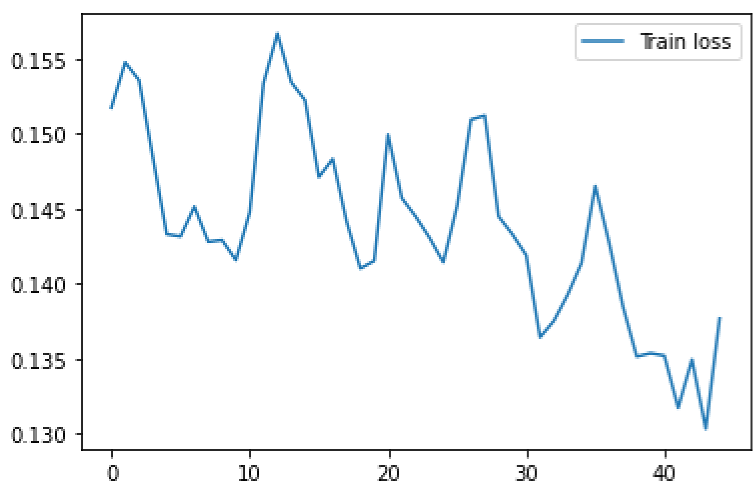
```
Train MSE minimum: 0.13030434690527548
Validation MSE minimum: 0.04278607918110724
```



In [42]:

```
plt.plot(model_fit.history['loss'], label='Train loss')
#plt.plot(model_fit.history['val_loss'], label='Validation loss')
plt.legend()
print('Train MSE minimum:', min(model_fit.history['loss']))
#print('Validation MSE minimum:', min(model_fit.history['val_loss']))
```

```
Train MSE minimum: 0.1303059160709381
```

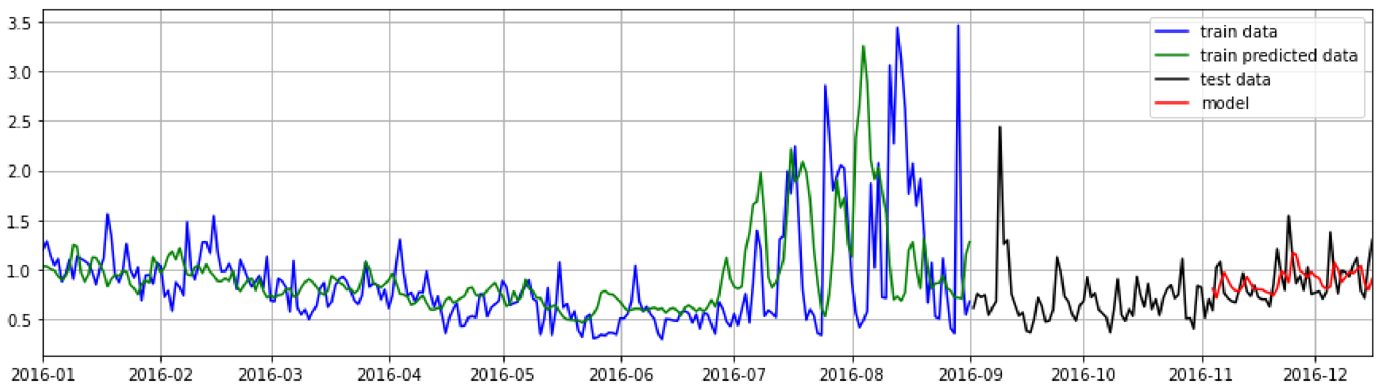


In [44]:

```
Train_pred = model.predict(X_train, verbose=0)
Y_pred = model.predict(X_test, verbose=0)
```

In [65]:
```
Y_pred_series = pd.Series(Y_pred.flatten().tolist(), index=data_daily[size+(np.abs(len(Y
Train_pred_series = pd.Series(Train_pred.flatten().tolist(), index=data_daily[:size].ind

#Plot
plt.figure(figsize=(15,4))
plt.plot(data_daily[:size], c='blue',label='train data')
plt.plot(Train_pred_series, c='green',label='train predicted data')
plt.plot(data_daily[size:], c='black',label='test data')
plt.plot(Y_pred_series, c='red', label='model')
plt.legend()
plt.grid(), plt.margins(x=0);

# calc error
print('MSE: %.5f' % (mean_squared_error(Y_pred, Y_test)))
print('RMSE: %.5f' % np.sqrt(mean_squared_error(Y_pred, Y_test)))
MAE = mean_absolute_error(Y_test, Y_pred)
MAPE = np.mean(np.abs(Y_pred - Y_test)/np.abs(Y_test))
print('MAE: %.3f' % MAE)
print('MAPE: %.3f' %MAPE)
print('MASE: %.3f' %MASE)
print('R^2 score: %.3f' % r2_score(Y_test, Y_pred))
```

```
MSE: 0.04671
RMSE: 0.21613
MAE: 0.170
MAPE: 0.203
MASE: 1.909
R^2 score: -0.099
```



In [ ]: