<u>BI Final Project</u>

Haaniyah Muhammad Mundia & Hurma Mehmood

Group Member	Name	ERP	Roles
1	Haaniyah Muhammad Mundia	14804	Feedback Colleague
2	Hurma Mahmood	14885	BI Analyst

Dataset: Smart Home Data

BI Tool: Tableau

Business Knowledge:

Since this data isn't of any business organization but of a smart home devices, there was not much available business knowledge available, the following information I found were through the different online dashboards available online.

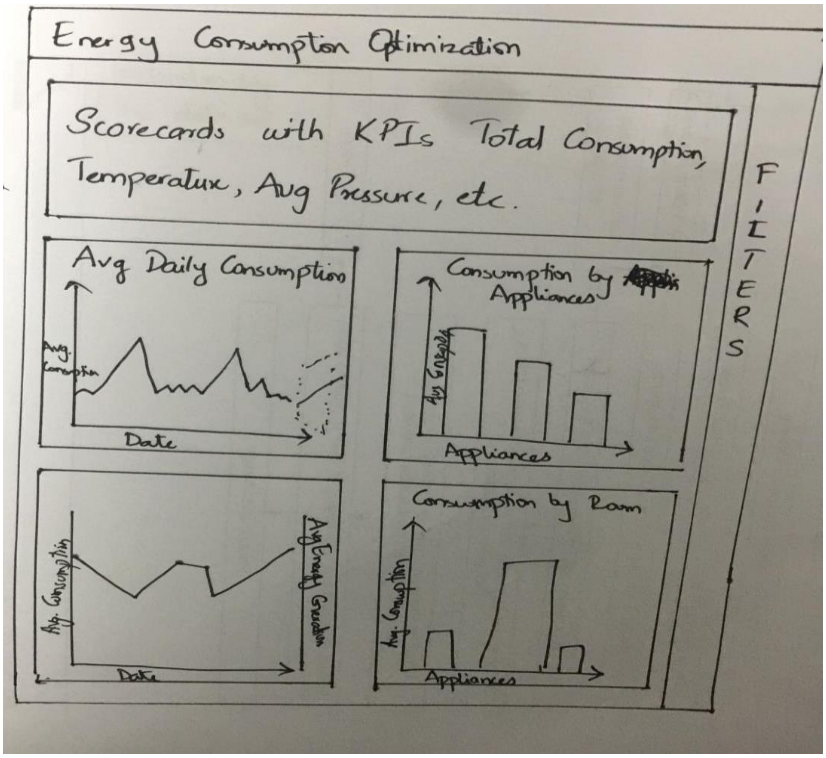
- A smart home is a modern home set up where all the room along with the appliance, devices even door are connected to an app/dashboards and they can be remotely controlled from anywhere through connecting internet using either your mobile or any other device.
- Power Consumed/Total Power: one the main KPI/metrics in any smart home application is to look for power consumed, this shows that total power that is consumed by the house as well as the appliances or devices.
- Power generated: In smart home, a power generation device is also setup which shows power generated per minute, here this metrics can help us record the power generated on average per day/minute/hour to help the user keep track for how much energy they are generating.
- Weather metric: Another thing that Smart home devices helps the user display is the weather in detail with precipitation, humidity, wind speed, temperature, etc.
- Spending/Energy Monitor: This metric displays a chart with energy consumed vs energy generated to compare if more energy is being used than generated and to help the user optimize their usage.
- Security: Some smart home dashboard provides the user with security function that locks their doors, or close garage or barn doors.

Problem:

“How can the overall energy consumption be optimized?”

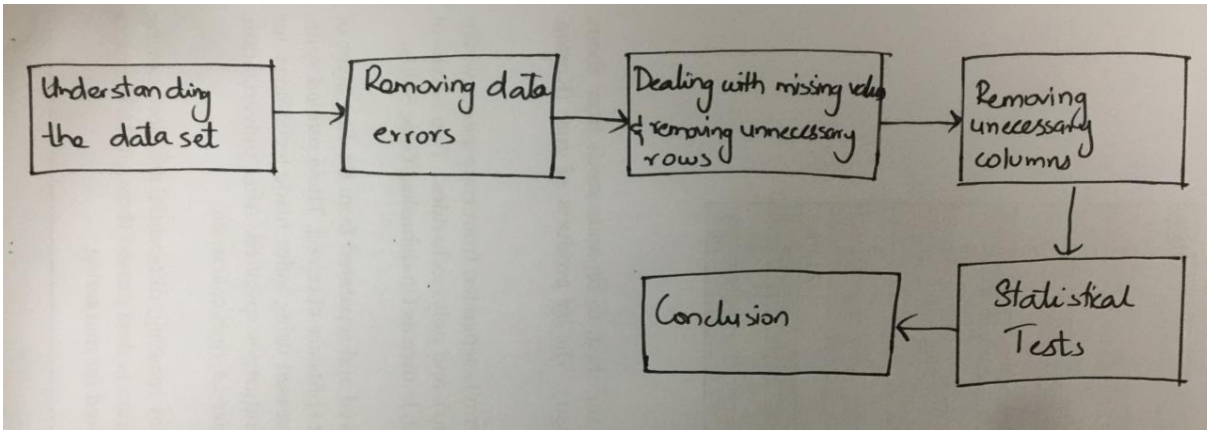
For any household, one of the major concerns is the energy consumption and its optimization. In order to achieve that one needs to monitor the energy consumed by each appliance and keep an eye on room-wise energy being utilized. Furthermore, the weather also plays a pivotal role in the usage of particular appliances. All these insights, together help determine the energy utilization patterns which in turn help in optimizing the energy usage.

BI Blueprint on paper:



- The scorecards on the top give concrete numbers of the total energy used and generated along with the weather details such as the temperature, pressure and precipitation probability.
- Next is the time series graph which gives the average daily consumption of the house highlighting the days when energy consumption was at its peak.
- The line chart below shows the energy used vs energy generated graph.
- Followed by a bar graph which gives a room wise energy consumption breakdown indicating the rooms that consume the most energy.
- Similarly, to the one above, the bar chart that follows gives an appliance wise energy consumption breakdown highlighting the appliance that uses the most energy.

Wrangling:



The wrangling notebook closely follows the wrangling pipeline shown. It is clearly divided amongst various headings with each step followed by a comment to explain the reason behind performing it. The major findings of wrangling are as follows:

- Important KPIs are: Energy Used and Energy Generated
- Important Dimensions are: Rooms, Appliances, Weather(temperature, precipitation, etc)

I've have explained each and every step in my jupyter notebook and their reasoning are mentioned in the wrangling notebook for which I'm attaching the snapshots below, which is attached as well is in the zip folder for this project.

jupyter SmartHomeDataset_FinalProject_HurmaMahmood (unsaved changes)   Logout

File Edit View Insert Cell Kernel Widgets Help   Not Trusted   Python 3

Run   Raw NBConvert

In [1]:
```python
#importing the libaries
import pandas as pd
import numpy as np
from matplotlib import pyplot as plt

from scipy.stats import shapiro
from scipy.stats import normaltest
from scipy.stats import anderson
from scipy.stats import chi2_contingency
from scipy.stats import chi2

import statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.multicomp import pairwise_tukeyhsd
from statsmodels.graphics.gofplots import qqplot

import seaborn as sns
sns.set()
```

In [2]:
```python
#import the data
home_df = pd.read_csv("HomeC.csv")
```

```
c:\python\python39\lib\site-packages\IPython\core\interactiveshell.py:3165: DtypeWarning: Columns (0,27) have mixed types.Sp
ecify dtype option on import or set low_memory=False.
  has_raised = await self.run_ast_nodes(code_ast.body, cell_name,
```

Understanding the dataset

In [3]:
```python
#printing top 5 rows of my dataframe to have an idea about the data
home_df.head()
```

Out[3]:

	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	apparentTemperature
0	1451624400	0.932833	0.003483	0.932833	0.000033	0.020700	0.061917	0.442633	0.124150	0.006983	...	10.0	Clear	29.26
1	1451624401	0.934333	0.003467	0.934333	0.000000	0.020717	0.063817	0.444067	0.124000	0.006983	...	10.0	Clear	29.26
2	1451624402	0.931817	0.003467	0.931817	0.000017	0.020700	0.062317	0.446067	0.123533	0.006983	...	10.0	Clear	29.26
3	1451624403	1.022050	0.003483	1.022050	0.000017	0.106900	0.068517	0.446583	0.123133	0.006983	...	10.0	Clear	29.26
4	1451624404	1.139400	0.003467	1.139400	0.000133	0.236933	0.063983	0.446533	0.122850	0.006850	...	10.0	Clear	29.26

5 rows × 32 columns

In [4]:
```
#printing the rows and columns of the data
home_df.shape
```
Out[4]:
```
(503911, 32)
```
In [5]:
```
#Printing all columns
home_df.columns
```
Out[5]:
```
Index(['time', 'use [kW]', 'gen [kW]', 'House overall [kW]', 'Dishwasher [kW]',
       'Furnace 1 [kW]', 'Furnace 2 [kW]', 'Home office [kW]', 'Fridge [kW]',
       'Wine cellar [kW]', 'Garage door [kW]', 'Kitchen 12 [kW]',
       'Kitchen 14 [kW]', 'Kitchen 38 [kW]', 'Barn [kW]', 'Well [kW]',
       'Microwave [kW]', 'Living room [kW]', 'Solar [kW]', 'temperature',
       'icon', 'humidity', 'visibility', 'summary', 'apparentTemperature',
       'pressure', 'windSpeed', 'cloudCover', 'windBearing', 'precipIntensity',
       'dewPoint', 'precipProbability'],
      dtype='object')
```
In [6]:
```
#Checking datatypes of the columns
home_df.dtypes
```
Out[6]:
```
time                    object
use [kW]               float64
gen [kW]               float64
House overall [kW]     float64
Dishwasher [kW]        float64
Furnace 1 [kW]         float64
Furnace 2 [kW]         float64
Home office [kW]       float64
Fridge [kW]            float64
Wine cellar [kW]       float64
Garage door [kW]       float64
Kitchen 12 [kW]        float64
Kitchen 14 [kW]        float64
Kitchen 38 [kW]        float64
Barn [kW]              float64
Well [kW]              float64
Microwave [kW]         float64
Living room [kW]       float64
Solar [kW]             float64
temperature            float64
icon                    object
humidity               float64
visibility             float64
summary                 object
apparentTemperature    float64
pressure               float64
```

```
apparentTemperature    float64
pressure               float64
windSpeed              float64
cloudCover              object
windBearing            float64
precipIntensity        float64
dewPoint               float64
precipProbability      float64
dtype: object
```

Removing Data Entry Errors

In [7]:

```
#As seen above the time column is an object whereas it should be in datetime.
#Dropping the original time column and replacing with a new one, Time Stamp, with the corrected datatype
time_index = pd.date_range('2016-01-01 05:00', periods=len(home_df),  freq='min')
time_index = pd.DatetimeIndex(time_index)
home_df['Time Stamp'] = time_index
home_df = home_df.drop(['time'], axis=1)
home_df.iloc[np.r_[0:5,-5:0]].iloc[:,0]
```

Out[7]:

```
0         0.932833
1         0.934333
2         0.931817
3         1.022050
4         1.139400
503906    1.599333
503907    1.924267
503908    1.978200
503909    1.990950
503910         NaN
Name: use [kW], dtype: float64
```

In [8]:

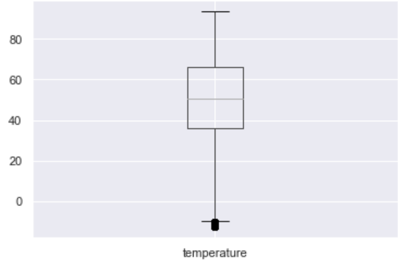
```
#Changing the datatypes of the column cloud cover to float
home_df['cloudCover'].replace(['cloudCover'], method='bfill', inplace=True)
home_df['cloudCover'] = home_df['cloudCover'].astype('float')
home_df['cloudCover'].unique()
```

Out[8]:

```
array([0.75, 0.  , 1.  , 0.31, 0.44, 0.13, 0.19, 0.25, 0.16, 0.21, 0.15,
       0.14, 0.27, 0.28, 0.17, 0.05, 0.1 , 0.26, 0.29, 0.11, 0.09, 0.12,
       0.06, 0.02, 0.08, 0.04, 0.35, 0.22, 0.23, 0.54, 0.39, 0.03, 0.07,
       0.76, 0.62, 0.18, 0.79, 0.48, 0.24, 0.57, 0.41, 0.78, 0.2 , 0.77,
       0.46, 0.55, 0.01, 0.51, 0.47, 0.5 , 0.4 , 0.3 , 0.43, 0.33, 0.6 ,
       0.68, 0.66, 0.45, 0.34, 0.52, 0.67, 0.49, 0.37, 0.36, 0.61, 0.38,
       0.42, 0.53, 0.63, 0.32, 0.56, 0.58, 0.72, 0.73, 0.71, 0.64, 0.59,
        nan])
```

In [10]:
```
#Plotting box plot to check for outliers
home_df.boxplot(column='temperature', sym='o', return_type='axes')
```

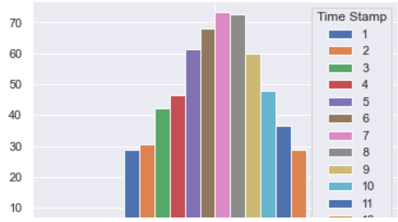
Out[10]: <AxesSubplot:>

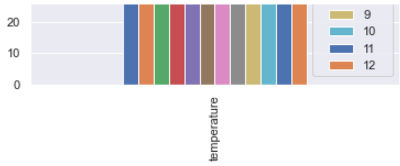


In [30]:
```
#Finding the average monthly temperature value
Month = []
Month = home_df['Time Stamp'].apply(lambda x: x.month)
avg_monthly_temp = home_df.pivot_table(columns=Month, values='temperature', aggfunc='mean').round(2)
print(avg_monthly_temp)
avg_monthly_temp.plot.bar()
```

```
Time Stamp      1      2      3      4     5      6      7      8      9  \
temperature  28.72  30.47  42.42  46.51  61.4  68.05  73.18  72.56  59.96

Time Stamp      10     11     12
temperature  47.92  36.55  28.67
```

Out[30]: <AxesSubplot:>





The above two plots are used to identify and understand the temperature variation throughout the year. The first one i.e. the box plot presents the 5 number summary of the temperature. The second one i.e. the bar chart finds out the trends in temperature over the months. It can be seen that the May-September is the summer season and it is warmer while in the remaining months are winters.

Dealing with Missing values and removing unnecessary columns and rows

In [12]:
```
#Dropping the Furnace 1 and Furnace 2 column and replacing it with the aggregated Furnace column
home_df['Furnace'] = home_df[['Furnace 1 [kW]','Furnace 2 [kW]']].sum(axis=1)

home_df = home_df.drop(['Furnace 1 [kW]','Furnace 2 [kW]'], axis=1)
```

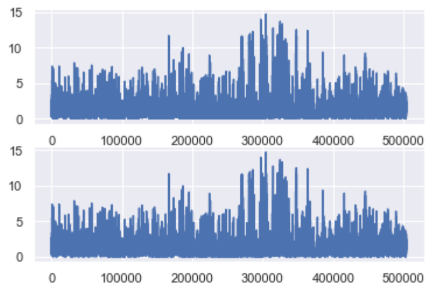
In [13]:
```
#Dropping the Kitchen 12, Kitchen 14 and Kitchen 38 column and replacing it with the sum and avg Kitchen column
home_df['sum_Kitchen'] = home_df[['Kitchen 12 [kW]','Kitchen 14 [kW]','Kitchen 38 [kW]']].sum(axis=1)

home_df = home_df.drop(['Kitchen 12 [kW]','Kitchen 14 [kW]','Kitchen 38 [kW]'],axis=1)
```

In [14]:
```
home_df.columns
```

Out[14]:
```
Index(['use [kW]', 'gen [kW]', 'House overall [kW]', 'Dishwasher [kW]',
       'Home office [kW]', 'Fridge [kW]', 'Wine cellar [kW]',
       'Garage door [kW]', 'Barn [kW]', 'Well [kW]', 'Microwave [kW]',
       'Living room [kW]', 'Solar [kW]', 'temperature', 'icon', 'humidity',
       'visibility', 'summary', 'apparentTemperature', 'pressure', 'windSpeed',
       'cloudCover', 'windBearing', 'precipIntensity', 'dewPoint',
       'precipProbability', 'Time Stamp', 'Furnace', 'sum_Kitchen'],
      dtype='object')
```

In [15]:
```
#Analyzing the trends in Total energy used and the overall house energy
fig, axes = plt.subplots(nrows=2, ncols=1)
home_df['use [kW]'].plot(ax=axes[0])
home_df['House overall [kW]'].plot(ax=axes[1])
```

Out[15]: <AxesSubplot:>



In [16]:

```
#checking if both the columns are identicial
home_df['use [kW]'].equals(home_df['House overall [kW]'])
```

Out[16]:

```
True
```

The above two plots were plotted to observe the trends in the values of energy used and the overall house energy used. It can be seen that both follow the exact same pattern. This observation was further confirmed by using the .equals function, which confirms that both the columns are identical. Hence dropping the House overall column to reduce redundancy.

In [17]:

```
#Dropping House overall column for the above mentioned reason.
home_df = home_df.drop(['House overall [kW]'], axis=1)
```

In [18]:

```
home_df.isnull().sum()
```

Out[18]:

```
use [kW]               1
gen [kW]               1
Dishwasher [kW]        1
Home office [kW]       1
Fridge [kW]            1
Wine cellar [kW]       1
Garage door [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
Time Stamp             0
Furnace                0
sum_Kitchen            0
dtype: int64
```

In [19]:

```
#Dropping the missing rows since there is only 1 row with missing values
home_df = home_df.drop(home_df[home_df['use [kW]'].isnull()].index)
```


In [19]:

```
#Dropping the missing rows since there is only 1 row with missing values
home_df = home_df.drop(home_df[home_df['use [kW]'].isnull()].index)
```


In [20]:

```
home_df.isnull().sum()
```

Out[20]:

```
use [kW]               0
gen [kW]               0
Dishwasher [kW]        0
Home office [kW]       0
Fridge [kW]            0
Wine cellar [kW]       0
Garage door [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
Time Stamp             0
Furnace                0
sum_Kitchen            0
dtype: int64
```

Anova Test

Here I'm conducting anova test on my two main KPIs i.e., Used, Generated and since anova is between one categorical and one numercial column I'll be conducting it with the categorical column Icon and summary which are the only other categorical column in this dataset.

In [21]:

```
y = home_df['use [kW]']
model = ols('y ~ C(Q("icon"))', data=home_df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Used [KW] - Icon")
display(anova_table)

model = ols('y ~ C(Q("summary"))', data=home_df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Used [KW] - summary")
display(anova_table)
```

```
Anova => Used [KW] - Icon
```

	sum_sq	df	F	PR(>F)
C(Q("icon"))	497.184470	8.0	55.547171	6.515651e-91
Residual	563781.530931	503901.0	NaN	NaN

```
Anova => Used [KW] - summary
```

	sum_sq	df	F	PR(>F)
C(Q("summary"))	1462.918537	17.0	77.044636	8.493588e-268
Residual	562815.796864	503892.0	NaN	NaN

From the above results, it can be said that since the p-value is less than 5% for both Icon and Summary with Used, we reject the null hypothesis and conclude that there us a significant difference between thier means.

In [22]:

```
y = home_df['gen [kW]']
model = ols('y ~ C(Q("icon"))', data=home_df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Generated [KW] - Icon")
display(anova_table)

model = ols('y ~ C(Q("summary"))', data=home_df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
print ("\nAnova => Generated [KW] - summary")
display(anova_table)
```


model = ols('y ~ C(Q("summary"))', data=home_df).fit()
anova_table = sm.stats.anova_lm(model, typ=2)


```
Anova => Generated [KW] - Icon
```

	sum_sq	df	F	PR(>F)
C(Q("icon"))	25.483047	8.0	193.718321	0.0
Residual	8285.827508	503901.0	NaN	NaN

```
Anova => Generated [KW] - summary
```

	sum_sq	df	F	PR(>F)
C(Q("summary"))	32.160311	17.0	115.139148	0.0
Residual	8279.150244	503892.0	NaN	NaN

From the above results, it can be said that since the p-value is less than 5% for both Icon and Summary with Gen, we reject the null hypothesis and conclude that there us a significant difference between thier means.

Chi-squared test

Since chi squared test can only be performed on categorical columns so here I'm using the only two categorical columns availbe in this datast i.e., summary and icon

In [23]:

```
from scipy.stats import chi2_contingency
from scipy.stats import chi2

data_crosstab = pd.crosstab(home_df['icon'], home_df['summary'],
margins = False)
print(data_crosstab)

stat, p, dof, expected = chi2_contingency(data_crosstab)
print('dof=%d' % dof)
print(expected)

# interpret p-value
alpha = 0.05
print('significance=%.3f, p=%.3f' % (alpha, p))
if p <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (fail to reject H0)')
```

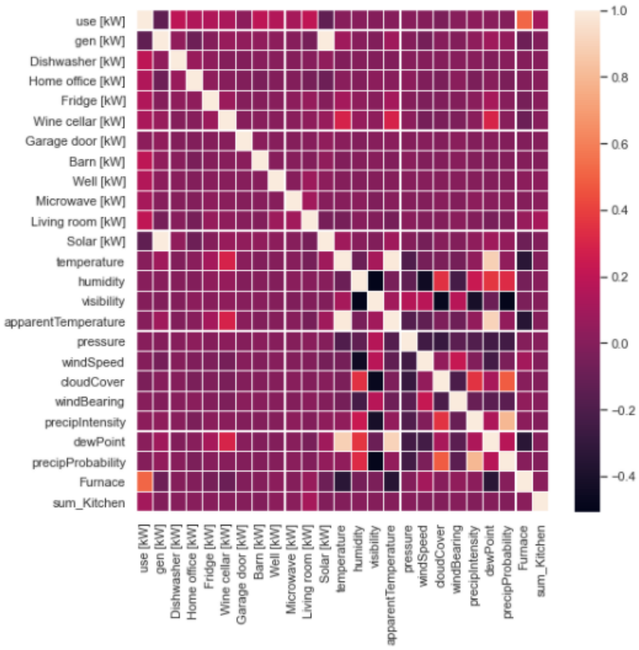
```
  2.46610609e+02 3.27566994e+02 3.37641807e+03 2.80283694e+02
  9.43497053e+00 6.24660118e+01]
 [1.33455262e+02 4.95861959e+00 8.89986724e+01 3.22079441e+04
  8.86566986e+02 4.95861959e+00 1.52947766e+02 9.83174575e+00
  8.32706118e+01 1.46193785e+01 2.33978450e+03 3.69588147e+02
  3.88824171e+02 5.16465879e+02 5.32350560e+03 4.41915598e+02
  1.48758588e+01 9.84884444e+01]
 [2.33882042e+01 8.69004386e-01 1.55971304e+01 5.64448314e+03
  1.55371991e+02 8.69004386e-01 2.68042904e+01 1.72302594e+00
  1.45932805e+01 2.56206465e+00 4.10050207e+02 6.47707924e+01
  6.81419301e+01 9.05113016e+01 9.32951122e+02 7.74462702e+01
  2.60701316e+00 1.72602250e+01]
 [8.24008255e+00 3.06165784e-01 5.49514794e+00 1.98865234e+03
  5.47403306e+01 3.06165784e-01 9.44363081e+00 6.07052847e-01
  5.14147368e+00 9.02661189e-01 1.44468020e+02 2.28199083e+01
  2.40076204e+01 3.18887500e+01 3.28695362e+02 2.72857058e+01
  9.18497351e-01 6.08108591e+00]]
significance=0.050, p=0.000
Dependent (reject H0)
```

From the above results, it can be said that since the p-value is less than 5%, therefore we reject the null hypothesis and conclude that there us a significant difference between thier means.

Corelation Matrix

In [26]:
```python
#let us check the correlation of all numerical columns with each other
corrmatrix = home_df.corr()
f, axis = plt.subplots(figsize =(8, 8))
sns.heatmap(corrmatrix, ax = axis, linewidths = 0.2)
```

Out[26]: <AxesSubplot:>

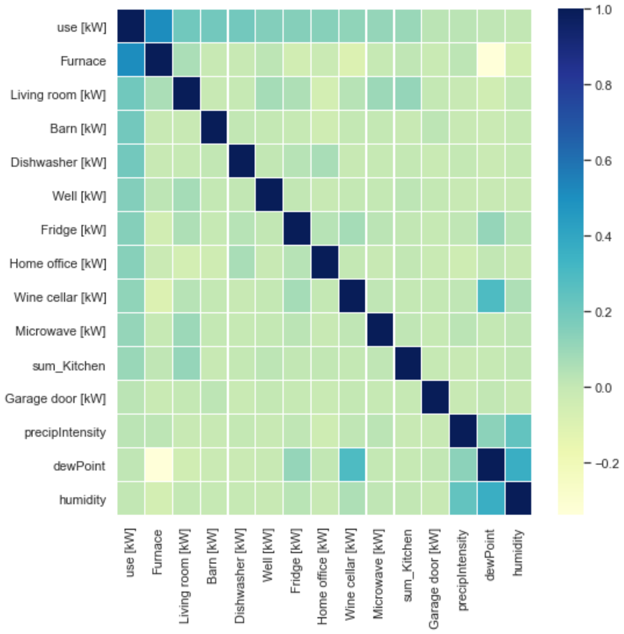


This heatmap shows the correlation of all numerical values with each other. From this heatmap we can clearly identify that energy generated is highly positively correlated with Solar. Apart from that no major consclusions can be drawn due to the large number of columns which causes difficulty in finding relations.

In [27]:

```python
# let us plot a correlation matrix only for Use (v vars with largest correlation with grand total)
v= 15

cols = corrmatrix.nlargest(v, 'use [kW]')['use [kW]'].index

coeffs = np.corrcoef(home_df[cols].values.T)
f, axis = plt.subplots(figsize =(8, 8))

sns.heatmap(coeffs, ax = axis, cmap ="YlGnBu",
            linewidths = 0.2, yticklabels = cols.values,
                              xticklabels = cols.values)
```

Out[27]: <AxesSubplot:>

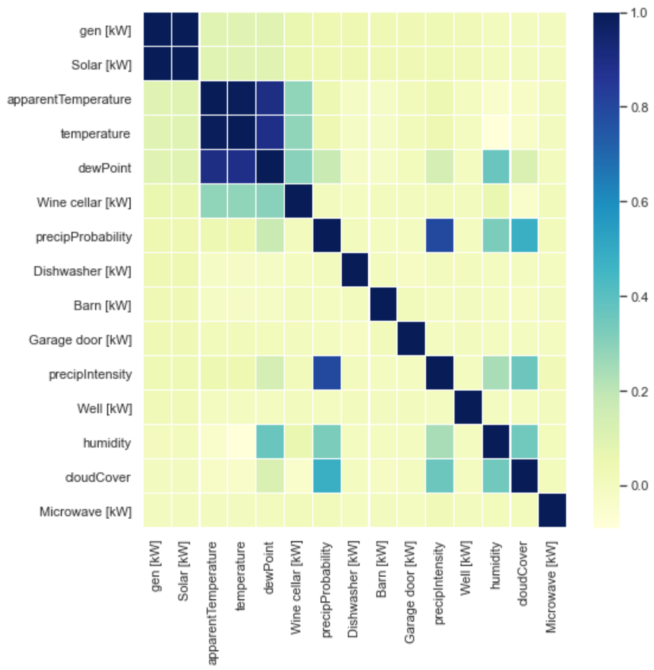


It can be clearly seen here that the most energy that is being used is by the Furnace as it is highly positively related with energy used

In [28]:

```
# let us plot a correlation matrix only for saleprice (v vars with largest correlation with grand total)
v=15

cols = corrmatrix.nlargest(v, 'gen [kW]')['gen [kW]'].index

coeffs = np.corrcoef(home_df[cols].values.T)
f, axis = plt.subplots(figsize =(8, 8))

sns.heatmap(coeffs, ax = axis, cmap ="YlGnBu",
            linewidths = 0.2, yticklabels = cols.values,
                             xticklabels = cols.values)
```

Out[28]: <AxesSubplot:>

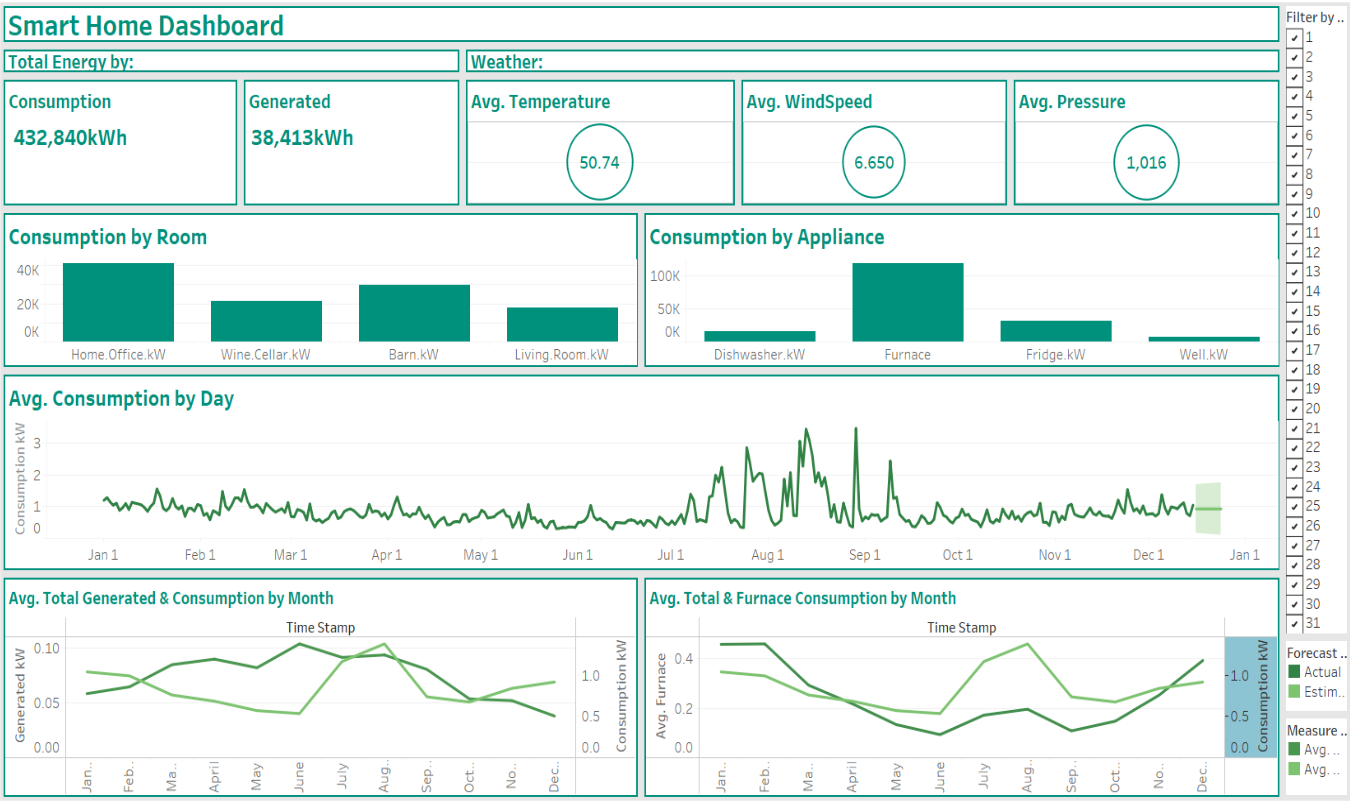


This heatmap shows that generation of energy is more dependant upon the weather conditions majorly the solar energy which again makes sense as well.

In [29]:

```
home_df.to_csv('home_data.clean.csv')
```

BI Dashboard:



The above dashboard effectively solves the problem as stated in the above problem statement i.e. “How can the overall energy consumption be optimized?”

The dashboard starts off with scorecards displaying the KPIs i.e. the total energy consumed and generated. This gives solid numbers about energy usage and generated in the year.

This is followed by scorecards that present a weather analysis i.e. average temperature, pressure and windspeed throughout the year, presenting an overall weather analysis of the year and the general climate of the area.

This is followed by bar charts that give room wise and application wise breakdown of the energy consumption, respectively. This presents insights about which room and which appliance is consuming the most energy hence allowing to identify the rooms and appliances, eventually finding the root cause of increased consumption, which can then be optimized accordingly.

Next, is a time series chart of the overall energy consumed on a day-to-day basis. This shows the trends of the daily energy consumption and allows to identify a specific day when more energy was consumed, which can further be drilled down to that day and energy consumption on that day can be analyzed hence optimized accordingly.

Furthermore, the next graph presents a comparison of the monthly average energy consumed vs the average energy generated. Looking at the trend it can be deduced that in winter (from months November to February) there is a clear pattern that the energy consumed in much greater than the energy generated. This greater consumption of energy can be due to the fact that heating appliance such as furnace are used more in winter which can be seen in the next chart as well as the pervious one.

Lastly, there is a line graph that depicts the monthly average energy used overall and the average energy used by the most energy consuming appliance i.e. the furnace. In line with the temperature trends understood in the wrangling phase, it can be seen that in the cooler months i.e. from October to February the energy consumed by the furnace starts to increase. Furthermore, it also shows that in summers due to the heat average energy consumed by the furnace is much less than that consumed overall in the house for instance in the month of August average energy consumed by the furnace is 0.1955 kW while that by the house is 1.3841.

With this dashboard, the user can easily keep a track of how much energy they are consuming on average, and the avg energy generated can help them keep a track especially on which months should they reduce/lower or control their energy consumption with this not only will they have smart home device but smart energy consumption.

Stakeholder's Contribution:

- Haaniyah suggested to make the Weather and the Energy Scorecards distinct by adding a heading for each of them so that both the KPIs are clearly visible and identifiable as different entities.
- She also suggested about changing the background color so as to enhance my charts and bring more focus on the charts and keeping the background a light shade such as grey.
- She also advised me to add the consumption vs generated chart on the dashboard.
- Haaniyah helped me identify via charts and functions that the use and House overall used columns were identical in the wrangling phase.
- She also helped since the data was quite tricky choose the best possible KPI i.e. daily energy use consumption, for time series analysis.

My Contribution as a stakeholder in Haaniyah's Project:

- I suggested to Haaniyah to Keep the KPI scorecards along with the filter on top of the dashboard so it grabs the attention of the user at once.
- In the wrangling phase, Haaniyah was a bit confused about the concept of the two statistical tests, I explained the basic concepts of Anova and chi test an as to why a BI analyst should use them in the wrangling phase.
- I advised Haaniyah on her color palette i.e. to keep a deep blue color in her overall dashboard instead of a medium blue so that it is easy on the eyes for the user and

show the negative or ‘bad’ values red in color so that there is a distinction between these two.

- Also, Haaniyah has used a normal gauge showing the number of orders completed and cancelled which did not help her display what information she was trying to convey so I suggested her to use a tachometer which can help her show the two values in different colors.
- Since, Haaniyah had two main KPIs one of which was order quantity, I suggested to her to add two bar charts in her dashboard which shows the top product category as well as the payment method w.r.t order quantity so that it is easier to understand the consumer behavior.