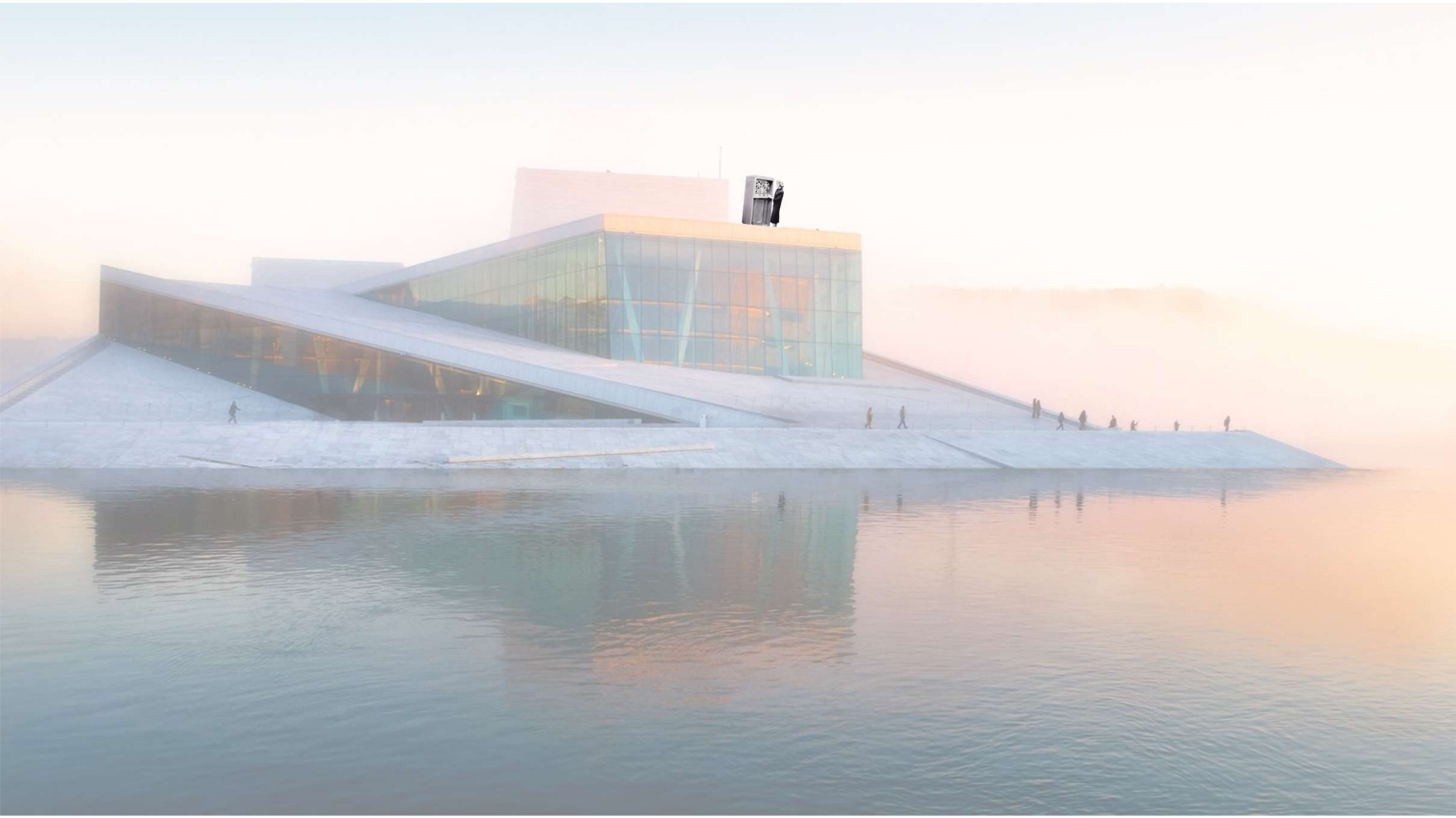
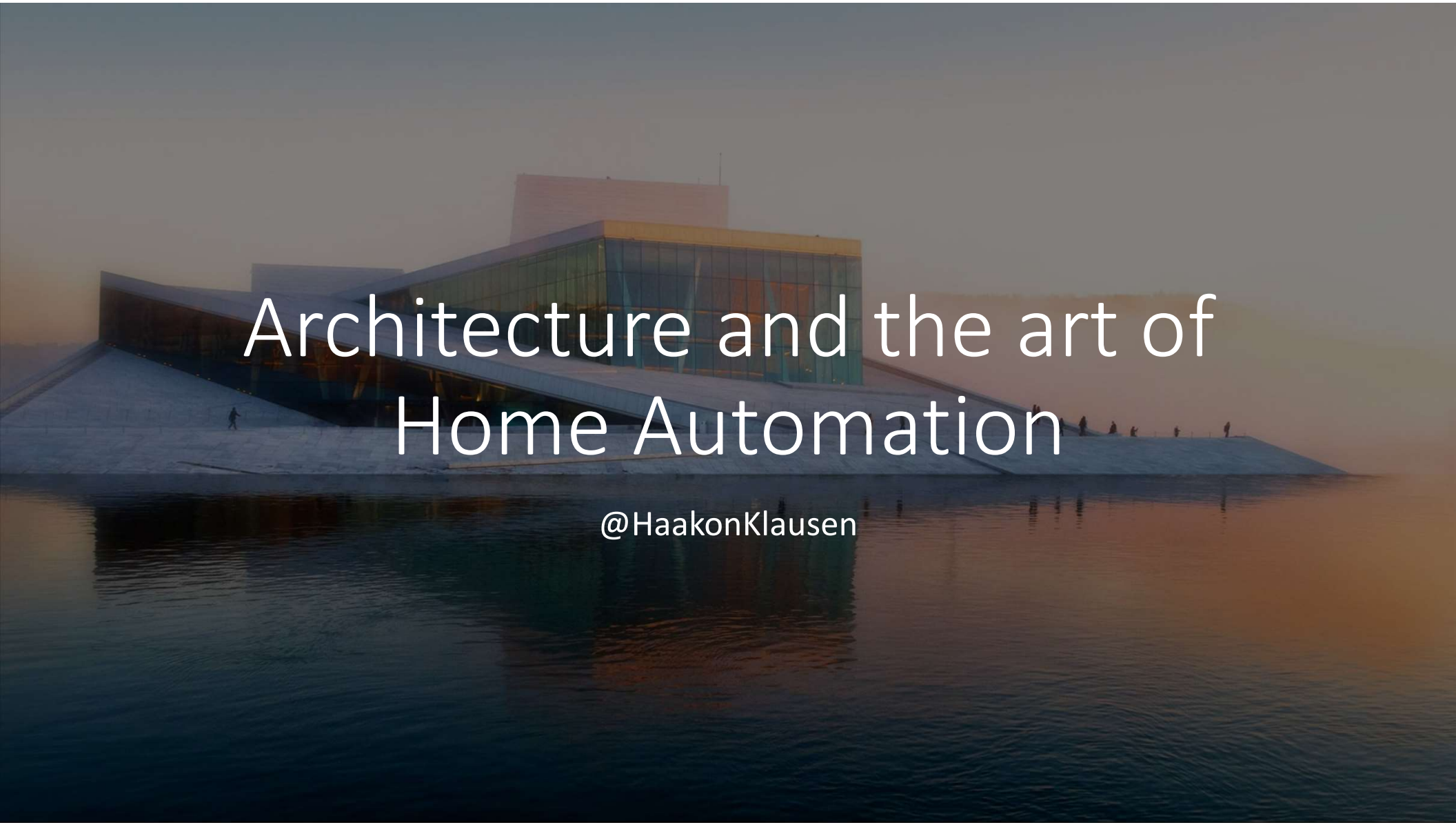
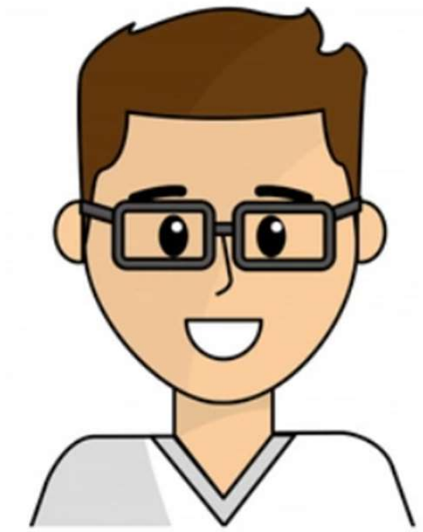




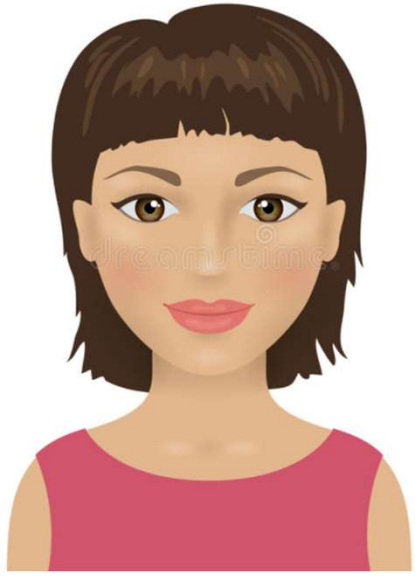
Architecture and the art of Home Automation

@HaakonKlausen

Where do I look for the Outside Temp?

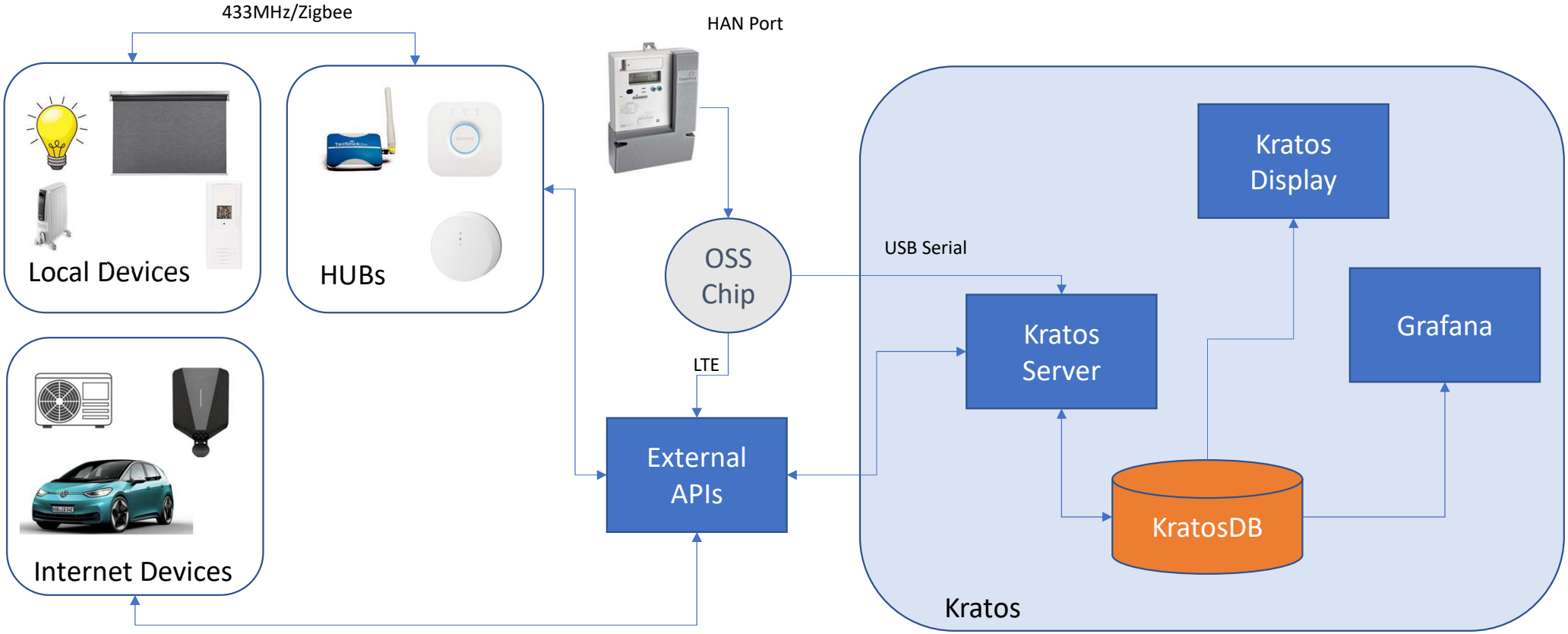


I HATE IT!



Vision:
The Switch-free house

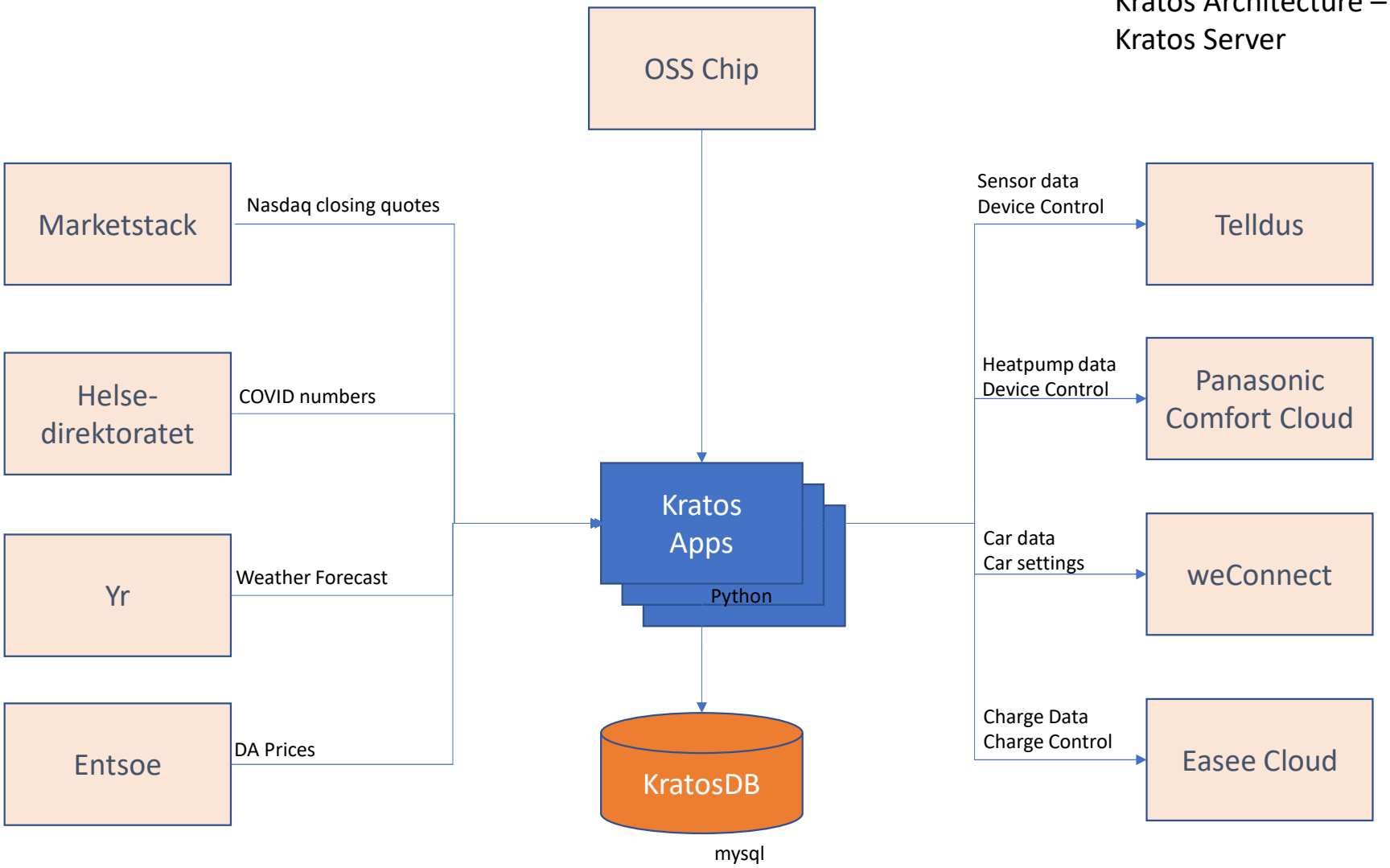
Kratos Architecture – Level 1



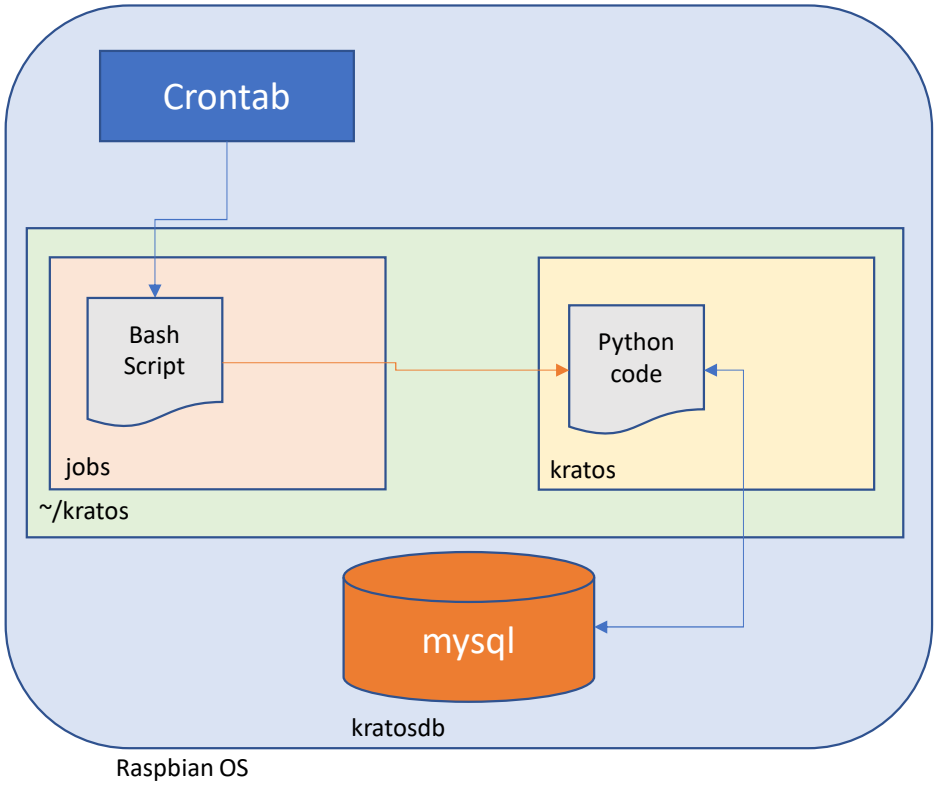
Architectural Principle:
Connect devices through external APIs

https://github.com/HaakonKlausen/kratos

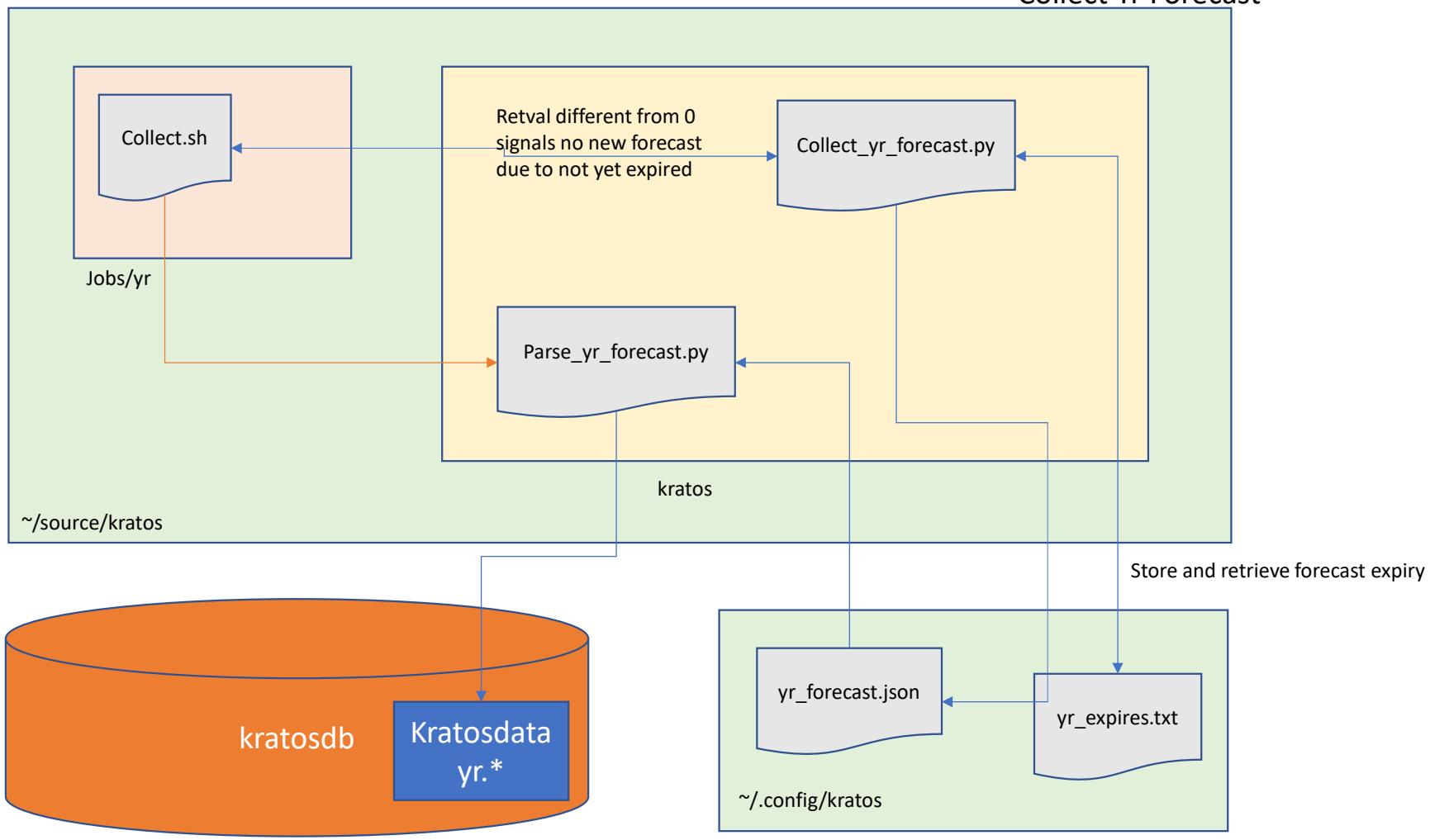
Kratos Architecture – Level 2
Kratos Server



Kratos Architecture – Level 3
Kratos App Principles



Kratos Architecture – Level 4
Collect Yr Forecast



Kratos Architecture – Infrastructure

Kratos Architecture – Infrastructure

OSS Chip

Raspberry Pi 4B
192.168.1.181

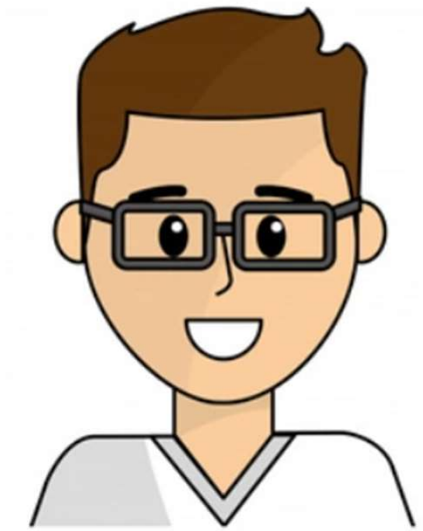
USB HUB

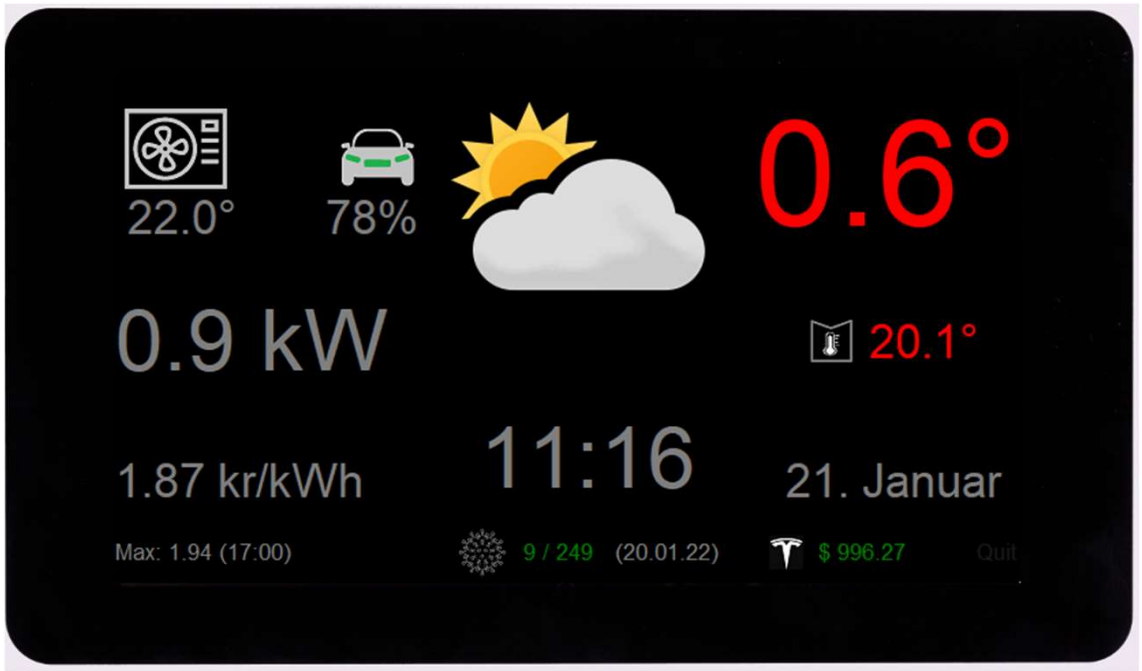
1 TB SSD

1 TB HDD

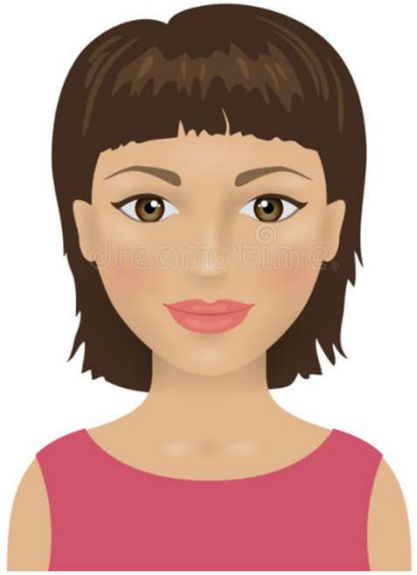
Raspberry Pi 3
7» Touch Screen
192.168.1.19

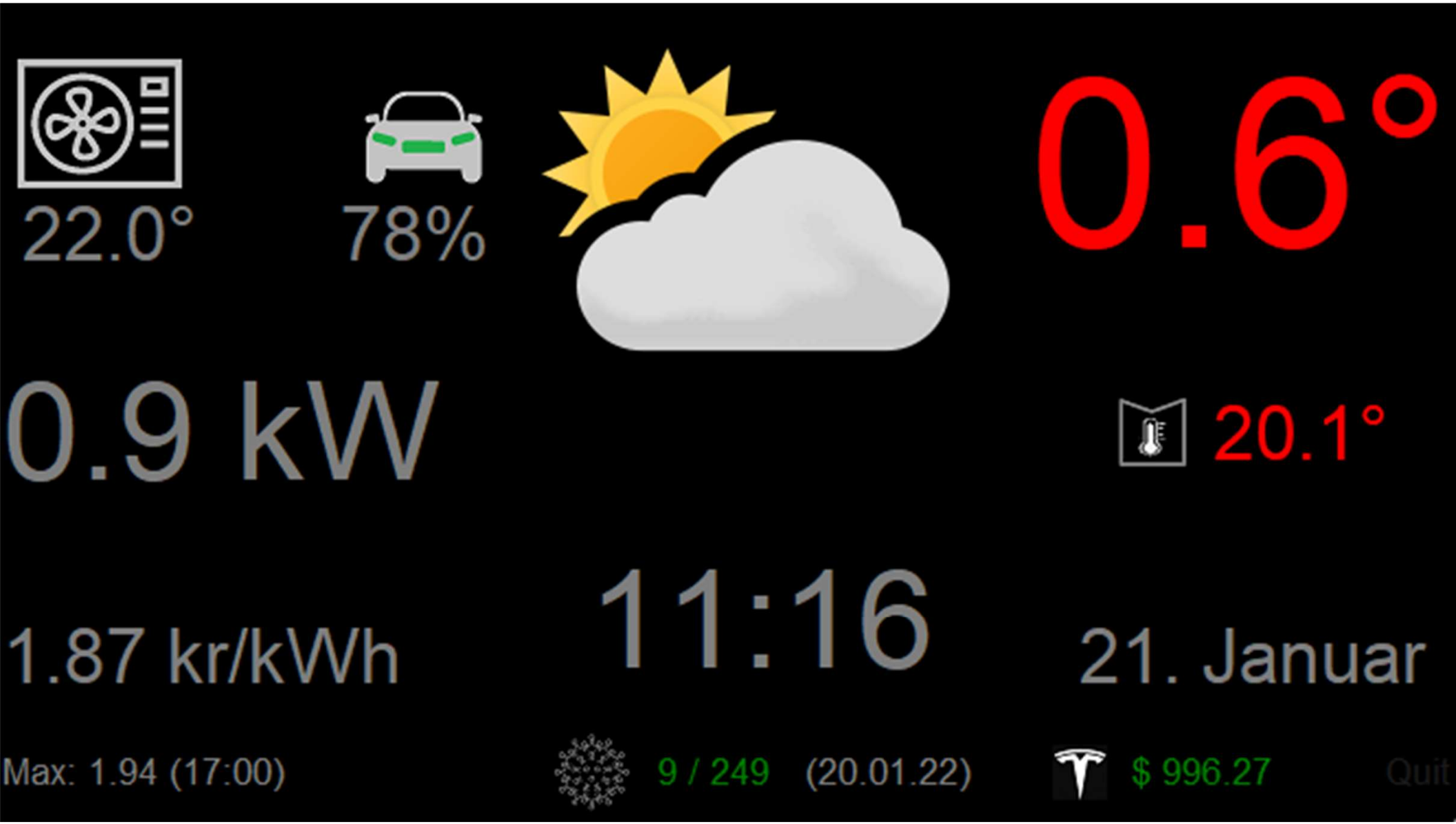
What more can I do here?





He is so clever!




22.0°
78%
0.6°
0.9 kW
20.1°
1.87 kr/kWh
11:16
21. Januar
Max: 1.94 (17:00)
9 / 249
(20.01.22)
$ 996.27
Quit

Kamstrup Binary Package

b'~\xa0\xe2+!\x13#\x9a\xe6\xe7\x00\x0f\x00\x00\x00\x00\x0c\x07\xe6\x02\t\x03\x14$(\xff\x80\x00\x0
0\x02\x19\r\n\x0eKamstrup_V0001\t\x06\x01\x01\x00\x00\x05\xff\r\n\x10570656727151107 6\t\x06\x
01\x01`\x01\x01\xff\r\n\x126841131BN243101040\t\x06\x01\x01\x01\x07\x00\xff\x06\x00\x00\x0c\x9
1\t\x06\x01\x01\x02\x07\x00\xff\x06\x00\x00\x00\x00\t\x06\x01\x01\x03\x07\x00\xff\x06\x00\x00\x0
0\x99\t\x06\x01\x01\x04\x07\x00\xff\x06\x00\x00\x00\x00\t\x06\x01\x01\x1f\x07\x00\xff\x06\x00\x00
\x02\x81\t\x06\x01\x013\x07\x00\xff\x06\x00\x00\x00\xcf\t\x06\x01\x01G\x07\x00\xff\x06\x00\x00\x0
2\xc7\t\x06\x01\x01
\x07\x00\xff\x12\x00\xe9\t\x06\x01\x014\x07\x00\xff\x12\x00\xea\t\x06\x01\x01H\x07\x00\xff\x12\x0
0\xe9E\xf7~\r\n'

1 1 1 7 0 255: Next 4 bytes are Active Power
1 1 1 8 0 255: Next 4 bytes are Active Energy (One such every hour)

Oss Norway has 3 year old Node.js example code for parsing, but it´s not updated and does not work

```
ser = serial.Serial('/dev/ttyUSB0', timeout=None, baudrate=115000, xonxoff=False, rtscts=False, dsrdtr=False)
```

Brute force decoding

```
while True:
    messages = messages + 1
    try:
        bytesToRead = ser.inWaiting()
        if bytesToRead > 0:
            data_raw = ser.read(bytesToRead)
            if bytesToRead > 2:
                for i in range (0, bytesToRead - 9):
                    parse_message(i)
        time.sleep(1)
```

The package is structured so it can be parsed, but I did not bother.

There are the same type of codes for Voltage and Current for all 3 phases

```
def parse_message(start_pos):
    message=''
    for i in range (0, 6):
        message = message + '.' + str(data_raw[start_pos + i])
    subtype = str(data_raw[start_pos + 7])
    if message == '.1.1.1.7.0.255':
        kratoslib.writeKratosData('oss.active_power', str(readUIntBE(start_pos+7, 4)))
        kratoslib.writeTimeseriesData('oss.active_power', float(str(readUIntBE(start_pos+7, 4))))
    if message == '.1.1.1.8.0.255':
        active_energy=float(str(readUIntBE(start_pos+7, 4))) / 100
        kratoslib.writeKratosData('oss.active_energy', str(active_energy))
```

Yr.no

To be a good citizen as an Yr client, you need to

- Send a User-Agent header with app name and email address
- Respect the Expires header and not call the API again until after that time

01d.png

Weather symbols are open source on github, but the cross reference between them is not. Forecast contains for instance the string «clearsky_day», the image is 01d.png.

```
symbol_codes = {'clearsky': '01',
                'cloudy': '04','fair': '02','fog': '15','heavyrain': '10','heavyrainandthunder':'11','heavyrainshowers':'41',
                'heavyrainshowersandthunder': '11', 'heavysleet': '48', 'heavysleetandthunder': '32', 'heavysleetshowers': '43',
                'heavysleetshowersandthunder': '27', 'heavysnow': '50', 'heavysnowandthunder': '27', 'heavysnowshowers': '45',
                'heavysnowshowersandthunder': '29', 'lightrain': '46', 'lightrainandthunder': '30', 'lightrainshowers': '40',
                'lightrainshowersandthunder': '24', 'lightsleet': '47', 'lightsleetandthunder': '31', 'lightsleetshowers': '42',
                'lightsnow': '49', 'lightsnowandthunder': '33', 'lightsnowshowers': '44', 'lightssleetshowersandthunder': '26',
                'lightssnowshowersandthunder': '28', 'partlycloudy': '03', 'rain': '09', 'rainandthunder': '22', 'rainshowers': '05',
                'rainshowersandthunder': '06', 'sleet': '12', 'sleetandthunder': '22', 'sleetshowers': '07', 'sleetshowersandthunder': '20',
                'snow': '13', 'snowandthunder': '14', 'snowshowers': '08', 'snowshowersandthunder': '21'}
```

Entsoe for Dayahead prices

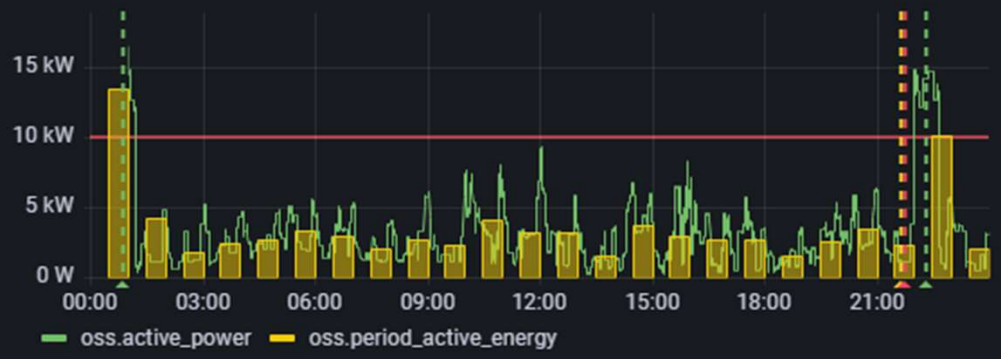
Need to send a support ticket to them to get access to the API – it´s free with some limitations. On the web page, you can then get the Security Token for the API call:

```
params = urllib.parse.urlencode({
    'documentType': 'A44',
    'in_Domain': '10YNO-2--------T',
    'out_Domain': '10YNO-2--------T',
    'periodStart': tomorrow_period + '0000',
    'periodEnd': tomorrow_period + '2300',
    'securityToken': config['api_token']
})
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Publication_MarketDocument xmlns="urn:iec62325.351:tc57wg16:451-3:publicationdocument:7:0">
  <mRID>d328774472f8441c9a4f607e3dea01a2</mRID>
  <revisionNumber>1</revisionNumber>
  <type>A44</type>
  <sender_MarketParticipant.mRID codingScheme="A01">10X1001A1001A450</sender_MarketParticipant.mRID>
  <sender_MarketParticipant.marketRole.type>A32</sender_MarketParticipant.marketRole.type>
  <receiver_MarketParticipant.mRID codingScheme="A01">10X1001A1001A450</receiver_MarketParticipant.mRID>
  <receiver_MarketParticipant.marketRole.type>A33</receiver_MarketParticipant.marketRole.type>
  <createdDateTime>2022-02-09T12:15:02Z</createdDateTime>
  <period.timeInterval>
    <start>2022-02-09T23:00Z</start
    <end>2022-02-10T23:00Z</end>
  </period.timeInterval>
  <TimeSeries>
    <mRID>1</mRID>
    <businessType>A62</businessType
    <in_Domain.mRID codingScheme="A
    <out_Domain.mRID codingScheme="
    <currency_Unit.name>EUR</curren
    <price_Measure_Unit.name>MWH</price_Measure_Unit.name>
    <curveType>A01</curveType>
    <Period>
      <timeInterval>
        <start>2022-02-09T23:00Z</start>
        <end>2022-02-10T23:00Z</end>
      </timeInterval>
      <resolution>PT60M</resolution>
      <Point>
        <position>1</position>
        <price.amount>117.23</price.amount>
      </Point>
      <Point>
        <position>2</position>
        <price.amount>116.84</price.amount>
      </Point>
```

```python
tree = ET.parse(kratoslib.getKratosConfigFilePath('da_forecast.xml'))
hour = int(datetime.datetime.now().strftime('%H'))
root = tree.getroot()
cursor=connection.cursor()
for period in range(24):
    print(root[9][7][period + 2][0].text, root[9][7][period + 2][1].text)
    period_data = (date, int(root[9][7][period + 2][0].text) - 1, root[9][7][period + 2][1].text)
```

Effekt og forbruk



Strømpriser

Aksjekurser

Elbil Statistikk

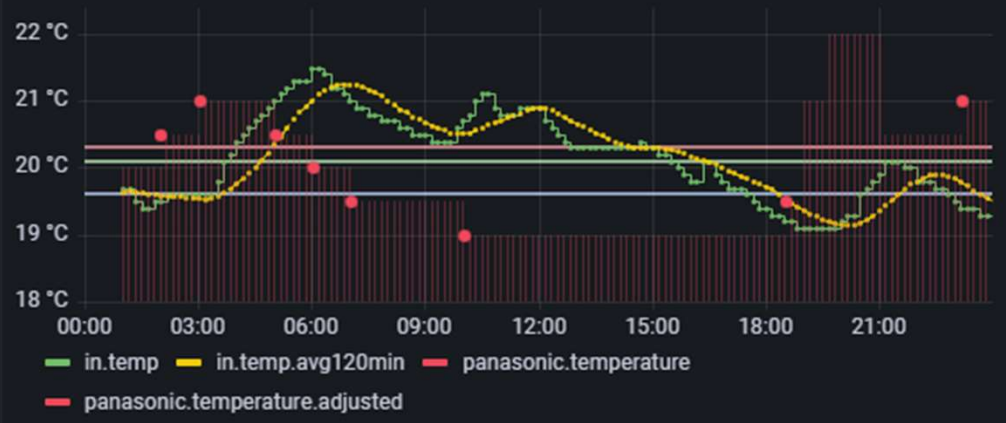
ID.3 S...

ID.3 R...

TSLA

996

Temperatur Inne



Temperatur Ute