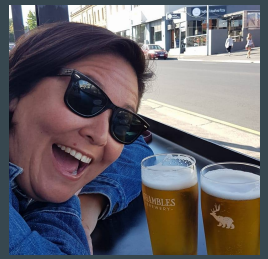


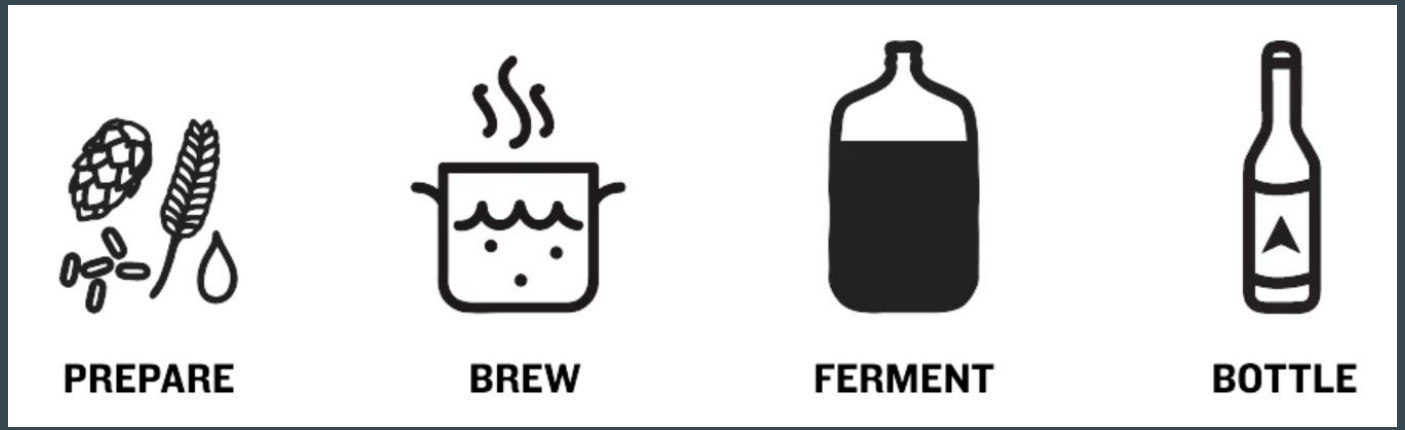
Sensors, Beers and Nerves

...

FIONA MCCAWLEY

GITHUB.COM/FIMAC

@SAUCERLIKE


PREPARE
BREW
FERMENT
BOTTLE

Goals

- See a history of temperature readings over time.
- Place the sensor in any location at home and get readings wirelessly.
- Trigger a ~~Solid State Relay and cooling/heating device~~ LED based on a temperature point.

Getting started....

Raspberry Pi Zero W

Bosch BME280 sensor

MicroSD card + Reader

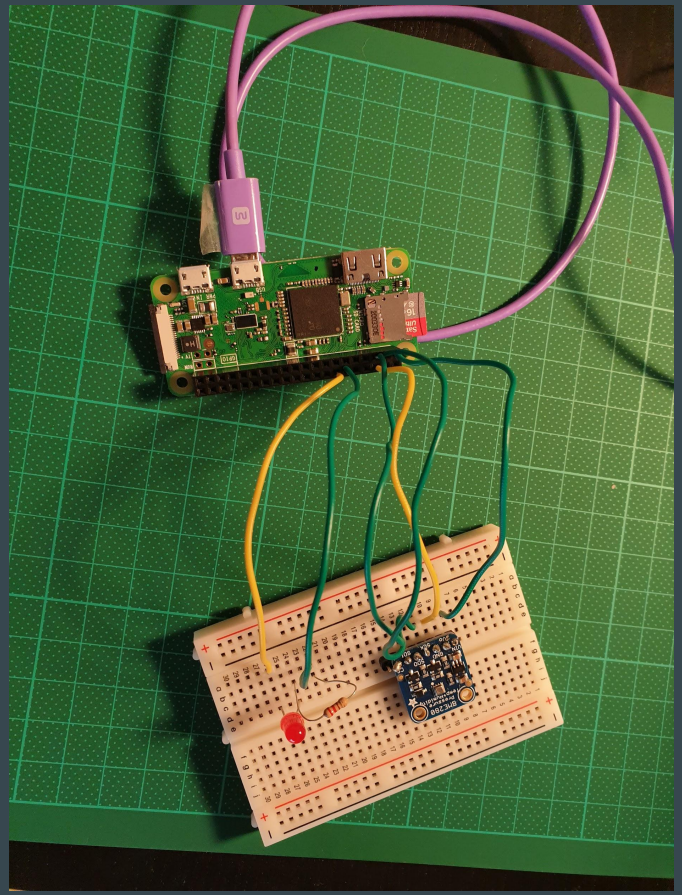
Micro USB Cable

LED

Breadboard

Resistor

Wires



Wiring Sensor

Pi 3V3 to sensor VIN
Pi GND to sensor GND
Pi SCL to sensor SCK
Pi SDA to sensor SDI

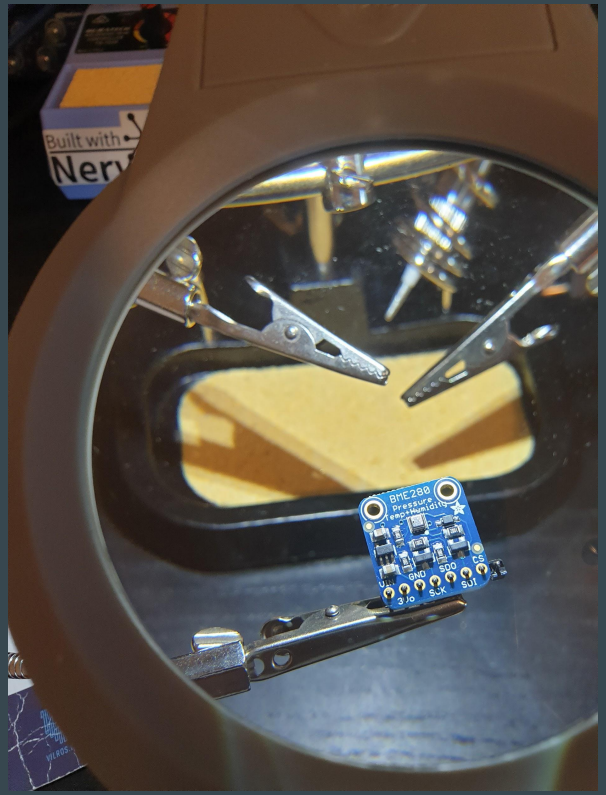
Nerves

- Builds a lightweight linux system specifically for a device eg Raspberry Pi 3 or Zero.

- Provides a bunch of Elixir modules, including networking and logging.

- Provides tooling enabling you to do things like update firmware wirelessly, ssh into a device.

Bosch BME280 - Temperature, Pressure and Humidity sensor



Using a library built by Frank Hunleth @fhunleth

https://github.com/fhunleth/bmp280

Getting started....

- Create new Nerves Project
- Add the BMP280 library as a dependency
- Set environment variable for the target device
- Install deps
- Set env vars for wifi and update configuration in config.exs file.
- Setup over the air firmware updates
- Ssh into the device. Run code directly, run logging

Install Nerves

https://hexdocs.pm/nerves/installation.html

Create new Nerves project and add BMP280 library as a dep

```
mix (beam.smp)
TS=64  -pipe -O2 -I/Users/fionamccawley/.nerves/artifacts/nerves_system_rpi0-portable-1.14.1/staging/u
sr/include -std=c99 -D_GNU_SOURCE -o /Users/fionamccawley/projects/nerves/sensor_project/_build/rpi0_d
ev/lib/muontrap/obj/muontrap.o muontrap.c
/Users/fionamccawley/.nerves/artifacts/nerves_toolchain_armv6_nerves_linux_gnueabihf-darwin_x86_64-1.4
.1/bin/armv6-nerves-linux-gnueabihf-gcc /Users/fionamccawley/projects/nerves/sensor_project/_build/rpi
0_dev/lib/muontrap/obj/muontrap.o --sysroot=/Users/fionamccawley/.nerves/artifacts/nerves_system_rpi0-
portable-1.14.1/staging  -o /Users/fionamccawley/projects/nerves/sensor_project/_build/rpi0_dev/lib/mu
ontrap/priv/muontrap
if [ -f test/Makefile ]; then /Applications/Xcode.app/Contents/Developer/usr/bin/make -C test; fi
Compiling 5 files (.ex)
Generated muontrap app
==> vintage_net
mkdir -p /Users/fionamccawley/projects/nerves/sensor_project/_build/rpi0_dev/lib/vintage_net/obj
mkdir -p /Users/fionamccawley/projects/nerves/sensor_project/_build/rpi0_dev/lib/vintage_net/priv
/Users/fionamccawley/.nerves/artifacts/nerves_toolchain_armv6_nerves_linux_gnueabihf-darwin_x86_64-1.4
.1/bin/armv6-nerves-linux-gnueabihf-gcc -c -I/Users/fionamccawley/.nerves/artifacts/nerves_system_rpi0
-portable-1.14.1/staging/usr/lib/erlang/erts-11.1.7/include -I/Users/fionamccawley/.nerves/artifacts/n
erves_system_rpi0-portable-1.14.1/staging/usr/lib/erlang/lib/erl_interface-4.0.2/include -D_LARGEFILE_
SOURCE -D_LARGEFILE64_SOURCE -D_FILE_OFFSET_BITS=64  -pipe -O2 -I/Users/fionamccawley/.nerves/artifact
s/nerves_system_rpi0-portable-1.14.1/staging/usr/include -std=c99 -D_XOPEN_SOURCE=600 -o /Users/fionam
ccawley/projects/nerves/sensor_project/_build/rpi0_dev/lib/vintage_net/obj/to_elixir.o src/to_elixir.c
```

```
{:bmp280, "~> 0.2.2"},
```

Getting started....

- ~~Create new Nerves Project~~
- ~~Add the BMP280 library as a dependency~~
- ~~Set environment variable for the target device~~
- ~~Install deps~~
- Set env vars for wifi and update configuration in config.exs file.
- Setup over the air firmware updates
- Ssh into the device. Run code directly, run logging

Update wifi config

```
config :vintage_net,
  regulatory_domain: "AU",
  config: [
    {"usb0", %{type: VintageNetDirect}},
    {"eth0",
     %{
       type: VintageNetEthernet,
       ipv4: %{method: :dhcp}
     }},
    {"wlan0",
     %{
       type: VintageNetWiFi,
       vintage_net_wifi: %{
         key_mgmt: String.to_atom(key_mgmt),
         ssid: System.get_env("NERVES_NETWORK_SSID"),
         psk: System.get_env("NERVES_NETWORK_PSK")
       },
       ipv4: %{method: :dhcp}
     }}
  ]
```

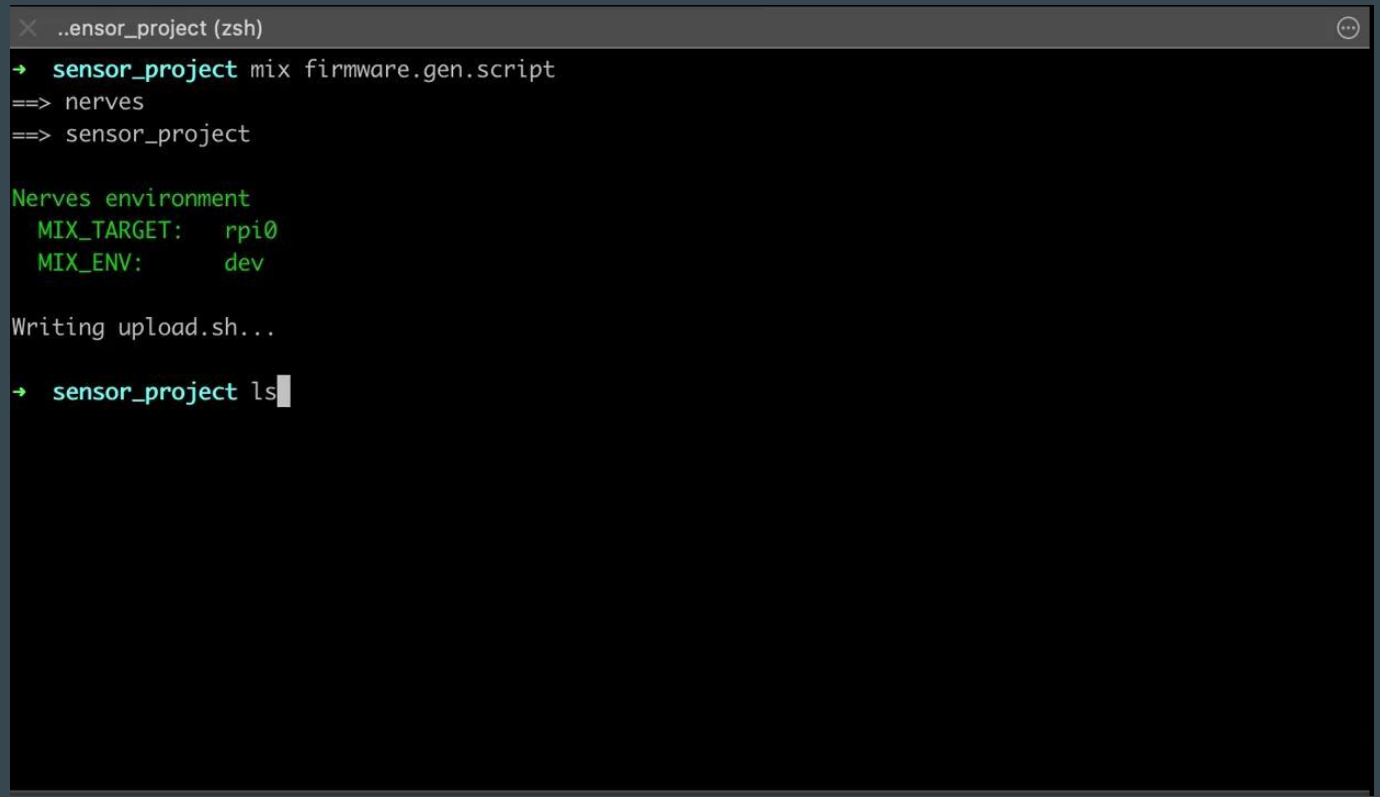
firmware > .env

```
1  export NERVES_NETWORK_SSID=
2  export NERVES_NETWORK_PSK=
3  export MIX_TARGET=rpi0
```

Getting started....

- ~~Create new Nerves Project~~
- ~~Add the BMP280 library as a dependency~~
- ~~Set environment variable for the target device~~
- ~~Install deps~~
- ~~Set env vars for wifi and update configuration in config.exs file.~~
- Setup over the air firmware updates
- Burn firmware
- Ssh into the device. Run code directly, run logging

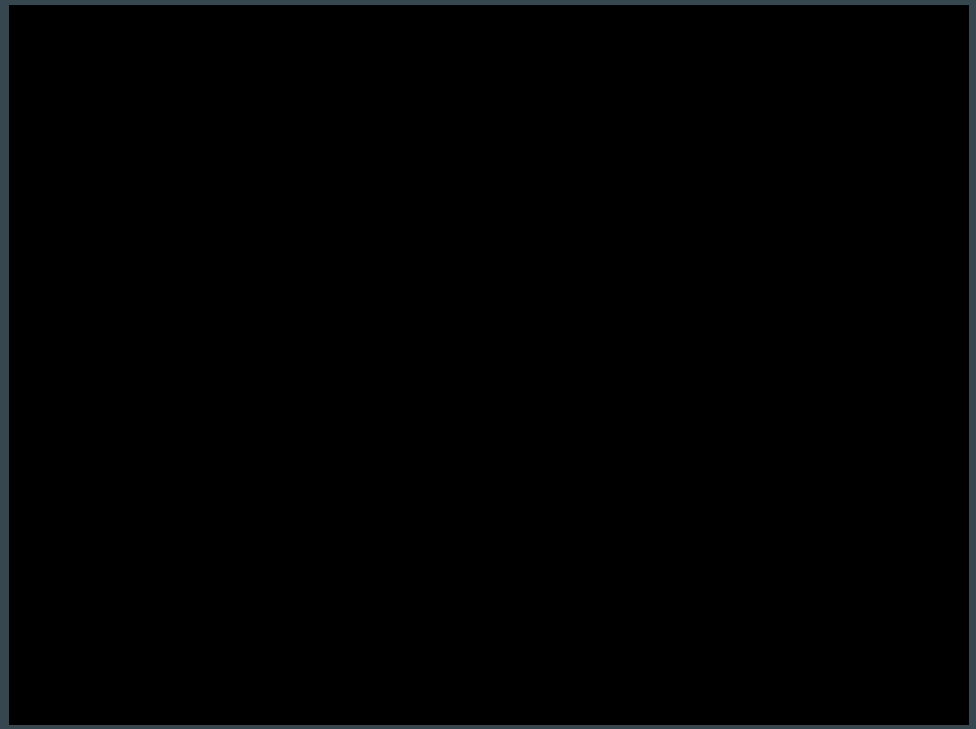
Setup network and usb cable firmware updates



Getting started....

- ~~Create new Nerves Project~~
- ~~Add the BMP280 library as a dependency~~
- ~~Set environment variable for the target device~~
- ~~Install deps~~
- ~~Set env vars for wifi and update configuration in config.exs file.~~
- ~~Setup over the air firmware updates~~
- Burn firmware
- Ssh into the device. Run code directly, run logging

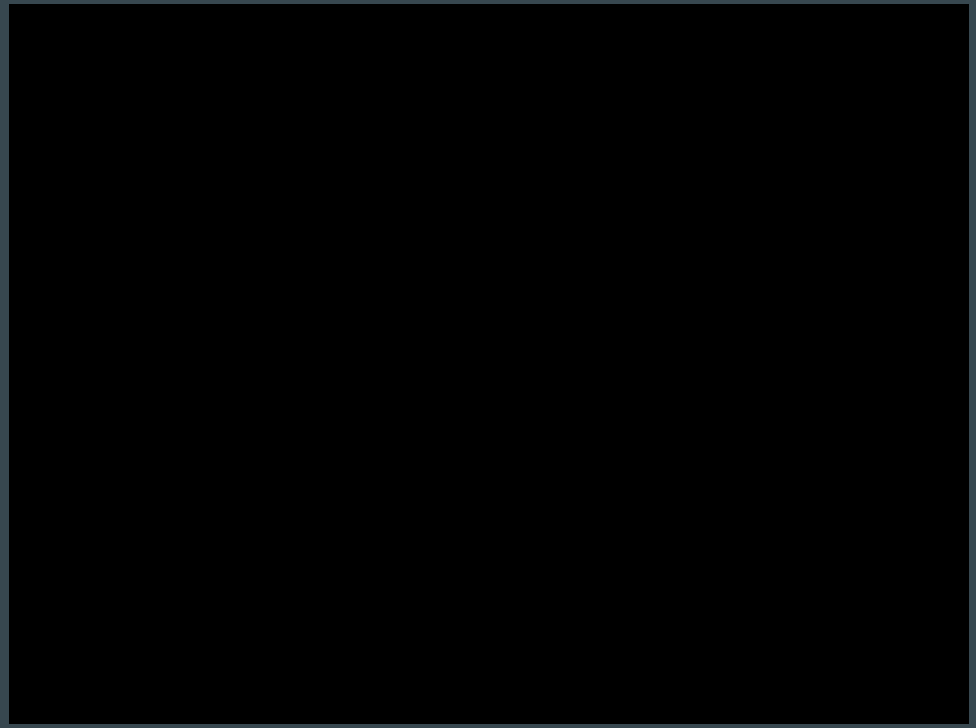
Burn to SD card



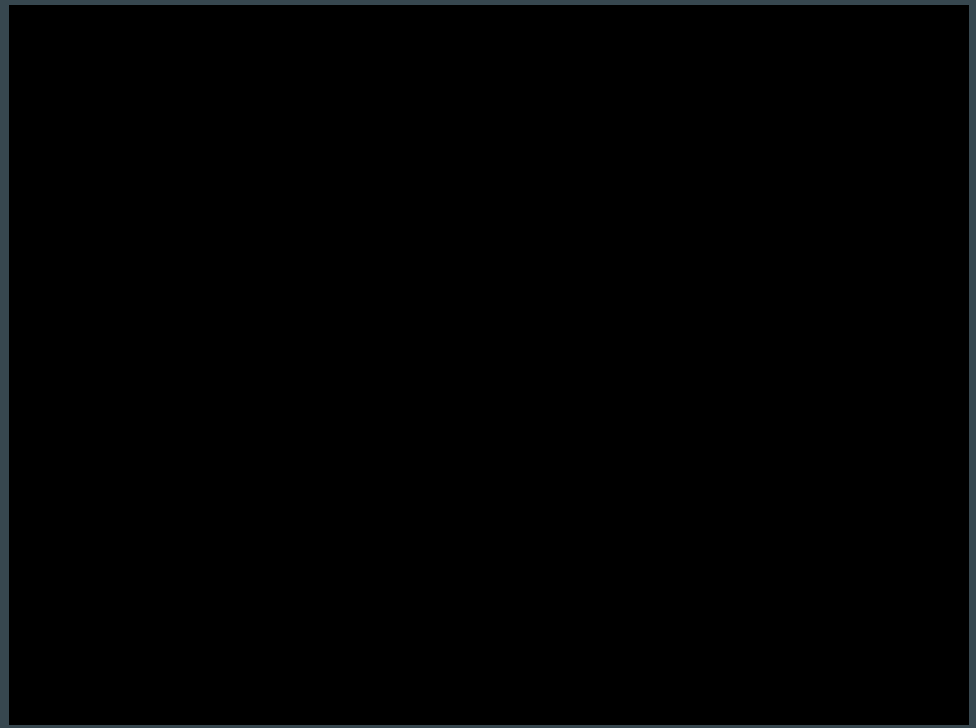
Getting started....

- ~~Create new Nerves Project~~
- ~~Add the BMP280 library as a dependency~~
- ~~Set environment variable for the target device~~
- ~~Install deps~~
- ~~Set env vars for wifi and update configuration in config.exs file.~~
- ~~Setup over the air firmware updates~~
- ~~Burn firmware~~
- Ssh into the device. Run code directly, run logging

SSH into device, logging and other tools.



Updating firmware over USB cable or WiFi



Start reading temperatures

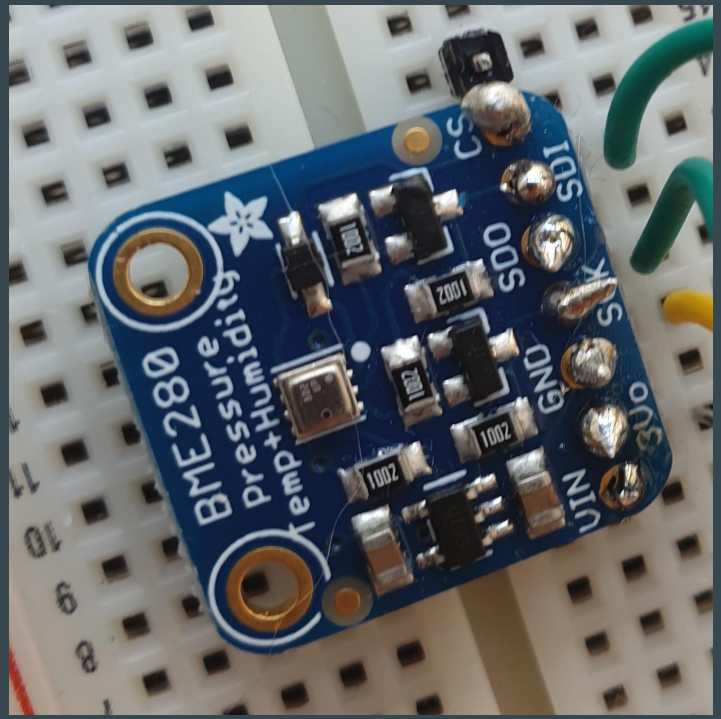
```
iex> {:ok, bmp} = BMP280.start_link(bus_name: "i2c-1", bus_address: 0x77)
{:ok, #PID<0.29929.0>}
iex> BMP280.read(bmp)
{:ok,
 %BMP280.Measurement{
   altitude_m: 13.842046523689644,
   dew_point_c: 18.438691684856007,
   humidity_rh: 51.59938493850065,
   pressure_pa: 99836.02154563366,
   temperature_c: 29.444089211523533
 }}
```

```
iex(1)> {:ok, bmp} = BMP280.start_link(bus_name: "i2c-1", bus_address: 0x77)
{:ok, #PID<0.2017.0>}
** (EXIT from #PID<0.2015.0>) shell process exited with reason: an exception was raised:
    ** (MatchError) no match of right hand side value: {:error, :i2c_nak}
        (bmp280 0.2.2) lib/bmp280.ex:191: BMP280.query_sensor/1
        (bmp280 0.2.2) lib/bmp280.ex:136: BMP280.handle_continue/2
        (stdlib 3.13.2) gen_server.erl:680: :gen_server.try_dispatch/4
        (stdlib 3.13.2) gen_server.erl:431: :gen_server.loop/7
        (stdlib 3.13.2) proc_lib.erl:226: :proc_lib.init_p_do_apply/3
```

Troubleshooting

Troubleshooting

```
iex(4)> Circuits.I2C.detect_devices
Devices on I2C bus "i2c-1":

0 devices detected on 1 I2C buses
iex(5)>
```




```
iex(1)> Circuits.I2C.detect_devices
Devices on I2C bus "i2c-1":
 * 119  (0x77)

1 devices detected on 1 I2C buses
iex(2)> {:ok, bmp} = BMP280.start_link(bus_name: "i2c-1", bus_address: 0x77)
{:ok, #PID<0.2050.0>}
iex(3)> BMP280.read(bmp)
{:ok,
 %BMP280.Measurement{
   altitude_m: 67.58225338243035,
   dew_point_c: 15.896116513153766,
   humidity_rh: 60.52678355101699,
   pressure_pa: 99201.45554209204,
   temperature_c: 24.000874632202613
 }}
```

Setup with Phoenix LiveView

```
..s/temp_sensor (zsh)
→  temp_sensor git:(main) mix nerves.new fw
```

```
→  temp_sensor git:(main) mix phx.new ui --live --no-ecto
```

```
{:ui, path: "../ui", targets: @all_targets, env: Mix.env()},
```

```
config :ui, UiWeb.Endpoint,
  url: [host: "nerves.local"],
  http: [port: 80],
  secret_key_base: "HEY05EB1dFVSu6KykKHuS4rQPQzSHv4F7mGVB/gnDLrIu75wE/ytBXy2TaL3A6RA",
  live_view: [signing_salt: "AAAABjEyERMkxgDh"],
  check_origin: false,
  root: Path.dirname(__DIR__),
  server: true,
  render_errors: [view: UiWeb.ErrorView, accepts: ~w(html json)],
  pubsub_server: Ui.PubSub,
  code_reloader: false
```

More info

https://github.com/nerves-project/nerves/blob/main/docs/User%20Interfaces.md

https://github.com/nerves-project/nerves_examples/tree/main/hello_phoenix

Use Liveview to view temps

```
scope "/", UiWeb do
  pipe_through :browser

  live "/", PageLive, :index
end
```

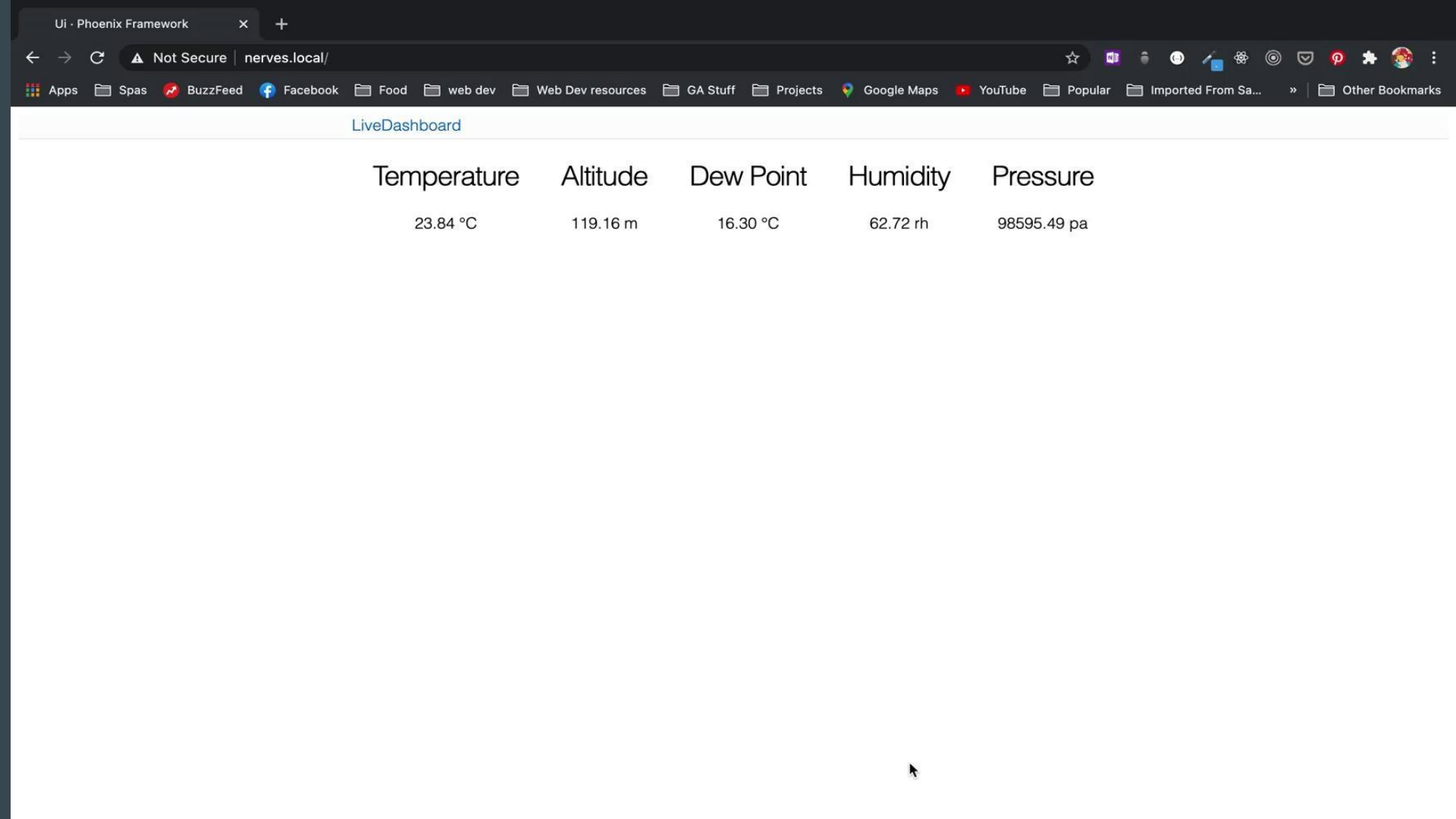
Use Liveview to view temps

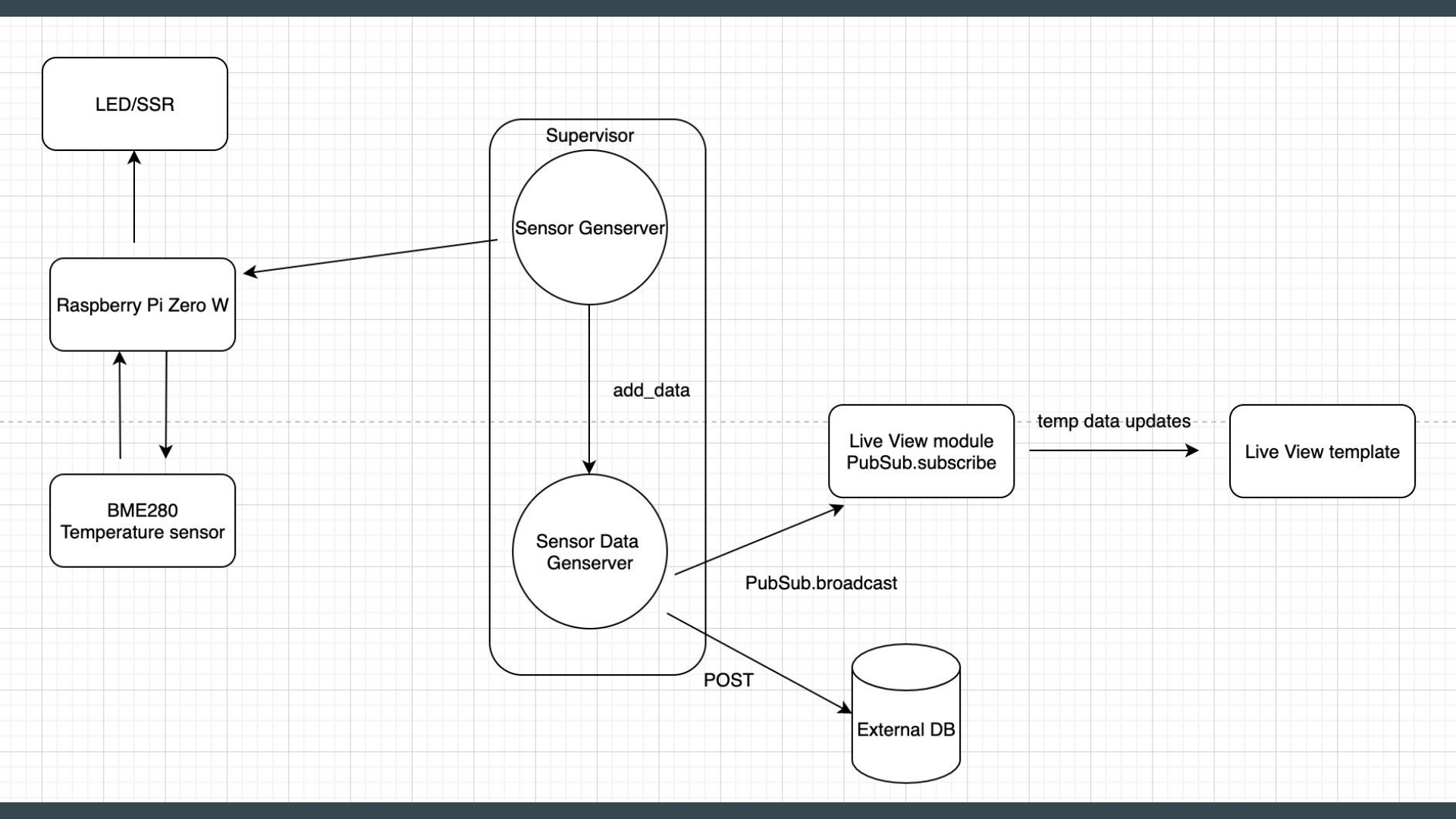
```elixir
defmodule UiWeb.PageLive do
  use UiWeb, :live_view

  @impl true
  def mount(_params, _session, socket) do
    if connected?(socket), do: Process.send_after(self(), :tick, 3000)
    {:ok, sensor} = BMP280.start_link(bus_name: "i2c-1", bus_address: 0x77)
    {:ok, assign(socket, sensor: sensor, temp: %{}, time: :calendar.local_time())}
  end


  @impl true
  def handle_info(:tick, %{assigns: %{sensor: sensor}} = socket) do
    Process.send_after(self(), :tick, 3000)
     {:ok, temp} =  BMP280.read(sensor)
     {:noreply, assign(socket, temp: temp, time: :calendar.local_time())}
  end
end
```

ib > ui_web > live > page_live.html.leex

You, seconds ago | 2 authors (You and others)

```
<section>
  <%= if Map.has_key?(@temp, :temperature_c) do %>
    <div class="reading">
      <div class="temp-data">
        <h2>Temperature</h2>
        <p><%= @temp.temperature_c |> Decimal.from_float() |> Decimal.round(2) |> Decimal.to_string() %> &#8451;</p>
      </div>
      <div class="temp-data">
        <h2>Altitude</h2>
        <p><%= @temp.altitude_m |> Decimal.from_float() |> Decimal.round(2) |> Decimal.to_string() %> m</p>
      </div>
      <div class="temp-data">
        <h2>Dew Point</h2>
        <p><%= @temp.dew_point_c |> Decimal.from_float() |> Decimal.round(2) |> Decimal.to_string() %> &#8451;
        </p>
      </div>
      <div class="temp-data">
        <h2>Humidity</h2>
        <p><%= @temp.humidity_rh |> Decimal.from_float() |> Decimal.round(2) |> Decimal.to_string() %> rh
        </p>
      </div>
      <div class="temp-data">
        <h2>Pressure</h2>
        <p><%= @temp.pressure_pa |> Decimal.from_float() |> Decimal.round(2) |> Decimal.to_string() %> pa
        </p>
      </div>
    </div>
  <%end %>
</section>
```


Ui · Phoenix Framework
Not Secure | nerves.local/
Apps
Spas
BuzzFeed
Facebook
Food
web dev
Web Dev resources
GA Stuff
Projects
Google Maps
YouTube
Popular
Imported From Sa...
Other Bookmarks
LiveDashboard
Temperature
Altitude
Dew Point
Humidity
Pressure
23.84 °C
119.16 m
16.30 °C
62.72 rh
98595.49 pa


LED/SSR
Supervisor
Sensor Genserver
Raspberry Pi Zero W
add_data
BME280
Temperature sensor
Sensor Data
Genserver
Live View module
PubSub.subscribe
temp data updates
Live View template
PubSub.broadcast
POST
External DB

Genservers

UiWeb.Sensor

- Starts link with sensor
- Polls every 5 seconds to get a reading
- Sends temp data to SensorData process
- Triggers GPIO port on/off based on temp < > a set temp.

UiWeb.SensorData

- Creates a queue and count in memory.
- Adds data to the queue.
- Removes data at a certain max count.
- Broadcasts data using Phoenix.PubSub.
- Sends data to an external DB.

Persisting Data and visualising

Contex - Elixir server-side data-plotting / charting system that generates SVG output.

InfluxDB - Time series database developed by InfluxData, written in Go.

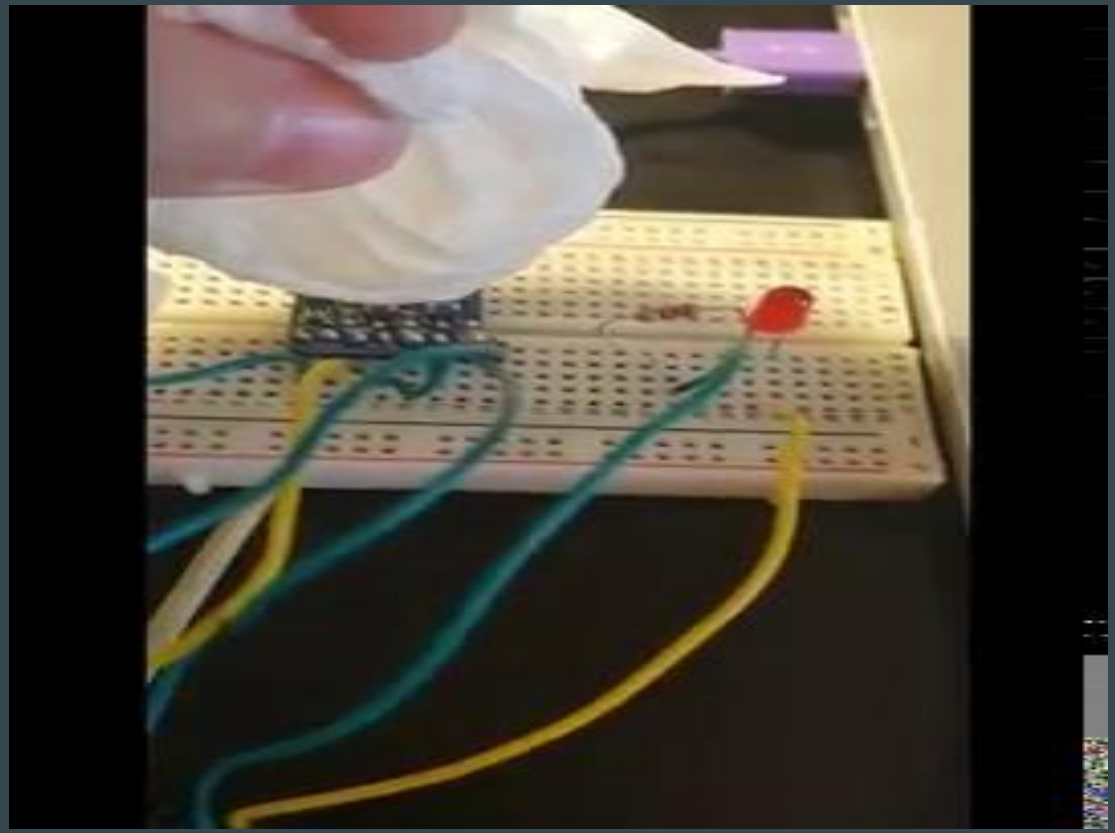
InfluxDB

https://docs.influxdata.com/influxdb/v2.0/get-started/

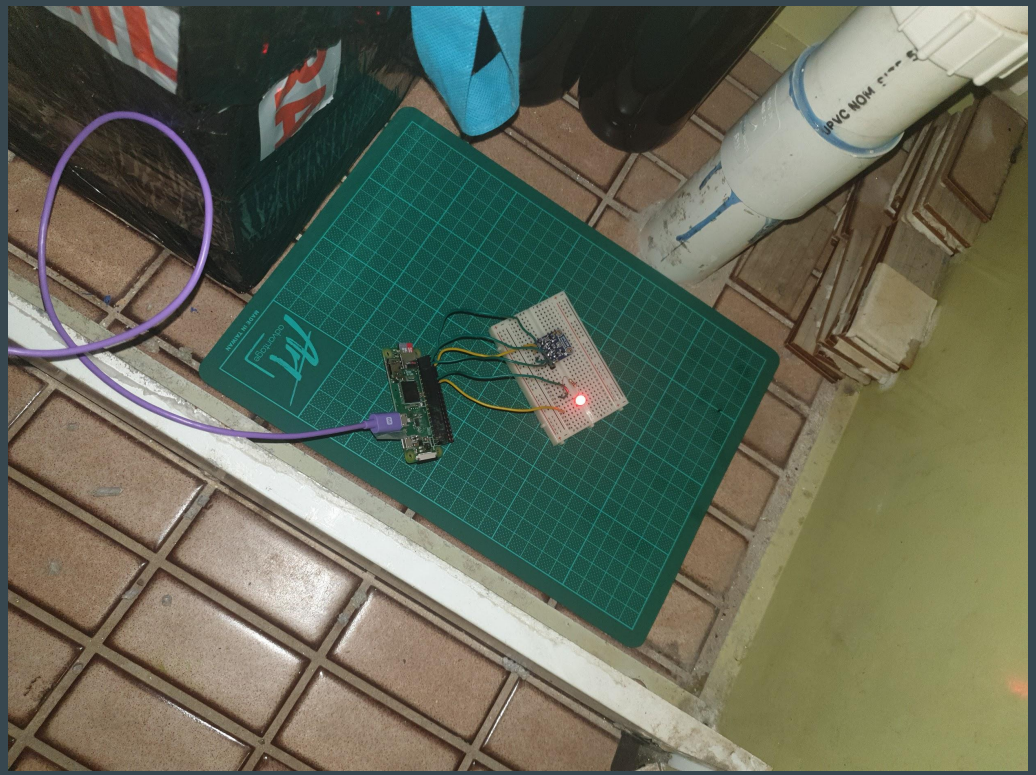
After setting up add the below to the .env file and create a new firmware image and upload.

```
export INFLUX_TOKEN=
export LOCAL_HOST_IP=
export INFLUX_ORG=
export INFLUX_BUCKET=
```

Demo

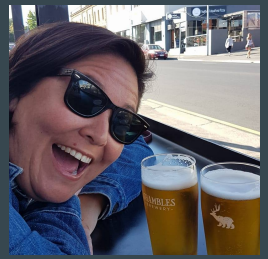


Sensor setup



Next steps

- Persist data locally to the device instead of to memory.
- Setup notifications.
- Learn more about connecting a Solid State Relay and implement a heating/cooling element.
- Use it to monitor my next batch of beer!



Sensors, Beers and Nerves

...

Slides: https://github.com/fimac/talks
Repo: https://github.com/fimac/temp_sensor_nerves

GITHUB.COM/FIMAC

@SAUCERLIKE