# Display on/off - shutdown (deutsch)

https://github.com/Archonw/Dynaframe-Stuff/blob/main/scripts/display-on-off-shutdown.py

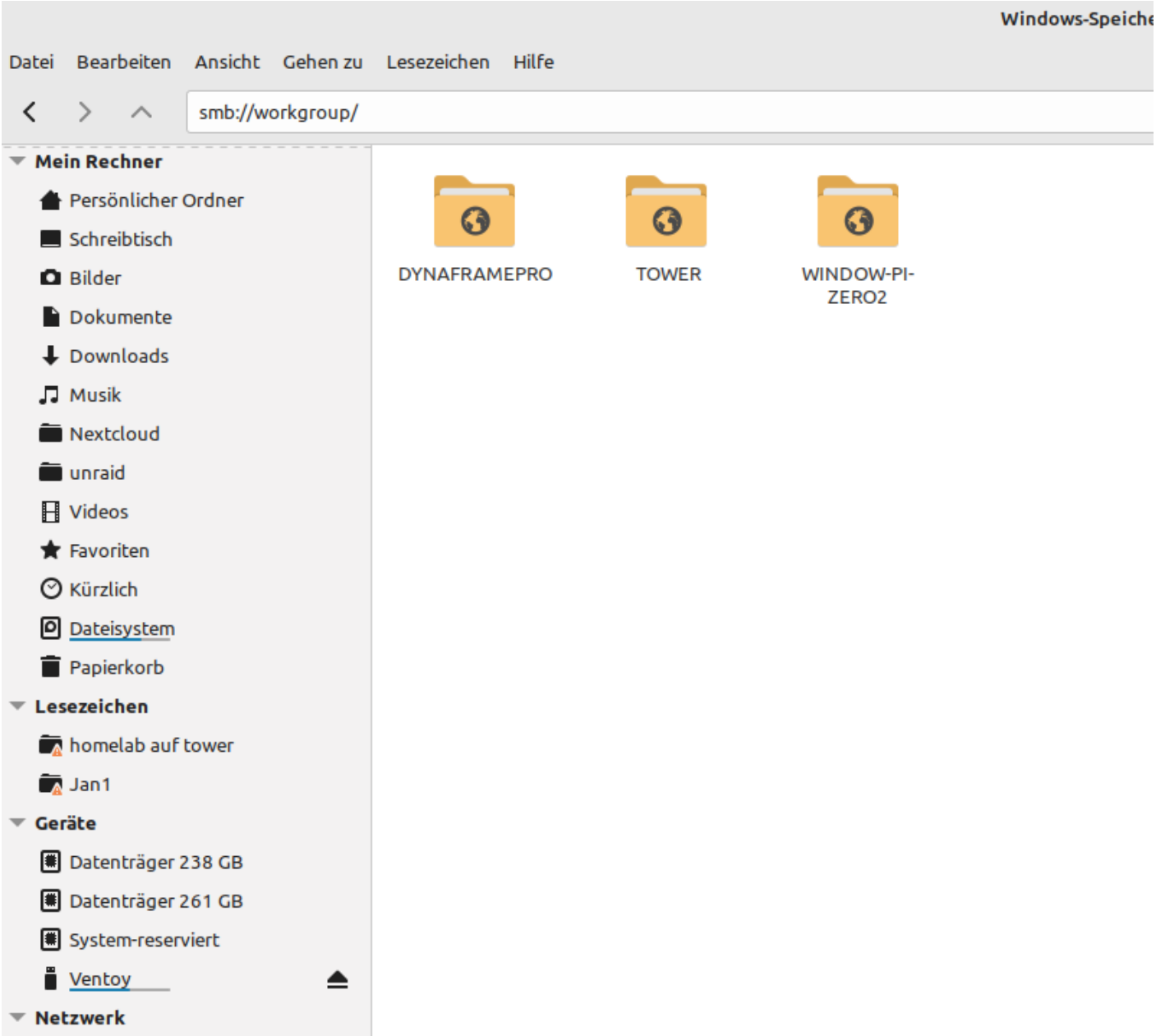
Von dieser Github Seite wird der Code verwendet, mit dessen Hilfe das HDMI-Sginal des Pi's ab- und angeschaltet werden kann, sowei bei langen Drücken der Pi herunter fährt.

Nachdem der Button am Pi angschlossen ist kann es mit der Software losgehen. Dafür wird ein Python-Script verwendet.

Im ersten Schritt erstellen wir auf dem Pi einen neuen Ordner "scripts" in dem wir dann eine Datei "button.py" anlegen werden welche wir dann im zweiten Teil über die "rc.local" beim Systemstart aufrufen lassen.

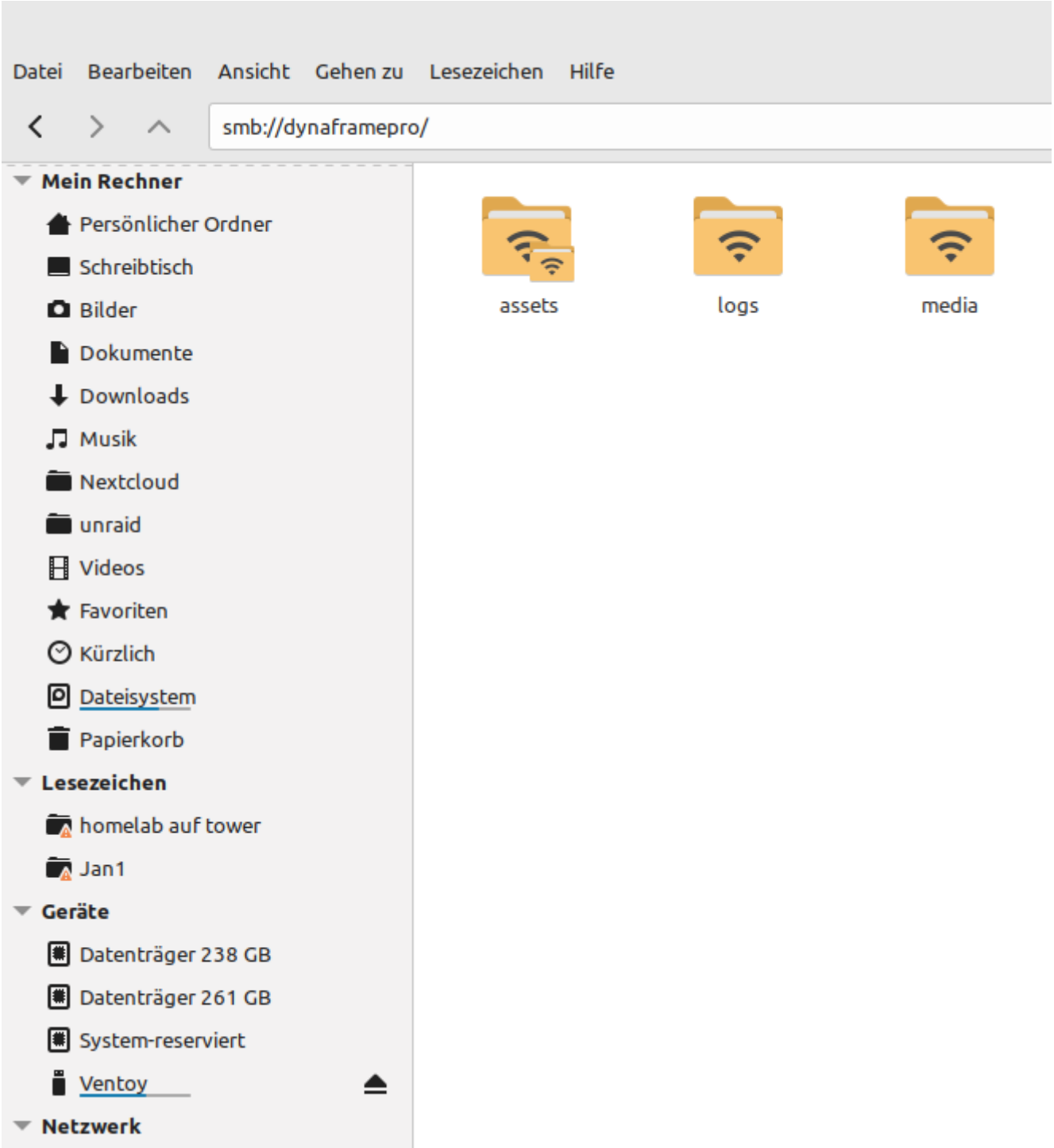
1. Teil

Der Raspberry Pi ist an und im Netwzerk angemeldet, so dass wir auf ihn über unser Netzwerk sehen und aufrufen können. Haben wir bei der Erst-Einrichtung des Pi den Namen nicht verändert, so finden wir ihn unter dem Namen DYNAFRAMEPRO.

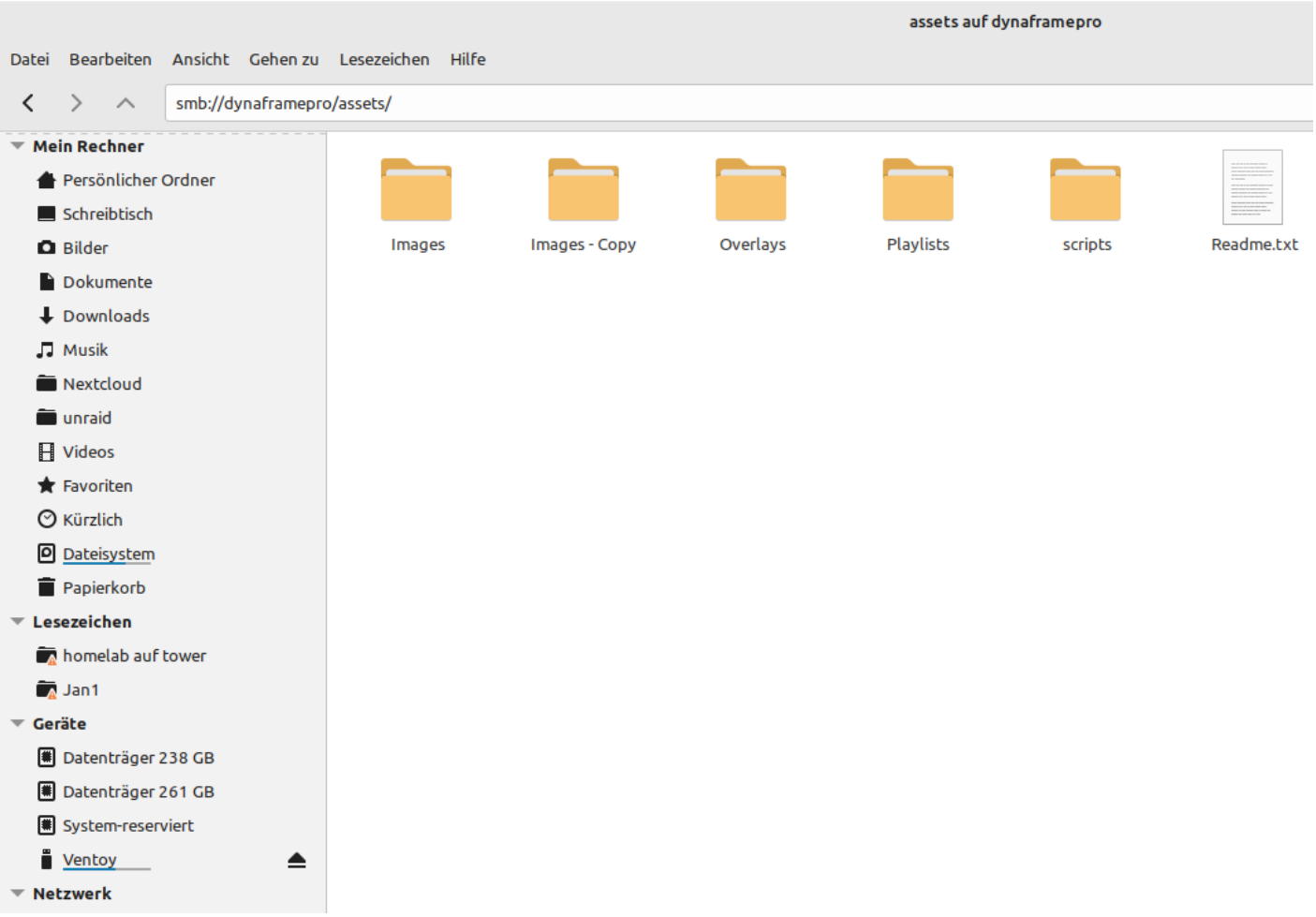



Schauen wir welche Ordner nun auf unserem Pi angezeigt werden so sehen wir unter anderem den Ordner "assets".

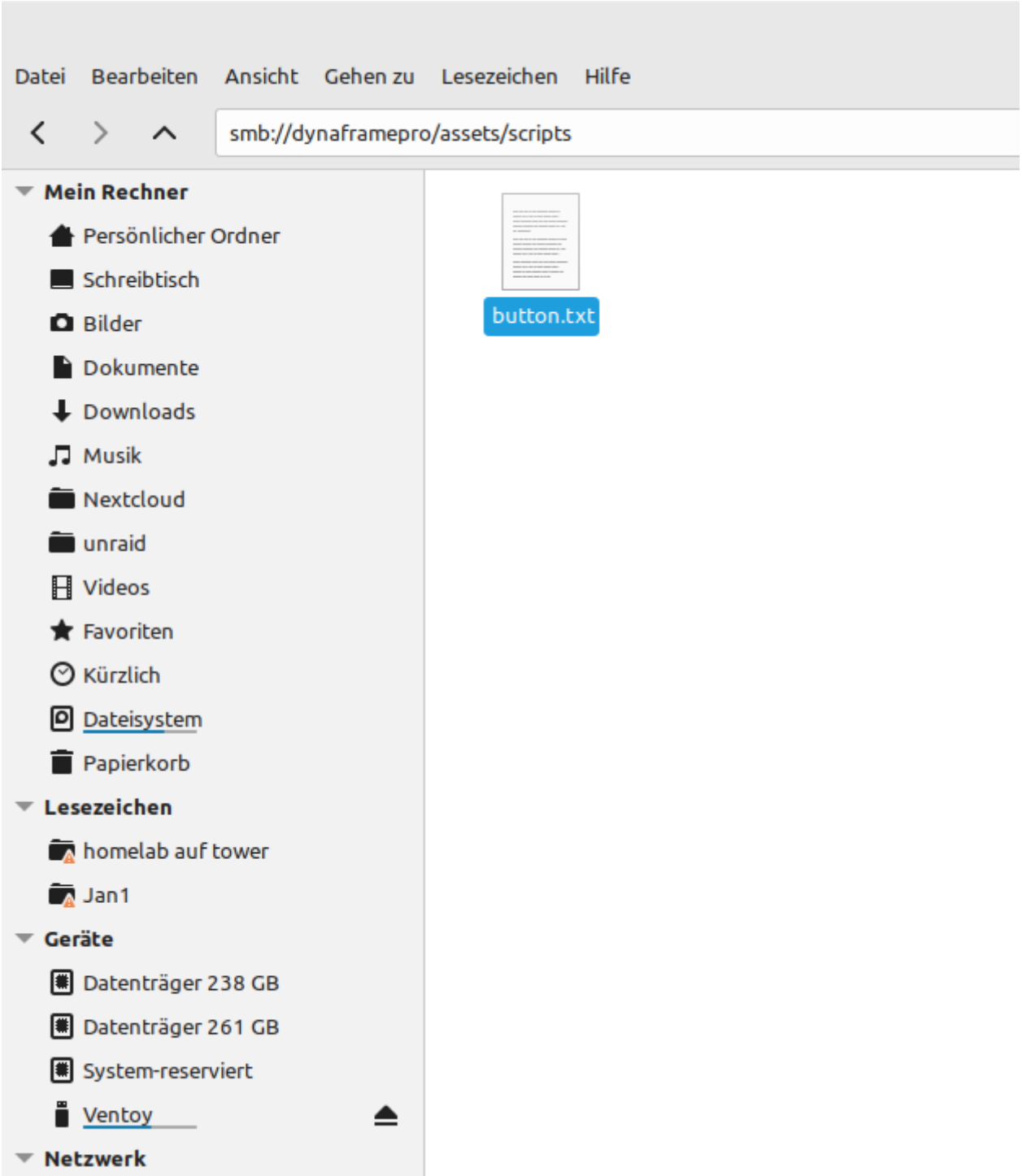
In diesen navigieren wir hinein.




Hier sind einige Ordner enthalten. Wir erstellen hier einen neuen Ordner "scripts" und navigieren und den neunen Ordner hinein.




In dem noch leeren Ordner erstellen wir eine Datei mit dem Namen "button.txt" und öffnen diese.




Hier kopieren wir folgenden Code hinein.

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

from gpiozero import Button
from subprocess import check_call
from signal import pause
import os
import requests
```

```
Button.was_held = False
button_pressed_time = None

def get_display_power_status():
    result = os.popen('vcgencmd display_power').read()
    return int(result.split('=')[1])

def toggle_display():
    current_status = get_display_power_status()
    if current_status == 1:
        response = requests.get("http://127.0.0.1/command/?COMMAND=SCREENSTATE&VALUE=false")
        print("Display off")
    else:
        response = requests.get("http://127.0.0.1/command/?COMMAND=SCREENSTATE&VALUE=true")
        print("Dislpay on")

def shutdown():
    print("Shutdown function called")
    os.system('sudo shutdown now')

def held(btn):
    btn.was_held = True
    print("Button was held, not just pressed")
    shutdown()

def released(btn):
    if not btn.was_held:
        toggle_display()
    btn.was_held = False

btn = Button(19, hold_time=4)  # Set your GPIO number and hold time here

btn.when_held = held
btn.when_released = released

# Warten
pause()
```

Wichtig: hier muss in Zeile 40 der Pin angegeben werden, an dem der Button am Pi angeschlossen wurde. In diesem Fall ist es der Pin 19.

Die fertige Datei sollte dann so aussehen:

*button.py (assets auf dynaframepro /assets/scripts)

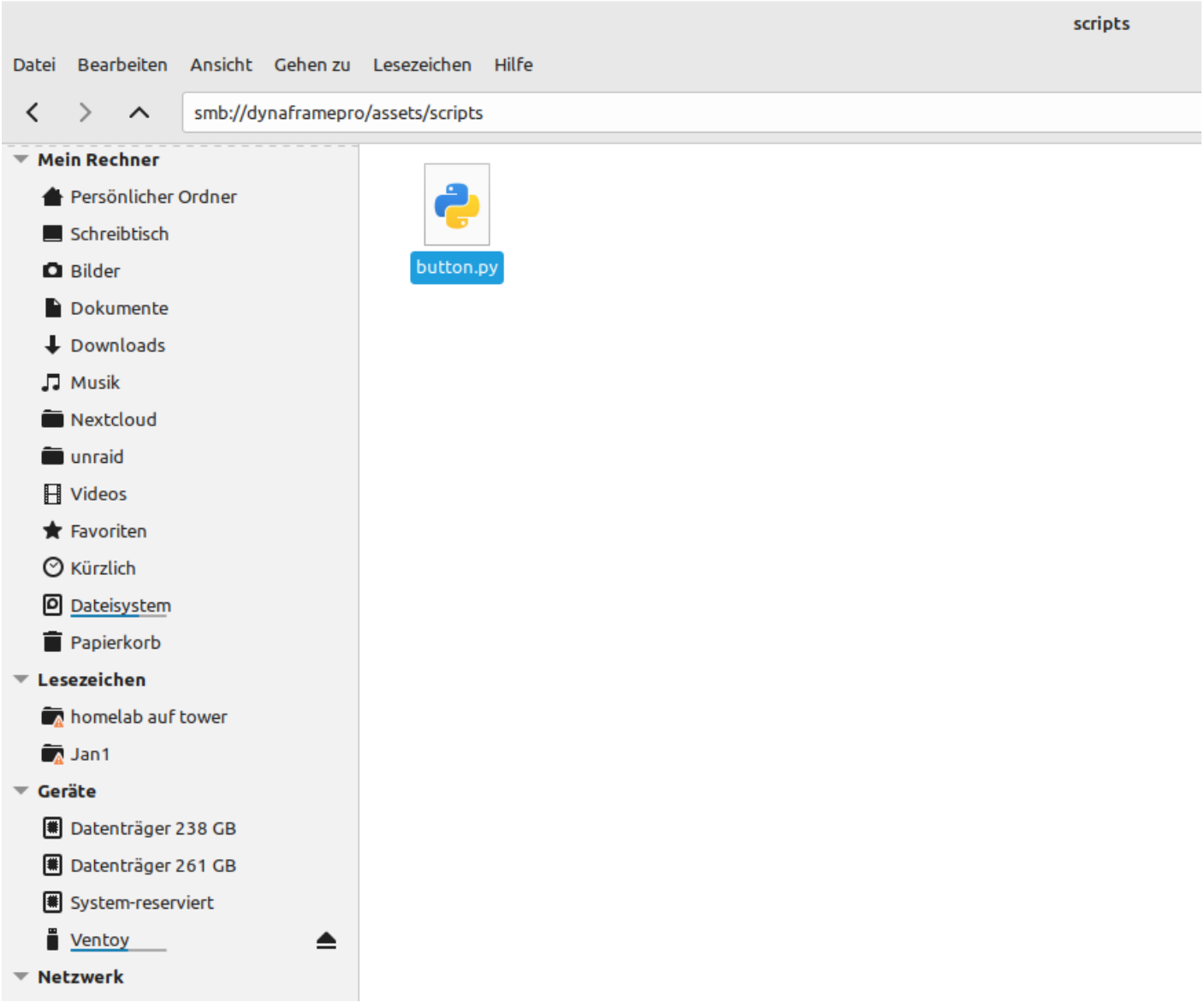
Datei Bearbeiten Ansicht Suchen Werkzeuge Dokumente Hilfe

*button.py

```
from signal import pause
import os
import requests

Button.was_held = False
button_pressed_time = None

def get_display_power_status():
    result = os.popen('vcgencmd display_power').read()
    return int(result.split('=')[1])

def toggle_display():
    current_status = get_display_power_status()
    if current_status == 1:
        response = requests.get("http://127.0.0.1/command/?COMMAND=SCREENSTATE&VALUE=false")
        print("Display off")
    else:
        response = requests.get("http://127.0.0.1/command/?COMMAND=SCREENSTATE&VALUE=true")
        print("Dislpay on")

def shutdown():
    print("Shutdown function called")
    os.system('sudo shutdown now')

def held(btn):
    btn.was_held = True
    print("Button was held, not just pressed")
    shutdown()

def released(btn):
    if not btn.was_held:
        toggle_display()
    btn.was_held = False

btn = Button(19, hold_time=4)  # Set your GPIO number and hold time here

btn.when_held = held
btn.when_released = released

# Warten
pause()
```

Diese können wir nun speichern und schließen.

Jetzt müssen wir sie noch kurz umbenennen in "button.py"




Mit diesem Fenster sind wir fertig und können es nun auch schließen.

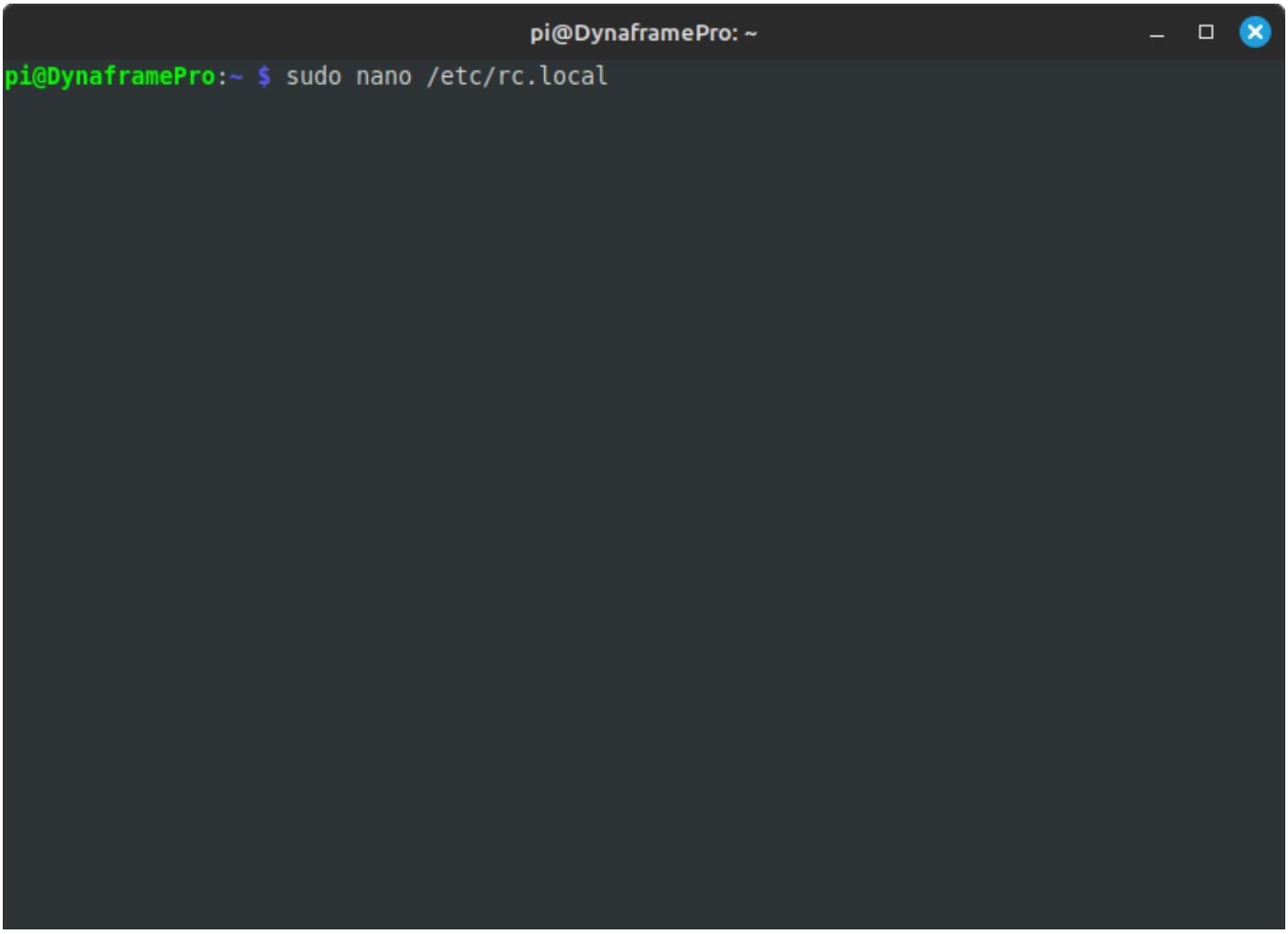
2. Teil

Wir haben nun die erforderliche Datei erstellt damit wir ein Python-Script beim start aufrufen und im Hintergrund laufen lassen können. Jetzt müssen wir noch dafür sorgen, dass dieses Script auch bei Systemstart aufgerufen wird. Dazu gibt es einige Wege; wir werden hier den Weg über eine Datei die "rc.local" nehmen.

Dazu müssen wir uns entweder per ssh auf den Pi verbinden, rufen über eine angeschlossene Tastatur und Maus ein Terminal auf oder verbinden uns über VNC auf den Pi und rufen dann ein Terminal am Pi auf.

Haben wir das gemacht geben wir folgenden Befehl im Terminal ein, um einen Editor zu starten, der uns die Datei rc.local öffnet, damit wir sie bearbeiten können.

```
sudo nano /etc/rc.local
```




Wir fügen diese Zeile kurz vor dem Ende ein.

```
/usr/bin/python3 /home/pi/Dynaframe/Assets/scripts/button.py
```

So sollte die Datei jetzt aussehen.

```
pi@DynaframePro: ~
  GNU nano 3.2                    /etc/rc.local                    Modified

#!/bin/sh -e
#
# rc.local
#
# This script is executed at the end of each multiuser runlevel.
# Make sure that the script will "exit 0" on success or any other
# value on error.
#
# In order to enable or disable this script just change the execution
# bits.
#
# By default this script does nothing.

# Print the IP address
_IP=$(hostname -I) || true
if [ "$_IP" ]; then
  printf "My IP address is %s\n" "$_IP"
fi

/usr/bin/python3 /home/pi/Dynaframe/Assets/scripts/button.py

exit 0


^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

Um den Editor zu schließen und die Datei zu speichern drücken wir "Strg+X" und danach "Y"

Jetzt fehlt nur noch ein Neustart des Pi und der Button ist einsatzbereit.

---

Revision #4
Created 1 June 2024 14:51:35 by Jan
Updated 1 June 2024 16:11:59 by Jan