Nordpool Scraper Mini Project

Ance Strazdina

CMP408

Introduction

Saving electricity has both environmental and financial benefits and IoT can help with monitoring energy costs and consumption.

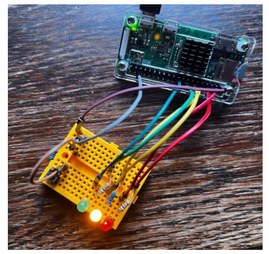
Nordpool is a Northern European power exchange that publishes its electricity tariff data. This is helpful to consumers; however, this data could be displayed in a clearer way that indicates the best times for electricity use cost-wise.

This project labels Nordpool tariffs as 'affordable', 'average', or 'expensive' based on their value and displays this information. It utilises:

- **Hardware:** Rpi Zero W, 3 LEDs, 1 button, 8 M/F and 1 M/M wire, and 4 resistors.
- **Software:** Scheduled scripts the retrieve and label Nordpool data, change LEDs to red, yellow, or green to reflect the value for the current tariff, and send data to a webserver. LKM and a supporting user space script that manually performs the scheduled operations with a button press.
- **Cloud:** AWS EC2 instance that acts as a webserver and displays the tariff data received from the RPi on a website.

This project addresses areas of IoT and cloud secure development by developing an LKM, automating processes, and utilising hardware and cloud infrastructure for displaying data. To ensure seamless and secure function, these aspects were thoroughly tested and configured with the appropriate security measures. Since a lot of IoT manufacturers overlook security for profit, proper configuration was important for this project, however, if optimal configuration was not implemented, the reasons, impacts, and countermeasures for this are acknowledged.

Methodology



Button - GPIO 16 and 3V3

LEDs:
Green - GPIO 24 and GND
Yellow - GPIO 18 and GND
Red - GPIO 14 and GND

1. Hardware. The RPi and corresponding hardware were wired.

2. Software. This phase utilised the nordpool Python library to obtain tariff information for two days and labelled it based on the price distribution with pandas. The upper 25% prices were considered expensive or red, the lower 25% - affordable or green, and the rest were average or yellow. The tariff and label data was saved as a CSV and sent to a webserver over MQTT with authentication.

Another script used these tariff labels in the CSV to determine which LED colour to turn on.

Both scripts were inserted in the crontab with the first executing daily after Nordpool releases day-ahead prices, and the latter – every hour.

An LKM handled button press interrupts and notified a user space service by writing to a character device to allow the mentioned scripts to be run outside of the scheduled times.

3. Cloud. An AWS EC2 Ubuntu Server 22.04 LTS t2.micro instance received the CSV with tariffs and labels from the RPi over MQTT with authentication. This data is displayed on a minimal Flask site in an easy to oversee way.

Time	12.12	13.12
23 - 00	0.095	0.086
00 - 01	0.095	0.086
01 - 02	0.095	0.077
02 - 03	0.094	0.077
03 - 04	0.095	0.095
04 - 05	0.15	0.063
05 - 06	0.184	0.15
06 - 07	0.2	0.15
07 - 08	0.2	0.15
08 - 09	0.2	0.15
09 - 10	0.184	0.148
10 - 11	0.184	0.15
11 - 12	0.155	0.15
12 - 13	0.15	0.149
13 - 14	0.171	0.149
14 - 15	0.195	0.146
15 - 16	0.181	0.151
16 - 17	0.172	0.148
17 - 18	0.155	0.15
18 - 19	0.152	0.13
19 - 20	0.147	0.113
20 - 21	0.128	0.101
21 - 22	0.111	0.104
22 - 23	0.098	0.091

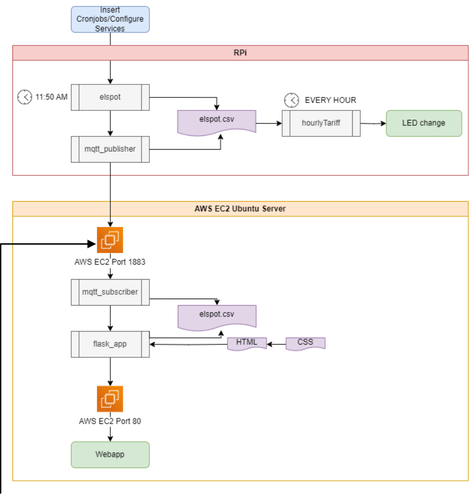
Nordpool Hourly Spot Prices for LV (and Baltics).

Prices are in EUR/Kwh. Timezone - UTC. VAT is not included.

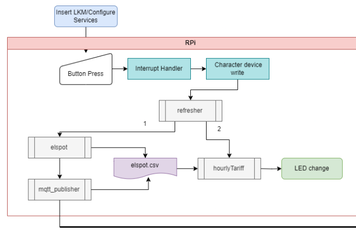
Red = highest 25% tariffs of the day, avoid heavy electricty usage; Green = lowest 25% tariffs of the day.

Author: Ance Strazdina

Automated Flow:



Manual button press on the RPi side:



Project Highlights

- Successfully implements the desired functionality: fetches tariff data every day, shows how affordable the current tariff is by the hour with LEDs, and displays colour-coded current and day-ahead data on a website.
- Does everything automatically with configured cronjobs and background services. There is an option to manually run the functionality scripts to refresh data and LED state with a button press thanks to an LKM.
- Utilises robust error handling to ensure seamless functionality. Security measures include authentication-based MQTT and AWS security groups.
- Mostly used free-tier AWS services so only spent $0.6 of the $100 AWS budget for reserving an Elastic IP.
- Additionally, the hardware used is easy to procure and inexpensive.

Future Work

Further security improvements such as SSL for MQTT and website and using Linux groups to address file permission issues that arose when some services accessed certain files.

Timezone and Nordpool region change option.

Automated setup.

References

Corbet, J., Rubini, A. & Kroah-Hartman, G., 2005. Chapter 3. Char Drivers. In: Linux Device Drivers, 3rd Edition. Sebastopol: O'Reilly, pp. 42-72. | Huiyeon, K., 2020. Step-by-step visual guide on deploying a Flask application on AWS EC2. [Online] Available at: https://medium.com/techfront/step-by-step-visual-guide-on-deploying-a-flask-application-on-aws-ec2-8e3e8b82c4f7 [Accessed 12 December 2023] | Huoman, K., 2022. nordpool. [Online] Available at: https://github.com/kipe/nordpool [Accessed 12 December 2023]. | Neshenko, N. et al., 2019. Demystifying IoT Security: An Exhaustive Survey. IEEE Communications Surveys & Tutorials, 21(3), pp. 2702 - 2733. | Pourbeik, P., 2021. How to use IoT for energy efficiency and sustainability. [Online] Available at: https://www.techtarget.com/iotagenda/feature/How-to-use-IoT-for-energy-efficiency-and-sustainability [Accessed 13 Decmeber 2023]. | Salzman, P. J. et al., 2023. The Linux Kernel Module Programming Guide. [Online] Available at: https://sysprog21.github.io/lkmpg/ [Accessed 12 December 2023].