Homne elektrihind

Raivo Kasepuu
matrikkel B710710

Millist probleemi lahendame?

- Elektrit börsihinnaga ostes on tihti olukordi, kus mõnel tunnil on hind mitmeid kordi suurem, kui eelneval või järgneval tunnil.
- Näiteks oli esmaspäeval (14.12.2020) Soome tuumajaamas üks reaktor välja lülitatud, mis tekitas elektridefitsiidi NordPool turul.
- Keskmine päeva hind kerkis 2 korda, päeva tipud olid 200€/MW tasemel.
- On tavapärane, et ühe päeva min ja max hinnad erinevad 4-5 korda.
- **Optimeerime oma elektritarbimist jälgides börsihinda!**

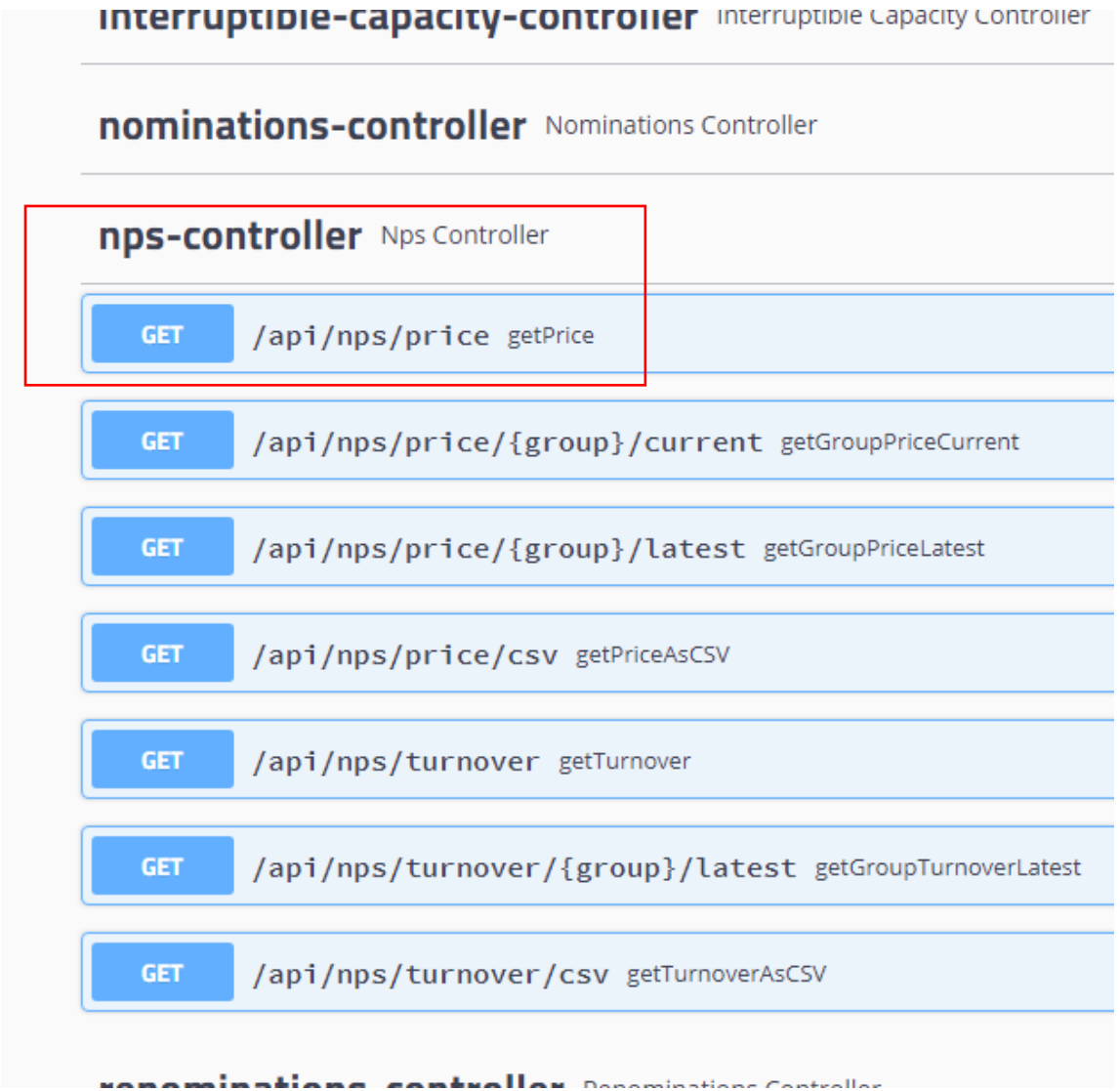
HOURLY | DAILY | WEEKLY | MONTHLY | YEARLY | 16 DEC 2020 | EUR

EUR/MWh

	16-12-2020	15-12-2020	14-12-2020	13-12-2020	12-12-2020	11-12-2020	10-12-2020	09-12-2020
00 - 01	21,00	19,38	47,39	41,96	44,77	30,02	20,06	19,30
01 - 02	19,93	19,97	25,71	35,07	28,04	28,05	19,49	18,99
02 - 03	19,82	19,02	27,28	30,09	19,67	26,56	18,96	18,65
03 - 04	19,90	18,91	26,23	30,07	19,25	20,24	19,47	18,59
04 - 05	21,07	21,88	33,67	35,03	25,06	30,61	35,32	20,02
05 - 06	54,95	54,56	94,24	34,25	30,03	57,48	41,48	28,60
06 - 07	68,57	63,05	199,96	35,01	30,38	68,48	58,95	51,22
07 - 08	74,79	70,91	199,93	46,22	44,73	75,70	72,93	69,00
08 - 09	77,50	70,85	186,08	47,33	46,19	76,51	96,43	81,66
09 - 10	79,94	68,87	131,43	48,50	46,61	77,93	93,94	78,23
10 - 11	67,89	58,10	92,26	55,00	56,94	75,72	90,26	69,96
11 - 12	67,88	59,97	92,27	60,12	50,07	76,76	90,76	73,71
12 - 13	81,99	66,12	147,95	53,19	52,66	78,00	86,82	78,27
13 - 14	72,18	66,87	174,92	65,81	49,95	76,76	93,10	79,16
14 - 15	76,44	68,85	199,94	68,25	74,37	76,24	85,24	84,40
15 - 16	78,74	70,74	199,97	68,81	75,04	76,55	89,68	89,83
16 - 17	85,41	68,89	199,91	77,01	77,20	78,46	101,00	90,19
17 - 18	72,98	68,80	108,54	77,21	72,85	75,22	105,41	99,92
18 - 19	67,90	65,37	90,20	74,29	70,63	75,03	93,56	80,52
19 - 20	49,54	56,20	75,68	69,71	68,10	54,93	75,22	52,33
20 - 21	42,36	48,99	71,94	47,55	48,27	40,49	60,26	50,74
21 - 22	44,98	48,05	58,54	60,26	46,39	50,50	49,78	46,65
22 - 23	40,93	40,77	60,53	54,60	44,79	30,73	31,00	32,93
23 - 00	25,02	25,06	33,58	35,09	39,96	19,87	22,11	20,02
Min	19,82	18,91	25,71	30,07	19,25	19,87	18,96	18,59
Max	85,41	70,91	199,97	77,21	77,20	78,46	105,41	99,92
Avg	55,49	51,67	107,42	52,10	48,41	57,37	64,63	56,37
Peak	73,20	65,80	141,60	63,77	61,72	74,84	91,79	79,85
Off-peak 1	37,50	35,96	81,80	35,96	30,24	42,14	35,83	30,55

- **Homne elektrihind**
  - Homne elektrihind määratakse NordPool-i poolt vastavalt pakkumiste – tellimuste suhtele ja avaldatakse eelneval päeval 13:45.
  - NordPool pakub ligipääsu ka API abil, kuid see on tasuline teenus – 3500€/aastas.
  - Elering võimaldab oma API-ga tasuta ligipääsu:

https://dashboard.elering.ee/swagger-ui.html



Mismoodi see API värk käib?

- Trükime näiteks oma veebilehitsejasse sellise aadressi:

https://dashboard.elering.ee/api/nps/price?end=2020-12-01%2021%3A00&start=2020-11-30%2022%3A00

- Saame vastuseks json-i ---------->

Näeme, et:

- päring on edukas: „success“: true
- Päringu vastus sisaldab erinevate turgude („ee“, „fi“, lv“) elektrihindasid(„price“).
- „timestamp“ tähendab täistunni algust ja on defineeritud sekundites alates 01.01.1970 kell 00:00 UTC


```
{"success":true,"data":{"ee":[{"timestamp":1606773600,"price":20.9900},{"time
stamp":1606777200,"price":23.1900},{"timestamp":1606780800,"price":25.0200},{
"timestamp":1606784400,"price":25.0200},{"timestamp":1606788000,"price":33.26
00},{"timestamp":1606791600,"price":35.3400},{"timestamp":1606795200,"price":
38.4300},{"timestamp":1606798800,"price":56.0300},{"timestamp":1606802400,"pr
ice":73.2500},{"timestamp":1606806000,"price":75.5700},{"timestamp":160680960
0,"price":119.9400},{"timestamp":1606813200,"price":70.2500},{"timestamp":160
6816800,"price":71.5800},{"timestamp":1606820400,"price":86.4500},{"timestamp
":1606824000,"price":82.8100},{"timestamp":1606827600,"price":75.5900},{"time
stamp":1606831200,"price":119.9300},{"timestamp":1606834800,"price":147.9700}
,{"timestamp":1606838400,"price":149.9400},{"timestamp":1606842000,"price":77
.8000},{"timestamp":1606845600,"price":71.3800},{"timestamp":1606849200,"pric
e":60.6600},{"timestamp":1606852800,"price":55.4700},{"timestamp":1606856400,
"price":50.1600}],"fi":[{"timestamp":1606773600,"price":15.0500},{"timestamp"
:1606777200,"price":13.1800},{"timestamp":1606780800,"price":12.4200},{"times
tamp":1606784400,"price":12.4100},{"timestamp":1606788000,"price":12.5400},{"
timestamp":1606791600,"price":12.9100},{"timestamp":1606795200,"price":38.430
0},{"timestamp":1606798800,"price":56.0300},{"timestamp":1606802400,"price":7
3.2500},{"timestamp":1606806000,"price":75.5700},{"timestamp":1606809600,"pri
ce":119.9400},{"timestamp":1606813200,"price":70.2500},{"timestamp":160681680
0,"price":71.5800},{"timestamp":1606820400,"price":86.4500},{"timestamp":1606
824000,"price":82.8100},{"timestamp":1606827600,"price":75.5900},{"timestamp"
:1606831200,"price":119.9300},{"timestamp":1606834800,"price":147.9700},{"tim
estamp":1606838400,"price":149.9400},{"timestamp":1606842000,"price":77.8000}
,{"timestamp":1606845600,"price":71.3800},{"timestamp":1606849200,"price":60.
6600},{"timestamp":1606852800,"price":55.4700},{"timestamp":1606856400,"price
":50.1600}],"lv":[{"timestamp":1606773600,"price":20.9900},{"timestamp":16067
77200,"price":23.1900},{"timestamp":1606780800,"price":25.0200},{"timestamp":
```

[{"timestamp":1606773600,"price":20.9900},{"timestamp":1606777200,"price":23.1900},{"timestamp":1606780800,"price":25.0200},{"timestamp":1606784400,"price":25.0200},{"timestamp":1606788000,"price":33.2600},{"timestamp":1606791600,"price":35.3400},{"timestamp":1606795200,"price":38.4300},{"timestamp":1606798800,"price":56.0300},{"timestamp":1606802400,"price":73.2500},{"timestamp":1606806000,"price":75.5700},{"timestamp":1606809600,"price":119.9400},{"timestamp":1606813200,"price":70.2500},{"timestamp":1606816800,"price":71.5800},{"timestamp":1606820400,"price":86.4500},{"timestamp":1606824000,"price":82.8100},{"timestamp":1606827600,"price":75.5900},{"timestamp":1606831200,"price":119.9300},{"timestamp":1606834800,"price":147.9700},{"timestamp":1606838400,"price":149.9400},{"timestamp":1606842000,"price":77.8000},{"timestamp":1606845600,"price":71.3800},{"timestamp":1606849200,"price":60.6600},{"timestamp":1606852800,"price":55.4700},{"timestamp":1606856400,"price":50.1600}]

[{'timestamp': 1607590800, 'price': 90.26, 'date': '2020-12-10', 'hour': 11, 'top_1': True, 'top_2': True, 'top_3': True}, {'timestamp': 1607594400, 'price': 90.76, 'date': '2020-12-10',

- **Mida minu programm main.py selle json-iga teeb?**
- Filtreerime välja meid huvitavad andmed: päringut tehes anname ette kuupäevad, saadud vastusest filtreerime välja Eesti hinnad.
- Json-ist saab sisuliselt järjend, mille elementideks on sõnastikud iga tunni kohta hinnainfo ja timestampiga.
- Teeme timestampi inimlikult loetavaks, ehk lisame kuupäeva võtme „date“ ja tunni võtme „hour“.
- Leiame 3 kõige kõrgema elektrihinnaga tundi. Lisame vastavad võtmed ka sõnastikku: top_1, top_2 ja top_3 ja anname neile *boolean* väärtused True/False.

Edasi…

- Talletame saadud andmed failis. Iga kord, kui küsime API-ga uusi andmeid, võtame aluseks vanad andmed, võrdleme, kas on tulnud uusi hindu, lisame uued hinnad, kustutame aegunud info ja kirjutame faili sisu üle.
- Meil on nüüd kenasti loetav, automaatselt uuenev fail maindata.txt, mille poole teised programmid, mis juhivad mingeid konkreetseid seadmeid, saavad pöörduda.
- Näiteks soojuspumba tööd juhib programm soojuspump.py:

- Jälgib andmefaili iga 5 min tagant;

- Kui võti top_3 on False, lülitab soojuspumba välja. Võti top_3 on False kolmel kõrgeima hinnaga tunnil.

Kliendiprogramm soojuspump.py:

```
import json
import time
from inputimeout import inputimeout, TimeoutOccurred

deviceOn = True
try:
   fileName = inputimeout('Sisesta andmefaili nimi (vaikimisi maindata.txt): ', timeout
except TimeoutOccurred:
   fileName = 'maindata.txt'

# Timeri:
while True:
   # Mida teha siis, kui faili ei ole:
   try:
      with open(fileName, 'r') as filehandle:
         mainData = json.load(filehandle)
         filehandle.close()
   except IOError:
      print('file not found')
      print('Raspberry puhul anname siin ühele väljundile pinge peale, et LED alarmeer
   if mainData[0].get('top_3') is False:
      deviceOn = False
   else:
      deviceOn = True
   time.sleep(300)
```

Põhiprogramm main.py:

```
# Projekt Raivo Kasepuu, matrikkel B710710, MTAT 03 256

import time
import datetime
import json
import requests
from os import path
from requests.exceptions import ConnectionError
from inputimeout import inputimeout, TimeoutOccurred

def getDateTimeNow():
    # käesoleva hetke inimkeeli loetava kuupäeva ja kellaaja leidmine
    return
datetime.datetime.fromtimestamp(time.time()).isoformat()


def getHumanDateTime(timestamp):
    # inimkeeli loetava kuupäeva ja kellaaja leidmine etteantud
timestamp-ist
    return
datetime.datetime.fromtimestamp(timestamp).isoformat()


def getDateForTomorrow():
    # homse kuupäeva leidmine
    return datetime.datetime.fromtimestamp(time.time() + 24 *
3600).isoformat().split("T")[0]


def getDateForYesterday():
    # eilse kuupäeva leidmine
    return datetime.datetime.fromtimestamp(time.time() - 24 *
3600).isoformat().split("T")[0]


def getDateForToday():
    # tänase kuupäeva leidmine
    return
datetime.datetime.fromtimestamp(time.time()).isoformat().split("T
")[0]


def getTimestampNow():
    # hetke timestampi leidmine
    return int(time.time())

def makeEleringUrl():
    # genereerib Eleringi turuhindade teenuse küsimiseks url-i
    # Näiteks https://dashboard.elering.ee/api/nps/price?end=2020-
12-01%2021%3A00&start=2020-11-30%2022%3A00
    # annab meile jsoni Eleringi hindadega 1.12.2020 -ks
    start = "start=" + str(getDateForYesterday()) + "%2022%3A00"
    end = "end=" + str(getDateForTomorrow()) + "%2021%3A00"
    return "https://dashboard.elering.ee/api/nps/price?" + end + "&"
+ start

def getEleringEePrices(url, market):
    while True:
        try:
            result = requests.get(url).json()["data"][market]
            if isinstance(result, list):
                return result
        except ConnectionError as e:
            # prindime erro9ri terminalile
            print(e)
            # ootame 10 sekundit ja proovime uuesti kogy try blokki
            time.sleep(10)


def addDateAndHourToPriceData(priceData):
    for i in range(len(priceData)):
        # lisame priceData sõnastikule kuupäeva:
        priceData[i]['date'] =
str(getHumanDateTime(priceData[i].get('timestamp')).split("T")[0])
        # lisame pricedata sõnastikule kellaaja tunnid integerina:
        priceData[i]['hour'] =
int(str(getHumanDateTime(priceData[i].get('timestamp')).split("T")[
1].split(":")[0]))
    return priceData


def addTopThreeHours(priceData):
    # Lisame priceData tänase päeva sõnastikele lisaväljad top_1 ..
top_3
    # markeerides sõnastikke vastavalt top3 hindadele
    topPriceOne = 0
    maxFirstHour = 0
    topPriceTwo = 0
    maxSecondHour = 0
    topPriceThree = 0
    maxThirdHour = 0
    for i in range(len(priceData)):
        if priceData[i].get('price') > topPriceOne and
priceData[i].get('date') == getDateForToday():
            topPriceThree = topPriceTwo
            maxThirdHour = maxSecondHour
            topPriceTwo = topPriceOne
            maxSecondHour = maxFirstHour
            maxFirstHour = priceData[i].get('hour')
            topPriceOne = priceData[i].get('price')

    for i in range(len(priceData)):
        if priceData[i].get('hour') == maxFirstHour:
            priceData[i]['top_1'] = False
            priceData[i]['top_2'] = False
            priceData[i]['top_3'] = False
        elif priceData[i].get('hour') == maxSecondHour:
            priceData[i]['top_1'] = False
            priceData[i]['top_2'] = False
            priceData[i]['top_3'] = True
        elif priceData[i].get('hour') == maxThirdHour:
            priceData[i]['top_1'] = False
            priceData[i]['top_2'] = False
            priceData[i]['top_3'] = True
        else:
            priceData[i]['top_1'] = True
            priceData[i]['top_2'] = True
            priceData[i]['top_3'] = True
    return priceData


def fileCreationIfNeeded(fileName):
    if path.exists(fileName) is False:
        emptyData = [{"timestamp": 1}]
        with open(fileName, 'w') as filehandle:
            json.dump(emptyData, filehandle)
        filehandle.close()


# Määrame failinime, kus hoiame oma andmeid:
try:
    fileName = inputimeout('Sisesta andmefaili nimi (vaikimisi
maindata.txt): ', timeout=10)
except TimeoutOccurred:
    print('valisin vaikimisi andmefailiks maindata.txt')
    fileName = 'maindata.txt'

# Kui selline fail puudub, siis tekitame selle koos triviaalsete
algandmetega:
fileCreationIfNeeded(fileName)

# Loeme faili sisu json-i:
with open(fileName, 'r') as filehandle:
    mainData = json.load(filehandle)
    filehandle.close()

# Timeri bloki algus:
while True:
    # küsime viimaseid hindu Eleringi API abil
    url = makeEleringUrl()
    market = "ee"
    priceData = getEleringEePrices(url, market)
    # arvutame ja lisame json-ile kuupäeva ja tunni
    addDateAndHourToPriceData(priceData)
    # markeerime top3 hindadega sõnastikud json-is
    addTopThreeHours(priceData)
    # uuendame mainData väärtusi:
    for i in range(len(priceData)):
        if priceData[i].get('timestamp') > mainData[-
1].get('timestamp'):
            mainData.append(priceData[i])
    # Kustutame üle 1h (3600 sec) vanad sõnastikud:
    while mainData[0].get('timestamp') < (getTimestampNow() -
3600):
        mainData.pop(0)
    # Prindime mainData terminalile:
    print(mainData)
    # Kirjutame andmete faili üle uute andmetega
    with open(fileName, 'w') as filehandle:
        json.dump(mainData, filehandle)
        filehandle.close()
    # kordame rutiini teatud aja jooksul sekundites.
    # Antud juhul 1h (3600sec). Testimiseks sobib näiteks 5 sekundit:
time.sleep(5)
    time.sleep(3600)
```

- **Koodist, projektist ja tulevikust...**
- Kõik kood töötab autonoomselt. Voolu katkemisel, internetiühenduse puudumisel, faili puudumisel ei juhtu midagi, mis nõuaks inimese sekkumist.
- Reaalses maailmas on kasutatud Raspberry Pi miniarvutit, mis lülitab seadmeid sisse - välja GPIO pin-ide abil.
- Minu projekti failid on saadaval GitHub-is:

https://github.com/RaivoKasepuu/UTpythonProject.git

- Erinevaid API-sid on maailm täis! Oska vaid googeldada!
- Võib lisada näiteks kontrolleri funktsiooni, mis süütab õues valgustuse, kui Sinu asukohas on Päike loojunud:

https://sunrise-sunset.org/api

Küsimused?

Aitäh!
API-likke Jõule!