1. Rühmatöö

# Elektri ööpäeva tunnihindade analüüs

Aines:

Objektorienteeritud projekteerimine

(IT mitteinformaatikutele)

Jaak Ivask, Peeter Niidas

2020/21 kevad

# Projekti kirjeldus

Projekti eesmärk on koduse säästva elektritarbimise automatiseerimine läbi elektri tunnihindade analüüsi. Koostatava programmi väljundina esitatav kõrgete ja madalate elektrihindade info kasutab ära koduautomaatika, mis vastavalt vähendab ruumide temperatuuri või ekstreemtippude korral võib kuni paariks tunniks soojuspumba ka välja lülitada. Samas info eeloleva kõrge elektrihinna kohta annab automaatikale aega reageerida ning vajadusel madalama elektrihinna perioodil lühiajaliselt teostada ka nn ülekütmist ehk tõsta ruumide temperatuure seadepunktist kõrgemale. Viimane tegevus on vajalik madalate välistemperatuuride korral mugava sisekliima tagamiseks. Samuti saab antud infot kasutada ruumide ventilatsiooni analoogiliseks juhtimiseks, valgustuse reguleerimiseks jne.

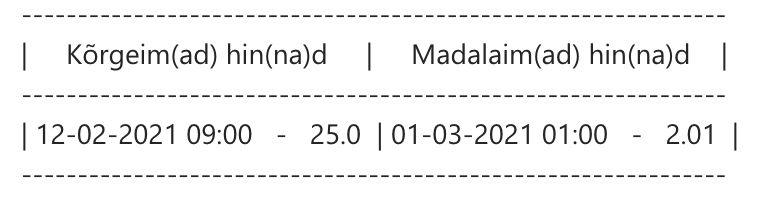
Reaalajas järgneva 24 tunni elektrihind määratakse Nord Pool elektriturul (<https://www.nordpoolgroup.com/>), kuid nendelt info saamine on tasuline teenus. Seega eeloleva perioodi elektri turuhinna info saame Elering Live veebileheküljelt (<https://dashboard.elering.ee/et>), kes tegelikult edastab ise sama Nord Pool’i informatsiooni.

Elering on kokku pannud oma API (<https://dashboard.elering.ee/v2/api-docs>), millest kasutame oma programmi juures *nps-controller*’i päringut *day-ahead price data*.

Tähelepanu peab juhtima asjaolule, et antud rakendusega saadav hind ei ole lõpphind tarbijate jaoks. Sellele arvestab iga elektrimüüja juurde oma marginali. Antud programmi kasutuseotstarbe seisukohalt aga ei ole see määrav, sest hindade relatiivne muutus ajas on info, mida saame kasutada automaatika juhtimisel juba järgmistes etappides. Näitena võiks tuua 12. veebruari käesoleval aastal, kus vahemikus 9:00 – 10:00 leiame oma rakendusega suure tipu (25 senti/kWh), kuid Eesti Energia müüdav börsihind oli antud hetkel 30 senti/kWh. Samas aga tipp oli ning mõistlik oleks olnud antud hetkel tarbimist piirata.



*Eesti Energia börsihind*



*Rakendusega leitud tarbimise tipp*

# Märkus programmi kasutamiseks

Antud rakendus toetub oma töös 3 library kasutamisele:

* json (json-simple-1.1.11.jar)
* junit (junit-4.101.jar)
* hamcrest (hamcrest-core-1.11.jar)

# Programmi klassid

## Klass EleringJsonApi

Pöördub Eleringi API poole ning teeb päringu vastavalt etteantud ajaperioodile ning tagastab JSON objekti. Kasutatavad meetodid on:

* Konstruktor (restEndPoint) - määrab vaikimisi Elering API endpointi.
* getEleringData() – kasutab väärtustatud isendivälju. Tagastab JSON objekti. Lisaparameetreid ei ole
* Private setEleringApiUrl() - Koostab vastavalt väärtustatud algus ja -lõppkuupäevale, baasurl ja API endpointile päringu URL. Kutsutakse välja meetodis getEleringData().
* Setter meetodid restEndPoint, alg ja lõpukuupäevale.

## Klass SisestatudKuupäevadeKontroll

**Klassi meetodeid kasutatakse:**

* Fikseeritud kuupäevade genereerimiseks
* Kasutajalt kuupäevade vahemiku küsimiseks
* Kasutaja sisestatud kuupäevade vahemiku kontrollimist vastavalt Eleringi API nõuetele.
* Kasutatud meetod salvestab kalkuleeritud või sisestatud kuupäevad isendi muutujasse.
* Hetkel kehtivad kuupäevad on kättesaadavad get meetodite kaudu.
* Algus ja lõpukuupäevade vahe peab olema väiksem, kui 365 päeva.

**Klassis on järgmises meetodid:**

* getKuupäevad() - küsib kasutajalt algus ja lõppkuupäeva ning saadab sisestud kuupäevad kontrolli. Kui leitakse viga, siis korratakse protsessi.
* getHomnePäev() - genereerib 24h aja. Algusaeg on kasutaja poolt käivitatud meetodi aeg. Sellele liidetakse juurde 24h
* getRandomPäev() - genereerib viimase aasta vahele suvalise kuupäeva.
* Private kuupäevadeKontroll() - Kontrollib kasutaja poolt sisestud kuupäevade vastavust nõuetele.

## Klass Elektrihind

Klass elektrihind sisaldab konstruktorit elektritarbimise aja ja hinnaga, antud parameetrite getter’eid ja setter’eid ning implementeerib Comparable liidest sorteerimaks andmeid elektrihinna alusel.

## Klass KuvaElektriHind

Kasutab vaikimisi konstruktorit. Lähteandmed antakse kasutatava meetodi parameetritega. Kasutab JSON andmed ning sellest selekteeritakse välja kõrgemad ning madalamad tunnihinnad. Soovi korral saab seda teha riikide kaupa. Kui päringu periood ei olnud *day-ahead* vaid pikem periood minevikust, siis leitakse keskmine elektrihinna kõikumine antud perioodil. Kasutatakse järgmisi meetodeid:

* Private loeJson
  + saab sisendina Json ojekti ja riigi tähise, loeb eeltoodud objektist timestamp’i, teisendab selle meile sobivasse formaati meetodiga tunnidTimestampist, loeb elektrihinna, arvutab selle ümber EUR/MWh -> senti/kWh ning kannab need ArrayList’i elektrihind klassi Elektrihind kasutades
* topUp
  + saab sisendina numbrilise parameetri, kui pikka Top’i on vaja ning leiab ArrayList’ist elektrihind vastava arvu kõrgeima hinnaga objekte, seejärel salvestab need ArrayList’i tipud
* topDown
  + saab sisendina numbrilise parameetri, kui pikka Top’i on vaja ning leiab ArrayList’ist elektrihind vastava arvu madalaima hinnaga objekte, seejärel salvestab need ArrayList’i põhjad
* leiaKeskmine
  + parameetrita meetod, mis kalkuleerib ArrayList’i elektrihind objektidest keskmise elektrihinna ning tagastab selle ümardatuna
* tunnidTimestampist
  + saab sisendina timestamp’i ning teisendab selle inimloetavaks kuupäevaks ja kellaajaks
* kuvaHomseElektriHinnaTabel
  + saab sisendina Json objekti mingi perioodi andmetega ja riigi tähisega, pöördub meetodite loeJson, topUp ja topDown poole, kogub andmed ja väljastab järgneva 24 tunni TOP 3 ning TOP -3 elektrihinnad
* määratudVahemikuMinMax
  + saab sisendina Json objekti mingi perioodi andmetega ja riigi tähisega, pöördub meetodite loeJson, topUp, topDown ja leiaKeskmine poole, kogub andmed ja väljastab määratud vahemiku kõrgeima, madalaima ja keskmise elektrihinna. Hardcoded parameetriga topPikkus on võimalik saada ka suuremate TOP’ide hindu
* kuvaSuvalineElektriHind
  + saab sisendina Json objekti mingi perioodi andmetega, pöördub korduvalt meetodi loeJson poole ning salvestab 4 erineva riigi etteantud perioodi elektrihinnad oma ArrayList’idesse, väljastab 4 riigi elektrihindade tabeli
* NulliListid
  + kustutab listidest elektrihind, tipud, põhjad eelmise tsükli jooksul kogutud andmed

## Klass Main

Selles toimub kasutajaga infovahetus, mille käigus küsitakse ajaperioodi ning väljastatakse Elektrihinnad klassi meetodite abil kalkuleeritud tulemused.

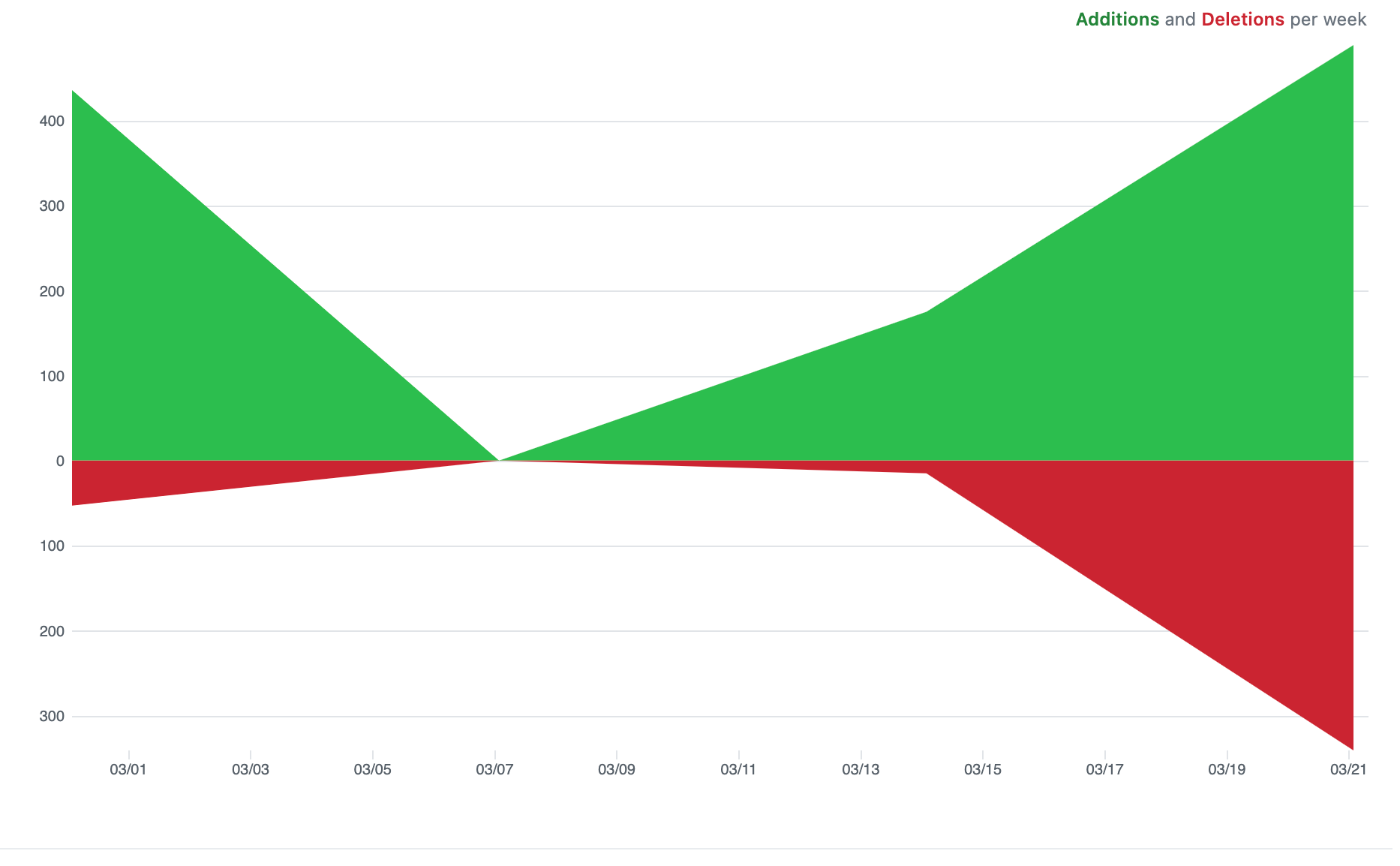
# Projekti tegemise protsessi kirjeldus

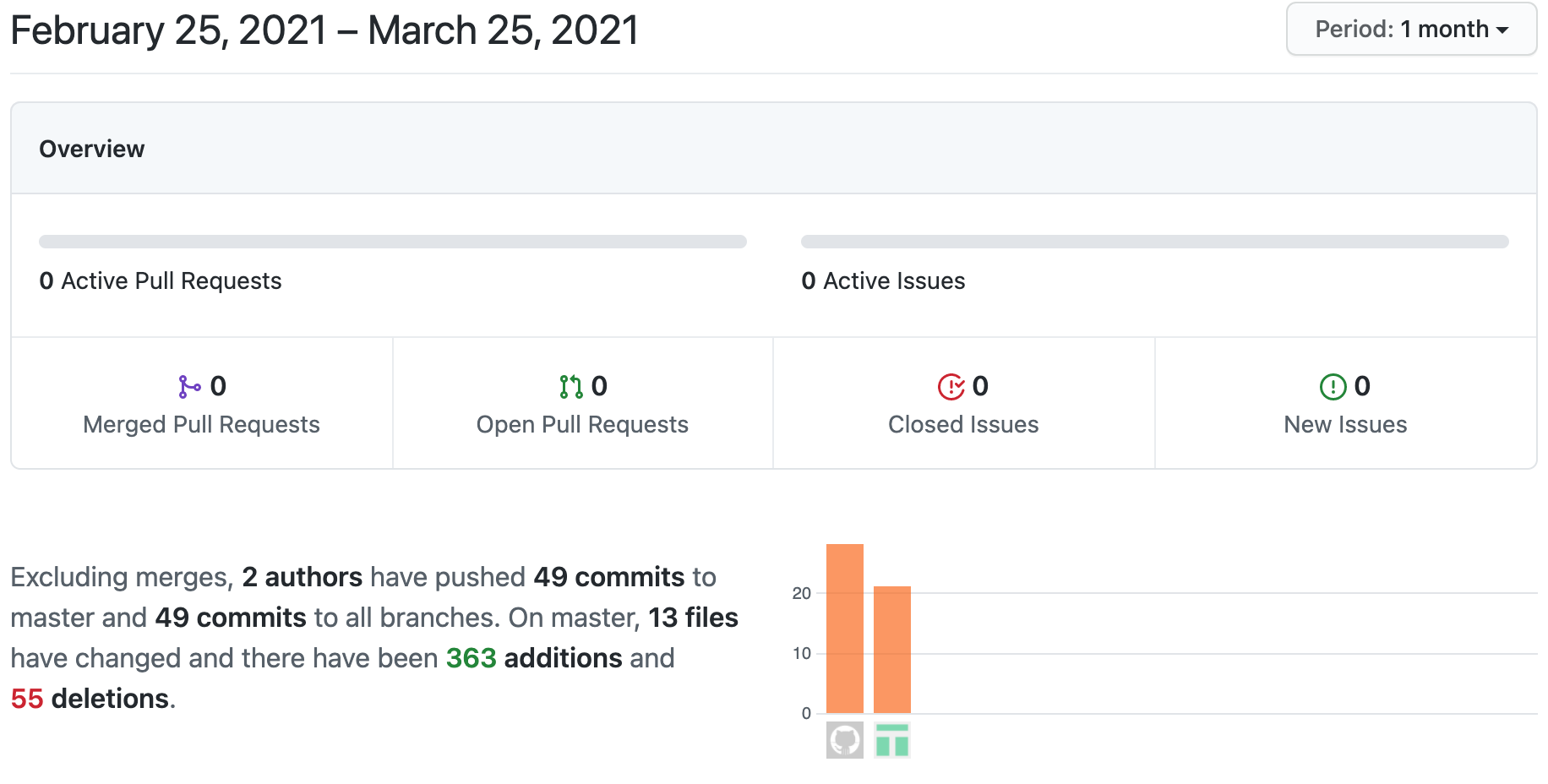
Projekt sai kirjutadud kasutades GitHub keskkonda. Kirjutamise ajal suhtlesid autorid omavahel ka konstantselt Teams’i rakenduses. Viimane oli eriti vajalik tagamaks, et erinevad kirjutatud koodid repo’s ei hakkaks üksteisega konfliki minema või ei läheks Update/push protsessides kaduma. Autorid üritasid tööd ühiselt teha, kuid jagatud olid klassid siiski põhiliselt järgnevalt:

|  |  |
| --- | --- |
| Jaak Ivask | Peeter Niidas |
| EleringJsonApi | Elektrihind |
| SisestatudKuupäevadeKontroll | KuvaElektriHind |
| Main | Main |

Klasside valmimine oli plaanitud alljärgnevas järjekorras: EleringJsonApi->SisestatudKuupäevadeKontroll->Main->Elektrihind->KuvaElektriHind->Main. Reaalsuses aga käis erinevate klasside arendamine ja muutmine vaheldumisi juba valmiskirjutatud lõikude kontrollimisega. Mitmel pool sai kasutatud ka vahetulemuste kontrollimiseks erinevate andmete ekraanile väljastamist.

Ajakulu projekti tegemisele hindavad mõlemad autorid orienteeruvalt 30 astronoomilise tunni piires. Lisatud indikaatorina ka 2 tabelit GitHub’i insight kokkuvõttest:





Projektiga antud staadiumis võivad autorid üldjuhul rahul olla. Kindlast oleks võimalik koodi lühemaks ja efektiivsemaks muuta. Lisaks peaks uuritava maa andmete leidmise olemasolevast loeJson meetodist eraldama teise meetodi. Vormistuse poolelt võiks tabelite väljastamine arvestada, et elektri tunnihind võib minna suuremaks kui 9,99 senti/kWh ning hinna kuvamiseks jäetud positsioonide arvu vastavalt sellele dünaamiliselt kohandada säilitamaks ilus ja ühtlane üldpilt.

# Raskused projekti koostamisel

Üheks suureks murekohaks oli vähene kogemus/oskused GitHub’i kasutamisel. Esines olukordi, kus ühe osapoole üles laetud versiooni ja teise osapoole muudatuste lisamine ei lõppenud oodatud tulemusega ja esines ka mõningat kirjutatud koodi kadumist. Teiseks takistuseks oli java keele vähene tundmine - suur hulk võimalusi kuupäevadega tegelemiseks, erinevad JSON teegid jm. Oli teada mida programm peab tegema, isegi mis sammud tuleb selleks teha, kuid kuidas seda javas välja kirjutada vajas proovimist, juhendmaterjalide ja näidiste läbiproovimist.