TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Andreas Saltsberg 155242IAPB

TÄNAVAVALGUSTUSE INTEGRATSIOONISÜSTEEMI KASUTAJALIIDES TARTU LINNALE JA HARKU VALLALE

Bakalaureusetöö

Juhendaja: Tanel Tammet
PhD

Tallinn 2018

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Andreas Saltsberg

21.05.2018

Annotatsioon

Käesoleva lõputöö eesmärgiks on luua Tartu linna ja Harku valla tänavavalgustite integratsioonisüsteemile kasutajaliides. Kasutajaliidese eesmärgiks on võtta valgustite andmed mitmest süsteemist ja näidata neid kombineeritult kujul. Kaks süsteemi edastavad andmeid valgustite häirete kohta ning üks süsteem annab nende valgustite detailandmed.

Töö esimeses osas analüüsitakse süsteeme, millega kasutajaliides end seob ja tuuakse välja, mis on nende eesmärgid. Tuuakse välja rakenduse funktsionaalsed ja mittefunktsionaalsed nõuded, mis peavad olema töö valmimisel täidetud. Järgmises osas räägitakse rakenduse realiseerimisest ja erinevate probleemide lahendamisest. Lisaks sellele kirjeldatakse ka kasutatud tehnoloogiaid. Viimases osas tuuakse välja võimalikud edasised arengusuunad, mille poole püüelda.

Töö tulemusena valmis kaardirakendus, kus on kuvatud valgustid ja nende staatused. Valgusteid on võimalik filtreerida ja ka muuta nende värve vastavalt andmetele. Lisaks sellele loodi juurkasutajatele profiilide haldamise vaade, kus on võimalik profiilide andmeid näha, neid lisada ja eemaldada. Kasutajaliidesesse sisenemiseks kasutatakse profiile ning igale profiilile vastavad mingi ala või mingi tootja valgustid.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 38 leheküljel, 4 peatükki, 9 joonist, 4 tabelit.

Abstract

Integrated streetlight system user interface for Tartu and Harku

The purpose of this thesis is to create a map application for street lights. The application is meant for two areas – city of Tartu and Harku parish. Before this project there was no way to visualize data in a way that combined street light alerts and their general data. The objective was to create a user interface that show status from all systems in one place. To get access to the map, user need to authenticate with a profile. Profiles are managed by local governments where they can view profiles, create new ones or remove them. Each profile shows specific data, for example Tartu local government only sees their city lights.

In the first part of the thesis author describes all the systems that are related to the project and brings out their purpose. Additionally author names all the functional and non-functional requirements, which must be met. Next section is about the implementation of the application and how each problem was solved. In the last part of the thesis author suggests new directions where the application can expand towards.

The result of this work is a web application that has main three views – authentication, map and profile management. Every street light on the map can be clicked to see details about the light. Lights can be filtered by their status and also color coded by different types of data.

The thesis is in Estonian and contains 38 pages of text, 4 chapters, 9 figures, 4 tables.

Lühendite ja mõistete sõnastik

SCSS	Stiililehe keel kasutajaliideste kujundamiseks
API	Rakendusliides
JWT	JSON-il baseeruv tõendi formaat
JSON	Javascript Object Notation, andmevahetuse formaat
HTTP	Hüperteksti Ülekande Protokoll
authorization	HTTP päis, milles on autentimisega seotud andmed
HTML	Hüperteksti märgistuskeel
XML	Laiendatav märgistuskeel
token	luba

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Antud bakalaureusetöö eesmärgiks on arendada kaardirakendus ja kasutajate haldamise süsteem. Rakendus arendatakse ettevõtte CGI Eesti AS jaoks ning selle ülesanne on kuvada Tartu linnavalitsuse ja Harku valla tänavavalgustite andmed. Tänavavalgusteid on linnades palju ja nende haldamine võib osutuda keerukaks. Kaardi peal peab olema võimalik lihtsasti valgusteid filtreerida ja eristada erinevate olukordade järgi, milleks on näiteks valgustite veateated, tootjad ning haldajad. Rakendus peab olema kasutaja jaoks võimalikult mugav ja andma hea ülevaate hetkeseisust. Kasutajateks hakkavad olema erinevad kliendid ja sellepärast on ka tähtis õiguste piiramine. See tähendab seda, et näiteks Tartu linnavalitsuse kasutajad näevad ainult enda linna valgusteid ja mitte teiste omi. Arendatud rakendus on mõeldud veebiplatvormile, et ligipääs oleks võimalikult lihtne ja paindlik.

1.1 Taust ja probleem

Tänavavalgustisüsteeme on linnades erinevaid ja tihtipeale on linnas kasutusel rohkem kui üks süsteem. Puudub ühtne formaat ja kõik antud süsteemide rakendusliidesed edastavad andmeid erineval kujul. Projektile, millele kasutajaliidest arendatakse, oli eelnevalt see probleem ära lahendatud – kõikide süsteemide andmed konverteeritakse ühtsele kujule. See võimaldab luua kasutajaliidese, kus on kõikide süsteemide andmed ühes koos ja kaardi peal ära visualiseeritud, andes ülevaate tervest linnast või vallast.

1.2 Ülesandepüstitus

Tänavavalgustid saadavad pidevalt andmeid oma signaalide kohta. Enne seda projekti puudus lahendus, mis annaks ülevaate kõikidest valgustitest ja nendega kaasnevatest andmetest. Hetkel koondatakse osad andmed ArcGIS-i andmehoidlasse ja kuna tänavavalgustite andmed saadetakse ArcGIS-i läbi käimasoleva projekti, on võimalik luua andmebaas, kuhu salvestatakse kõik süsteemist läbikäivad andmed. Loodava andmebaasi eesmärgiks on andmete pärimise kiiruse parandamine, et andmed jõuaksid kasutajaliidesesse nii väikse viivitusega kui võimalik.

Autor on loonud süsteemi, mis annab lahenduse eelnevalt kirjeldatud probleemidele. Rakenduse põhiosaks on kaart, kus on ära märgitud kõik piirkonna valgustid ning mida on võimalik filtreerida veateadete kaudu ja eristada mitme erineva loogika järgi:

- Eristamine veateadete kaudu – võimaldada eristada valgusteid, mille kohta on teada veasignaale ning eristada neid teistest õigesti töötavatest valgustitest.
- Eristamine tänavavalgustite tootja ja haldaja kaudu
- Eristamine valguslampide järgi – eesmärgiks on anda ülevaade, milliste valgustitega on tegu. See on kasulik valgustite haldajate jaoks, kes tegelevad nende parandamisega.

Kõik andmed valgustite kohta saab kätte kaardil mingi kindla valgusti peale vajutades, kus on ära märgitud ka kõik filtreeritavad andmed.
Teine osa tööst on autentimise loomine ja võimaldada lisada alamkasutajaid, kellel on piiratud ligipääs.

1.3 Metoodika

Kasutajaliides arendatakse kasutades veebiraamistikku nimega ReactJS. Rakenduse kompileerimisel genereeritakse HTML ja Javascript klassikalisel kujul. Küljenduskeeleks kasutatakse SCSS-i, millest tekib kompileerimisel CSS. SCSS lisab küljenduskeelele kasulikke funktsionaalsusi, muutes keele paindlikumaks. Lisaks sellele on kasutusel ka Google Maps ning veebikomponentide kogu AntDesign. Kasutajaliides arendatakse nii, et seda oleks võimalik kasutada ka mobiilseadmes. Serveris jookseb Spring Boot rakendus ning andmed salvestatakse MongoDB andmebaasisüsteemi.

Antud töö arenduse tagasisidet saadi CGI projekti liikmete käest, kellega arutati läbi kõik funktsionaalsused ning kontrolliti, kas arendatud töö vastab oodatud tulemustele.

1.4 Ülevaade tööst

Esialgselt analüüsitakse tööga seotuid süsteeme. Uuritakse, mis on olemas ning mis veel võiks olemas olla. Tuuakse välja loodava rakenduse kõik funktsionaalsed kui ka mittefunktsionaalsed nõuded, kus on ära kirjeldatud, kuidas rakendus peab töötama. Lisaks sellele kirjeldatakse süsteemi objekte, et aru saada, milliste andmetega üldse on tegu ja kuidas on need omavahel seotud.

Teiseks sammuks on kirjeldada kõik funktsionaalsused ning need realiseerida. Töös tuuakse välja, mis on realiseeritud ja kuidas erinevad funktsionaalsused töötavad. Kirjeldatakse, kuhu mahub arendatud osa tervikusse süsteemi ning kuidas rakendus suhtleb teiste süsteemiosadega. Lisaks sellele nimetatakse erinevad raskuskohad ja kuidas need lahendati.

Viimases osas tuuakse välja arendussuunad, kuidas võiks projektiga edasi käituda ning mida veel oleks võimalik arendada.

2 Analüüs

Antud peatükis tuuakse välja ning analüüsitakse tööga peamiselt seotud süsteeme, milleks on ArcGIS, Gridens ja CityIntel.

Kirjeldatakse funktsionaalsed ja mittefunktsionaalsed nõuded, mida rakendus peab täitma. Tuuakse välja peamised objektid, mida süsteemis kasutatakse ja selgitatakse lühidalt, mis midagi on. Viimases alampeatükis kirjeldatakse töö komponendid ja mis eesmärki nad täidavad.

2.1 Tööga seotud süsteemid

2.1.1 ArcGIS

ArcGIS on geograafiliste informatsiooni talletamiseks mõeldud süsteem. ArcGIS on loodud Esri poolt ning seda kasutatakse mingi ala infrastruktuuri kaardistamiseks ja andmete salvestamiseks. Süsteemi kasutavad antud töö raames Tartu linnavalitsus ja Harku vallavalitsus. Süsteemis on võimalik hoida informatsiooni näiteks tänavavalgustite, veetorude, liinide ja elektrikappide kohta. Süsteemis on sisemine andmebaas, millest on võimalik pärida andmeid ning samuti saab andmed kätte ka läbi rakendusliidese. Päringud on väga paindlikud – saab valida formaadi (JSON, XML, HTML) ning täpsustada, milliseid andmeid on vaja. Lisaks sellele saab ka pärida ainult mingi kindla ala andmeid. Üheks tähtsaks ArcGIS-i funktsionaalsuseks on andmete visualiseerimine. Üheks visualiseerimise meetodiks on kaart, kus on ära näidatud kõik süsteemis olevad andmed – näiteks kõik valgustid ning valgusti peale vajutades on näha selle antud valgusti andmeid.
ArcGIS-is on võimalik andmeid analüüsida ning koostada aruandeid. Süsteem annab hea ülevaate olemasolevast infrastruktuurist ning muudab probleemide tuvastamise lihtsamaks.

2.1.2 Gridens

Gridens on juhtimissüsteem, mis võimaldab tänavavalgustite energiatarbimist vähendada. Gridens-i valgusteid on võimalik dünaamiliselt muuta. See tähendab seda, et valgustitele saab luua profiile – mis kell mingi valgustus peaks põlema, mis valgustasemega peaks valgusti töötama ja nii edasi. Lisaks sellele on võimalik ka valgusteid käivitada liikumisandurite abiga, mis omakorda aitab ka säästa energiat.

Läbi Gridens-i rakendusliidese on võimalik pärida valgustite hetkeseisu. ArcGIS annab üldised andmed valgustite kohta – näiteks asukoha, lambi tüübi ja haldaja. Gridens lisaks saadab valgustite kohta täpsemaid andmeid – näiteks veateateid, palju voolu tarbitakse ja valgustite temperatuure. Antud rakenduses on just kõige olulisemad veateated. Kontrollerite käest on võimalik küsida temaga ühendatud valgustite staatust – kas esineb mingeid teadaolevaid probleeme.

2.1.3 CityIntel

CityIntel-i eesmärgiks on ka luua nutikaid tänavavalgusteid, mida on võimalik konfigureerida vastavalt oludele. Täpselt nagu Gridens, CityIntel-i kontrolleritega on võimalik muuta valgustite profiile. Profiile saab ka muuta vastavalt ilmastikuoludele – näiteks vihmase ilma korral saaks valgustite valgustugevust tõsta. Ka CityIntel-i kontrollerid jälgivad valgustite signaale ja tagastavad vajalikku statistikat – kas valgustiga esineb mingeid probleeme ning milliseid.

2.1.4 Tänavavalgustuse API

Töö taustaks on projektis varem arendatud rakendusliides, mille eesmärgiks on andmed erinevatest tänavavalgustisüsteemidest muuta samaks formaadiks, et edasi saata andmed ArcGIS-i. Seda on vaja selleks, et kõik valgustite andmed, milleks on näiteks valgustite veateated, oleks ühtselt mõistetav.

2.2 Veebirakenduse nõuded

Nõuded jagunevad kaheks – funktsionaalseteks ja mittefunktsionaalseteks nõueteks. Funktsionaalsed nõuded kirjeldavad funktsionaalsust, mis veebirakenduses peab olema realiseeritud ning mittefunktsionaalsed nõuded pigem kirjeldavad seda, kuidas süsteem peaks töötama tervikuna. Nõuete loetelu sõnastati koos CGI arendustiimiga, kes ka kontrollivad nõuete täidetavust.

2.2.1 Funktsionaalsed nõuded

- Veebirakendusse on võimalik sisse logida.
- Autentimise ebaõnnestumisel peab nägema põhjust veateatena.
- Veebirakenduses on võimalik välja logida.
- Lehe taasavamisel säilitatakse sessioon tingimusel, et see pole aegunud.
- Rakenduses on kaardivaade.
- Kaardi peal kuvatakse autentitud profiili valgustite andmed.
- Kaardil valgusti peale vajutades kuvatakse valgusti detailandmed.
- Kaardi peal on võimalik valgusteid filtreerida.
- Filtreerida peab saama valgusti staatuse järgi.
- Kaardil peab olema võimalik valgusteid eristada tootja, lambipirni ja staatuse järgi.
- Valgustite andmeid ei tohi saada veebiakenduses muuta ega kustutada
- Valgusti detailandmete vaates peab kuvama kõik teadaolevad andmed.
- Valgustite päringu ebaõnnestumisel peab olema kuvatud veateade.
- Rakenduses on profiilivaade.
- Profiilivaates kuvatakse autentitud kasutaja kõiki andmeid.

- Autentitud kasutaja parooli on võimalik muuta.
- Autentitud kasutaja ülejäänud andmeid ei ole võimalik muuta.
- Rakenduses on alamprofiilide vaade.
- Alamprofiili vaade avaneb ainult vajalike õigustega kasutajatel.
- Kõik kasutaja alamprofiilid kuvatakse tabelis.
- Alamprofiile on võimalik kustutada.
- Kasutajal on võimalik lisada alamprofiile, täites ära kõik vajalikud lüngad.
- Lisatud alamprofiiliga peab olema võimalik sisse logida.
- Alamprofiilid ei saa omakorda enda alamprofiile luua.

2.2.2 Mittefunktsionaalsed nõuded

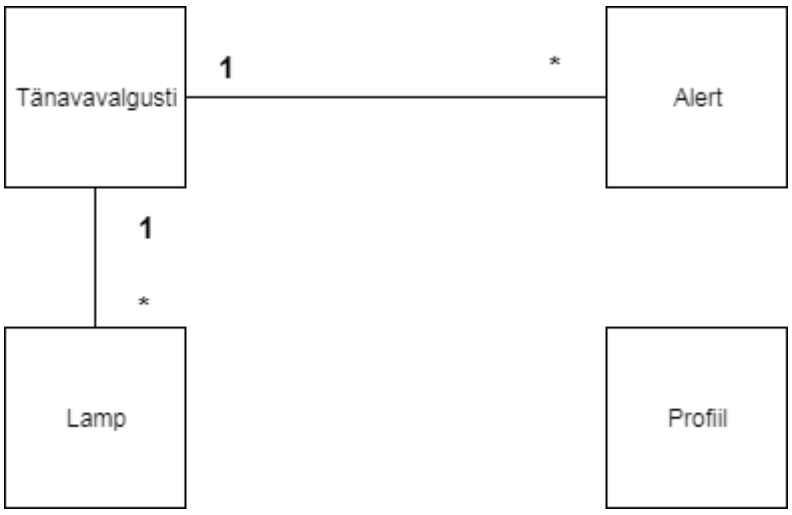
- Veebirakenduses kasutatav keel on inglise keel.
- Rakendus peab töötama igas modernses brauseris – Mozilla Firefox, Edge ning Google Chrome.
- Rakendus peab olema avatav ka mobiilseadmes.
- Kasutajaliides peab olema võimalikult lihtne ja minimalistlik.
- Kaardivaade peab olema kiire ning reageeriv – päritakse ainult seda, mida vaja.

2.3 Süsteemi objektid

Peamiseks süsteemi objektiks on vara, milleks võib olla nii valgusti kui ka valgustuskilp. Valgustuskilbis on ära märgitud kõik tema alla kuuluvad valgustid, märkides ära nende ID-d. Valgustitel võivad olla ka mitu lampi ning iga lambi kohta hoitakse andmeid eraldi.

Valgustite kohta käivad häired hoitakse eraldi objektina, kus on ära märgitud informatsioon häire kohta. Neil on ka erinevad tasemed – hoiatus, viga või häire puudub. See võimaldab need lihtsasti ära filtreerida prioriteetide järgi.

Autentimise jaoks on andmebaasis eraldi profiili objektid, mis viitavad, kus kohast andmeid pärida. Profiilis hoitakse veebirakenduses autentimiseks vajalikke andmeid ning lisaks ka välissüsteemidest andmete pärimiseks vajalikke andmeid.



Joonis 1. Süsteemi objektid ja kuidas need omavahel seotud on

3 Realiseerimine

Peatükis tuuakse välja arendatud funktsionaalsused ja näidatakse realiseerimise tulemust. Tuuakse välja, kuidas ja millisel kujul andmed kasutajaliidesesse jõuavad ja kuidas need andmed välisüsteemides välja näevad.

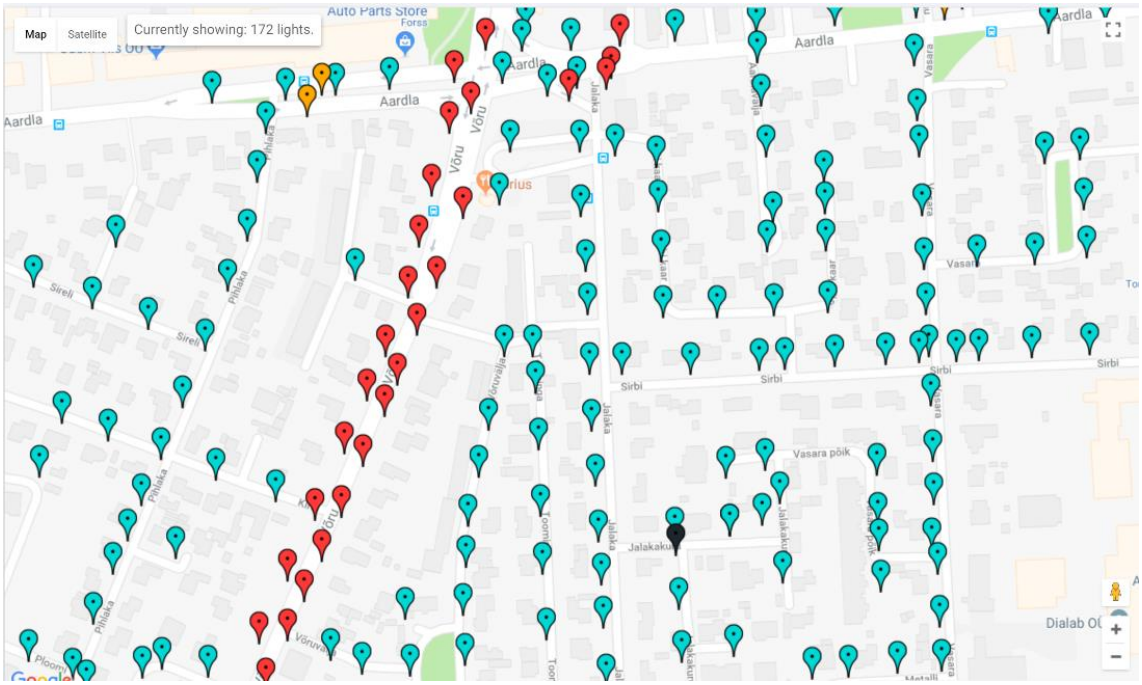
Antud peatükis tuuakse välja rakenduse arendamisel kasutatavad tehnoloogiad, kirjeldatakse ära rakenduse struktuur ning tuuakse välja erinevad keerulised kohad ja kuidas neid lahendati.

3.1 Rakenduse struktuur

Arendataval veebilehel on peamiselt kolm vaadet, milleks on kaardi, autentimise ja profiilide vaade. Kolme vaadet seob menüü, millega on võimalik nende vahel navigeerida. Autentimise vaatele saab väljalogimisega või siis, kui sessioon aegub. Peamine vaade on kaardivaade, kuhu kasutaja suunatakse automaatselt edukal sisselogimisel või kui kasutaja on juba eelnevalt sisse logitud.

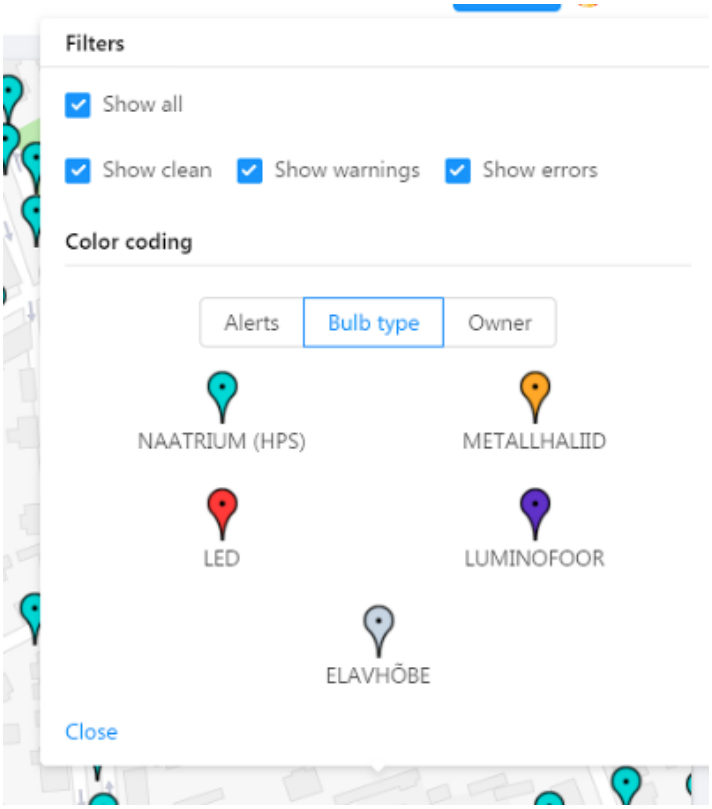
3.1.1 Kaardivaade

Rakenduse põhieesmärgiks on anda hea ülevaade linna või mingi muu ala valgustitest. Lehe laadimisel kuvab kaart kohe ala, kus profiili valgustid asuvad. See tagab selle, et kasutaja ei pea hakkama otsima, kus kohast tema valgustid algavad. Lisaks sellele on kaardi peal kuvatavatel valgustitel ka kindlad värvid, mida on võimalik ka konfigureerida. Värve saab muuta rippmenüüst, kus kuvatakse ära kõik võimalused ning ka näidatakse, mis värv midagi tähendab. See täidab ka legendi eesmärki. Samas rippmenüüs saab ka filtreerida valgusteid, näiteks kas kuvada ilma vigadeta töötavaid valgusteid.



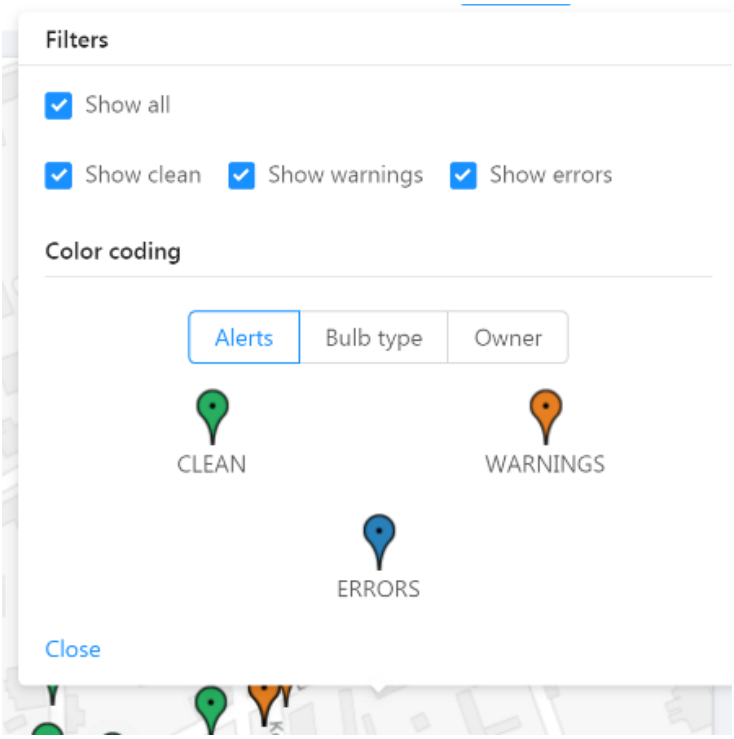
Joonis 2. Kaardivaade, kus on näha ala valgusteid

Joonisel on näha valgusteid, mis kuuluvad profiilile. Antud juhul on autenditud Tartu linnavalitsuse kasutajaga, nii et on näha kõiki Tartu valgusteid. Iga markeri peale on võimalik vajutada ning selle peale kuvatakse valgusti kõik andmed, mis asuvad kaardist allpool. Hetkel on valitud markerite värvideks valgustite lambipirnid ning kuvatakse kõik valgusteid – neid, millel ei ole veasignaale ja neid, millel on.



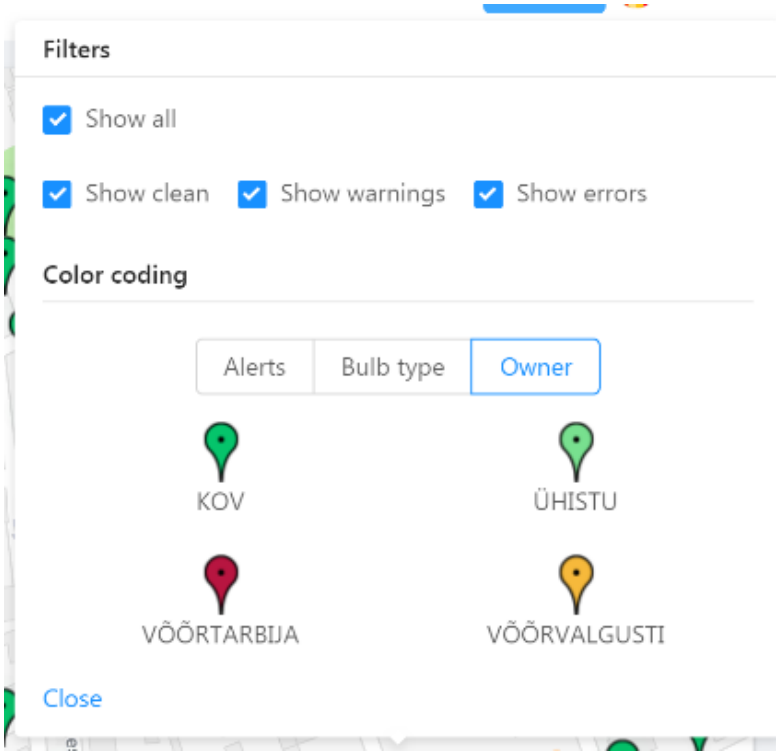
Joonis 3. Valgustite kuvamine lampide järgi

Rippmenüü avamisel kuvatakse kasutajale selline vaade. Vaade jaguneb kaheks osaks – filtreerimine ja värvide valik. Punased markerid viitavad valgusdioodi lampidele, helesined naatriumlampidele ja kollased halogeenilampidele. Valiku muutumisel ei toimu lisapäringuid ja tulemused on koheselt nähtavad.



Joonis 4. Valgustite kuvamine nende staatuse järgi

Üheks võimaluseks valgustite kuvamisel on valida värvid vastavalt valgustite staatusele. Valgustitel on kolm staatust – töötavad, hoiatusega ja hooldust vajavad. Täpselt sama asja saab teha ka teiste värvkoodide valimisel, aga siis toimub see hoopis filtreerimise kaudu. Filtreerimisel ei kuvata üldse valgusteid, mis tingimustele ei vasta, näiteks peites ära kõik korrektselt töötavad valgustid.

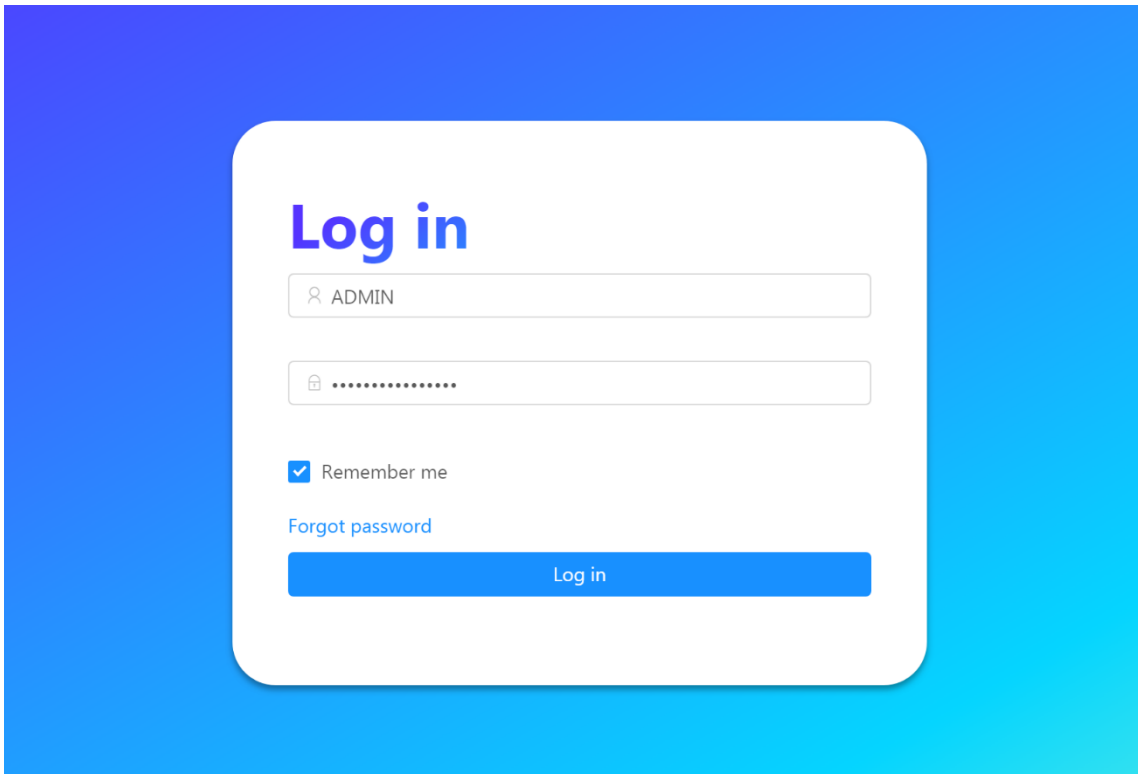


Joonis 5. Valgustite kuvamine nende omanike järgi

Kolmandaks variandiks saab valgusteid kuvada vastavalt nende omanikele. Suur osa valgusteid kuulub ikkagi kohalikule omavalitsusele, aga on valgusteid, mis kuuluvad ka kellelegi teisele. Lisaks kohalikule omavalitsusele, võib valgusti kuuluda ka ühistule. Kui valgusti ei kuulu neile kahele, siis valgusti kategoriseeritakse võõrvalgustiks või võõrtarbijaks.

3.1.2 Autentimine

Autentimise vaatel on kaks välja – kasutajanimi ja parool. Lisaks sellele on võimalik valida, kas jätta kasutajanimi meelde, muutes järgmisel korral sisse logimise kiiremaks.

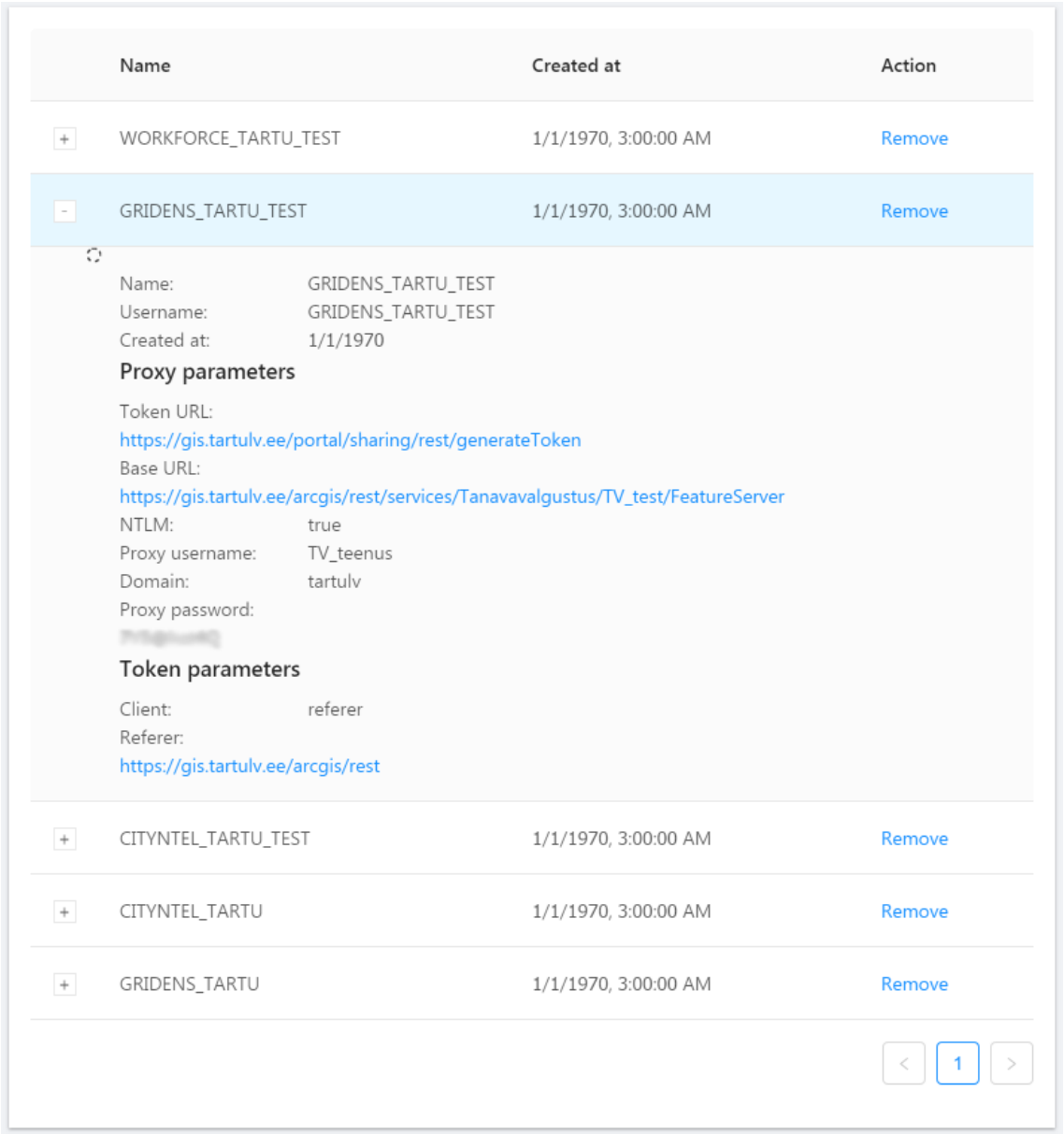


Joonis 6. Autentimine

Edukal autentimisel luuakse sessioon ning suunatakse kasutaja kaardivaatesse. Kui kasutaja sisestab vale parooli või kui sellist profiili ei eksisteeri, kuvatakse kasutajale veateade.

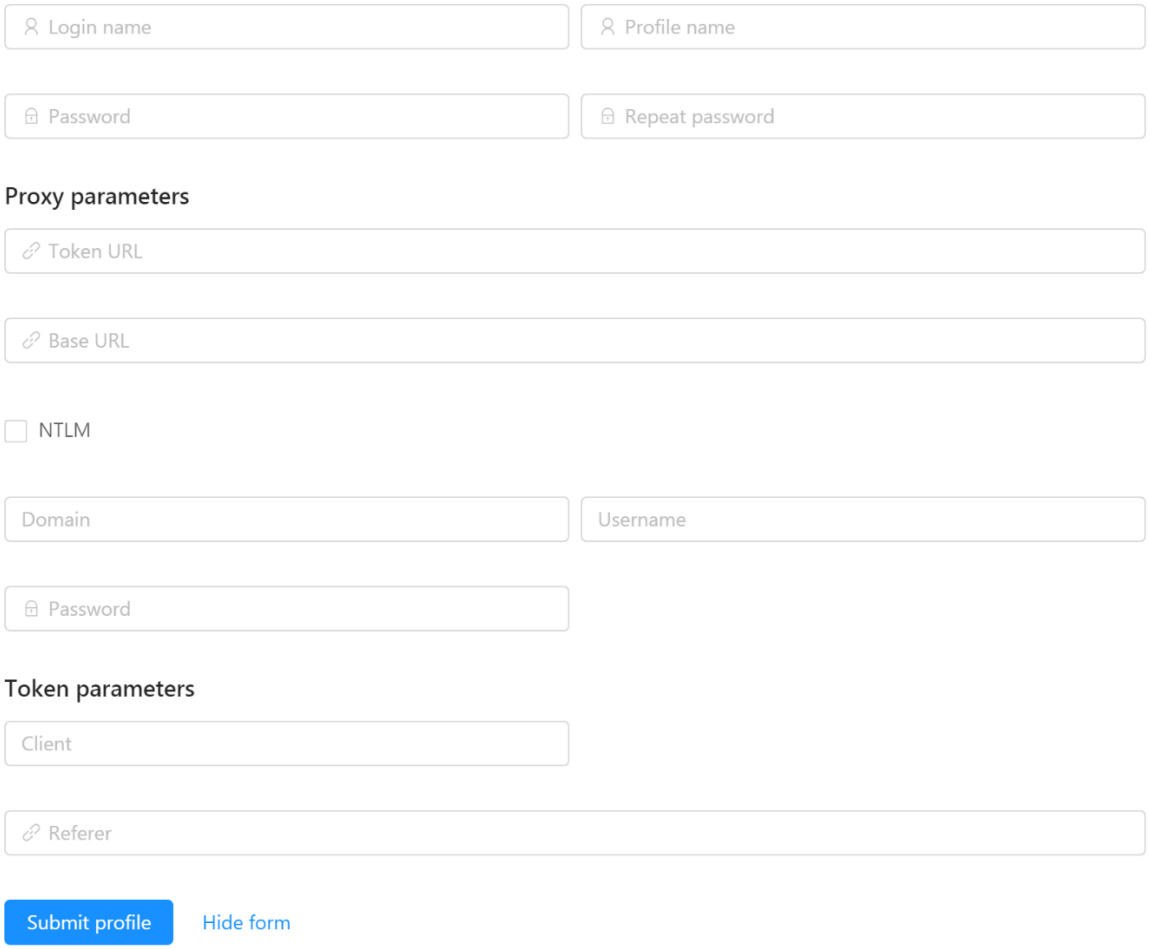
3.1.3 Alamprofiilide haldamine

Süsteemis on kahte sorti profiile – haldavad profiilid ja nende alamprofiilid. Haldavaid profiile ei ole võimalik rakenduses lisada, aga alamprofiilide haldamiseks on eraldi vaade. Vaatele saab ligi ainult siis, kui autenditud profiil on haldaja staatusega. Vaates on põhiliselt kaks komponenti – olemasolevate alamprofiilide tabel ning lisaks ka vorm, kus on võimalik uusi profiile luua. Uue profiili loomisel on võimalik koheselt ka sellega sisse logida ning alamprofiilid näevad ainult enda profiili ulatuses andmeid kaardil – näiteks Gridens-i test profiil näeb ainult nende testandmeid kaardil.



Joonis 7. Alamkasutajate tabel

Alamprofiilide tabelis on näha profiili kohta kõiki andmeid välja arvatud kasutajaliidesesse autentimiseks kasutatav parool. Parool hoitakse andmebaasis nagunii räsitul kuju, nii et sellega ei oleks võimalik sisse logida. Paroolide räsimiseks kasutatakse SHA-512 räsifunktsiooni.



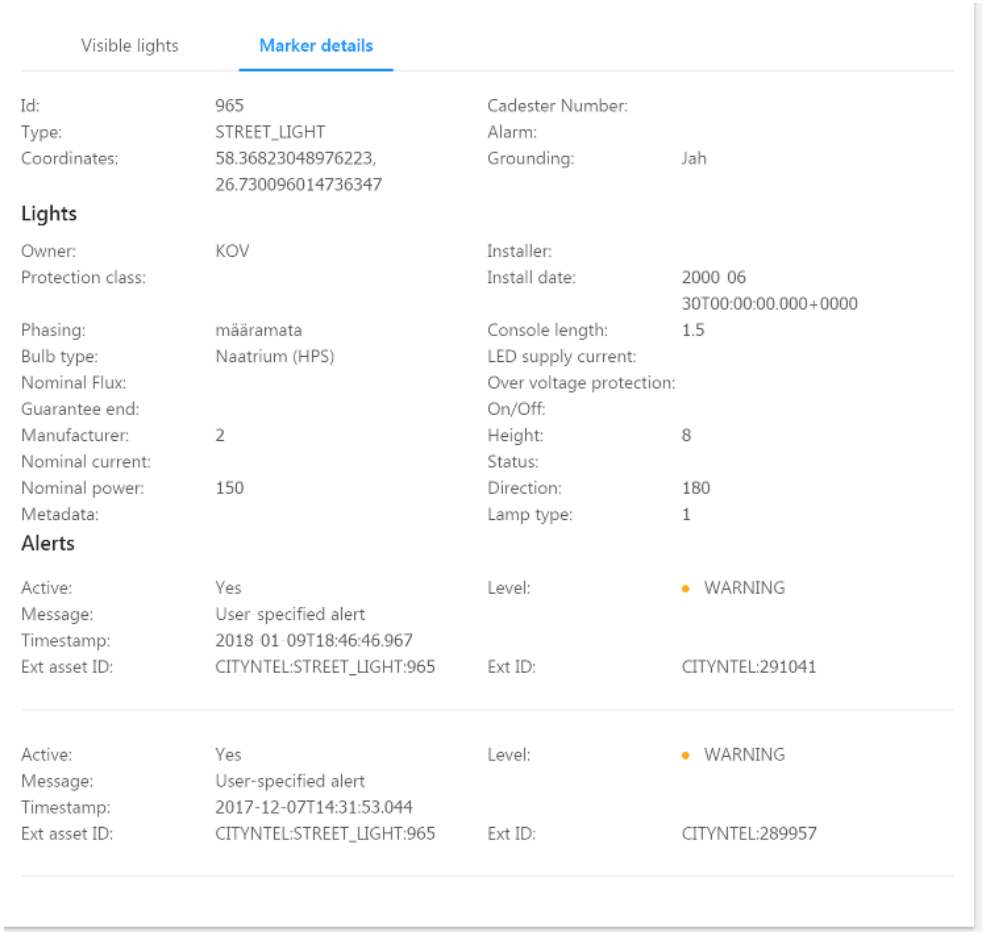
Joonis 8. Alamkasutajate lisamine

Uue profiili loomisel tuleb määrata profiili üldised andmed, millega on võimalik kasutajaliidesesse sisse logida. Järgmises sektsioonis tuleb sisestada andmed, millega on võimalik välisüsteemis ennast autentida ja aadressi, kust on võimalik valgustite andmeid pärida. Viimaseks sektsiooniks on sessioonitõendi konfigureerimiseks vajalikud väljad, sest tõendi lahendused on süsteemides erinevad.

3.1.4 Valgustite detailvaade

Valgustite detailvaade on osa kaardivaatest, mida kuvatakse valgusti peale vajutades või valgustite tabelist vaate avamisel. Vaate eesmärgiks on ära kuvada kõik teadaolevad andmed selle valgusti kohta. Andmed on jaotatud loogilisteks sektsioonideks, millel on ka oma alampealkiri.

Esimeses sektsioonis tuuakse välja valgusti üldised andmed, milleks on näiteks tema koordinaadid ja ID. Sellele järgneb lampide sektsioon, mida on tavaliselt üks, aga võib olla ka rohkem. See on sellepärast, et on olemas tänavavalgusteid, mille küljes on rohkem kui üks lamp. Lambi kohta tuuakse välja näiteks kes on selle paigaldaja, tootja, milline pirn on kasutusel ja nii edasi. Need andmed on igal profiilil erinevad, näiteks testprofiilidel on osad väljad tühjad või vananenud. Viimaseks sektsiooniks on häired. Häireid võib olla null kuni lõpmatus tükki, aga tähtis on see, kas häire on aktiivne või mitte. Kõigepealt kuvatakse ikkagi aktiivsed häired ja nendest allpool mitteaktiivsed. Iga häire kohta kuvatakse ära mis on häire tase, sisu, millal häire tekkis jne.



Joonis 9. Detailandmete vaade

3.2 Rakendusliidesed ja andme arhitektuur

Antud peatüki eesmärgiks on tuua välja kuidas ja millisel kujul andmed kasutajaliidesesse jõuavad ja kuidas need andmed välisüsteemides välja näevad. Tuuakse välja, kuidas välisüsteemidest andmete pärimine välja näeb ja millise vastuse tagastatakse.

3.2.1 Gridensi rakendusliides

Gridens-i juhtimissüsteemist tuleb aja tagant küsida, kas on mingeid muutusi esinenud. Kui on muutused esinenud, tagastatakse nimekiri häiretest ja nende andmetüübid muudetakse ümber ühtsele kujule ja salvestatakse MongoDB andmebaasisüsteemi. Esialgselt jõuavad rakendusse Gridens-i häired kujul:

Tabel 1. Gridens'i häirete kuju

Andmeväli	**Andmetüüp**	**Näidisväärtus ja tüüp**	**Kirjeldus**
id	täisarv	2483765 -	Unikaalne häire identifikaator
cabinet	täisarv	317 -	Valguskapi identifikaator, mis häire teavitas
terminalGroup	täisarv	1950 -	Terminaligrupi identifikaator
lamp	täisarv	15590-	Tänavavalgusti identifikaator, millest häire tekkis
since	kuupäev ja kellaaeg	2018-05-09T01:44 –	Aeg, millal häire tekkis
until	kuupäev ja kellaaeg	2018-05-14T16:19:12 –	Aeg, millal häire lõppes. Võib ka tühi olla
ongoing	kahendmuutuja	false -	Näitab, kas häire oli ajutine või mitte
level	sõne	"warning" -	Häire tase, on *warning* või *error*
type	JSON	{"description":"Not responding"} –	Häire sisu

3.2.2 CityIntel’i rakendusliides

Täpselt nagu Gridens-i korral, tuleb ka CityIntel-i süsteemist küsida muutusi. Kui on muutused esinenud, tagastatakse nimekiri nendest CityIntel-i andmetüübi kujul, mis tuleb ka muuta ühtsele kujule, et andmed oleksid ühtselt mõistetavad kasutajaliideses. CityIntel-i rakendusliidesest jõuavad häired kujul:

Tabel 2. CityIntel'i häirete kuju

Andmeväli	**Andmetüüp**	**Näidisväärtus**	**Kirjeldus**
ack	kahendmuutuja	true	Kas häirest on teavitatud
active	kahendmuutuja	false	Näitab, kas häire oli ajutine või mitte
alertTypeName	sõne	"Missing status (never received)"	Häire tüüp
alertType	sõne	"new_without_response"	Aeg, millal häire tekkis
id	täisarv	288122	Häire unikaalne identifikaator
level	täisarv	0	Häire tase 0 – teavitus 1 – vajab lahendamist
ts	kuupäev ja kellaaeg	2016-12-13T10:04:52.650226	Häire tekkimise kuupäev ja kellaaeg

3.2.3 ArcGIS'i rakendusliides

Kui CityIntel ja Gridens tagastavad andmeid häirete kohta, siis ArcGIS-i rakendusliidese käest saab teada üldiseid andmeid valgustite kohta, milleks on asukoht, lambipirni andmed ja nii edasi. Kuna tänavavalgustite üldandmed tulevad alati ühest kohast, ei pea neid ümber muutma. Ühe valgusti andmed on kujul:

Tabel 3. Tänavavalgusti andmekuju

Andmeväli	**Andmetüüp**	**Näidisväärtus**	**Kirjeldus**
id	täisarv	12110	Tänavavalgusti unikaalne identifikaator
type	sõne	"STREET_LIGHT"	Vara tüüp, antud töös tegeletakse ainult tänavavalgustitega, aga samas formaadis võib tulla ka teist tüüpi varad.
location	geograafilised koordinaadid JSON kujul	{"lat":58.368076342349916, "lng":26.72795010790143}	Valgusti asukoht, teada on laiuskraad ja pikkuskraad
lights	nimekiri lambi objektidest	[{"owner":"KOV", bulbType":"Naatrium (HPS)", "manufacturer":2, "nominalPower":150.0, "installer":"Eesti Elekrivõrkude Ehituse AS Tartu Ettevõte", "installDate":"2000-06-30T00:00:00.000+0000", "consoleLength":1.5, "height":8.0, direction":135, "lampType":1}]	Lampe võib ühe valgustiposti küljes olla rohkem kui üks. Iga lambi kohta on objekt, kus on andmed selle lambi andmetest (lambi tüüp, tootja jne)

3.2.4 Gateway

Gateway on teenus, mis võtab vastu ArcGIS-i, CityIntel-i ja Gridens-i andmed ja muudab nad üheks formaadiks. Kõik andmed jõuavad lõpuks just sinna ning see on ka teenus, kus kohast saab kasutajaliides oma andmed. Kui andmed jõuavad teenusesse, muudetakse need ühtseks formaadiks ja salvestatakse need andmebaasi. Valgustite häired saadetakse ka tagasi ArcGIS'i.

Tabel 4. Häirete ühtne kuju

Andmeväli	**Andmetüüp**	**Näidisväärtus**	**Kirjeldus**
externalId	sõne	"GRIDENS:2483839"	Häire unikaalne identifikaator, mille eesliide näitab seda, millise süsteemi valgustiga on tegemist
externalAssetId	sõne	"GRIDENS:STREET_LIGHT:1943"	Vara unikaalne identifikaator, koosneb kolmest osast. Esimene osa näitab, millise süsteemi varaga on tegemist, teine osa näitab vara tüüpi. Vara tüüp on antud projekti raames alati tänavavalgusti. Viimane osa näitab valgusti identifikaatorit.
level	sõne	"WARNING"	Häire tüüp
active	kahendmuutuja	false	Näitab, kas häire oli ajutine või mitte
message	JSON	{"lastBrightness": 0.7843011, "lastCurrent": 434}	Häire sisu, näitab mis häire põhjustas

3.3 Tehniliste probleemide lahendamine

Töö arendamisel tuli lahendada mitmeid erinevaid keerulisi probleeme, et nõuded oleks täidetud ning rakendus oleks optimeeritud. Peamisteks probleemideks oli tänavavalgustite pärimine vastavalt kaardi piiridele, minimaalsete andmete edastamine kasutajaliidesele ja turvalisuse tagamine, et igaüks ei saaks ligi linna või valla andmetele. Lisaks sellele tuuakse välja, kuidas ehitati teenus, mis automaatselt andmeid uuendab. Igas alampeatükis kirjeldatakse probleemi ja kuidas sellele lahendus leiti.

3.3.1 Valgustite pärimine

Optimaalse lahenduse jaoks ei ole hea pärida kõik tänavavalgustid kohe lehe laadimisel, sest sellisel juhul peab kaardil kuvama kümneid tuhandeid valgusteid korraga, muutes rakenduse kasutuskõlbmatuks. Tuleb ka arvestada sellega, et kasutajat võib huvitada ainult mingi kindel ala, sundides kasutajat laadima ebavajalikke andmeid. Nende kahe probleemi lahendamiseks otsustati pärida andmeid ainult siis, kui kasutaja on suuminud kindla tasemini ja päritakse ainult valgusteid alale, mis on ekraanil nähtaval. See tähendab seda, et võetakse kaardil nähtava kõige ülemise vasaku nurga koordinaadid ja alumise parema nurga koordinaadid. Nende andmetega saab kindlaks määrata ristküliku, mis on kasutajal nähtaval. Selgunud koordinaadid saadetakse rakendusserverisse. MongoDB'l on sisse ehitatud georuumilised päringud, mis tähendab seda, et andes ette need eelnevalt kirjeldatud koordinaadid, oskab MongoDB süsteem ise kõik valgustid üles otsida, mis selles alas on.

3.3.2 Valgustite andmemahu optimeerimine

Ühe tänavavalgusti kohta on teada nii tema veasignaalid, lampide andmed kui ka üldised andmed, mis on näiteks asukoht, kõrgus ja nii edasi. Reaalsuses on otstarbekas ainult saata minimaalsed andmed, milleks on filtreeritavad andmed (valgusti staatus, tootja, lambi tüüp) ja kindlasti ka tänavavalgusti asukoht. Täpsemate andmete teadasaamiseks kasutaja klõpsab valgusti peale ja siis päritakse kõik andmed. Nii on nõutav andmemaht minimaalne, aga kasutajani jõuab ikka sama hulk andmeid.

3.3.3 Turvalisus – JWT token ja *authorization* päis

Rakendusliidese ja kasutajaliidese kasutamiseks tuleb end ära autentida, selleks on kaks põhjust. Esimene põhjus on kindlasti see, et kõik ei saaks andmetele ligi ning teiseks et, autentimisel on võimalik tuvastada, milliseid andmeid tagastada. Antud projektis on tegemist kahe alaga – Tartu linn ja Harku vald. Harku valla kasutaja autentimisel peab olema tagatud, et kasutajaliideses kuvatakse ikkagi ainult Harku valla tänavavalgustite andmeid. Pärast edukat autentimist tagastatakse kasutajale JSON kujul sessioonitõend.

Sessioonitõendi formaadiks on JWT (JSON Web Token), mis genereeritakse edukal autentimisel. Tõend salvestatakse kasutaja brauseri lokaalsesse andmehoidlasse, nii ei pea kasutaja uuesti autentima lehele tagasi tulles, juhul kui sessioon ei ole aegunud. Rakendusserver saab lihtsasti ära tuvastada, kes on sessioonitõendi omanik ja valideerida selle legitiimsust. Päringute tegemiseks pannakse tõend HTTP *authorization* päisesse ja juhul kui päis puudub või tõend on vale, tagastatakse veateade.

3.3.4 Valgustite andmete uuendamine

Kuna tänavavalgustite andmed muutuvad, tuleb kuidagi tagada selle, et antud töö rakenduses oleksid ka värsked andmed. Muudatust võivad tekitada näiteks vanade valgustite väljavahetamine ja uute valgustite lisamine. Selle lahendamiseks loodi teenus, mis käib mingi iga kindla aja tagant uusi andmeid pärimas. Aeg on konfigureeritav, aga see ei peaks toimuma väga tihti, sobiks ka korra päevas. Andmebaasis duplikaatide vältimiseks on otstarbekas kasutada MongoDB *Save()* meetodi, mis kirjutab vana valgusti objekti üle uuega.

Teine olukord on aga küll veateadete korral. Kui mingil valgustil tekivad veasignaalid, ei oleks ideaalne sellest alles järgmine päev teada saada. Sellises olukorras tuleb tihti küsida, kas on midagi muutunud. Kui midagi on muutunud, siis tuleb valgusti eelnev veateade ära nullida ja salvestada ära uus seis. Tähtis on see, et kasutajani jõuaks alati viimane seis, aga oleks ka juurdepääs vanadele veateadetele. Kuna veateateid on märgatavalt vähem kui valgusteid, siis jõudluse probleeme see ei tekita.

3.4 Kasutatud tehnoloogiad

Tehnoloogiate valikul lähtuti tehnilistest vajadustest ning arendaja kogemustest ja huvist. Valikud räägiti läbi ka arendustöö juhendajaga, võttes vastu ka ideid ja soovitusi.

3.4.1 MongoDB

MongoDB on andmebaasisüsteem, kus hoitakse andmeid JSON kujul. Antud andmebaasisüsteemis puuduvad relatsioonid, mis tähendab, et kindel struktuur puudub ja andmed võivad olla väga paindlikud. MongoDB-s on päringud minimaalsed ja võimalikud lihtsad, tagades hea kiiruse ka suurte andmekoguste korral. Süsteem ise on avatud lähtekoodiga ja hästi dokumenteeritud. MongoDB valimise põhjuseks oli huvi andmebaasisüsteemi vastu, lihtne kasutatavus ning sisse ehitatud georuumilised päringud. [1]

3.4.2 ReactJS

ReactJS on komponentidel põhinev Javascript'i raamistik, mis on loodud Facebook'i arendajate poolt. See raamistik on ülimalt populaarne ning on kasutusel veebilehtedel nagu Facebook, Netflix, IMDB ja paljudes muudes. React'i peamisteks eelisteks on täielik kontroll üle komponentide elutsükli ning lai valik mooduleid. Antud projektis oli see raamistik ideaalne just sellepärast, et andmed muutuvad pidevalt ja kasutajaliides reageerib andmemuudatustele lihtsalt ja efektiivselt. [2]

3.4.3 AntDesign

AntDesign on React'i raamistiku komponentide kogum. [3] See tähendab seda, et nende poolt on arendatud mitmeid kõrgekvaliteetseid komponente, milleks on näiteks külgriba, modaal, kalender ja nii edasi. Kogum valiti just sellepärast, et sellega arendamine on palju kiirem, tagab ilusa ja intuitiivse kasutajaliidese ning see on React'i kommuuni poolt kõrgelt hinnatud. Komponentide kujundamisel on lähtutud Material Design printsiipidest.

3.4.4 Spring Boot, Spring Security

Spring on raamistik Java rakenduste loomiseks. Raamistik on arendatud ettevõtte Pivotal Software poolt ja on avatud lähtekoodiga. [4] Spring'ist on saanud üks populaarsemaid rakendusserveri raamistikke ja selle kohta leidub lugematu arv näiteid ja probleemide lahendusi.

Spring Boot muudab Spring raamistike ülesseadmise lihtsaks ja probleemivabaks, eemaldades üleliigsed konfigureerimisnõuded ja seadistab automaatselt projektis vajalikud moodulid, milleks on näiteks Spring Security. [5]

Antud projekti töös kasutatakse autentimiseks Spring Security raamistikku. Raamistiku üheks eeliseks on see, et enamus töö tehakse automaatselt arendaja eest ära. Saab seadistada, millele saab juurde autentimata kasutaja ja millele edukalt autentinud kasutaja. [6]

3.4.5 Google Maps

Google Maps on Google poolt arendatud veebipõhine kaardirakendus. Kaarti on võimalik integreerida enda veebilehele ning lisaks sellele on täielik kontroll selle üle, mida kaardi peal otsustatakse kuvada. Antud projektis lisatakse kaardile erinevaid tänavavalgustite markereid. [7]

Veebirakenduses kasutatakse React'i komponenti *react-google-maps*, mis viib kokku React'i ja Google Maps funktsionaalsused, võimaldamaks muuta kaardi peal kuvatavaid andmeid automaatselt siis, kui muutub komponendi olek. Lisaks sellele saab kuulata kasutaja tegevusi, näiteks kui kasutaja muudab kaardi asukohta või suumib sisse.

4 Võimalikud arengusuunad

Antud peatükis tuuakse välja võimalikud rakenduse arengusuunad. Võimalikest lisatavatest funktsionaalsustest oleks kasulik lisada kasutajaliidesesse tänavavalgustite reguleerimine, statistika vaade ning valgusti ajalooline vaade. Tulevikus on planeeritud avalikustada rakendusliides nii, et kõik saaksid mingitel tingimustel andmetele ligi. Sellisel juhul oleks otstarbekas kasutada Swagger rakendusliidese raamistikku.

4.1 Valgustite reguleerimine

CityIntel-i ja Gridens-i tänavavalgusteid on võimalik kaugelt reguleerida. Tänavalgusteid saab reguleerida ka hetkel, aga see on palju keerulisem protsess, kuna puudub liides selle jaoks. Ideaalselt oleks võimalik seda integreerida kasutajaliidesesse, kus on võimalik määrata, mis kellaajal ja kuidas mingi valgusti töötab. Võimalik on muuta aega, millal valgusti töötab, reguleerida vastavalt ilmaoludele ning ka muuta valgustaset.

4.2 Swagger rakendusliidese avalikustamisel

Swagger on rakendusliidese raamistik, mis võimaldab automaatselt genereerida dokumentatsiooni ning demonstreerida, kuidas mingi päring välja näeb ja milline on vastus. Dokumentatsioonis tuuakse välja, mis tüüpi sisendid peavad olema ning isegi autentimise meetodid. [8]

Swagger'i kasutamine on kasulik kahel põhjusel. Esiteks, kui rakendusliides avalikustada, siis on raske aru saada, kuidas andmeid pärida. Selle lahendamiseks tuleks käsitsi kirjutada valmis dokumentatsioon või lasta seda Swagger'il ise teha. Teine olukord oleks siis, kui arendama hakkab mingi uus arendaja, siis ta ei pea ise vana koodi lugema, vaid saab Swagger'ist kohe teada, kuidas mingit ressurssi kätte saada.

4.3 Statistika ning detailne ülevaade hetkeseisust

Hetkel puudub vaade, kus oleks võimalik näha aruannet, mis seis on. Aruandes võiks olla ära kuvatud, paljudel valgustitel esinevad mingid häired. Kaardil on raske näha kõiki valgusteid, mis vajavad hooldamist. Selle lahendamiseks võiks olla mingi tabel või kaardivaade, kus on terve ala näha, aga nähtaval on ainult vigased valgustid. Lisaks sellele võiks aruande vaates olla veel näha seda, kuidas vigade protsent on ajas muutunud. Näiteks kas olukord on paranenud või mida aeg edasi, seda rohkem häired on hakanud esile tekkima.

4.4 Kaardil juhtmete ja elektrikappide kuvamine

Antud töö raames oli vaja kaardil kuvada ainult tänavavalgusteid. ArcGIS-is on andmeid ka näiteks juhtmete ja elektrikappide kohta. Hea oleks näha, millised valgustid mingi elektrikappi alla kuuluvad ja näha, mis kaudu need juhtmed jooksevad. See aitaks ka paremini vigu tuvastada, tegu võib olla ka näiteks vigase kaabliga või olukorraga, kus elektrikapp saadab valesid andmeid.

5 Kokkuvõte

Käesoleva töö eesmärgiks oli luua integratsioonisüsteem, mis võtab kokku andmed kolmest süsteemist, milleks olid ArcGIS, CityIntel ja Gridens. Töö käigus uuriti, mis eesmärki eelnevalt nimetatud süsteemid täidavad ja kuidas nende andmed kokku viia ja neid ühes kohas visualiseerida.

Töö käigus valmis kaardirakendus, mis kuvab tänavavalgustite andmeid kombineeritult kolmest süsteemist. CityIntel ja Gridens tagastavad valgustite veateateid ja ArcGIS valgustite üldiseid andmed. Rakendusse sisenemiseks peab teadma profiili kasutajanime ja parooli ning iga profiili kohta käivad erinevad andmed. Profiile on kahte tüüpi – haldavad profiilid ja nende alamprofiilid. Haldavad profiilid kuuluvad enamasti linna- ja vallavalitsusele ja neil on juurdepääs oma ala kõikidele andmetele. Need profiilid saavad eemaldada ja ka lisada uusi alamprofiile. Kaardi peal kuvatavaid tänavavalgusteid saab filtreerida ja markerite värve saab ka muuta vastavalt hooldajale, lambitüübile ning staatusele.

Uuriti ka töö edasisi arengusuundi, mida võiks tulevikus implementeerida. Võeti arvesse kliendi vajadusi, võimalusi ning ka töötegija kogemusest tulenevaid soovitusi. Uurimise käigus selgus, et kasulik oleks Swaggeri kasutuselevõtt, et dokumentatsioon genereerida ning lisaks oleks võimalik lisada ka kasutajaliidesesse erinevaid funktsionaalsusi. Rakendust saaks täiendada mitmel moel – valgustite reguleerimise vaate lisamine, juhtmete ja elektrikappide kuvamine kaardil ning teha ka eraldi vaade, kus on näha statistikat hetkeseisust.

Kasutatud kirjandus

[1] „What is MongoDB,“ MongoDB, [Võrgumaterjal]. Available: https://www.mongodb.com/what-is-mongodb. [Kasutatud 15 05 2018].

[2] „React - A JavaScript library for building user interfaces,“ Facebook, [Võrgumaterjal]. Available: https://reactjs.org/. [Kasutatud 15 05 2018].

[3] „Ant Design - A UI Design Language,“ AFX, [Võrgumaterjal]. Available: https://ant.design/. [Kasutatud 16 05 2018].

[4] „Spring,“ Pivotal Software, [Võrgumaterjal]. Available: https://spring.io/. [Kasutatud 05 16 2018].

[5] „Spring Boot,“ Pivotal Software, [Võrgumaterjal]. Available: https://projects.spring.io/spring-boot/. [Kasutatud 15 05 2018].

[6] „Spring Security,“ Pivotal Software, [Võrgumaterjal]. Available: https://projects.spring.io/spring-security/. [Kasutatud 15 05 2018].

[7] „Google Maps Platform,“ Google, [Võrgumaterjal]. Available: https://cloud.google.com/maps-platform/. [Kasutatud 2018 05 16].

[8] „World's Most Popular API Framework | Swagger,“ SmartBear Software, [Võrgumaterjal]. Available: https://swagger.io/. [Kasutatud 15 05 2018].