TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Marina Ivanova 179532IAIB
Roman Bondarev 179570IAIB

KAPILLAARELEKTROFOREESI ANDMETÖÖTLUSE TÖÖLAUA- JA VEEBIRAKENDUS

Bakalaureusetöö

Juhendaja: Evelin Halling
MSc

Tallinn 2020

Autorideklaratsioon

Kinnitame, et oleme 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.

Autorid: Marina Ivanova, Roman Bondarev

25.05.2020

Annotatsioon

Käesoleva bakalaureusetöö eesmärgiks on edasi arendada Tallinna Tehnikaülikooli Keemiainstituudis asuvale kapillaarelektroforeesi seadmele töölauarakendus ja veebirakendus erinevate tasemetega kasutajatele kapillaarelektroforeesi käigus tekkivate andmete lugemiseks, töötlemiseks ja tehtavate katsete täpsemaks tõlgendamiseks, kasutades selleks eelnevalt tehtud katsete tulemusi ja testida päris andmete peal.

Kapillaarelektroforees on analüütilises keemias kasutatav elektriliselt laetud osakeste lahutamise meetod. Kapillaari lõpuosas asuvas detektoris registreeritakse komponentide kontsentratsioonile vastavad signaali ajalises järjestuses, mille saab kätte Arduino mikrokontrolleri peal töötav seade ja saadab signaali töölaurakendusele, kus kuvatakse selle graafikul ja eksperemedi lõppedes salvestakse graafikul olevad punktid tekstifaili, katse seadistused metadatana teise faili, graafiku pildifaili ning interneti olemasolul saadaks kogu katse ajal kogutud informatsiooni läbi REST API andmebaasi.

Veebirakendus, mis on genereeritud JHipsteri abil ja mis kasutab Spring Boot'i ja Angular'i, saab andmeid töölaurakenduse kaudu läbi REST API ja salvestab neid andmebaasi. Veebirakendus võimaldab teha analüüsi, matemaatilisi arvutusi, mis põhinevad algandmete peal.

Rakendus võimaldab kehtestada kasutaja rolli, millel on määratletud erinevad võimalused. Antud rakendus kasutab kaks peamist rolli: teadlane ja tavakasutaja. Teadlase roll võimaldab moodustada meetodeid, teha katseid ja analüüse, otsustada missugused katsed on positiivsed ja negatiivsed ehk kas nad lähevad kalibratsioonikõvera arvutamiseks. Tavakasutaja saab katse läbi viia olemasolevate meetodite peal ja määrata elektriliselt laetud osakesi.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 36 leheküljel, 6 peatükki, 30 joonist.

Abstract

Desktop and web application for capillary electrophoresis data processing

The aim of this bachelor's thesis is to further develop desktop and web applications for capillary electrophoresis device at the Institute of Chemistry of Tallinn University of Technology for users of different levels to read, process and more accurately interpret data generated by capillary electrophoresis.

Capillary electrophoresis is a method of separating electrically charged particles used in analytical chemistry. The detector at the end of the capillary records the signal in chronological order received by the device running on the Arduino microcontroller and sends the signal to the application to display it in real time and save the points in the text file, experiment settings as metadata to another file and sends all information collected during the experiment through the REST API database.

A web application is generated using JHipster that uses Spring Boot and Angular which receives data through the desktop application through the REST API and stores it in a database. The web application allows you to perform analysis, mathematical calculations based on raw data.

The application allows you to set up a user role with different options and privileges. This application uses two main roles: scientific and regular. The scientific role allows to make new methods, perform experiments and analysis, decide which experiment are positive and negative. The average user can perform the experiment using existing methods.

The thesis is in Estonian and contains 36 pages of text, 6 chapters, 30 figures.

Lühendite ja mõistete sõnastik

API	Application Programming Interface, rakendusliides
CE	Capillary Electrophoresis, kapillaarelektroforees
CRUD	Create, Read, Update and Delete, looma, lugema, värskendama ja kustutama
GUI	Graphical user interface, graafiline kasutajaliides
JDL	Hipster-specific domain language, JHipsteri-spetsiifiline domeenikeel
JWT	JSON Web Token, JSON veebitunnus
HTTP	HyperText Transfer Protocol, hüperteksti edastusprotokoll
IDE	Integrated development environment, integreeritud arenduskeskkond
LOD	Limit of detection, avastamispiir
MVC	Model, View and Controller, mudel, vaade ja kontroller
RSS	Residual sum of squares, ruutude jääksumma

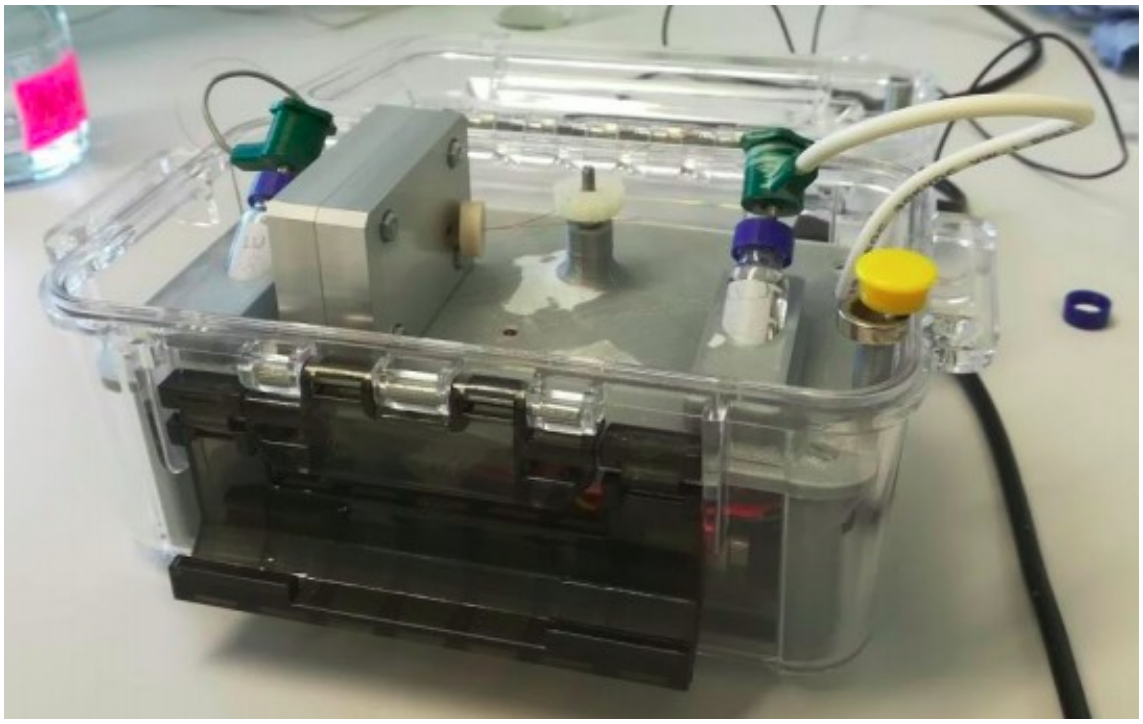
Sisukord

Jooniste loetelu

1 Sissejuhatus

Elektroforees on laetud osakeste ehk ioonide liikumine välise allika poolt loodud elektriväljas. Elektroforeesi füüsikalist protsessi kasutatakse tänapäeval laialdaselt erinevates tööstusharudes. Kõige sagedamini kasutatakse seda analüütilises keemias bioloogiliste ainete tuvastamiseks uurimismeetodites. [3]

Käesolevas lõputöös keskendutakse elektroforeesi alammeetodile: kapilaarsele elektroforeesile (Joonis 1).



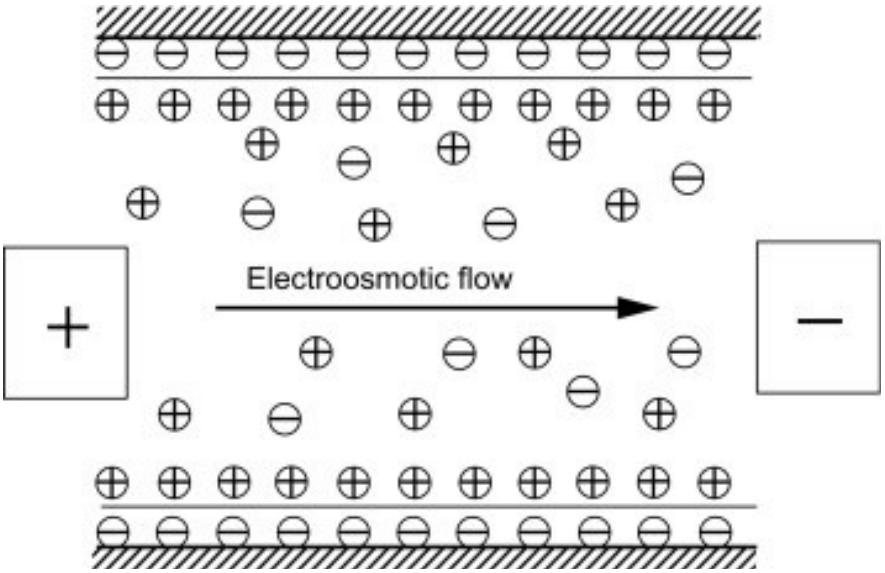
Joonis 1. Kapillaarelektroforeesi seade.

1.1 Kapillaarelektroforeesi mõiste

Kapillaarelektroforeesis ehk KE (inglise keeles Capillary Electrophoresis ehk CE) on lahutusmeetod, mis põhineb laetud osakeste liikumisel erineva kiirusega elektrivälja mõjul kapillaarkolonnis. Analüüdi osakeste migratsiooni kiirus on erinev ja seda määratakse elektroosmootilise liikuvuse abil. [3]

Lahuse laengut kandva osakeste liikuvuse kiirus sõltub nende omadustest. Näiteks elektrilaeng, molekulide suurus, kuju. Mida suurem on iooni laeng ja elektrivälja tugevus, seda kiiremini liigub ioon. Mida suuremad on iooni raadius ja puhvri viskoossus, seda suurem on takistusjõud. Kapillaari lõpuosas asuvas detektoris registreeritakse komponentide kontsentratsioonile vastavad signaalid ehk piigid ajalises järjestuses ning saadakse elektroferogramm. [2]

Selle meetodi abil saab uurida anorgaanilisi ioone, valke, oligosahhariide, vitamiine ja teisi aineid. [3]

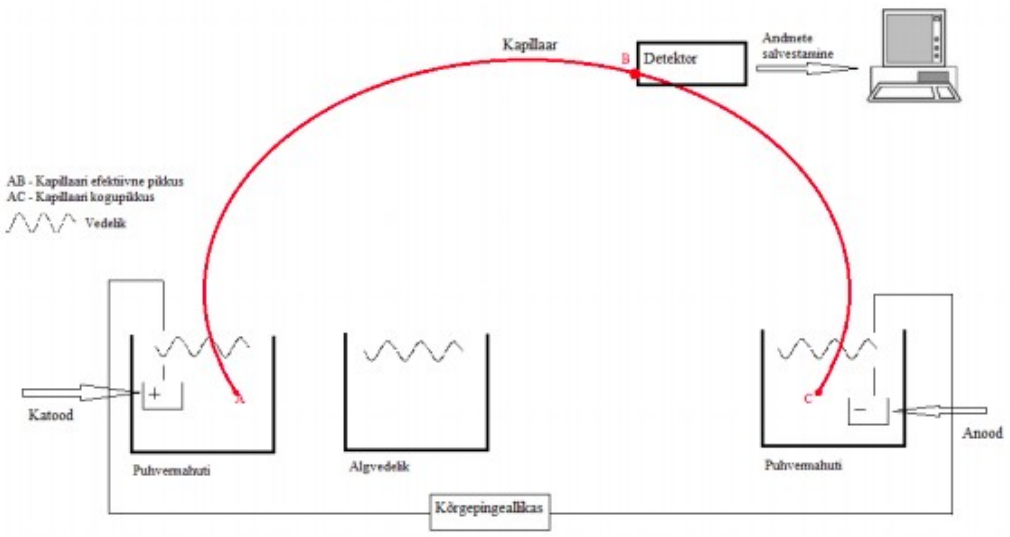


Joonis 2. Elektroosmootiline voog.

1.2 Kapillaarelektroforeesi seadme kirjeldus

Kapillaarelektroforeesi seade on päris lihtsa ehitusega ja koosneb:

- reguleeritavast kõrgepinge allikast;
- kahest puhverlahustega anumitest, mis asuvad ühel ja samal tasemel ja sisaldavad anoodseid ja katoodseid lahuseid;
- kahest elektroodist (katood ja anood), mis on pandud anumissse puhverlahusega ja ühendatud toiteallikaga;
- kapillaarist, milles osakeste eraldamine toimub;
- detektorist, mis võimaldab kontrollida tuvastatud ainete arvu;
- termostaadist, mis suudab hoida konstantset temperatuuri kapillaari sees;
- arvutist, mis registreerib eksperimendi tulemusi. [3]



Joonis 3. Kapillaarelektroforeesi seadme ehitus.

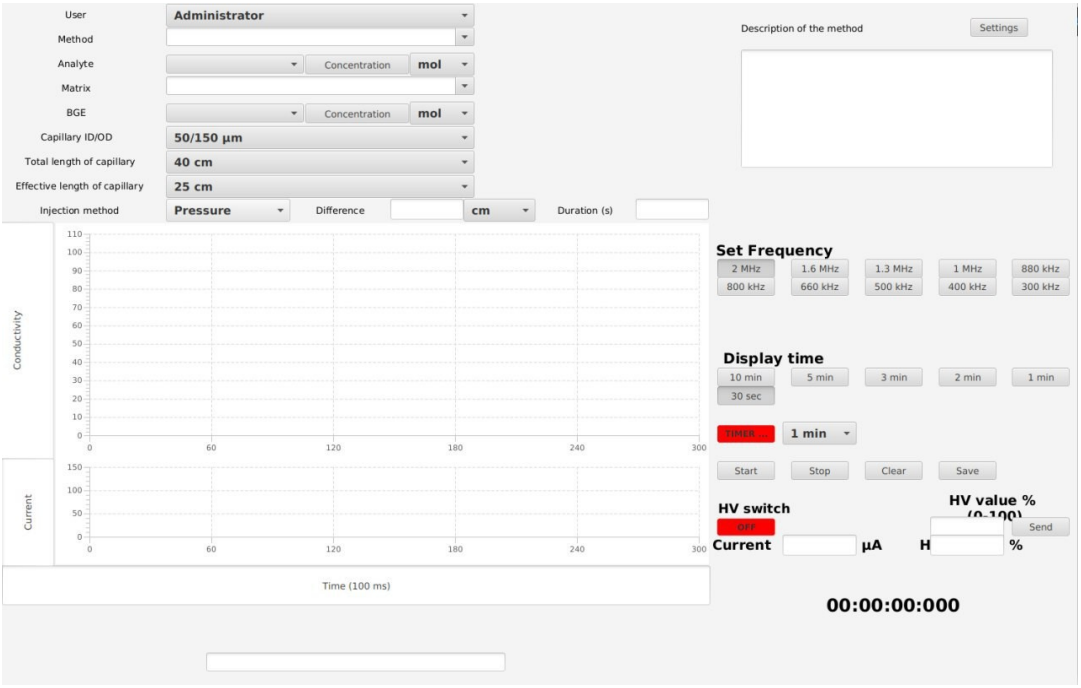
2 Probleemi püstitus

Käesolevas bakalaureusetöös jätkame töölauarakenduse ja veebirakenduse arendamist. Töölaua rakenduse kaudu saadetakse kapillaarelektroforeesi seadmelt signaal, mis kuvatakse elektroferogrammina. Peale eksperementi salvestatakse andmed failidesse. Veebirakenduse kaudu saab kapillaarelektroforeesi käigus tekkivaid mõõtmisandmeid kuvada neid kasutajatele arusaadaval kujul.

2.1 Olemasolev rakendus

2.1.1 Töölauarakendus

Töölauarakenduses on tehtud enamus tähtsast funktsionaalsusest. On võimalik alustada uut katset ja selle jaoks valida nõutud seadistuste vahel: katse tegija nimi, kapillaari mõõtmed, testitavad analüüdid, testitava analüüdi meetod, testimiskeskkond, elektrivoolu sagedus, kõrgepinge ning selle protsentuaalne kasutus, katse kestvus ja ka graafikul nähtav ajaline osa. Katse jooksul tekib graafik, mis on reaalajas muutuv. Kui internetiühendus on olemas, siis salvestakse graafikul olevad punktid tekstifaili, ning interneti olemasolul saadetakse kogu katse ajal kogutud informatsiooni läbi REST API andmebaasi. Pärast salvestamist saadetakse uued andmed tagasi töölauarakendusele. Töölauarakenduses olev informatsioon uuendatakse nii rakenduse käivitamisel kui ka pärast andmebaasi info saatmist. [5]



Joonis 4. Algne töölauarakendus.

2.1.2 Veebirakendus

Veebirakendus on loodud kasutades JHipsteri platvormi. Rakenduses kasutatakse Angulari ja Springi Booti raamistikku. Rakendus võimaldab kasutajal sisendandmeid töödelda, mille jooksul tehakse järgmiseid samme: pööratakse negatiivne ferogramm positiivseks, määratakse elektrivooluvõnkumiste müra ruutkeskmine ning sellest lähtuvalt määratakse võnkumiste lävi. Kui lävi on määratud, leitakse ferogrammile nulljoon ning alustatakse piikide leidmist. Leitud piikide pindalad salvestatakse. [4]

2.2 Probleem

Probleemiks aga osutus töölaua- ja veebirakenduse omavahelise suhtlemise puudus. Kuna mõlemad osad olid tehtud erinevate autorite poolt, siis andmete salvestus oli realiseeritud erinevatesse kohtadesse ning mõlemad rakendused laadisid andmeid nendele sobival viisil. Lisaks oli põhifunktsionaalsus katsete andmete analüüsi tegemiseks puudu.

2.2.1 Töölauarakendus

Osa tähtsast funktsionaalsusest oli enne meie töö algamist realiseeritud - oli kirjutatud kood, mis luges andmed Arduino peal jooksvast rakendusest ning parsis ja salvestas neid andmemudeli sisse ning on olemas GUI, et neid andmeid saaks töölauarakenduse kasutajale näidata ja katse meta andmeid seadistada.

Kliendi probleemi analüüsi käigus osutus, et peab töölauarakendusele lisama funktsionaalsust ja parandama teadaolevaid vigu. Samas pidi töölauarakenduse andmemudeleid üle viima veebirakenduses olevatele andmemudelitele, et rakenduste omavaheline suhtlus oleks võimeline.

Probleemiks oli, et töölauarakendus kasutas ainult ühte andmemudelit ning terve kood oli ühe Java klassi sees, mis ei võimaldanud mugavat täiendamist ja parandamist. Võeti vastu otsus rakendus üle kirjutada, mis sai alguse aine “Meeskonna projekt” raames.

2.2.2 Veebirakendus

Antud töö alguses saime veebirakenduse, millel oli realiseeritud algne funktsionaalsus. Valmis olid esialgsed andmemudelid ja paika pandud nende omavahelised seosed. (Lisa 4)

Veebirakenduses oli implementeeritud esialgne algandmete töötlus, aga ei saanud seda nii iseseisvalt ega ka kliendiga kuidagi tööle panna. Kliendi nõuetest jäi puudu ka analüütide tuvastamine ja kontsentratsiooni arvutamine, mille jaoks on vaja realiseerida kalibratsiooni kõvera arvutamist, mis sai alguse aine “Meeskonna projekt” raames.

2.3 Töö eesmärgid

Antud töö eesmärgiks on luua töötav veebirakendus, erineva tasemega kasutajatele, kapillaarelektroforeesi käigus tekkivate andmete töötlemiseks ja tehtavate katsete täpsemaks tõlgendamiseks, kasutades selleks eelnevalt tehtud katsete tulemusi ja testida päris andmete peal.

2.3.1 Töölauarakendus

Töölauarakenduses saab eristada väiksemaid töö eesmärke, mida tuleb realiseerida:

- andmemudelite ülekirjutamine veebirakenduses kasutatavatele;
- töölaurakenduses tekkivate andmete veebirakendusse saatmise realiseerimine;
- graafiku reaalajas joonestamise parandamine;
- sageduse reaalajas muutmise graafiku lisamine;
- uute andmeväljade lisamine kliendi nõudel;
- meetodi plaani koostamise realiseerimine ja selle kasutamise võimaldamine.

2.3.2 Veebirakendus

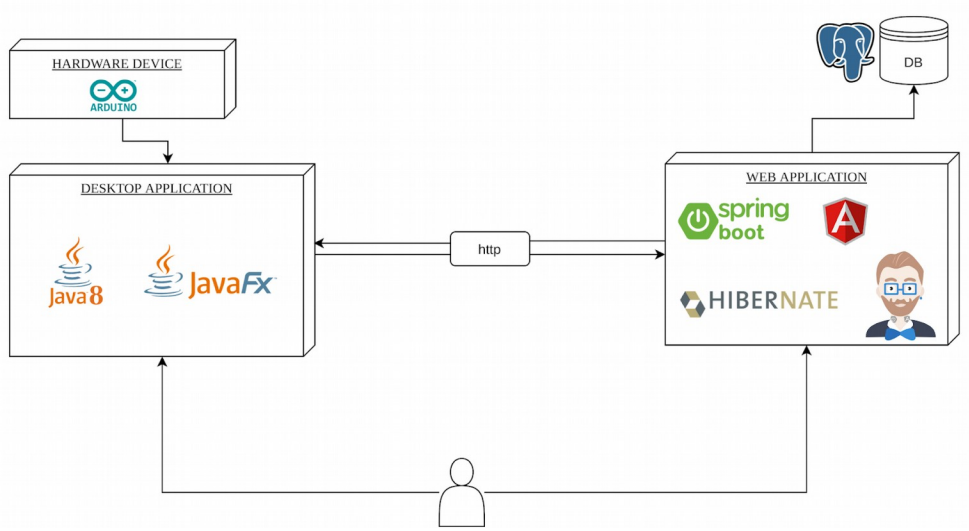
Veebirakenduse väiksemad eesmärgid:

- analüütide tuvastamise realiseerimine;
- analüütide teadlase poolt määramise realiseerimine;
- kalibratsioonikõvera arvutamine, mõõtühiku loogika realiseerimine;
- analüüsi raportide koostamine teadlase ja tavakasutaja jaoks;
- ainete riskipiiride määramine ja analüüsitulemustes kuvamine;
- mugava kasutajavoo realiseerimine;
- analüüsi parameetrite ja nende tulemuste salvestamine ja taaskasutamise realiseerimine;
- listide vaatel filtrite lisamine;

- eksperementide ja piikide staatuse lisamise ja negatiivseks muutmise võimaluse realiseerimine;
- meetodi *default* analüüsi parameetrite salvestamise ja kasutamise realiseerimine.

3 Projekti disain

Terviklikku süsteemi moodustavad töölauarakendus ja veebirakendus. Töölauarakenduse kaudu töötab teadlane välja meetode ja viib läbi katseid. Andmete lugemiseks on vaja ühendust riistvaraseadmega, mida kasutatakse katsete läbiviimiseks. Seade töötab Arduino platvormi peal ning sealt läbi USB pordi loetakse andmeid töölauarakendusse. Eduka katse korral salvestatakse andmed kasutaja arvuti peale ning interneti olemasolul saadetakse need veebirakendusse edasiseks analüüsimiseks.



Joonis 5. Projekti disain.

3.1 Töölauarakendus

3.1.1 Riistvaraseade

Arduino on elektroonikaplatvorm, mis põhineb hõlpsasti kasutataval riist- ja tarkvaral. Arduino tahvlid on võimelised lugema sisendeid (valgust anduril, sõrme puudutamist) ja muutma neid väljundiks. Java rakenduse ja Arduino vahelise suhtluse ülesseadmiseks kasutatakse Arduino programmeerimiskeelt ja töötlemist toetavat Arduino tarkvara (IDE). [6]

3.1.2 Kontrollerid (*controller*) ja teenused (*service*)

Iga kasutajale näidetava vaate kujundamise jaoks on vaja FXML *layout*'i, mille sees kirjeldatakse mis komponendid seal on ja kuidas need peavad asetsema ja välja nägema. Suuremad vaated võivad koosneda mitmetest *layout'idest*.

Iga FXML *layout'iga* seadistatakse kontroller, mis võimaldab iga vaate komponenti seostada meetodiga, mis käivitub komponendi muutmisel. Kuna vaateid võib olla palju ja on vaja sama koodi osasid kasutada erinevates kontrollerites, on loodud teenused *(service)*, mis võimaldavad koodi taaskasutada.

3.1.3 Autentimine ja kasutajad

Töölauarakenduse algses versioonis ei olnud autentimist ning mõiste "kasutaja" omaette ei eksisteerinud. Uue töörakenduse jaoks on autentimine ja kasutaja eeskätt vajalik, et saaks andmeid laadida ja salvestada, kuna ilma autentimiseta ei luba veebirakendus mingeid andmeid salvestada või lugeda.

Kasutajad on realiseeritud viisil, et enne töölauarakendusse sisenemist peab rakenduse kasutaja läbima autentimise protsessi - sisestama kasutaja nime ja parooli, seejärel interneti olemasolul teostatakse kasutaja kontroll, mille tulemusena tagastatakse JWT token või mitte. JWT tokeni abil saab rakendus iga HTTP päringu autentiseerida ja andmeid saada.

Eduka kasutaja autentimisel luuakse igale kasutajale oma töökoht, kuhu kasutaja saab lisada või muuta katsete seadistusi.

3.1.4 Gradle

Projekti kiiremaks ja mugavamaks arendamiseks oli võetud Gradle. Gradle on paindlikkusele ja jõudlusele keskenduv automatiseerimise tööriist. Selle abil ei pea arendaja sõltuvusi käsitsi alla laadima ja installeerima. Gradle võimaldab scripte luua, mida saab kas käsitsi või CI/CD tööriista abil käivitada. Sai kirjutatud skript, mis pakib projekti kliendi arvuti peal käivitavasse JAR faili.

3.2 Veebirakendus

Veebirakendus on genereeritud JHipsteri abil ja kasutab PostgreSQL andmebaasi toodangversiooni jaoks ja H2 testide käivitamiseks. Veebirakendus jookseb Spring Boot 2 raamistiku peal kasutades Java 8. Java mudeli kaardistamiseks traditsiooniliste relatsiooni andmebaasidega kasutatakse Hibernate 5 raamistikku. Eessüsteemi (*front-end*) arendamiseks kasutatakse Angulari raamistikku.

3.2.1 Jhipster

Veebirakenduse baas on genereeritud kasutades tasuta ja avatud lähtekoodiga rakenduste generaatorit JHipster. JHipsteri koodi genereerimise jaoks on vajalik andmemudeleid ja nendevahelist kirjeldust, mille peal saaks JHipster luua Java objekte. Andmemudeleid kirjeldatakse kasutades JDL (JHipster-spetsifiiline domeenikeel). Mugavama andmemudelite suhete visualiseerimiseks kasutatakse JDL studio. [7]

Rakenduse genereerimise käigus saab seadistada andmebaasi, front-end raamistikku ja muid parameetreid. Genereerimise käigus luuakse jooksutamiseks valmis projekt baasfunktsionaalsusega:

- luuakse Spring Boot rakendus;
- seadistatakse Spring Security, meie projekti korral kasutatakse JWT autentimist;
- luuakse põhiobjektide andmemudelid;
- iga andmemudeli jaoks luuakse Spring Data JPA repository;

- iga andmemudeli jaoks luuakse Spring MVC REST controller, mis võimaldab CRUD operatsioone;
- iga andmemudeli jaoks luuakse vaated CRUD operatsioonide tegemiseks;
- iga andmemudeli jaoks seadistatakse *cache*.

3.2.2 PostgreSQL andmebaas

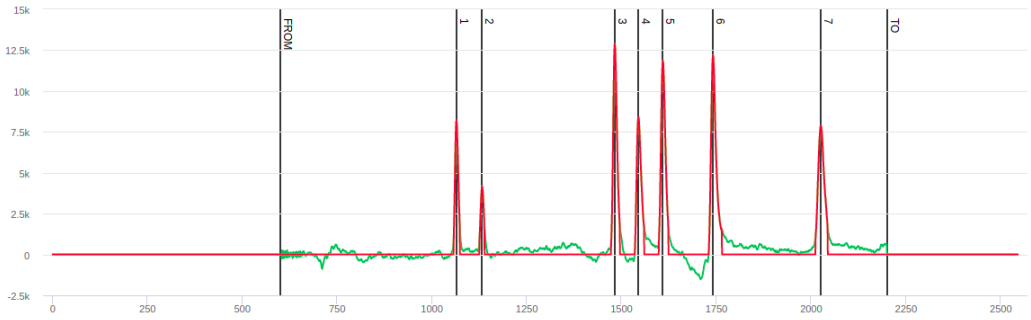
Veebirakenduses kasutatakse PostgreSQL andmebaasi, mis võimaldab andmete salvestamist, lugemist, muutmist ja kustutamist ehk kõike CRUD funktsioone. Andmebaasi kujundamisel on kasutatud JDL Studio't. JDL on JHipsteri-spetsiifiline domeenikeel, kus saab kirjeldada failis kõiki olemeid ja nende suhteid spetsiifilise süntaksiga. (Lisa 2)

3.2.3 Angular

Veebirakenduses kasutatakse Angular, mis on JavaScripti *front-end* raamistik. Raamistik kohandab ja laiendab traditsioonilist HTML-i spetsiaalsete atribuutidega, et esitada dünaamilist sisu. Kasutatakse ka Bootstrapi, mis on tasuta tööriistakomplekt veebirakenduste loomiseks ja sisaldab HTML- ja CSS-kujundusmalle. [8]

3.2.4 Hightcharts

Algse projekti versioonis oli kasutusel Plotly, mis on JavaScripti teek graafikute ja jooniste kujutamiseks, kuid sellel oli piiratud funktsionaalsus, mis ei võimaldanud kliendinõuete täitmist. Võeti vastu otsus minna üle Highcharts teeki kasutamisele, mis võimaldab mugavalt kasutajale kuvada erinevaid diagramme ja graafikuid.



Joonis 6. Hightcharts graafik.

3.2.5 Dom-to-image ja jsPDF

dom-to-image on JavaScriptis kirjutatud teek, mis võimaldab HTML DOM sõlmistikke muuta vektoriteks või piltideks, mida saab salvestada SVG, PNG või JPG formaatides. jsPDF on JavaScript teek, mille abil saab genereerida PDF'e kasutaja brauserist. Mõlemaid teeke kasutatakse raportite genereerimiseks.

3.3 Töölaua- ja veebirakenduse omavaheline liidestus

Töölaua- ja veebirakenduste suhtlus toimub HTTP protokolli abil. Veebirakendus on arendatud REST arhitektuuri stiili kasutades, pakkudes API veebirakendust kasutatavatele rakendustele. Saadetavaid ja vastuvõetavaid andmeid formeeritakse JSON kujuks päringu kehas.

4 Kapillaarelektroforeesi rakendus

4.1 Töölauarakendus

Rakenduse ümberkirjutamise eesmärk oli luua rakendus, mida on tulevikus lihtsam toetada ja lisafunktsionaalsust juurde lisada. Selleks otsustati kood võimalikult väikesteks tükkideks jagada.

Ümberkirjutamise käigus pidi varem kirjutatud funktsionaalsus jääma muutumatuks. Eelmise töölauarakenduse veersioonist saab lugeda Aivar Loopalu bakalaureusetöös.

Joonis 7. Välja arendatud töölauarakendus.

4.1.1 Andmemudelite ülekirjutamine

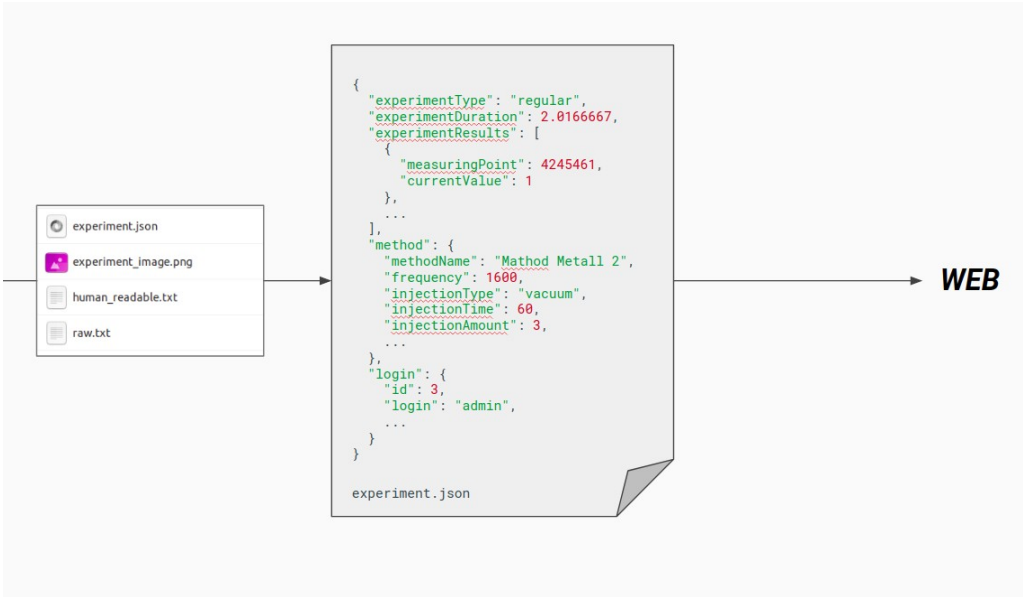
Analüüsi ajal tulime otsusele, et peame andmemudeleid parandama ja täiendama ehk muutma andmebaasi arhitektuuri, kuna antud andmemudelid ei sobinud äriloogika implementeerimiseks ning oli tarvis lisada uusi seoseid.

4.1.2 Töölauarakenduses tekkivate andmete veebirakendusse saatmine

Arduino seadmes katse tegemisel kogutud andmed salvestatakse vahemällu ning kui kasutaja on rahul välja tulnud katsega, saab ta need andmed salvestada edasiseks kasutamiseks.

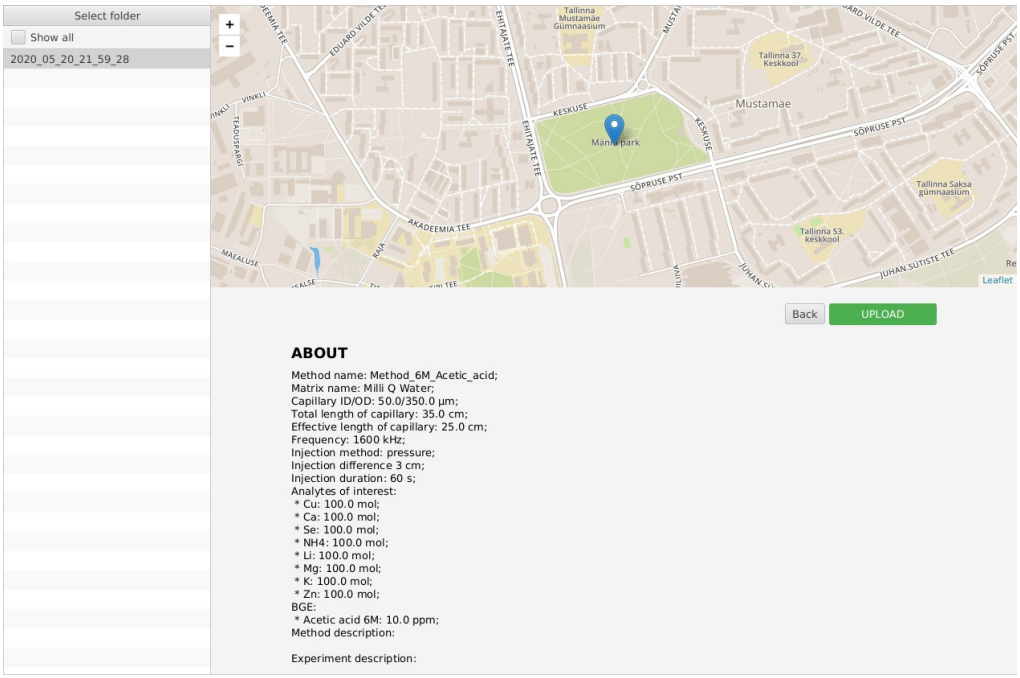
Andmete salvestamine toimub nii kasutaja kettale kui ka veebirakendusse. Kettale salvestamisel luuakse kaust, mis on määratud rakenduse seadetes. Kausta sisse lisatakse 4 faili: experiment.json, experiment_image.png, human_readable.txt ja raw.txt.

- experiment.json - JSON kujul salvestatud andmed, mida on võimalik saata veebirakenduse API kaudu süsteemi;
- experiment_image.png - mõõdetud andmed graafikuna;
- human_readable.txt - inimese jaoks mõeldud katseandmete esitamise viis;
- raw.txt - algandmed, mida saab kasutaja kasutada teistes süsteemides edasiseks analüüsiks, võrdlemiseks jne.



Joonis 8. Andmete saatmise skeem.

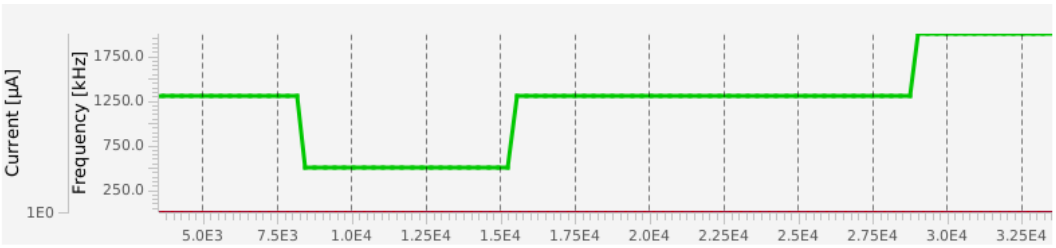
Interneti olemasolul saadakse experiment.json faili sisu läbi veebirakenduse API süsteemi edasiseks analüüsiks. Sõltumata interneti olemasolust või salvestamise edukusest (näiteks salvestamisel tekkis veebirakenduses viga) salvestatakse alati andmed kettale, mida kasutaja saab läbi töölauarakenduse hiljem üles laadida.



Joonis 9. Salvestamata katse vaade.

4.1.3 Sageduse reaalaja muutmise graafik

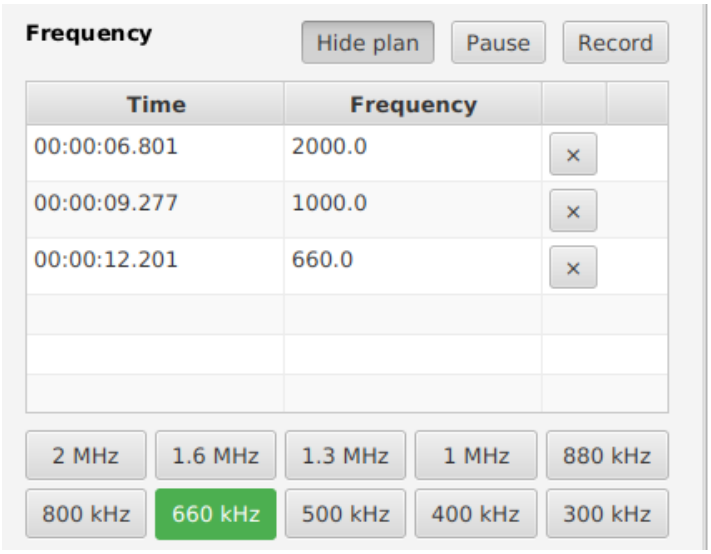
Kliendi soovil lisasime algandmete ja elektrivoolu tugevuse graafikute juurde ka sageduse graafiku. Sageduse muutmise visualiseerimine on tähtis, kuna sageduse muutumisel muutub analüütide liikumine, mille tulemusena elektroferogramm tuleb teistsugune.



Joonis 10. Sageduse muutmise graafik.

4.1.4 Meetodi sageduse muutmise plaan

Sageduse muutmine on tähtis, aga mitme katse korral ei suuda kasutaja alati sageduste ümberlülitumist jälgida. Tekivad olukorrad, kus sagedust muudetakse liiga hilja või vara. Sellise olukorra vältimiseks loodi sageduse muutmise plaan, mis näeb välja töölauarakenduses järgmiselt:



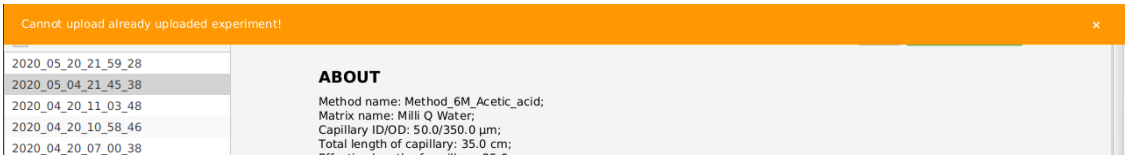
Joonis 11. Meetodi sageduse muutmise plaan.

Kasutajal on võimalus meetodi väljatöötamisel sagedusi salvestada ning lisada loodud plaani meetodile. Joonisel välja toodud plaanil, katse 6-ndal sekundil, muudab rakendus sagedust 2 MHz peale, 9-ndal sekundil 1 MHz peale ja 12-ndal sekundil 660 kHz peale. Kasutajal on võimalik meetodis salvestatud plaani ignoreerida vajutades “Pause” nupu peale.

4.1.5 Teatised ja hoiatused

Eelmises töölauarakenduses ei olnud mingit viisi kasutajale teada anda rakenduse sees toimuvast. Näiteks kui tekkis mingi viga, ei saanud kasutaja seda märkida. Meie ülekirjutatud rakenduses iga ootamatu vea korral ilmub kasutaja ekraanile veateade ning täpsemat selgitust saab vaadata logifailis (log.txt). Lisada saab ka teisi infosõnumeid, et paremini informeerida kasutajat rakenduses toimuvast.

Joonis 12. Veateade näidis.



Joonis 13. Hoiatuse näidis.

4.2 Veebirakendus

Veebirakenduse töö tagamiseks peavad süsteemis olema andmed. Andmete tekitamine on võimalik läbi:

1. veebirakenduse kasutajaliidese;
2. API.

Katse ja katsekomponentide loomine läbi kasutajaliidese on mugav ja vähese kogemusega kasutajale arusaadav. Kuigi kõikide andmete loomine on ajamahukas ning

vea tekkimise tõenäosus on suur, siis kasutajaliides sobib eeskätt andmete parandamiseks ja muutmiseks.

Kõigi andmete salvestamiseks on loodud API sise lõpp-punkt */api/internal/experiments.* Andmeid saadetakse Experiment objekti sees ning seejärel salvestatakse katsekomponente eri loogika järgi. Üldine reegel on: iga komponendi jaoks peab tegema päringu andmebaasi, et leida vastavate parameetrite järgi süsteemis salvestatud objekt, mille puudumisel peab uue looma. On erandeid millal ei pea seda reeglit järgima, näiteks *ExperimentResults* või *AnalyteOfInterst* salvestamisel. Kuna salvestamisel on lisatud eri loogika ja sama päringu saatmisel on erinevad kõrvaltoimed, ei saa kasutajaliideses lõpp-punkti kasutada katse salvestamiseks.

4.2.1 Andmete struktuur

Rakenduse põhiobjektiks on katse ehk Experiment klassi objekt. Katse tegemisel määratakse katsele aeg, tüüp, staatus, kommentaar, kestus, mõõdetavad väärtused (*ExperimentResults*) ja uuritavad analüüdid (*AnalytesOfInterest*). Lisaks igal katsel on meetod (*Method*) millega see katse oli tehtud. Meetodis kirjeldatakse parameetreid mida kasutatakse katse tegemisel: sagedus, süstimisviisi tüüp, süstimise kestus, süstimise väärtus ja mõõtühik vastavalt süstimisviisi tüübile, kõrgepinge väärtus protsentides, kirjeldus, BGE ja analüüdid. (Lisa 1)

Kõik parameetrid on vajalikud järgmise analüüsi tegemisel, aga osadel parameetritel on suurem tähtsus kui teistel, näiteks tüüp ja staatus. Praeguses süsteemis on olemas kaks tüüpi:

- scientific (teaduslik) - näitab, et katse tegija oli teadlane ning tehtud katse võib täiendada positiivsete katsete andmebaasi;
- regular (tavaline) - näitab, et tegu on tavalise katsega ning tulemused ei lähe kuskile [3]

ja kolm staatust:

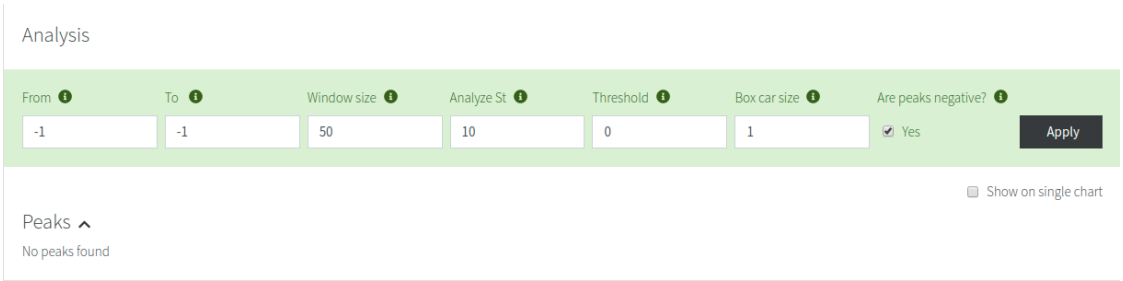
- positive (positiivne) - näitab, et katsetulemused täiendavad positiivsete katsete andmebaasi ning tehtud katse andmeid ja analüüsi hakatakse kasutama tulevastes arvutustes;
- negative (negatiivne) - näitab, et tegemist oli vigase katsega, katse lisatakse negatiivsete katsete andmebaasi ning tulemus ei laeku kuskil;
- new (uus) - näitab, et tegemist on uue katsega, mida peab analüüsima.

4.2.2 Andmete analüüs

Analüütide määramine ja tuvastamine on kaks erinevat kasutajavood, eri tüüpi katsete jaoks. “Scientific” tüübiga katsetel ei tehta automaatset analüütide tuvastamist, vaid teadlane peab ise analüüsi läbi viima ja analüüte määrama. “Regular” tüübiga katsete jaoks tehakse automaatsed analüüsid ja analüütide määramised, kasutades varem teadlase poolt läbiviidud ja analüüsituid katseid.

4.2.2.1 Analüütide määramine

Analüütide määramiseks peab algandmete peal analüüsi teostama katse analüüsimise lehe kaudu, mis analüüsimata katse korral näeb välja järgmisena:



Joonis 14. Analüüsimata eksperiment.

Analüüsi tegemiseks peab määrama seitse parameetrit:

- From ja To - määravad analüüsi piirid ning andmed, mis jäävad väljapoole vahemikku kärbitakse ära;
- Window size - määrab suurust, mida kasutatakse elektrimüra eristamiseks piigidest;

- Anlyze St - määrab kohta, kust peab piike hakkama tuvastama;
- Threshold - määrab mürataseme. Kõik allpool on müra ja ülalpool on huvipakkuv analüüt;
- Box Car Size - andmehulga vähendamise tegur. Keskmistatakse N punkti ja luuakse N punkti asemel ainult üks punkt;
- Negative peaks - määrab kas tuleb tuvastada negatiivseid piike või mitte.

Seejärel teostatakse analüüs algandmete peal seitsme sammuna:

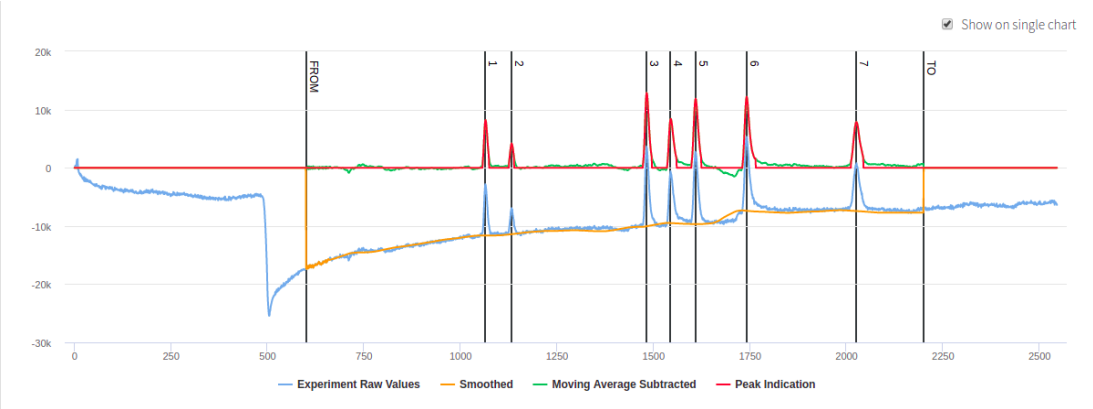
1. Esimese sammuna rakendatakse punktide keskmistamise meetod, kasutades “Box Car Size” parameetri väärtust, et vähendada selle abil algandmete hulka ning siluda ebatäpsusi.
2. Teise sammuna vastavalt “Negative peaks” parameetrile, muudetakse andmete polaarsus vastupidiseks. Sõltuvalt olukorrast võivad piigid olla x-telje suhtes kas all- või ülalpool. Negatiivseid piike (x-telje suhtes allpool) on vaja pöörata positiivseks, et neid saaks kasutada järgmistes arvutuskäikudes.
3. Kolmandal sammul “From” ja “To” parameetri olemasolul, lõigatakse mõõtmistulemuste osa, mida hakatakse analüüsima ehk edasine analüüs rakendub ainult “From” ja “To” parameetritega määratud osale.
4. Neljanda sammuna eristatakse müra signaalist ning silutakse mõõtmistulemusi.
5. Viies samm on nulljoone leidmine. Nulljoone leidmine on vajalik hiljem piikide tuvastamisel, et piikide pindala arvutamist parandada.
6. Kuuendal sammul arvutatakse sobiv müratase “Threshold” parameetri puudumisel.
7. Viimane samm on piikide pindala arvutamine ja piikide eeltuvastus. Tulemusena saadakse loetelu piikidest, kus on kirjeldatud piigi algus- ja lõppkoht ning selle ekstreemumi koht. [4]

Pärast anlüüsi näeb kasutaja järgmist vaadet:

Peak Number	Peak Start	Peak End	Peak Highest	Peak Area
1	458 (1058)	474 (1074)	464 (1064)	83543
2	526 (1126)	539 (1139)	532 (1132)	39580.766
3	873 (1473)	896 (1496)	882 (1482)	158866.23
4	936 (1536)	960 (1560)	944 (1544)	119131.14
5	999 (1599)	1024 (1624)	1009 (1609)	165348.33
6	1131 (1731)	1165 (1765)	1141 (1741)	194338.14
7	1412 (2012)	1444 (2044)	1425 (2025)	153579.89

Joonis 15. Analüüsitud eksperiment.

Pildil on näha, et ilmusid kaks graafikut. Mõlemaid graafikuid on võimalik ühendada üheks vajutades “Show on single chart” nupule.



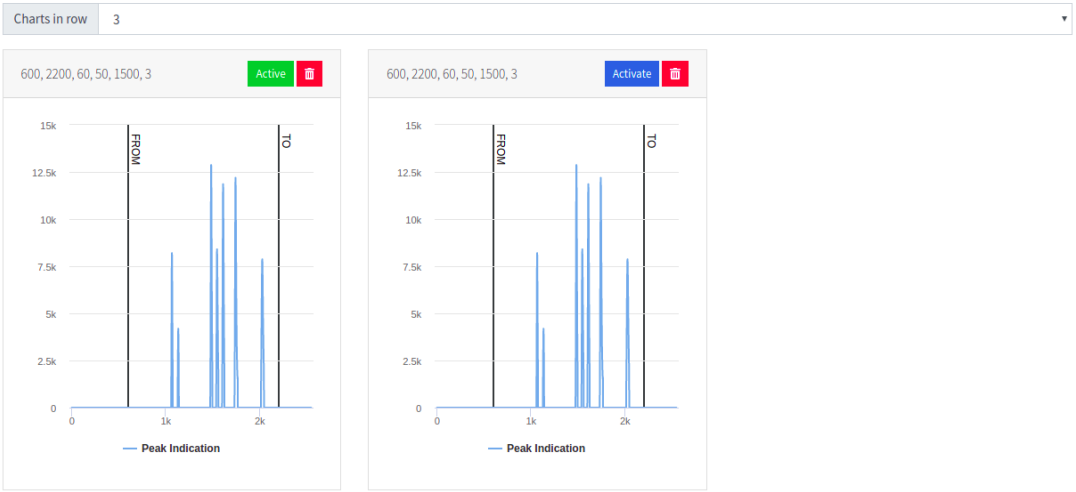
Joonis 16. Ühendatud analüüsi graafik.

Lehe allpool asuvas tabelis näidatakse analüüsi käigus tuvastatud piike. Iga piik on näidatud graafikul vertikaalse joonega koos piigi reanumbriga. Teadlane peab ise määrama igale piigile, mis analüüdiga on tegu. Analüütide loetelu võetakse katse juures salvestatud analüütide nimekirjast. Selleks, et piigile saaks määrata analüüti, peab analüüsi kinnitama "*Apply*" nuppu vajutades parameetrite sisestamise riba lõpus ning seejärel saab analüüte valima hakata. Iga piigi juures saab määrata kas ta on aktiivne või mitte. See staatus on vajalik järgmiste arvutuste jaoks:

Peak Number	Peak Start	Peak End	Peak Highest	Peak Area	Analyte Conc.	Analyte	Active
1	458 (1058)	474 (1074)	464 (1064)	83543	100 µmol	NH4	Active
2	526 (1126)	539 (1139)	532 (1132)	39580.766	100 µmol	K	Active
3	873 (1473)	896 (1496)	882 (1482)	158866.23	100 µmol	Ca	Active
4	936 (1536)	960 (1560)	944 (1544)	119131.14			Inactive
5	999 (1599)	1024 (1624)	1009 (1609)	165348.33	100 µmol	Mg	Active
6	1131 (1731)	1165 (1765)	1141 (1741)	194338.14	100 µmol	Zn	Active
7	1412 (2012)	1444 (2044)	1425 (2025)	153579.89	100 µmol	Li	Active

Joonis 17. Tuvastatud piigid.

Lehe lõpus saab näha mitu graafikut. Süsteem salvestab eelnevad analüüsid ja soovi korral saab taastada vajaliku versiooni vajutades “*Activate*” nupu peale.



Joonis 18. Eelnevate analüüside loetelu.

4.2.2.2 Analüütide tuvastamine

Analüütide tuvastamiseks kasutatakse kalibratsioonikõverat, mis oli arvutatud teadlase poolt tehtud positiivsete katsete peal. Tuvastamise aluseks on analüüdi tuvastamine tema väljumisaja põhjal. Teiseks sammuks on selle aine ja meetodi kalibratsioonikõvera leidmine ja kontsentratsiooni arvutamine lineaarfunktsiooni abiga.

4.2.3 Kalibratsioonikõvera arvutamine

Analüütilises keemias kalibratsioonikõver ehk standardkõver on üldine meetod proovis oleva aine kontsentratsiooni määramiseks, võrreldes teadaolevate kontsentratsioonide andmetega.

Kõigepealt peab teadlane koostama rea katseid, mis satuvad kindla kontsentratsiooni vahemikku ja on tehtud kindla meetodiga. Minimaalne arv tehtud positiivseid katseid peab olema 3, ainult siis saab arvutada kalibratsioonikõverat. Arvestusse lähevad ainult aktiivsed piigid, millel on positiivse staatusega katse, mis on tehtud teadlase poolt.

Kalibratsioonikõvera arvutamise sisendiks on sobivate piikide koordinaadid. Arvutuse käigus tehakse mitu sammu:

- Esimene samm on mõõtühikute teisendamine. Kuna erinevatel piikidel võivad olla erinevad mõõtühikud, siis arvutuse jaoks on vaja need ühtlaseks teha. Piikide mõõtühikud teisendatakse kalibratsioonis määratud ühikusse. Kui ühikut muuta, siis toimub ülearvutus. Hetkel on võimalus kasutada järgmiseid ühikuid: *mol*, *mmol*, *µmol*, *ppm*, *ppb*.

- Teine samm on minimaalsete ja maksimaalsete konsentratsioonide leidmine, et hiljem saaks määrata piire, kus on võimalik kontsentratsiooni järgi ainet tuvastada.

- Kolmas samm on T-jaotuse (T-Distribution) arvutamine. T-jaotust kasutatakse hüpoteesi testimisel, kui on vaja teada saada kas peaks nullhüpoteesi aktsepteerima või tagasi lükkama. Selle arvutamise jaoks kasutati *org.apache.commons.math3.distribution.TDistribution* klassi.

- Neljas samm on keskmiste leidmine. Otsitakse *x*, *y*, *xx*, *yy* ja *xy* väärtuste keskmist, kasutades järgmisi valemeid:

$$\bar{x}=\frac{\sum_{i=1}^{n} x_i}{n} \qquad \bar{y}=\frac{\sum_{i=1}^{n} y_i}{n}$$

$$S_{xx}=\sum (x_i-\bar{x})^2 \qquad S_{yy}=\sum (y_i-\bar{y})^2 \qquad S_{xy}=\sum (x_i-\bar{x})(y_i-\bar{y})$$

- Viies samm on lineaarse funktsiooni $y=ax+b$ a koefitsienti (*slope*) ja b koefitsienti (*intercept*) väärtuste arvutamine kasutades keskmiste väärtusi, mis leiti neljandas sammus.

$$b=\frac{S_{xy}}{S_{xx}} \qquad a=\bar{y}-b\,\bar{x}$$

- Kuues samm on ruutude jääksumma ja standardvea arvutamine valemiga:

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

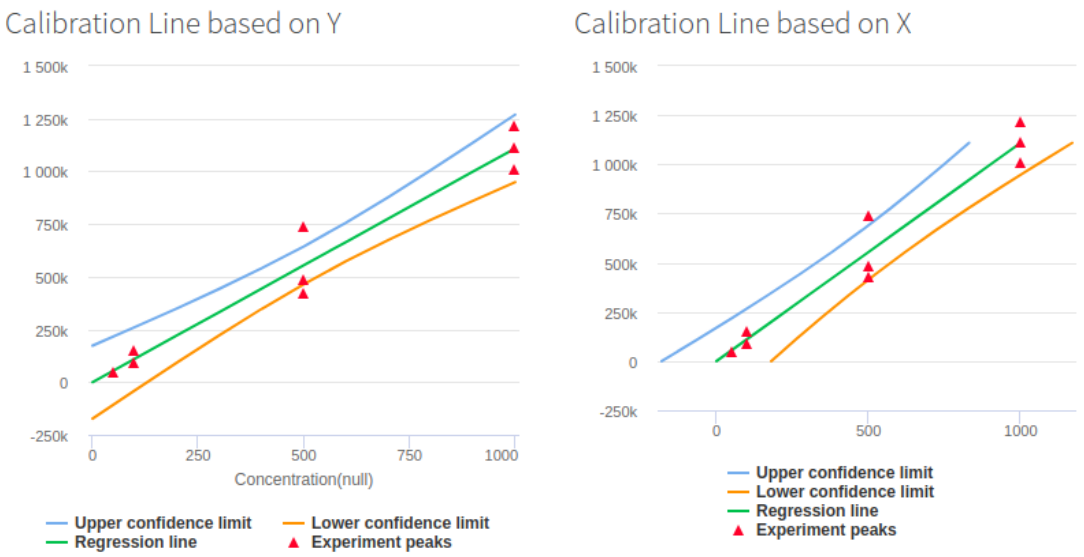
$$SY = \sqrt{\frac{RSS}{N-2}}$$

- Seitsmes samm on delta ja usalduspiiride arvutamine. Esimene etapp on delta y ja delta x arvutamine.

$$\Delta y' = t * SY \sqrt{\frac{1}{N} + \frac{(x'_i - \bar{x})^2}{\sum_{i=1}^{N} (x' - \bar{x})^2}}$$

$$\Delta x' = \frac{t * SY}{b} \sqrt{\frac{1}{N} + \frac{(x'_i - \bar{x})^2}{\sum_{i=1}^{N} (x' - \bar{x})^2} + \frac{1}{M}}$$

Siis arvutatakse usalduspiire *y* ja *x* jaoks kasutades lineaarfunktsiooni, mis arvutati viiendas sammus. Arvutatud väärtuse juurde lisatakse ja lahutatakse delta ning kätte saadakse usalduspiirid. Arvutatud andmete põhjal saab joonistada kalibratsioonikõverat *y* ja *x* jaoks.



Joonis 19. Kalibratsioonikõverad.

Gaafiku peal on samuti näha aktiivsete piikide punkte. Kui punkt ei satu usalduspiiride sisse, siis teadlasel on võimalus teha see mitteaktiivseks tabeli kaudu, kus on loetletud

kõik olemasolevad aktiivsed punktid graafikul. Mitteaktiivseks muutmine eemaldab antud piigi kalibratsioonikõvera arvutamisest.

Active peaks

Experiment	Peak ID	Peak Start	Peak End	Peak Highest	Peak Area	Analyte Conc.	
120 - 21.03.2020 20:17	1883	2102	2121	2110	44630.016	50 µmol	Deactivate
113 - 21.03.2020 17:38	1869	1930	1991	1946	736291.56	500 µmol	Deactivate
112 - 21.03.2020 15:51	1860	1948	2026	1965	1113167.9	1000 µmol	Deactivate
122 - 21.03.2020 15:23	1890	1953	2036	1969	1213725.5	1000 µmol	Deactivate
127 - 21.03.2020 17:11	1912	1922	1975	1936	483480.16	500 µmol	Deactivate
132 - 21.03.2020 18:46	1919	2072	2123	2086	424612.1	500 µmol	Deactivate
133 - 21.03.2020 15:36	1969	1937	2009	1952	1008052.7	1000 µmol	Deactivate
111 - 21.03.2020 19:30	2024	2012	2044	2025	153579.89	100 µmol	Deactivate
115 - 21.03.2020 19:51	2031	2069	2095	2080	89468.984	100 µmol	Deactivate

Joonis 20. Aktiivsed kalibratsiooni piigid.

4.2.4 Analüüsi raportid

On olemas kahte tüüpi raporteid: teadlase raport ja tavakasutaja raport. Raportites näidatakse eksperimentide ja meetodite kokkuvõtet. Raportitel on olemas võimalus neid välja printida.

4.2.4.1 Teadlase raport

Teadlase raportis saab vaadata meetodi ja eksperemendi detaile. Meetodi detailides on näha kõik selle meetodiga tehtud eksperementide infot: nende analüütide kõrgemate piikide väljastuse aeg, selle keskmine, standardhälve, suhteline standardhälve, minimaalne ja maksimaalne väärtus. Koefitsientid *a* ja *b* väärtused on võetud selle meetodi ja analüüdi ühisest kalibratsioonikõvera lineaarfunktsioonist ja nende avastamispiirid (LOD) on arvutatud valemiga:

$$LoD_1 = \frac{3{,}3 * SY}{b}$$

$$LoD_2 = \frac{m-b}{a}$$, kus *m* on esimene y kõrgema usalduspiiri väärtus

Method report

		Time heights							
Analysis ID	**Experiment ID**	**Li**	**Mg**	**K**	**Cu**	**Ca**	**Zn**	**Se**	**NH4**
1169	**120**	2110	1657	1151	2689	1520	1795		1080
1170	**122**	1969	1568	1119	2496	1452	1702		1053
1178	**133**	1952	1558	1109	2478	1041	1691		
1167	**113**	1946	1556	1108	2473	1440	1690		1038
1172	**126**		1613	1139	2579	1489	1750		1071
1212	**111**	2025	1609	1132		1482	1741		1064
1174	**132**	2086	1653	1160	2669	1524	1794		1089
1166	**112**	1965	1636	1119	2494	1568	1701		1049
1173	**127**	1936	1541	1080	2480	1421	1672		1012
1213	**115**	2080	1638	1142		1506	1777		1073
		Li	**Mg**	**K**	**Cu**	**Ca**	**Zn**	**Se**	**NH4**
Average c. migration time		2007.6667	1602.9	1125.9	2544.75	1444.3	1731.3	0	1058.7778
Standard deviation, STDEV		68.4598	43.6767	23.7321	89.5764	148.2798	46.1207		23.6948
Relative standard deviation, RSD, %		3.4099	2.7249	2.1078	3.52	10.2666	2.6639		2.2379
Min		1936	1541	1080	2473	1041	1672		1012
Max		2110	1657	2032	2689	1568	1795		1089
Slope		1109.2869	1022.0633	323.5238	1300.3788	869.0626	1266.4271		592.5478
Intercept		-233.7689	24500.5815	3922.814	54794.0925	9634.2431	-20529.215		-8070.9747
LOD by slope		372.8369	672.8054	513.1049	437.0813	807.9151	415.4972		333.7538
LOD by UCL		156.2806	259.3374	182.9392	199.5269	325.5125	160.1562		119.3415

Joonis 21. Teadlase meetodi raport.

Eksperimenti detailide tabelist saab samuti näha infot analüütide kohta, aga seekord kindla eksperimendi juures. Selles tabelis on andmed, mille kaudu saab teada analüüdi väljamineku aja, tema pindala ja lineaarfunktsiooni kaudu arvutatud kontsentratsiooni. Samuti saab teada analüütide usalduspiire. Mõnede analüütide juures on määratud nende kontseratsiooni riskipiirid ja arvutatud nende ohuvõimalus. Riskipiire saab määratleda ja muuta meetodi juures.

Experiment (111) report

	Li	Mg	K	Cu	Ca	Zn	Se	NH4
Time	2025	1609	1132		1482	1741		1064
Area	153579.89	165348.33	39580.766		158866.23	194338.14		83543
Concentration	138.66 µmol	137.81 µmol	110.22 µmol		171.72 µmol	169.66 µmol		154.61 µmol
Concentration confidence limit	154.8907	258.3028	183.524		312.2838	159.5172		126.4091
Lowest possible concentration	254.8907	358.3028	283.524		412.2838	259.5172		226.4091
Highest possible concentration	-54.8907	-158.3028	-83.524		-212.2838	-59.5172		-26.4091
Law regulations min		10	0					
Law regulations max		1000	1					
Danger	NO	NO	YES	NO	NO	NO	NO	NO

Joonis 22. Teadlase katse raport.

4.2.4.2 Tavakasutaja raport

Tavakasutaja saab näha oma raportis tehtud eksperimentide tulemust ehk analüütide kontsentratsioone koos nende ohuvõimalusega.

Experiment (111) report

Acid	Li	Mg	K	Cu	Ca	Zn	Se	NH4
Concentration	110.22 µmol	138.66 µmol	137.81 µmol	171.72 µmol	154.61 µmol			169.66 µmol
Danger	YES	NO	NO	NO	NO	NO	NO	NO

Joonis 23. Tavakasutaja raport.

4.2.5 Listide vaade (filtrid ja paging)

Kuna andmeid ja atribuute tekib palju, siis kliendi soovil lisasime listide vaadetel filtreid täpsema otsingu jaoks. Parameetrid, mille järgi peab otsinguid teostama saadetakse URL parameetritena, seejärel server võtab need vastu ja koostab dünaamiliselt andmebaasipäringu kasutades saadetuid parameetreid. Lisatud on ka *paging* ehk lehekülgedeks jagatud tagastatav vastus päringule, mis järsult tõstab andmete laadimise kiirust, kuna ei laeta kõiki andmeid korraga.

4.2.6 ID jada genereerimine

Kuna katse käigus mõõdetakse ja salvestatakse suurel hulgal andmeid baasi eraldi ridadena ning iga rea jaoks on vaja ID, siis mitme katse pärast ületab ID väärtus üle 50 000. Projekti JHispteriga genereerimise etapil genereeritakse vaikimisi tabeleid niiviisi, et kõik tabelid kasutavad üht ID jada generaatorit. Seega oli tavaline, et kahe analüüdi loomisel said nende ID-d eristuda mitmekordselt. Sellise olukorra vältimiseks lõime ID

jada generaatorid kõrge ID genereerimisvajadusega tabelitele: experiment, experiment_results, experiment_analysis ja experiment_peaks, ülejäänud tabelid kasutavad ühist ID jada generaatorit.

4.2.7 Gitlab CI/CD

Projektis on kasutusele võetud tarkvara arendamiseks Gitlab poolt pakutav tööriist Gitlab CI/CD. Pidev integratsioon (ingl. *continuous integration*) ja pidev edastamine (ingl. *continuous delivery*) võimaldavad koodimuudatusi sagedamini ja mugavamalt arenduskeskondadesse edastada.

Gitlab CI/CD abil saab kirjeldada töid (ingl. *jobs*) moodustades torujuhtmeid (ingl. *pipeline*), mille abil saab koodi paigaldada arenduskeskkonda. Iga töö sees kirjeldatakse käske ja samme, mida *gitlab runner* peab automaatselt käivitama, et töö õnnestuks. Gitlab runner on masinal jooksev programm, mis tegeleb tööde käivitamisega ja tööde tulemuste saatmisega tagasi Gitlab keskkonda. (Lisa 3)

Meil on loodud 3 tööd:

1. testide jooksutamise *job;*
2. koodi kokkupanemise ja Docker konteineri ehitamise *job;*
3. Docker konteineri jooksutamise arendus masinal.

Igale tööle saab seadistada piiranguid, et saaks määrata töö käivitamise tingimused. Testide jooksutamise töö käivitub igal koodimuudatusel, sõltumata valitud harust. Docker konteineri ehitamine ja selle jooksutamine toimub ainult *master* harus muudatuse toimumisel.

5 Valideerimine

Valideerimine oli tehtud võrdluse põhjal. Samad andmed olid sisendiks nii meie valmis rakenduses, kui ka ülikoolis kasutusel olevas rakenduses, mis on ülesehitatud Microsoft Excelis ning kasutusel on erinevad Visual Basicu makrod.

5.1 Kalibratsioonikõvera valideerimine

Tulemuste valideerimiseks kasutame kalibratsioonikõvera andmeid, kus võrdleme pindala väärtust, mis põhineb välja arvutatud lineaarfunktsioonist. Nagu näha, siis joonistel pindala väärtuste erinevus on kas väga väike või üldse puudub.

Järgmine väärtus mida tuleks kontrollida on *delta y*, kuna sellest väärtusest sõltuvad usalduspiirid. Delta väärtused erinevad rohkem, kuna enne lõplikku väärtust on tehtud palju teisi arvutusi, kus Microsoft Excelis kasutati ümardamist ja veebirakenduses ei kasutatud. Võib öelda, et veebirakenduses saadud tulemus on täpsem.

Regression line data					
ci'	yi'	(ci-ck)^2	delta y'	Upper confidance limit	Below confidance limit
0	20624.047	28476.56	65495.335	86119.382	-44871.288
50	169852.186	14101.56	58586.276	228438.462	111265.909
100	319080.324	4726.56	53602.719	372683.043	265477.606
150	468308.463	351.56	51111.036	519419.499	417197.427
200	617536.602	976.56	51474.375	669010.977	566062.226
250	766764.740	6601.56	54635.808	821400.548	712128.932
300	915992.879	17226.56	60155.786	976148.665	855837.093
350	1065221.018	32851.56	67457.782	1132678.800	997763.235
400	1214449.156	53476.56	76030.080	1290479.237	1138419.076
450	1363677.295	79101.56	85491.407	1449168.702	1278185.888
500	1512905.434	109726.56	95578.109	1608483.543	1417327.324

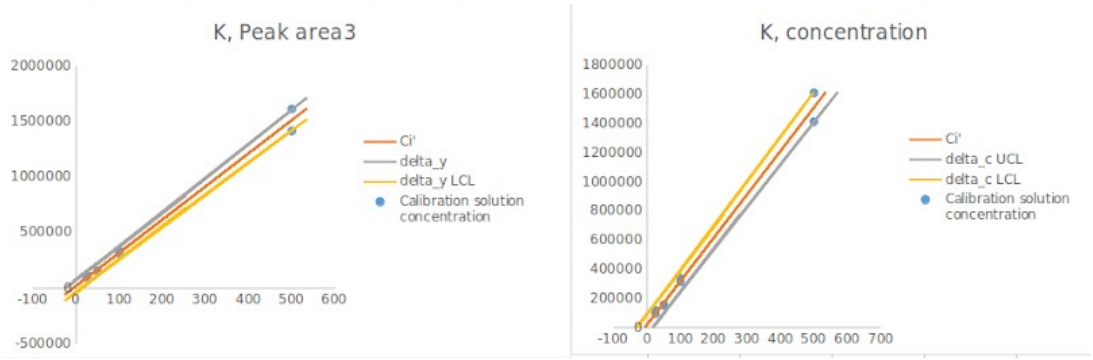
Joonis 24. Microsoft Excelis arvutatud *x*, *y*, *delta y* ja usalduspiirid.

Calibration Line Data

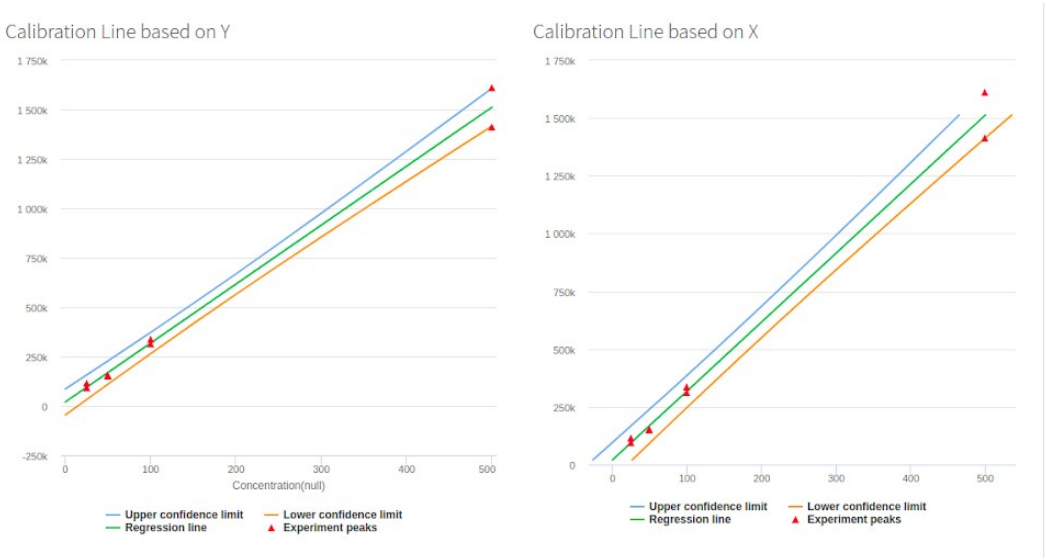
Concentration	Area	Delta Y	Upper Y Confidence	Below Y Confidence
0	20624.0496	65680.8581	86304.9077	-45056.8085
50	169852.1868	58752.2286	228604.4154	111099.9582
100	319080.324	53754.5545	372834.8786	265325.7695
150	468308.4612	51255.8136	519564.2748	417052.6477
200	617536.5985	51620.1825	669156.781	565916.416
250	766764.7357	54790.5702	821555.3058	711974.1655
300	915992.8729	60326.1837	976319.0566	855666.6892
350	1065221.0101	67648.8642	1132869.8743	997572.1459
400	1214449.1473	76245.444	1290694.5913	1138203.7033
450	1363677.2845	85733.5708	1449410.8553	1277943.7137
500	1512905.4217	95848.8451	1608754.2669	1417056.5766

Joonis 25. Veebirakenduses arvutatud *x*, *y*, *delta y* ja usalduspiirid.

Veel üks viis kalibratsioonikõvera valideerimiseks on graafikute võrdlemine. Tulemuseks on kaks graafikut, kus üks põhineb pindala ehk y väärtusest ja teine kontsentratsiooni ehk x väärtusest.



Joonis 26. Microsoft Exceli kalibratsioonikõverate graafikud.



Joonis 27. Veebirakenduse kalibratsioonikõverate graafikud.

Teine aspekt, mis oli võetud kontrollimiseks on avastamispiir (LOD) . Neid on kahte tüüpi: avastamispiir, mis on arvutatud regressiooni tõusu põhjal ja avastamispiir, mis on arvutatud delta y abil. Mõlemad väärtused on piisavalt täpsed.

Joonis 28. Veebirakenduses arvutatud LOD väärtused.

6 Kokkuvõte

6.1 Kommentaarid

Saime tööga hästi hakkama - klient jäi rahule ja kiitis meie tööd. Samas see ei tähenda, et saime tööga ideaalselt hakkama või oleme rahul saadud tulemusega, sest alati jääb asju mida saaks parandada ja täiendada.

Süsteemi arendamisel oli kõige raskem Hibernate raamistiku kasutamine. Hibernate lisas keerukust ja lahendas probleeme ainult osaliselt. Tihti pidi mitu tundi otsima vastust lihtsale küsimusele või leida teisi viise probleemi lahendamiseks.

Hibernate'is on puudu universaalne ja ainuõige viis kuidas andmebaasiga suhelda: Criteria API, HQL, native query või otse Entity Manager'i kaudu. Iga teostus on ehitatud JDBC liidese ümber, aga ükski neist ei suuda pakkuda täisfunktsionaalsust. Seega projektis on näha, et erinevate ülesannete jaoks on valitud erinevad viisid, mis oleks lihtsad ja loetavad.

Projekti genereerimisel JHipster genereerib iga olemi jaoks *controller'i* ja *repository'i.* S*ervice* kihti ei ole. Lihtsa rakenduse jaoks ei peagi *service'it* olema, sest äriloogika puuduse tõttu saab ilma selleta hakkama. Meie lisasime *service'id* tähtsamate olemite jaoks ja võiks ka ülejäänud olemid viia üle samale struktuurile, mis võimaldaks koodi taaskasutada.

Praeguse töölauarakenduse disaini asemel võiks välja mõelda midagi teistsugust. Terve JavaFX rakenduse asemel saaks jätta ainult koodi, mis suhtleks riistvaraga ning suudaks kasutajaliidest ja funktsionaalsust avada *brauseris*. See lihtsustaks arendamist, sest arendaja ei peaks oskama JavaFX ning veebis saaks rohkem ja mugavamat funktsionaalsust lisada.

6.2 Edasine töö

Selles töös oli realiseeritud üks identifikatsiooni viis, milleks on analüüdi väljumisaeg. On olemas teised viisid tuvastamiseks: sisestandardi ja analüüdi väljumisaja suhe (korrelatsioon ühe standardiga) ning sisestandardite ja analüüdi väljumisaja suhted (korrelatsioon kahe standarditega) .

Samuti oleks vaja realiseerida teisi kalibreerimise viise. Hetkel on kasutusel kalibratsioon, mis on tehtud y – piigi pindala ja x – analüüdi kontsentratsiooni peal. On olemas ka teised võimalused:

- y – analüüdi ja sissestandardi piikide pindalate suhe ning x - analüüdi kontsentratsioon;
- y – analüüdi piigi pindala ja väljumisaja suhe ning x - analüüdi kontsentratsioon;
- y - analüüdi ja sisestandardi piikide pindalade suhe;
- y – analüüdi ja sisestandardide piikide pindalade suhe.

Kasutatud kirjandus

[1] “LibreTexts”, [Võrgumaterjal]. Available:

https://chem.libretexts.org/Bookshelves/Analytical_Chemistry/Supplemental_Modules_(Analytical_Chemistry)/Instrumental_Analysis/Capillary_Electrophoresis. [Kasutatud 10 mai 2020]

[2] Weinberger, R., Practical Capilllary Electrophoresis, [Võrgumaterjal]. Available:

https://books.google.ee/books?hl=en&lr=&id=cX8KAQAAQBAJ&oi=fnd&pg=PP8&dq=capillary+electrophoresis&ots=SnfxZTdgRo&sig=gsdfmkJ2RcEM94cyLGxy5o8ELFg&redir_esc=y#v=onepage&q=capillary%20electrophoresis&f=false. [Kasutatud 11 mai 2020]

[3] Gorbatsova, J., Kapilaarelektroforeetiliste katsete digitaliseerimise võimalused, Research work, 2019, pp. 8-22.

[4] Kivimägi, G., Kapillaarelektroforeesi andmetöötluse veebirakendus, Research work, 2019, pp. 8-26.

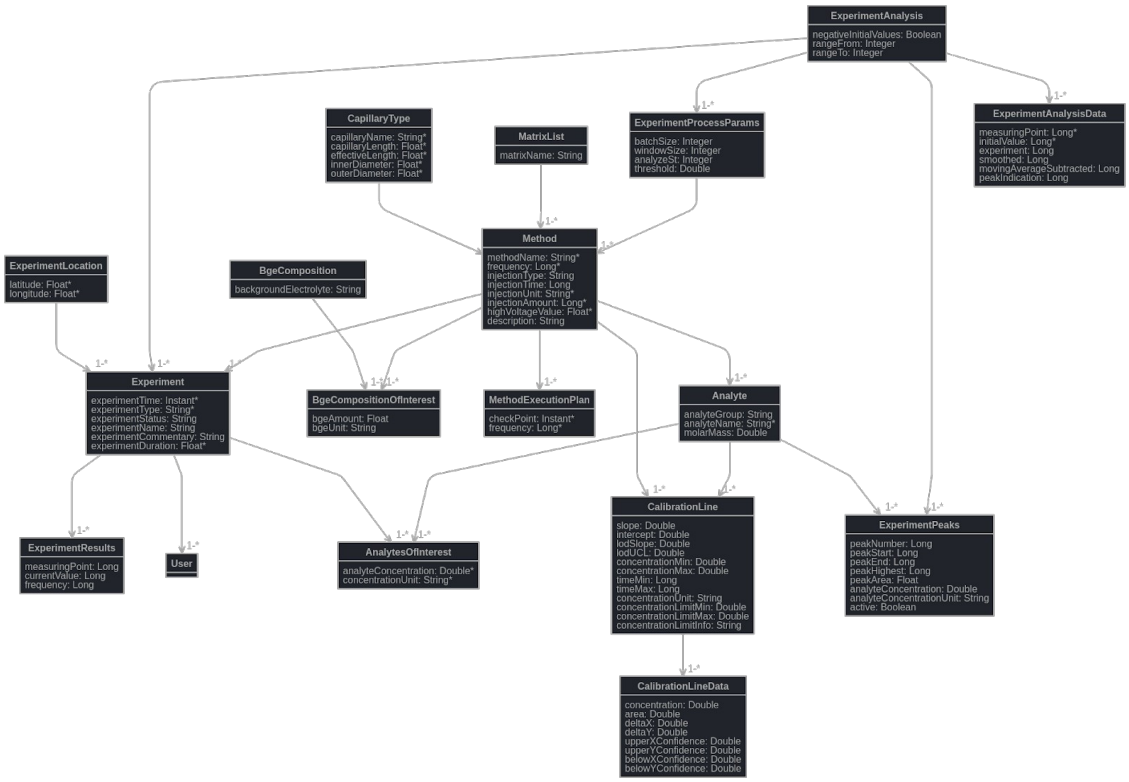
[5] Loopalu, A., Töölauarakendus kapillaarelektroforeesi teostamiseks, Research work, 2018, pp. 11-46.

[6] “Arduino”, [Võrgumaterjal]. Available: https://www.arduino.cc. [Kasutatud 15mai 2020]

[7] “Jhipster”, [Võrgumaterjal]. Available: https://www.jhipster.tech. [Kasutatud 15 mai 2020]

[8] “Angular”, [Võrgumaterjal]. Available: https://angular.io. [Kasutatud 15 mai 2020]

Lisa 1 – Andmebaasi struktuur



Joonis 29. Andmebaasi struktuur.

Lisa 2 – Andmebaasi olemid ja nende suhted

```
entity Experiment {
  experimentTime Instant required, /* start time of the experiment */
  experimentType String required,
  experimentStatus String,
  experimentName String,
  experimentCommentary String,
  experimentDuration Float required /* seconds */
}

entity ExperimentLocation {
  latitude Float required,
  longitude Float required
}

entity ExperimentAnalysis {
  negativeInitialValues Boolean,
  rangeFrom Integer,
  rangeTo Integer
}

entity ExperimentAnalysisData {
  measuringPoint Long required,
  initialValue Long required, // initial values
  experiment Long , // converted value (negative to positive ex.)
  smoothed Long ,
  movingAverageSubtracted Long ,
  peakIndication Long
}

entity Method {
  methodName String required,
  frequency Long required, /* 300, 400, ... 2000 kHz */
  injectionType String /* electricity, pressure, vacuum */
```

```
  injectionTime Long , /* in seconds */
  injectionUnit String required, /* kV for electricity; mbar for pressure; cm for vacuum.
*/
  injectionAmount Long required, /* see above */
  highVoltageValue Float required, /* % */
  description String
}

entity MethodExecutionPlan {
  checkPoint Instant required
  frequency Long required
}

entity BgeComposition {
  backgroundElectrolyte String
}

entity BgeCompositionOfInterest {
  bgeAmount Float,
  bgeUnit String,
}

entity CapillaryType {
  capillaryName String required,
  capillaryLength Float required, /* in cm */
  effectiveLength Float required, /* in cm, from sample injection to detector */
  innerDiameter Float required,
  outerDiameter Float required,
}

entity ExperimentPeaks {
  peakNumber Long,
  peakStart Long, /* measuring Point */
  peakEnd Long,
  peakHighest Long,
  peakArea Float,
  analyteConcentration Double,
  analyteConcentrationUnit String,
  active Boolean
```

```
}

entity MatrixList {
  matrixName String,
}

entity ExperimentResults{
  measuringPoint Long,
  currentValue Long,
  frequency Long,
}

entity AnalytesOfInterest {
  analyteConcentration Double required,
  concentrationUnit String required
}

entity Analyte {
  analyteGroup String,
  analyteName String required,
  molarMass Double
}

entity CalibrationLine {
  slope Double,
  intercept Double,
  lodSlope Double,
  lodUCL Double,
  concentrationMin Double,
  concentrationMax Double,
  timeMin Long,
  timeMax Long,
  concentrationUnit String,
  concentrationLimitMin Double,
  concentrationLimitMax Double,
  concentrationLimitInfo String
}

entity CalibrationLineData {
```

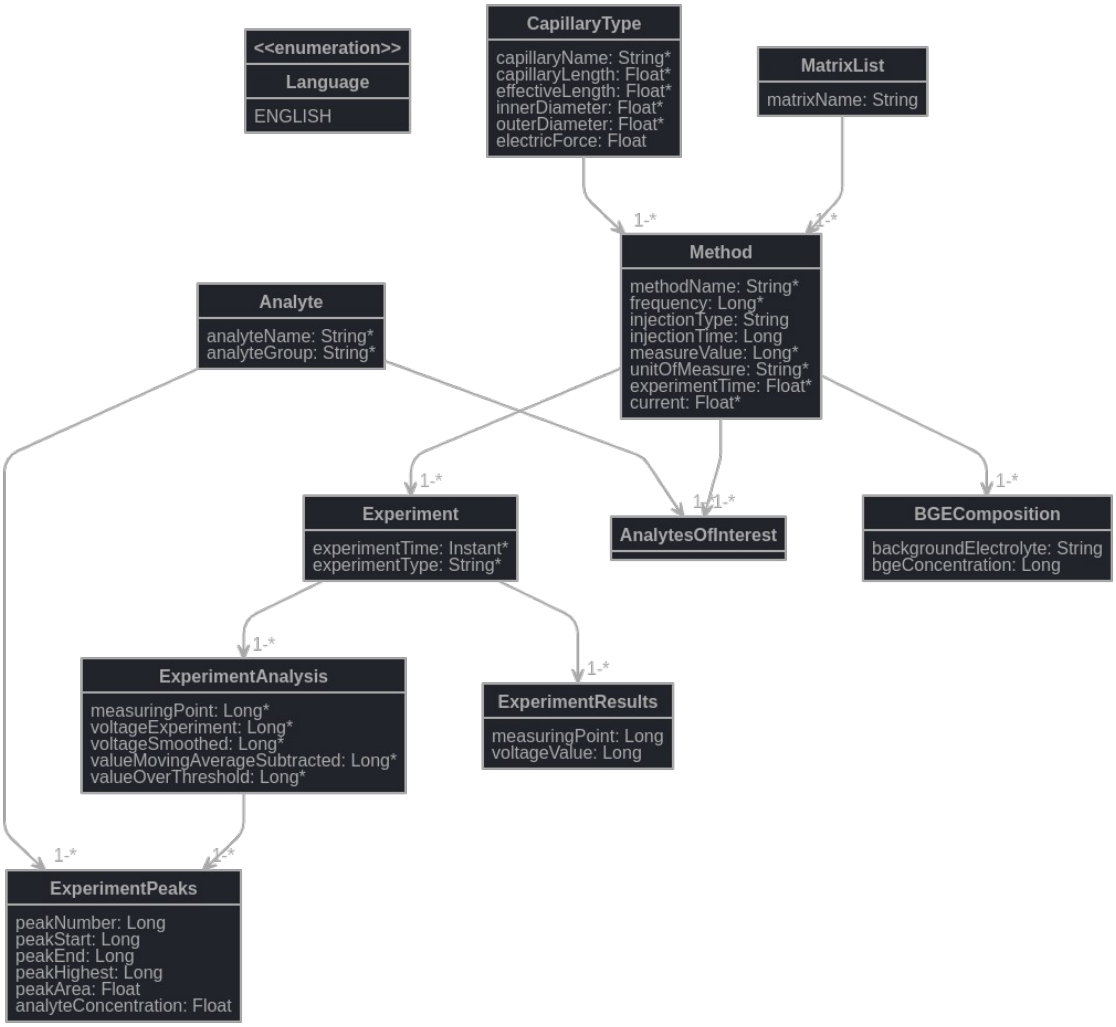
```
  concentration Double,
  area Double,
  deltaX Double,
  deltaY Double,
  upperXConfidence Double,
  upperYConfidence Double,
  belowXConfidence Double,
  belowYConfidence Double
}

entity ExperimentProcessParams{
    batchSize Integer,
    windowSize Integer,
    analyzeSt Integer,
    threshold Double
}

relationship OneToOne {
 ExperimentAnalysis to Experiment
 ExperimentAnalysis to ExperimentProcessParams
}

relationship OneToMany {
  BgeComposition to BgeCompositionOfInterest
  Method to BgeCompositionOfInterest
  Method to Experiment
  Method to MethodExecutionPlan
  CapillaryType to Method
  MatrixList to Method
  ExperimentAnalysis to ExperimentPeaks
  Experiment to ExperimentResults
  Experiment to AnalytesOfInterest
  ExperimentAnalysis to ExperimentAnalysisData
  ExperimentLocation to Experiment
  Analyte to AnalytesOfInterest
  Analyte to CalibrationLine
  Method to CalibrationLine
  CalibrationLine to CalibrationLineData
  Analyte to ExperimentPeaks
  ExperimentProcessParams to Method
```

```
}

relationship ManyToOne {
  Experiment{login} to User
}

relationship ManyToMany {
  Method{analyte} to Analyte{method}
}
```

Lisa 3 – Gitlab CI/CD tööde kirjeldus

```
stages:
  - test
  - build
  - deploy

run-tests:
  stage: test
  script:
    - sudo mvn verify

build-docker-image:
  stage: build
  script:
    - sudo mvn -Dprod -q compile jib:dockerBuild
  only:
    - master

deploy-image:
  stage: deploy
  script:
    - sudo docker-compose -f src/main/docker/app.yml up -d
  only:
    - master

after_script:
  - sudo git clean -ffxd
```

Lisa 4 – Algne andmebaasi struktuur



Joonis 30. Algne andmebaasi struktuur.