TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Raul Altmäe 179309IAIB
Robert Altmäe 179259IAIB
Karl-Joosep Karp 185398IAIB
Rihard Liiva 185767IAIB

Digitaalse mikrofluidika töölauarakendus

Bakalaureusetöö

Juhendaja: Evelin Halling
PhD,
Jelena Gorbatšova
PhD

Tallinn 2021

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: Raul Altmäe, Robert Altmäe, Karl-Joosep Karp, Rihard Liiva

18.05.2021

Annotatsioon

Käesoleva bakalaurusetöö eesmärgiks on edasi arendada Tallinna Tehnikaülikooli Loodusteaduskonna jaoks digitaalse mikrofluidika töölauarakendust, et võimaldada mikrovedelike liigutamist elektroodplaadil läbi mugava ja kaasaegse kasutajaliidese.

Digitaalne mikrofluidika on keemia valdkond, mis võimaldab indviduaalselt juhtida tilku mööda avatud elektroodide järjendit. See võimaldab teha muuhulgas täpset reaktiivide analüüsi, antikehade ja -geenide tuvastamist, DNA-proovidega manipuleerimist ja kliiniliste proovide diagnostikat. Vedeliku analüüsimiseks tehtava eksperimendi läbi viimiseks kasutatakse mitmest alamosast koosnevaid elektroonikakomponente, mida juhitakse läbi arvutis oleva rakenduse.

Antud töö raames on välja arendatud Electron, React ja Spring raamistikel põhinev töölauarakendus vedelikuanalüüsi eksperimendi juhtimiseks. Rakenduselt käskude elektroonikamoodulile saatmiseks kasutatakse Flaski raamistikku ning *serial* suhtlust.

Rakendus võimaldab teadlasel teostada vedelikuanalüüsi eksperimente, võimaldades rakenduses konfigureerida elektroodplaate, luua alamprogramme, seadistada vedelikupumpasid ning seda kõike käivitada ühe salvestatava eksperimendina.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 53 leheküljel, 8 peatükki, 20 joonist.

Abstract

The aim of this bachelor's thesis is to further develop the desktop application of digital microfluidics for the School of Science of Tallinn University of Technology in order to enable the movement of microfluids on an electrode plate through a convenient and modern user interface.

Digital microfluidics is a field of chemistry that allows individual droplet conduction along a sequence of open electrodes. This allows, among other things, accurate analysis of reagents, detection of antibodies and -genes, manipulation of DNA samples, and diagnosis of clinical samples. The fluid analysis experiment uses electronic components consisting of several sub-components, which are routed through a computer application.

In the scope of this work, a desktop application based on Electron, React and Spring frameworks has been developed for conducting a liquid analysis experiment. Flask framework and serial communication are used to send commands from the application to the electronics module.

The application allows researchers to perform fluid analysis experiments, allowing them to configure electrode plates, create subprograms, configure fluid pumps, and run it all as a single saveable experiment.

The thesis is written in Estonian and contains text on 53 pages, 8 chapters, 20 figures.

Lühendite ja mõistete sõnastik

CE	Capillary Electrophoresis, kapillaarelektroforees
CI/CD	Continuous Integration and Continuous Delivery, pidev integratsioon ja pidev tarne
CRUD	Create, Read, Update and Delete, looma, lugema, värskendama ja kustutama
DOM	Document Object Model, dokumendi objektimudel
HTTP	HyperText Transfer Protocol, hüperteksti edastusprotokoll
JDBC	Java Database Connectivity, Java andmebaaside ühenduvus

Sisukord

Jooniste loetelu

Sissejuhatus

Digitaalne mikrofluidika on teadusvaldkond, mis tegeleb elektrivälju kasutades vedelikutilkade analüüsiga. Sellisel viisil vedelikke analüüsides on potensiaalselt suur kasu meditsiinis, füüsikas, keemias ja isegi kosmosereisidel.

Antud bakalaureusetöö kirjeldab, kuidas on valmis arendatud töölauarakendus vedelikutilga analüüsimise eksperimendi läbi viimiseks. Töö on jaotatud mitmeks peatükiks. Neis analüüsitakse kasutusel olevat riistvara, kirjeldatakse tellija seatud nõudmisi, tuuakse välja kasutatud tehnoloogiad ja arhitektuur ning seletatakse, kuidas kogu tööprotsessi käigus rakendus valmis arendati. Autorid avaldavad ka arvamust, mil viisil oleks arendus sujuvamalt läinud ning toovad välja töö jooksul esinenud probleemid. Lisaks tuuakse välja projekti võimalikud tulevikuarendused.

Töö hüpotees:

Kas on võimalik arendada töötav töölauarakendus, et parandada vedelikutilga eksperimendi töövoogu?

Töö ülesanded:

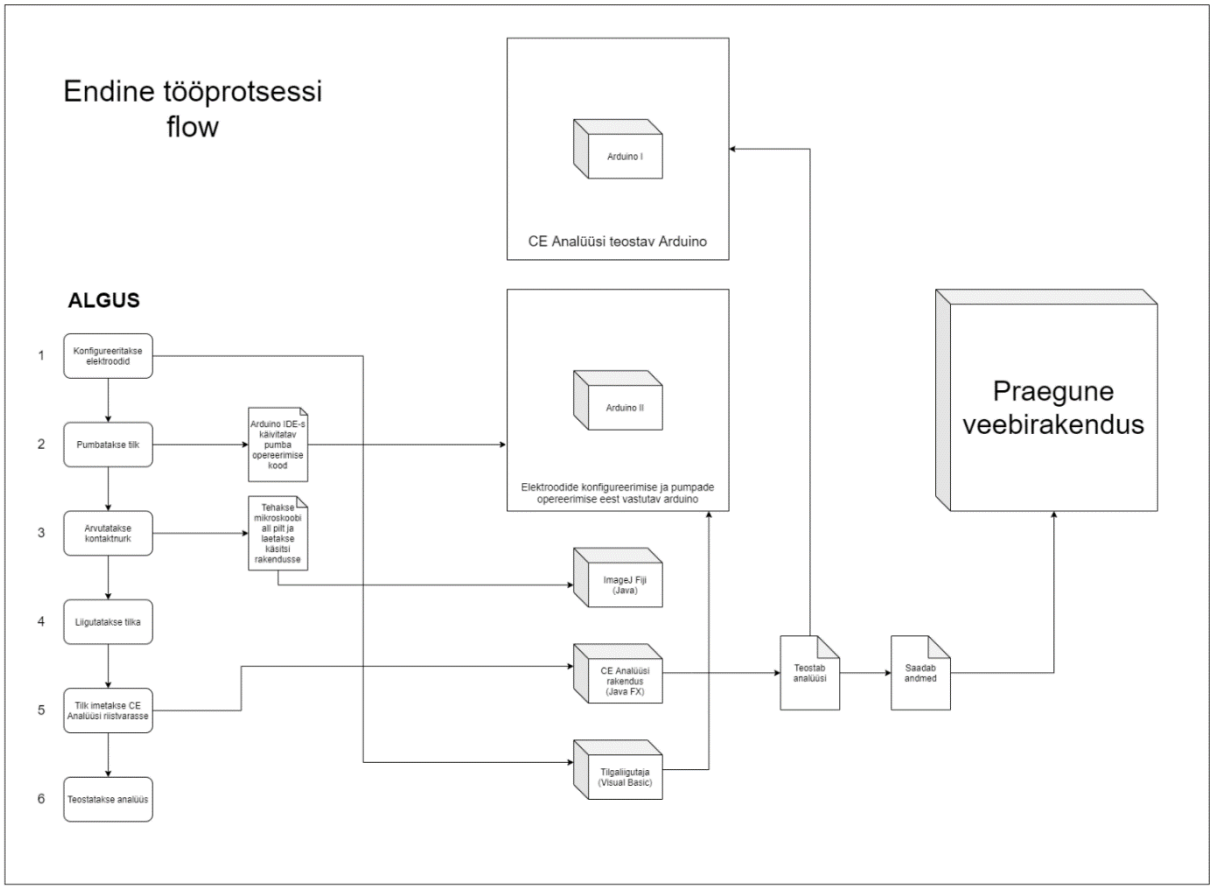
- Teha kindlaks tellija soovitud lahendus
- Arendada uus töölauarakendus
- Valideerida töövoog rakenduse ja olemasoleva füüsilise mooduliga

1. Ülesande püstitus

Antud töö raames oli meie ülesanne lihtsutada ning moderniseerida olemasolevat lahendust. Meie eesmärk oli teha töölauarakendus, mis ühendaks endasse mitmed erinevad rakendused ja katseprotsessid. Nendeks olid näiteks Tilgaliigutaja rakendus, mis tehti 2016. aastal Tallinna Tehnikaülikooli vanemlektori Martin Jaanuse poolt ning tilga pumpamine, mida varem tuli teostada käsitsi läbi Arduino IDE paarirealist koodi jooksutades. Olemasolev lahendus küll toimis, kuid eksperimendi läbiviimine oli äärmiselt aeganõudev ning seda ei olnud võimalik automatiseerida.

1.1. Olemasolev lahendus

Varasemalt olemasolevas lahenduses tuli esmalt konfigureerida elektroodid, kasutades eelmainitud Tilgaliigutaja rakendust. Seejärel tuli jooksutada Arduino IDEs paarirealist koodi, millega pumbati elektroodplaadile tilk. Seejärel arvutati ImageJ'ga, mis on avatud lähtekoodiga pilditöötlustarkvara [1], tilga kontaktnurk elektroodplaadi suhtes ning pärast seda kasutati Tilgaliigutaja rakendust, et panna tilk elektroodplaadil liikuma. Elektroodplaadi elektroodide pingestamise eesmärk on liigutada tilk punktist A ehk sealt, kuhu tilk esialgu pumbatakse, punkti B ehk torusse, mis imeb selle kapilaarelektroforeesi (edaspidi CE) analüüsi teostavasse masinasse, kus peale analüüsi teostamist saadetakse andmed kapillaarelektroforeesi andmetöötlust teostavasse veebirakendusse. (Joonis 1)



Joonis 1. Endine tööprotsessi voog

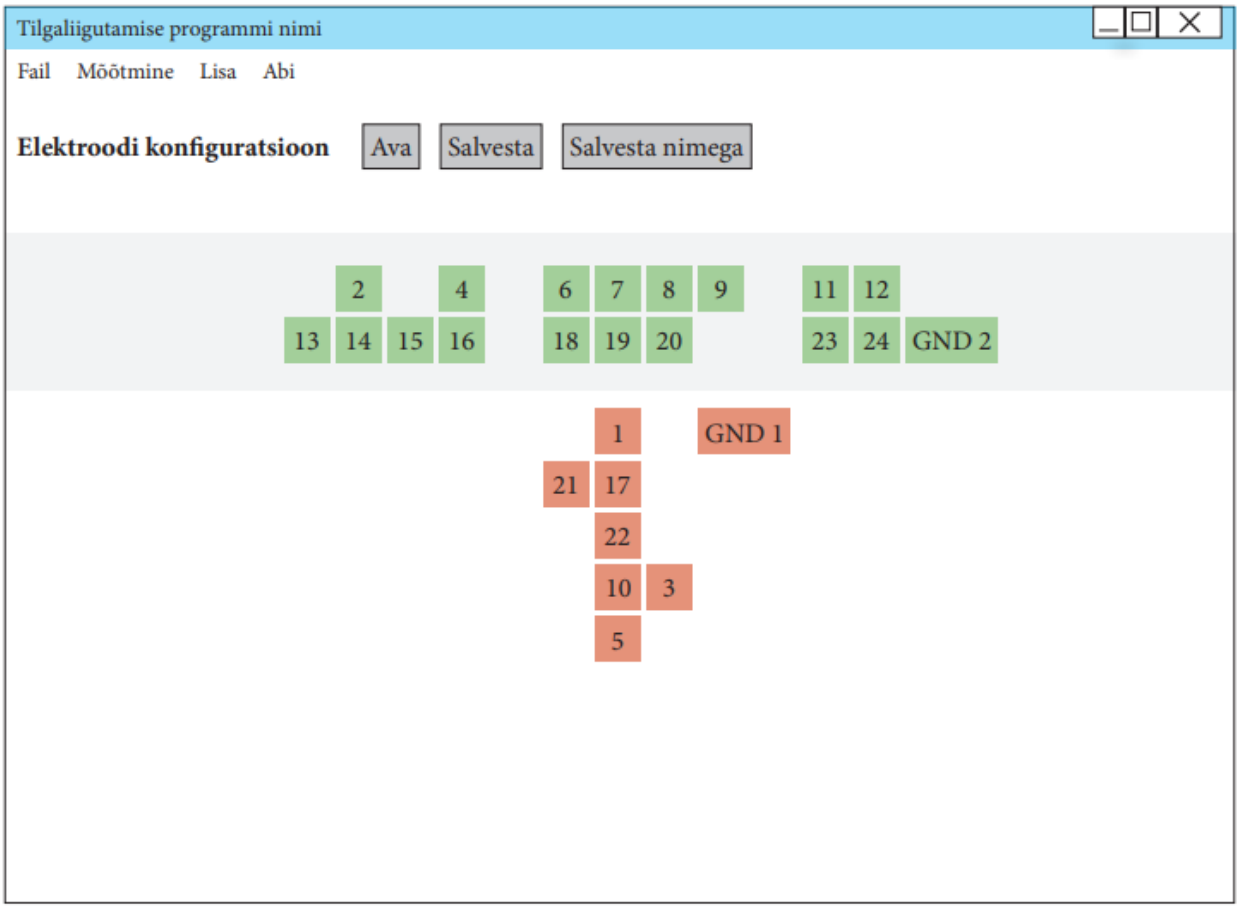
1.2. Probleem

Olemasoleva lahenduse probleem oli eksperimendi läbiviimisega kaasnev suur ajakulu ning manuaalse töö maht. Tellija kirjelduse järgi võis ühe katse läbiviimine võtta aega päevi, kuna pidevalt tuli erinevates rakendustes tegutseda, käsitsi andmeid ja tulemusi üles märkida, käsitsi testida, kas tilk üldse hakkab liikuma.

1.3. Tellija soovid

Tellijapoolsed soovid ja nägemus tulevasest rakendusest oli antud meile PDF faili kujul, kus asus esialgne rakenduse makett. Selle abil saime hakata planeerima rakenduse disaini ning kasutajavooge. Arvestasime kõikide otsuste langetamisel tellijapoolsete soovidega ning kõik küsimused said alati ühiselt lahendatud meie iganädalastel kohtumistel, kus osales nii meie juhendaja kui ka tellija ehk Jelena Gorbatšova. Samuti arvestasime ka jooksvalt tellija poolt

tulnud tagasisidega, mille tõttu tegime töölauarakenduse kasutajaliideses nii mõnegi suurema muudatuse. Alljärgnevalt on kujutatud tellija esialgset nägemust rakendusest. (Joonis 2)



Joonis 2. Esialgne tellijapoolne töölauarakenduse makett

2. Projekti kirjeldus

2.1. Digitaalne mikrofluidika

Digitaalne mikrofluidika on mikrovedelike käitlemisega tegelev tehnoloogia. Digitaalne mikrofluidika võimaldab indviduaalselt juhtida tilku mööda avatud elektroodide järjendit. Pikoliitrist kuni mikroliitrini ulatuvaid tilku saab panna liikuma, ühinema, eralduma. Tänu oma ainulaadsetele eelistele, vähekeerukatele seadmetele ning integratsiooni lihtsusele on digitaalne mikrofluidika kasutuses paljudes valdkondades [2].

2.1.1. Seadme disain ja implementatsioon

Harilikud digitaalse mikrofluidika seadmed sisaldavad üldiselt nelja komponenti: substraadid, elektroodid, dielektriline kiht ja hüdrofoobsed kihid. Laialdaselt on substraatidena kasutuses klaas ja räni. Meie seadmed kasutatavad substraadina trükkplaati, mis on madalate tootmiskuludega. Elektroodid moodustatakse tavaliselt metallidest või muudest juhtivatest materjalidest. Meie rakendusega seotud seadmete elektroodplaatide elektroodid on tehtud vasest. Elektroodide peale pannkse dielektriline kiht, mis hõlpsustab tilkade käitlemist. Hüdrofoobne kiht aitab vähendada pinnaenergiat, kiht on tavaliselt fluoropolümeer, näiteks Teflon® AF [2].

2.1.2. Rakendused

Kuna digitaalsel mikrofluidikal on palju unikaalseid ja kasulikke funktsioone, siis saab seda rakendada paljudes valdkondades, millest olulisimad on kirjeldatud järgnevalt [2]:

- **Keemilised ja ensümaatilised reaktsioonid** - Nõuab reaktiivide täpset doseerimist pumbast ning seejärel segamist mikroreaktorite loomiseks.
- **Immunotestid –** Lihtsustatult antikehade ja antigeenide tuvastamine
- **DNA-põhised rakendused –** DNA proovide manipuleerimine ja iseloomustamine
- **Kliiniline diagnostika –** Neli järjestikust etappi: proovide kogumine, proovide ettevalimstamine, analüütiline töötlemine ja tuvastamine

- **Proteoomika –** Nõuab mitmeastmelist proovianalüüsi erinevate detektoritega. Digitaalne mikrofluidika aitab erinevate reaktiividega individuaalselt tegeleda.

- **Rakupõhised rakendused –** Rakkude miniatuursel kujul testimine, katsete tegemine

2.1.3. Valdkonna tulevik

Koos tehnoloogia arenguga muutab ka digitaalne mikrofluidika odavamaks ja kättesaadavamaks nii tööstuses kui ka teadustöös tekivad elektroodivabad aktiveerimismeetodid, mis tuginevad optilistel jõududel, magnetismil, termakapillaarsetele jõududel [2].

2.2. Kapillaarelektroforees

Kapillaarelektroforees on lahutusmeetod, mis põhineb elektrivälja mõjul laetud osakeste liikumisel erineva kiirusega kapillaarkolonnis. Analüüsitavate osakeste migratsioonikiirus on erinev ja seda määratakse elektroosmootilise liikuvuse abil [3, p. 11].

Tänu digitaalsele mikrofluidikale, kogu loodud riistvarasüsteemile ja meie poolt arendatud rakendusele toimub kapillaarelektrofreesi analüüs täpsemalt, kuna tilk on jõudnud masinasse puhtas ja puutumata keskkonnas.

3. Projekti disain

Töölauarakenduse kasutamiseks peab kasutajal olema välja valitud vedelik või vedelikud, mis pannakse elektroodplaadi peal liikuma. Rakenduse tööks peab olema loodud ühendus läbi Arduino platvormi nii elektroodplaadi kui ka pumpadega. Rakenduses konfigureerib kasutaja vastavalt valitud elektroodplaadi järgi omale sobivad väljundid, sobiva kujuga ning määrab väljundite vahelised seosed. Katset alustades valib teadlane konfigureeritud elektroodplaadi konfiguratsiooini ning kirjeldatud eletroodide vahelised seosed. Katse jaoks on vaja uuritavat vedelikku, mis pumbatakse plaadile läbi töölauarakenduse liidese. Eduka katse korral liigub tilk kapillaarelektrofeesi masina juurde, mille toru imeb tilga endasse ning kus toimub edasine analüüs.

Digitaalmikrofluidika töölauarakenduse moodustavad alamsüsteemid, mis on implementeeritud järgnevate tehnoloogiate abil:

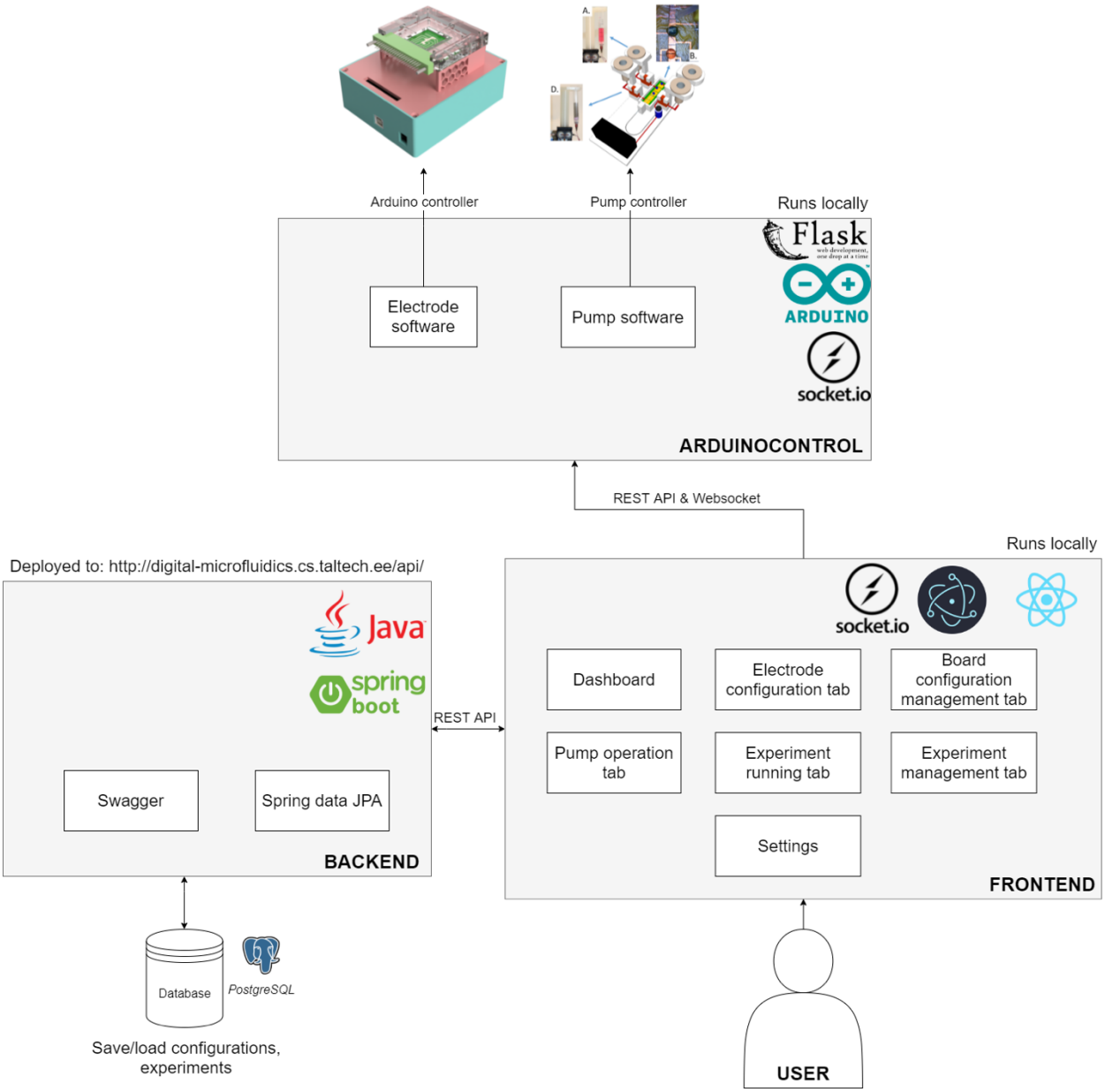
- Java
- PostgreSQL
- Electron, React
- Python
- Arduino

Projekti jaoks on implementeeritud kolm põhilist moodulit:

- Töölauarakendus – installeeritakse kasutajale lokaalselt; suhtleb HTTP päringutega Java rakendusliidesega; suhtleb HTTP ja WebSocketi kaudu Arduino juhtimismooduliga;
- Java rakendusliides – lähetatud Tallinna Tehnikaülikooli serverisse; suhtleb lokaalselt PostgreSQL andmebaasiga;

- Arduino juhtimistarkvara – installeeritakse kasutajale lokaalselt; käivitub töölauarakenduse avamisega automaatselt; suhtleb *serial* ühenduse kaudu Arduino mooduliga;

Järgneval joonisel on kujutatud projekti arhitektuurijoonis, kus on näha eelmainitud süsteemide ülesehitust. (Joonis 3)



Joonis 3. Projekti arhitektuur.

3.1. Arduino juhtimine

3.1.1. Flask

Flask on Pythonil baseeruv mikroraamistik, mille kasutamiseks pole vaja kindlaid tööriistu ega teeke. Flaski abil on lihtsasti võimalik implementeerida HTTP suhtluseks vajalikke otspunkte (*endpointe)*, olles samaaegselt kasutaja masinasse lihtsasti paigaldatav. [4]

3.1.2. Arduino

Arduino on vabavaralise riist- ja tarkvara arendusega tegelev firma, mis disainib ja toodab programmeeritavate mikrokontrolleritega trükkplaate. Plaatidel on lisaks erinevatele mikroprotsessoritele ka analoog- ja digitaalpinid, mida saab kasutada sisendi ja väljundina. Arduino mikrokontrollereid on võimalik programmeerida C-keelel põhinevas Arduino keeles, kasutades firma arendatud programmeerimisrakendust. [5]

3.1.3. Firmata

Firmata on teek, mis võimaldab suhelda Arduino plaadiga, kasutades Firmata protokolli. Eelmainitud teegi abil on võimalik lihtsasti üles seada suhtlus teadlase arvuti ja Arduino Mega plaadi vahel. [6]

3.1.4. PySerial

PySerial on vabavaraline teek, mis võimaldab Pythonis implementeerida suhtlust läbi serial-portide, seda nii Windows, OSX, Linux ja BSD operatsioonisüsteeme kasutatavatel masinatel. [7]

3.2. Java rakendusliides

3.2.1. Spring raamistik

Java rakendusliideses kasutatakse Spring raamistiku, mis põhineb Java platvormil. Spring pakub põhjaliku programmeerimis- ja konfiguratsioonimudelit Java-põhistele rakendustele. Saab kasutada alguskeelena ka näiteks Kotlin-it või Groovy-t. Projekti juures aitab Spring lihtsustada HTTP päringute tegemist ning JDBC ühenduse loomist andmebaasiga [8].

3.2.2. Gradle

Gradle on avatud lähtekoodiga koodiehitusautomaatika tööriist. Gradle lihtsustab sõltuvuste manageerimist, tänu millele ei pea sõltuvusi alla laadima ja installeerima [9].

3.2.3. Hoidla-teenuse muster

Java kood on jaotatud kiht-arhitektuuri ja domeenide järgi kolmeks kihiks:

- Kontrollerid (*controller*) – esitluskiht, presenteerib teenuse kihtis tulevat andmestikku.
- Teenused (*service*) – rakendusteenuse kiht või äriloogika kiht, määratakse transaktsiooni piirid
- Hoidlad (*repository*) – domeeni kiht, abstraktne kiht andmebaasi peal, mis suhtleb andmeallikaga

Valitud arhitektuur on Springi rakendusi luues väga populaarne ning aitab hästi koodi struktureerida ja vajalikud aspektid lahus hoida.

3.3. PostgreSQL andmebaas

Rakendus on liidestatud PostgreSQL andmebaasiga, mis võimaldab andmete salvestamist, lugemist, muutmist ja kustutamist. Andmebaasi olemid genereeritakse Hibernate abil, defineerides väljad Java domeeni objektide peal ning mudelitevahelised seosed üks ühele (*OneToOne*) või üks paljudele (*OneToMany*) jne.

3.4. Töölauarakendus

Töölauarakenduse arendamisesse läks meie põhiline aeg ja fookus. Tegemist on Electroni platvormile ehitatud ning Reacti Javascripti teeki kasutades Typescriptis kirjutatud rakendusega, mis töötab teoorias nii Windowsil, macOS-il kui ka Linuxil, kuid tellija on seda testinud ja kasutanud hetkel Windowsi platvormil. Meie eesmärk oli teha töölauarakendus võimalikult kasutajasõbralik ning töökindel.

3.4.1. Installeerimine

Töölauarakenduse paigaldusfaili maht on ligikaudu 85 MB ning see paigaldatakse kasutaja arvutisse, Windowsi puhul AppData kausta. AppData kaust on kasutajapõhine ning see tähendab, et rakendus installeeritakse ainult kindlale operatsioonisüsteemi kasutajale. Rakendus on allalaaditav Gitlabi repositooriumi väljalasete sektsioonist[1].

Rakenduse installeerimiseks vajalikud süsteemi miinimumnõuded ei ole täpselt määratletud, kuid Electron on välja toonud platvormid, mida nad toetavad. Järelikult on meie rakenduse kasutamiseks vajalik üks järgnevatest operatsioonisüsteemidest: [10]

- Windows 7 (32- või 64-bit) või uuem
- macOS 10.11 (64-bit) või uuem
- Ubuntu 18.04 (32- või 64-bit) või uuem

Installeerimise lihtsustamiseks koostasime ka repositooriumi wiki sektsiooni installeerimise juhise. Ainuke meiepoolne nõue rakenduse jooksutamisele on lisaks veel Python 3.0 (või kõrgema) versiooni olemasolu. Kasutaja peab installeerimisjuhise järgi ka käsurealt paigaldama omale järgnevad Pythoni teegid [11]

- Flask
- Flask-socketio
- Flask_cors
- Serial
- Gevent-websocket

[1] https://gitlab.cs.ttu.ee/water-analyzer/digital-microfluidics/-/releases

3.4.2. Electron

Electron on tööriist, millega on võimalik ehitada platvormist sõltumatut töölauarakendust, mille sisu on ehitatud Javascripti, HTMLi ja CSSiga. Electroniga koos paigaldatakse ka kasutaja arvutisse Chromium ning Node.js [12].

Chromium on avatud lähtekoodiga brauser [13], millel põhinevad ka mitmed teised tuntud brauserid, näiteks Google Chrome, Microsoft Edge, Opera ja Brave [14]. Kokkuvõtvalt, Electron kasutab Chromiumi kuvamaks kasutajale rakenduse sisu, mis Javascripti, CSSi ja HTML'iga kirjutatud on.

Electroni rakendus on olemuselt Node.js rakendus, see omab endas package.json faili ning seda saab käivitada arendades käsurealt nagu iga teist Node.js'i rakendust. Rakendusele määratakse „package.json" failis ära sisenemispunkt, kuhu kirjutatakse loogika rakenduse akna avamiseks ning seal kuvatava sisu juhtimiseks [15].

Meie rakenduse sisenemispunkt Electroni mõistes on „electron.js" fail frontend/public kaustas, kus määratakse akna suurus, laius, mõned lisaparameetrid ning kaust, kus asub Reacti poolt valmis ehitatud „index.html" fail, mida aknas kuvama hakatakse.

3.4.3. React

React on Facebook'i poolt loodud ning hallatav Javascripti teek kasutajaliideste ehitamiseks [16]. Otsustasime kasutada antud projektis Reacti, kuna meil oli sellega varasem kogemus, see on tööturul nõutud oskus ning see on väga levinud teek praegusel ajahetkel ehk selle kohta on väga lihtne vajadusel lisainfot otsida.

Meie töölauarakenduses on kasutusel Reacti versioon 17, mis tuli algselt välja 20. oktoobril 2020. aastal. Otsutasime kasutada projektis ainult Reacti funktsionaalseid komponente ning nendega kaasnevaid „*hooks*" funktsioone.

3.4.4. Material-UI

Material-UI on meie projektis kasutusel olev Reacti kasutajaliideste komponentide raamistik. Valisime selle seetõttu, et sellel on väga palju valmiskomponente ning see on väga laialdaselt kasutusel olev raamistik, tänu millele leidub foorumites selle kohta palju informatsiooni.

Material-UI pakub lihtsasti stiilitavaid, valmiskujul olevaid Reacti komponente, mida saab atribuutidega juhtida. Material-UI raamistik on kasutusel näiteks nii NASA, J.P.Morgani kui Netflixi ettevõttes [17].

3.4.5. Socket.IO

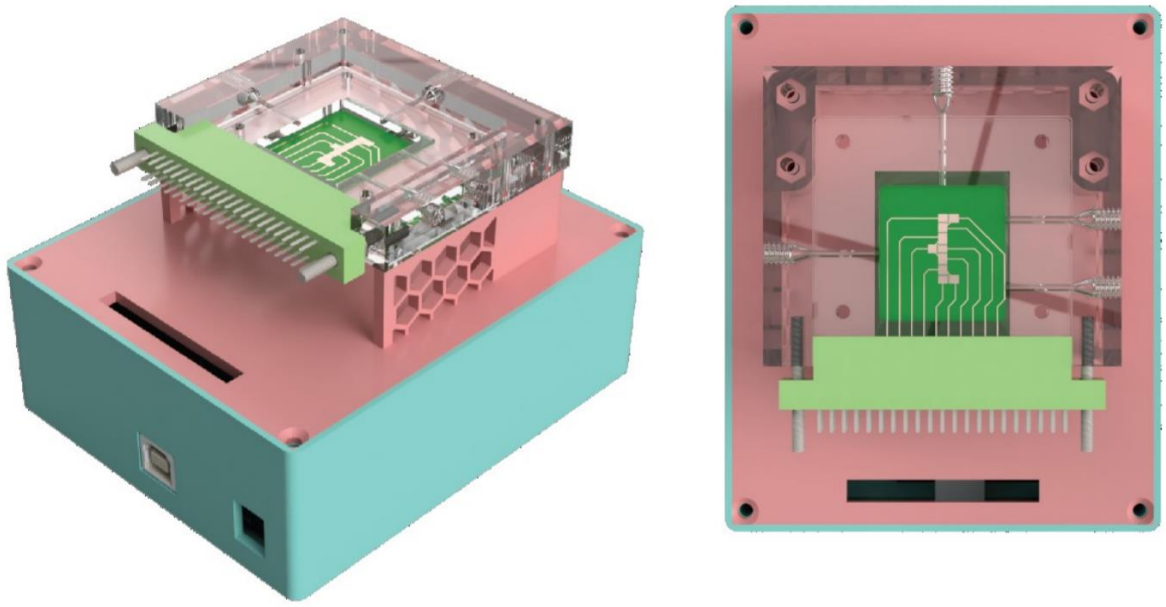
Meie rakenduses on kasutusel ka Socket.IO Javascripti ning Pythoni teek, mis võimaldab reaalajas bidirektsionaalset suhtlust serveri ja kliendi vahel [18]. Meie puhul on serveriks Arduinot juhtiv Pythoni rakendus ning kliendiks meie töölauarakendus.

Vajasime reaalajas toimuvat ühendust just seetõttu, et me kuvame kasutajale mitmes erivaates aktiivseid elektroodide ühendusi ning nendele mõjuvat pinget ning sagedust. Kuna katse käigus võib see info muutuda väga kiiresti ning tihti, siis otsustasime tavaliste HTTP päringute asemel selle teegi kasuks.

3.5. Riistvara ülevaade

Järgnevas peatükis on ülevaade projektis kasutusel olnud riistvarast, sealhulgas digitaalse mikrofluidika platvorm, Arduino Mega, Microchip HV507 ja elektroodplaat.

3.5.1. DMF platvorm



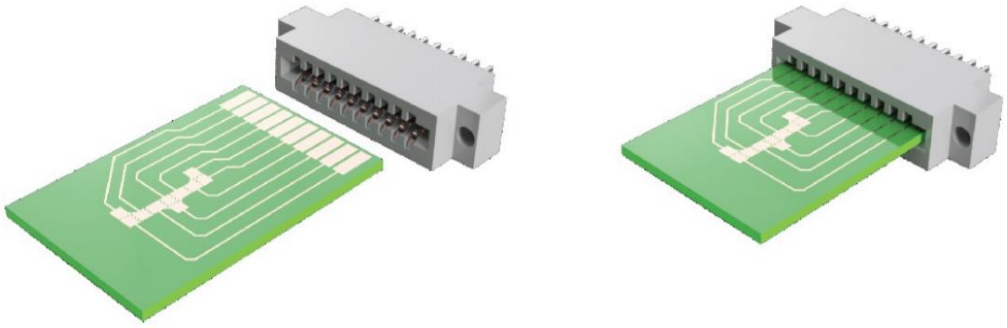
Joonis 4. Tilgaliigutaja instrumendi disain [19]

Digitaalse mikrofluidika uurimiseks disainitud instrument koosneb elektroonikakomponentidest, mis on ehitatud *shield*-tüüpi mooduliks (Joonis 4). Moodul on paigutatud 3D-prinditud plastikust karpi. Seda on võimalik ühendada arvutiga läbi USB-B kaabli ning vajab töötamiseks 15 V, 1 A toiteallikat [19]. Alljärgnevalt on kirjeldatud mooduli olulisemad komponendid.

Arduino Mega on trükkplaat, mis baseerub ATmega2560 mikrokontrolleril. Sellel on 54 digitaalväljundi- ja sisendina kasutatavat pini, neist 15 saab kasutada PWM väljunditena. Analoogsisendiks on võimalik kasutada 16 sisendpini. Plaati on võimalik läbi USB ühendada arvutiga ning sealt programmikoodi peale laadida [20]

Elektrilise signaali kommuteerimiseks on kasutusel Microchip HV507 seeria-paralleel konverter, millel on 64 kõrgepingeväljundit. See võimaldab väljundpinget seadistada vahemikus 5-300 V sagedusel kuni 100 kHz [19]. Kõrgepingeallikaks kasutatakse EMCO C03 elementi, mis võimaldab reguleerida pinget vahemikus 0-300 V [19].

Moodul ühendatakse EDAC Card Edge ühendusi kasutades vasest elektroodidega elektroodplaadi külge [19]. Alloleval joonisel on illustreeritud elektroodplaadi ühendamine kontaktidega. (Joonis 5)



Joonis 5. Elektroodplaadi ühendamine kontaktidega [19]

4. Projekti sisu

4.1. Rakenduse vaated

Rakendus koosneb mitmest erinevast vaatest. Käivitades kuvatakse kasutajale vaikimisi vaadet „Töölaud“, kus tulevikus hakatakse kasutajale kuvama eelnevate eksperimentidega seotud infot ning üldandmeid. Rakenduse vasakus ääres asub sahtel-menüü, mille abil saab navigeerida erinevatele vaadetele.

4.1.1. Elektroodplaadi konfigureerimine

Elektroodplaadi konfigureerimise vaatesse saab navigeerida läbi peamenüü. Vaate avanedes kuvatakse kasutajale kolmest sammust koosnev voog. Igas sammus näeb ka vajadusel kahte ühendatud kaamera pilti.

Esimeses sammus kuvatakse kasutajale 4x10 ruudustik pin-idest, kus kasutaja valib vastavalt elektroodplaadile soovitud pinid.

Teises sammus näeb kasutaja vasakul olemasolevaid varem konfigureeritud kujundeid ning paremal saab ta luua uue eelnevas sammus valitud pinidega. Elektroodide paigutus on oluline, et tilk jõuaks lõpuks õigele kohale või teeks katses läbi soovitud raja.

Kolmandas sammus suunatakse kasutaja alamprogrammi konfigureerimise vaatesse, mille kirjeldus on järgmises punktis.

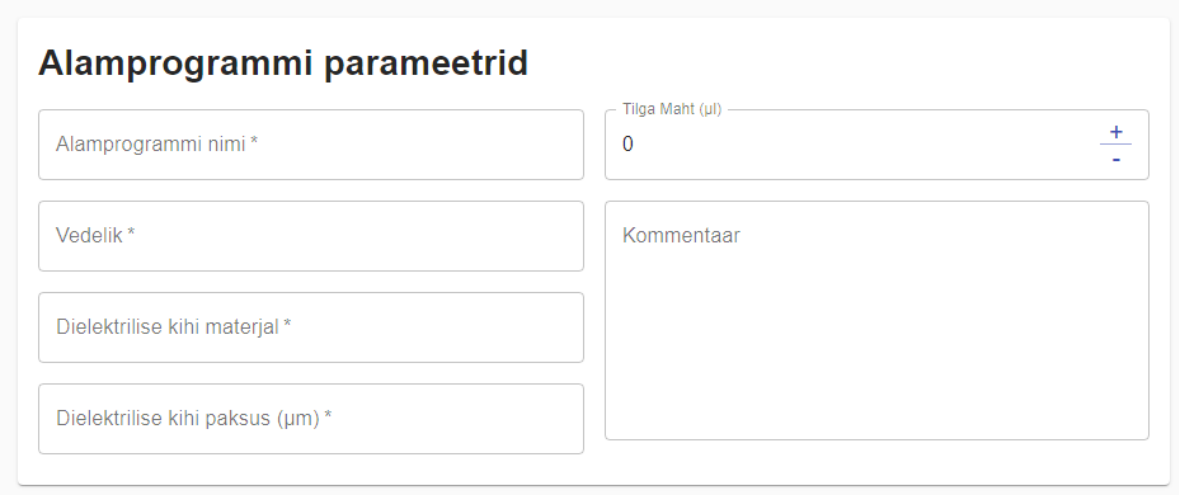
4.1.2. Alamprogrammi konfigureerimine

Alamprogrammi konfigureerimisvaade on elektroodplaadi konfigureerimise vaate kolmandas sammus asuv põhikomponent. Vaate avanedes kuvatakse kasutajale kuuest osast koosnev vorm:

1. Alamprogrammi parameetrid
2. Pinge ja sageduse seadistamise
3. Testimise paneel

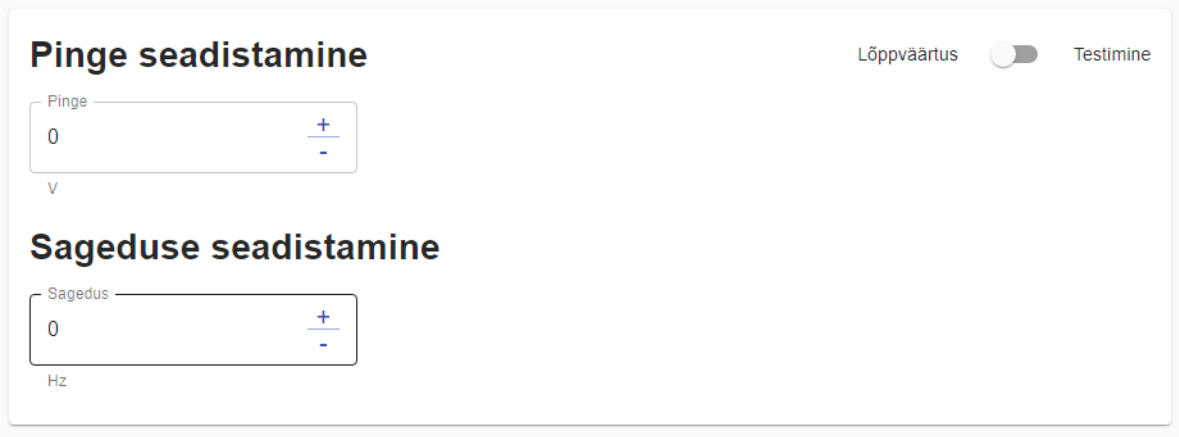
4. Elektroodide ühendused
5. Elektroodplaadi paigutus
6. Konfiguratsiooni lõpetamine

Alamprogrammi parameetrites asuvad väljad, mille eesmärgiks on kirjeldada antud alamprogrammi. Need on olulised, et kasutaja tunneks ära, mille jaoks ta antud alamprogrammi konfigureeris. (Joonis 6)



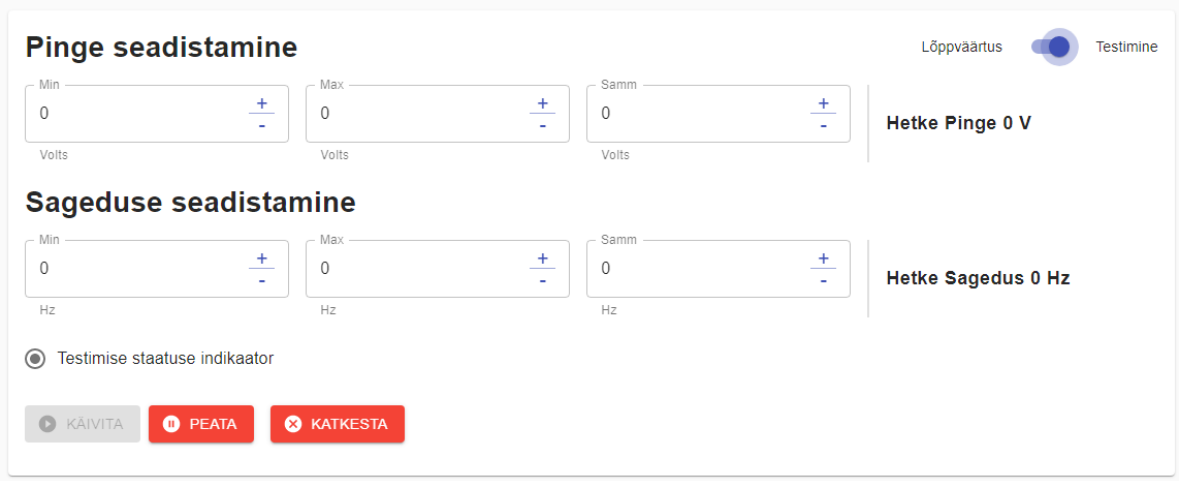
Joonis 6. Alamprogrammi parameetrid

Pinge ja sageduse seadistamise paneelis saab seadistada kõikidele elektroodide ühendustele korraga sageduse ja pinge. Eraldi igale elektroodi ühendusele pinge ja sageduse määramine on aega nõudev ja ebamugav tegevus. Paneeli eesmärk on, et kasutaja saaks kiiremini ja mugavamalt elektroodide ühendustele määrata pinge ja sageduse. (Joonis 7)



Joonis 7. Pinge ja sageduse seadistamise paneel

Testimise paneelile saab ligi, kui kasutaja vajutab pinge ja sageduse seadistamise paneelis testimise nupule. Testimispaneelis saab määrata pingele kui ka sagedusele miinimum-, maksimum- ja sammu väärtused. Kui kasutaja testimise käivitab, siis saadetakse edasi kõikidele elektroodide ühendustele määratud väärtused, alustades miinimumpingest ja -sagedusest. Lisaks käivitatakse ka ekraani salvestamine pärastiseks analüüsiks. Edasi minnakse määratud sammu alusel, kuni jõutakse maksimumini. Testimispaneeli eesmärgiks on anda kasutajale võimalus leida õige pinge ja sagedus, millel veetilk elektronplaadil liikuma hakkab. (Joonis 8)

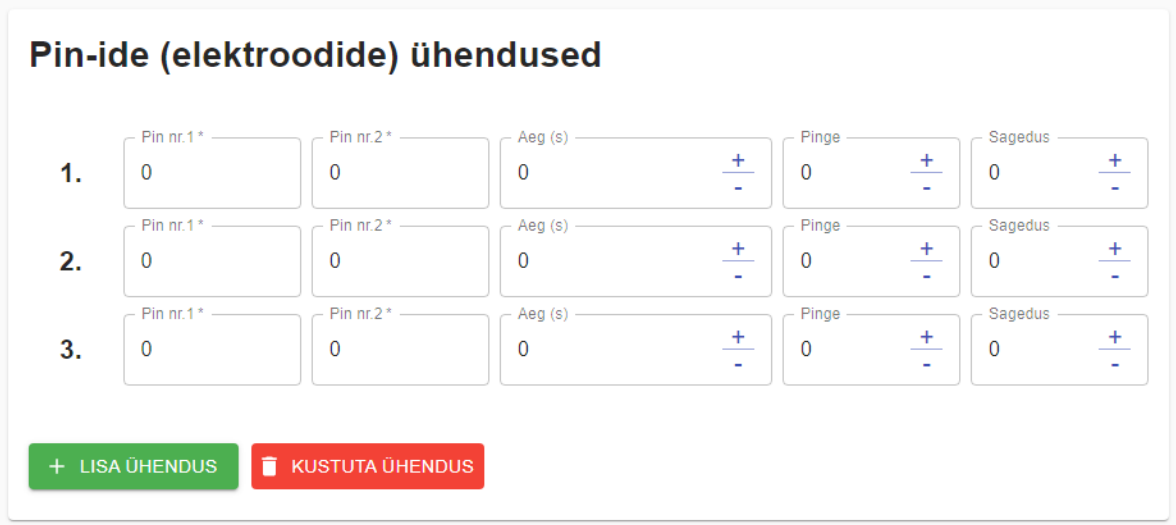


Joonis 8. Testimise paneel

Elektroodide ühenduste paneelis saab määrata elektroodide ühendusi, mis koosnevad:

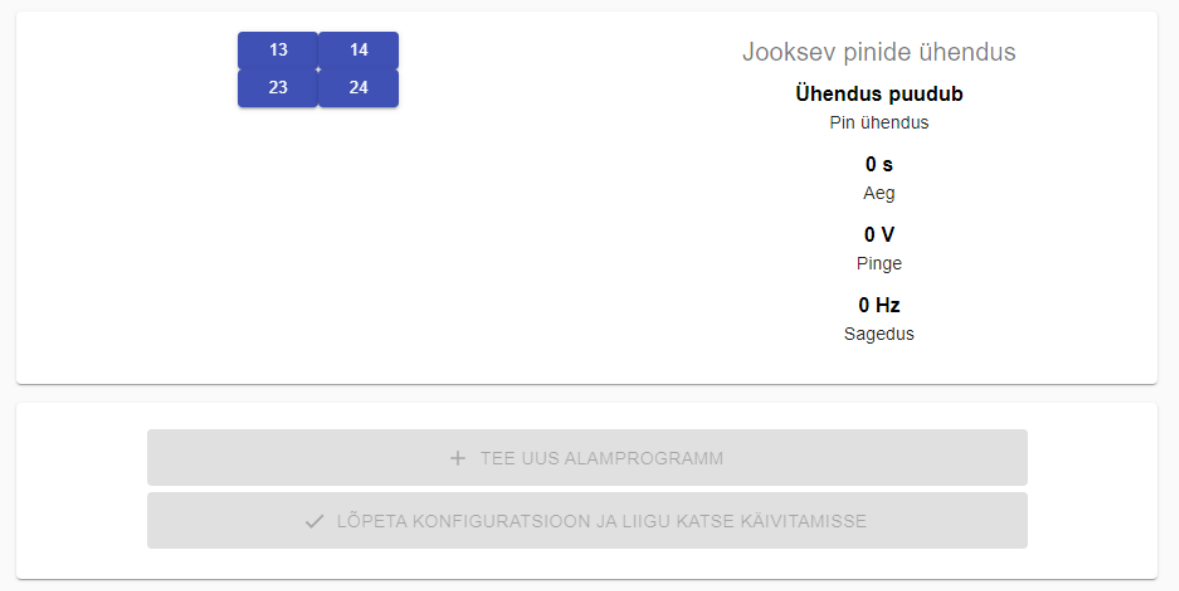
- Elektroodidest, mida käivitatakse
- Pinge ja sageduse ajast
- Pinge väärtusest
- Sageduse väärtusest

Paneelis saab kustutada ja lisada elektroodide ühendusi. Elektroodide ühenduste paneeli eesmärgiks on elektroodide vahelise ühenduse seadistamine, mille tulemusena hakkab vedelikutilk elektroodplaadil liikuma. (Joonis 9)



Joonis 9. Elektroodide ühenduste paneel

Elektroodide konfiguratsiooni paneelis kuvatakse kasutajale hetkel pooleliolevat elektroodkonfiguratsiooni ja jooksvat pinide ühendust. Jooksva pinide ühenduse all näidatakse kasutajale aega, pinget ja sagedust, mida testimispaneel hetkel välja saadab. Paneeli eesmärgiks on kasutajale teha elektroodide konfigureerimine mugavamaks, kuna siis on kasutajal elektrood konfiguratsioon visuaalselt pidevalt kujutatud. Lisaks teeb see rakenduse töö läbipaistvamaks. (Joonis 10)



Joonis 10. Elektroodide konfiguratsiooni paneel

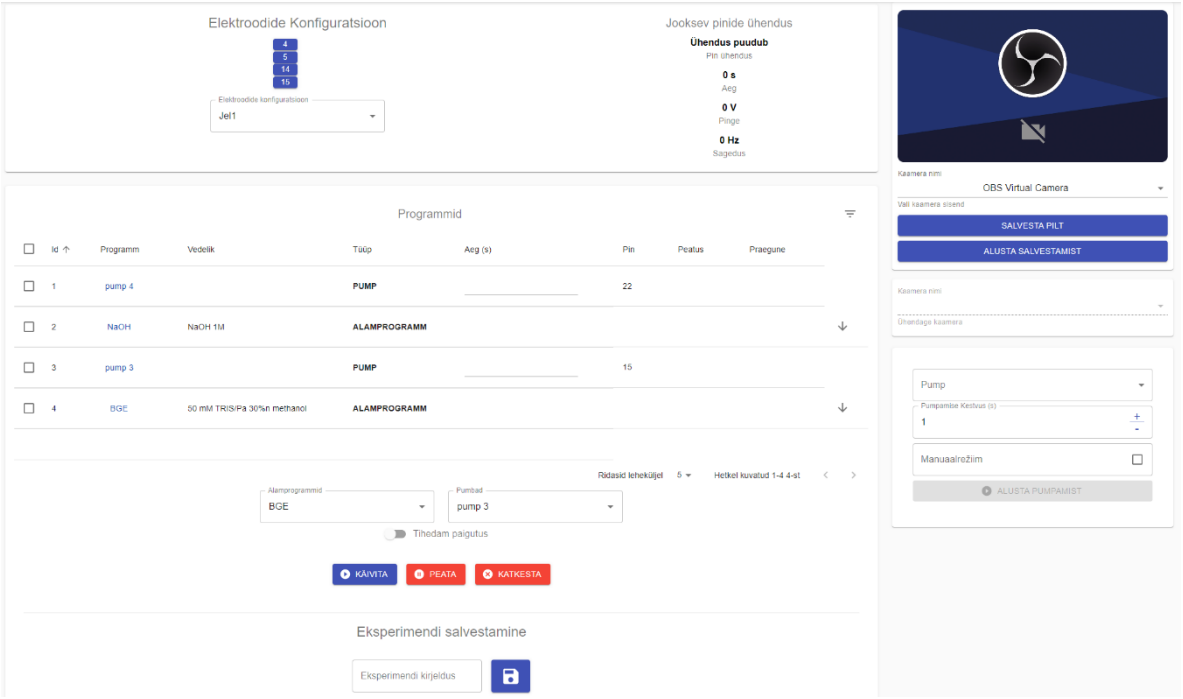
Konfiguratsiooni lõpetamise paneelis on võimalik kasutajal olemas olev alamprogramm salvestada ja alustada uue alamprogrammi lisamist või lõpetada konfiguratsiooni lisamine.

4.1.3. Eksperimendi käivitamine

Eksperimendi käivitamise vaatesse jõuab kasutaja kas menüüst navigeerides või elektroodide konfigureerimise vaatest vajutades „Lõpeta konfiguratsioon ja liigu katse käivitamise vaatesse“. Siin vaates toimub reaalse eksperimendi konfigureerimine ning erinevate alamprogrammide käivitamine. Meie rakenduses moodustab eksperimendi erinevate alamprogrammide ja pumpamiste käskude loetelu. Esimesena peab kasutaja valima elektroodide konfiguratsiooni, mida ta kasutada soovib. Seejärel on tal võimalik laadida olemasolev või luua uus eksperiment, mida konfigureerima hakata.

Eksperimendi konfigureerimine toimub tabelvaates. Kasutaja saab lisada järjest alamprogramme ning pumpamiskäsklusi, mida järjest ülevalt alla Arduinos käivitama hakatakse. Tellija erisoovil lisasime ka peatumispunktide lisamise võimekuse, mis tähendab seda, et eksperimendi käsklusi käivitatakse järjest, kuni jõutakse peatumispunktini ning seejärel peab kasutaja käsitsi „Jätka“ vajutama, et jätkata. Tabelvaate eesmärk on, et kasutaja saab kombineerida mitu erinevat alamprogrammi ning pumbakäsklust ning selle abil suure mahuka katse läbi viia ilma ise vahepeal sekkumata.

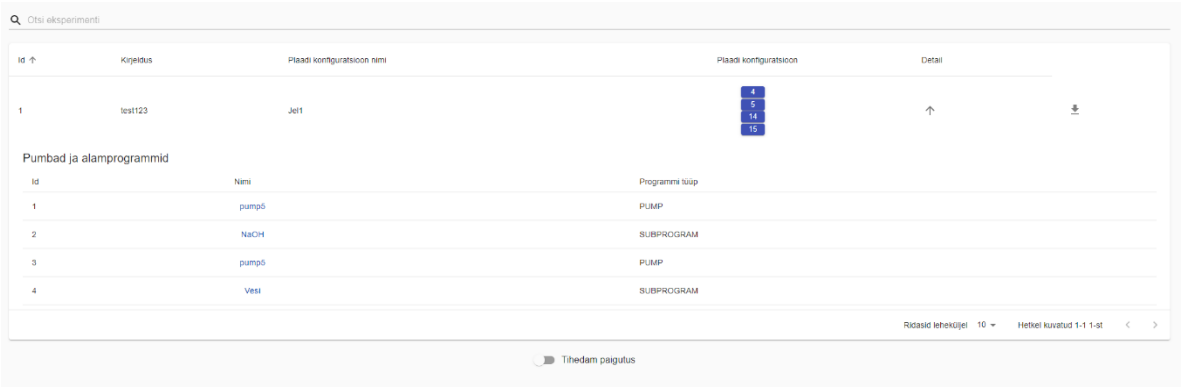
Tabelvaates olevat konfiguratsiooni on võimalik ka tulevikus kasutamiseks salvestada. Andes talle tabeli allosas oleva vormi kaudu nime ning vajutades salvesta nuppu, salvestatakse see andmebaasi ning seda saab hallata „Eelnevate eksperimentide haldamise“ vaates. Samuti on võimalik seda ka tabeli kohal asuvas sektsioonis sisse laadida. (Joonis 11)



Joonis 11. Eksperimendi käivitamise vaade.

4.1.4. Eksperimentide loendivaade

Eksperimentide loendivaates kuvatakse kasutajale loend varem salvestatud eksperimentidest. Iga rea peal näeb infot ekperimendi kohta. Kui eksperiment avada, siis kuvatakse kasutajale eksperimendiga seotud alamprogrammid ja pumbad. Alamprogrammi alt näeb näeb ka konfigureeritud pini ühendusi ning saab navigeerida ka alamprogrammi muutmise vaatesse. Eksperimente kuvatakse lehtede kaupa. Loendi peal asub ka otsinguriba, mille järgi saab eksperimente filtreerida. (Joonis 12)

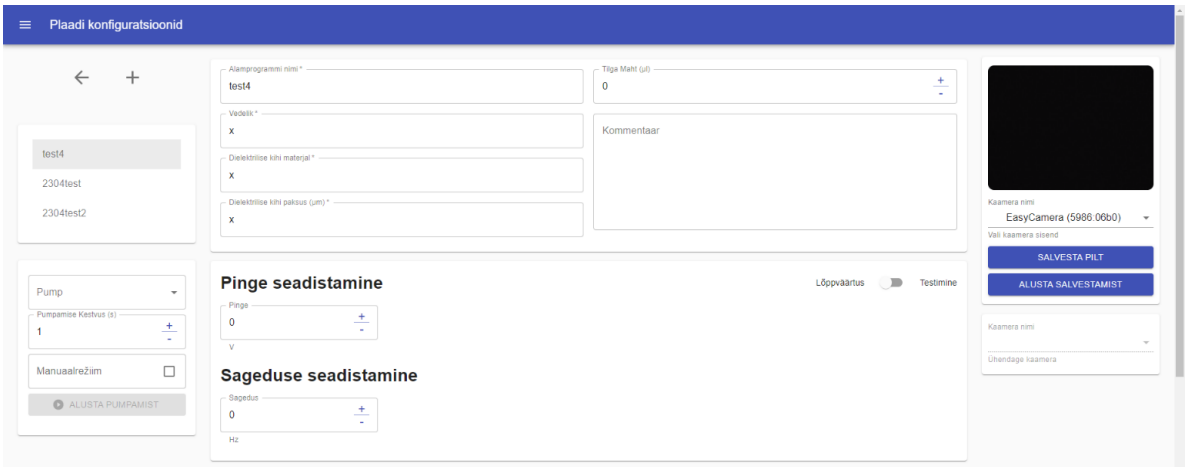


Joonis 12. Eksperimentide loendivaade

4.1.5. Alamprogrammi modifitseerimine

Alamprogrammi modifitseerimisvaatesse saab kasutaja ligi läbi plaadi konfiguratsiooni või läbi eksperimendi loendivaate, kui vajutada alamprogrammi nime peale. Alamprogrammi modifitseerimisvaate põhikomponendiks on alamprogrammi konfigureerimise vaade. Lisaks kuvatakse vasakul tagasimineku võimalus ja uue alamprogrammi lisamise võimalus. Selle all kuvatakse antud plaadi konfiguratsiooni alamprogrammide nimekirja ja pumba opereerimiskomponenti. (Joonis 13)

Alamprogrammi modifitseerimisvaate eesmärgiks on kasutaja mugavus, selle vaate abile on kasutajal võimalik plaadi konfiguratsioonile lihtsalt lisada ja muuta alamprogramme.



Joonis 13. Alamprogrammi modifitseerimise vaade

4.1.6. Kaamera konfigureerimine

Kaamera konfigureerimise komponenti kuvatakse järnevates vaadetes:

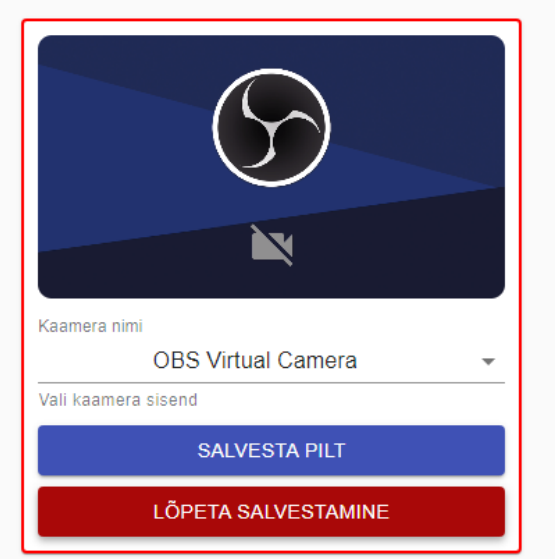
- Töölaud
- Elektroodplaadi konfigureerimine
- Eksperimendi käivitamine
- Alamprogrammi konfigureerimine
- Alamprogrammi modifitseerimine

Kaamera konfigureerimise komponendis saab valida erinevate kaamerate vahel, kuna enamikel vaadetel on kaks kaamera komponenti. (Joonis 14)

Komponendis saab hetkel valitud kaamera pilti salvestada, mis avab kasutajale pildi allalaadimisakna. Lisaks on võimalik komponendis ka panna kaamera videot salvestama. Kui vajutada „Lõpeta salvestamine“, siis avatakse kasutajale video allalaadimisaken. (Joonis 15)

Kaamera komponendi eesmärk on, et kasutaja saab lihtsalt jälgida elektroodplaadi peal toimuvat ning vajadusel saab toimuva peatada.

Joonis 14. Kaamera konfigureerimise komponent



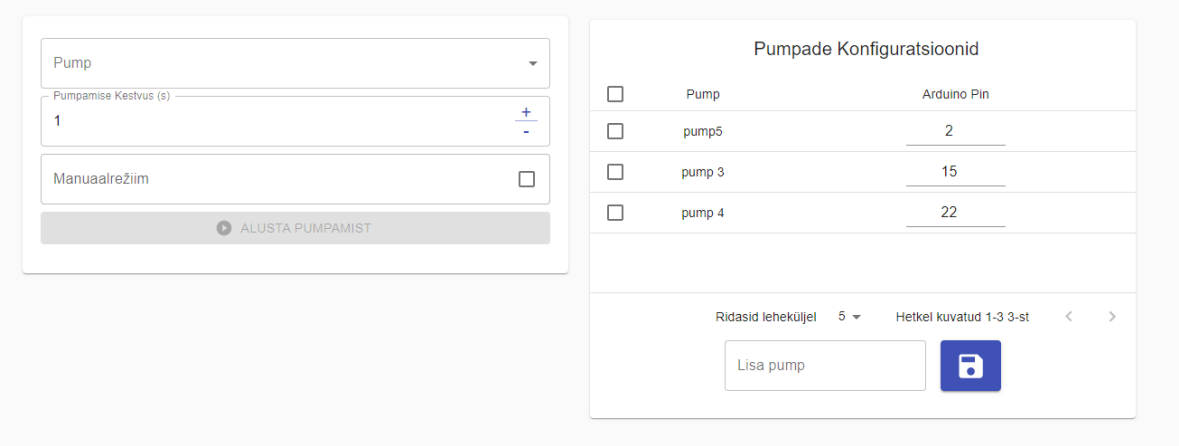
Joonis 15. Kaamera konfigureerimise komponent salvestamas videot

4.1.7. Pumba opereerimise vaade

Pumba opereerimise vaate eesmärk on võimaldada kasutajal hallata olemasolevate pumpade konfiguratsiooni ning opereerida pumpasid. Pumbale määrab kasutaja nime ning Arduino pini, mille külge see pump ühendatud on.

Antud vaate üks põhiosa on pumba juhtnuppude sektsioon. See on lihtsasti taaskasutatav Reacti komponent, mille saab ilma suurema vaevata teistesse vaadetesse liidestada. Hetkel on see kasutuses nii eksperimendi alustamise vaates kui ka alamprogrammi konfigureerimise vaates. Kasutaja peab antud paneelis valima aktiivse pumba, mida soovitatakse kasutada, määrama pumpamise kestvuse ning käivitama pumpamise. Ajalise pumpamise käivitamisel kuvatakse kasutajale ka edenemisriba, mis näitab pumpamise progressi. Samuti on kasutajal võimalik ka kasutada manuaalset režiimi. Manuaalses režiimis toimib pumpamine senikaua, kuni kasutaja selle lõpetab. Pumpamise juhtpaneeli eesmärk oli anda kasutajale võimalus enne päris eksperimendi käivitamist testida pumpamist.

Teine pumba opereerimise vaate osa on pumpade konfigureerimise tabel. Tabelis kuvatakse kõikide kasutaja poolt tehtud pumpade nimed ning nendele vastavad Arduino pinid. Tabelis olevaid ridu saab mitmekaupa kustutada ning nimele vastavat Arduino pini muuta. Uue pumba sisestamine toimub sama tabeli allosas, kuhu kasutaja peab sisestama pumba nime. (Joonis 16)

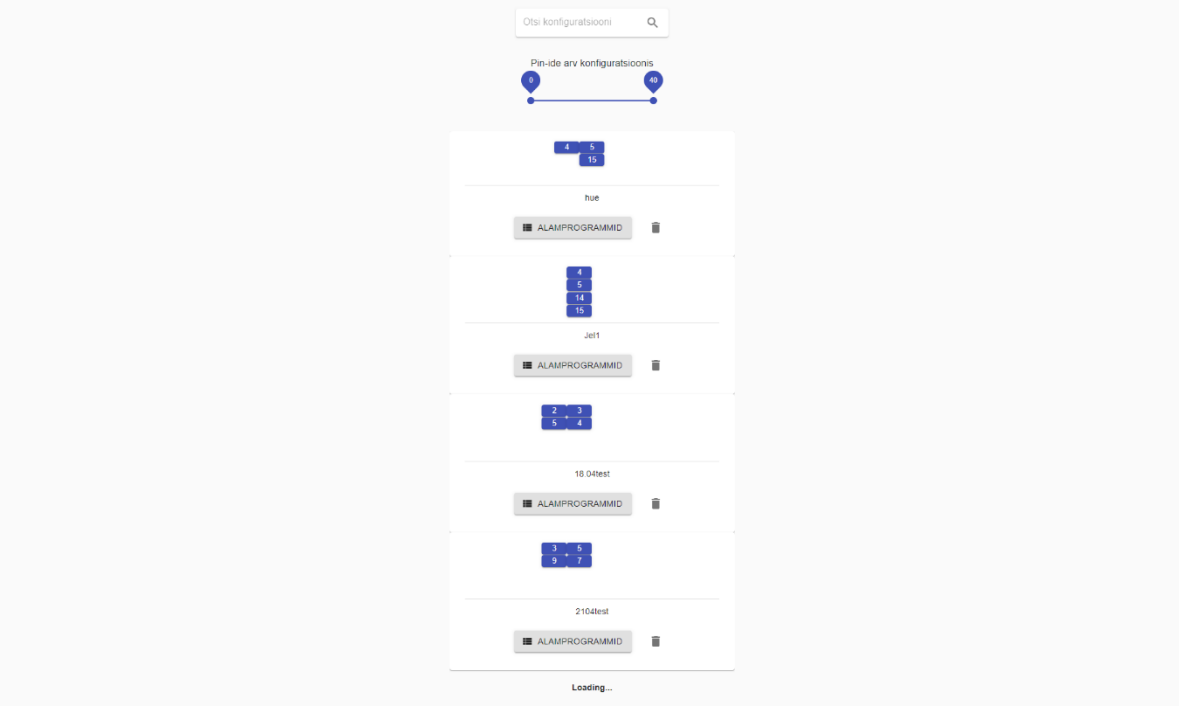


Joonis 16. Pumba opereerimise vaade.

4.1.8. Plaadi konfiguratsioonide loendivaade

Plaadi konfiguratsioonide loendi vaates kuvatakse kasutajale varem konfigureeritud plaadi kujundid. Konfiguratsioonid kuvatakse kasutajale lehtedena lõpmatu kerimise läbi. Iga kuvatud konfiguratsiooni kaardi peal on kustutamise nupp ja nupp, mille peale vajutades navigeeritakse kasutaja alamprogrammide vaatesse, kus ta näeb vastava plaadi

konfiguratsiooniga seotud alamprogramme. Loendit saab filtreerida, kasutades otsinguriba või määrates liuguri peal pinide vahemiku. Näiteks kui kasutaja paneb miinimumiks 3 ja maksimumiks 6, siis kuvatakse talle ainult sellised plaadi konfiguratsioonid, mille pin-ide arv jääb sinna vahemikku. (Joonis 17)



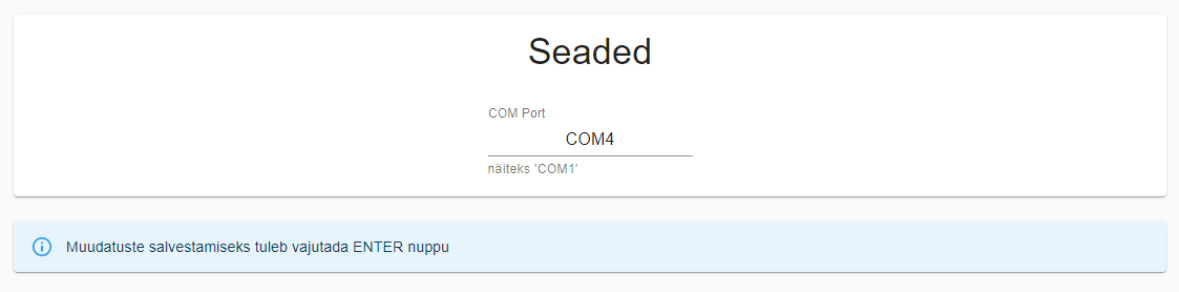
Joonis 17. Plaadi konfiguratsioonide loendivaade.

4.1.9. Seadete vaade

Seadete vaates on kasutajal võimalik seadistada kõige olulisemaid seadeid. Hetkel on seal ainult üks väli, mida seadistada saab, kuid vaade teostati eraldi vaatena, eeldades, et sellele rakendusele tehakse tulevikus lisaarendust ning konfigureerimisvajadus võib suureneda. Kasutajal on võimalik praeguse seisuga seadete vaates määrata, mis kommunikatsioonipordil Arduino ühendus asub. See port antakse päringuparameetrina kaasa igale Arduino suunas tehtavale päringule. (Joonis 18)

Seadeid hoitakse JSON kujul lokaalselt iga kasutaja kettal Appdata/Local kaustas. Selleks on kasutusel meil Electron-json-storage-nimeline teek. Iga kord kui rakendus käima pannakse, kontrollitakse selle JSON konfiguratsioonifaili olemasolu ning kui see puudub ehk

tegemist on esmakordse käivitusega või on konfiguratsioonifail kustutatud kasutaja poolt, luuakse see konfiguratsioonifail vaikesätetega.



Joonis 18. Seadete vaade.

4.2. Arduino juhtimine

Elektroodplaadil vedelikutilga liigutamiseks on elektroode vaja pingestada. Selle jaoks kasutatakse Arduino Mega plaadil olevat *shield*-tüüpi moodulit.

Elektroodide pingestamiseks saadetakse Arduinole läbi serial pordi sõnumeid. Sõnumid saadakse Pythonis implementeeritud Flaski otspunktidest, kus need sobivaks töödeldakse. Alljärgnevalt on kirjeldatud, kuidas implementeeriti elektroodide pingestamine, Arduinole käskude saatmine ning Flaskiga andmete vastuvõtmine.

4.2.1. Arduino

Elektroodplaatide pingestamiseks kasutati Microchip HV507 kiipi, mille abil on võimalik elektroodi pingestada kuni pingeni 300 volti. Mooduli külge on ühendatud 13 juhet, mida on võimalik ühendada elektroodplaadi külge.

Elektroodide pingestamiseks on kasutatud varasemalt kirjutatud Arduino koodi, mis võtab vastu infot Pythonis implementeeritud koodist läbi serial ühenduse. Arduino koodi on projekti jooksul täiendatud, et võimaldada servomootorit juhtides tilga pumpamist.

4.2.2. Python serial

Rakendusest tulevate andmete sobivale kujule töötlemiseks, eksperimendi loogika haldamiseks ja eksperimendi ja pumpade nii automaatseks kui ka manuaalseks juhtimiseks on kasutusel Pythonis implementeeritud kood.

Eksperimendi käivitamise otspunktilt käsu vastuvõtmisel töödeldakse andmed Arduinole sobivaks sõneks. Seejärel luuakse ühendus läbi serial pordi ning saadetakse andmeid. Sellele lisaks on juurde arendatud võimekus juhtida eksperimenti ja pumpa manuaalselt, see tähendab, et eksperimenti või pumpa käivitades on see võimalik poole töö pealt peatada. Selle jaoks on kasutatud paralleellõimesid (*thread*).

Eksperimendi juhtimiseks saadetakse käskude abil infot, et seadistada alljärgnevad parameetrid:

- Kõrgepinge

- Elektroodi number
- Elektroodi pinge
- Elektroodi sagedus

Pumpade juhtimiseks saadetakse infot, et seadistada parameetrid:

- Pumba pini number
- Pumba pinge

4.2.3. Flask

Käskude rakenduselt vastuvõtmiseks kasutati Flask raamistikku, mille abil implementeeriti Pythonis võimekus vastu võtta HTTP päringuid. Otspunktidest võib eristada eksperimendiga, pumpadega ning ühenduse kontrolliga seotud otspunkte.

4.2.4. Firmata

Projekti algusfaasis kasutati Arduino juhtimiseks PyFirmata teeki. See võimaldab Firmata protokolli kasutades lihtsasti juhtida Arduinot läbi teadlase arvuti. Elektroodplaadi juhtimismooduliga katsetades tuli välja mitu probleemi, sealhulgas ühenduse halb kiirus ning raskused HV507 kiibi juhtimisel. Kuna eksisteeris varasema projekti raames kirjutatud Arduino kood, kus oli juba implementeeritud eelmainitud võimendi juhtimine, siis otsustati PyFirmata teek kõrvale jätta ning teha kogu Arduino ja Pythoni vaheline suhtlus läbi serial ühenduse.

4.3. Java rakendusliides ja andmebaas

Java rakendus, mis suhtleb andmebaasiga, on klassikaline CRUD rakendus, mis kasutab Springi teeki. Domeeni objekte on 7 ning andmebaasi olemid ning nende seoseid aitab genereerida Hibernate. Üks peamiseid domeeniklasse on BoardConfiguration, mille küljes on väljad „id“ ja „description“. BoardConfigurationil on üks mitmele seos CircuitBoardPanel ja SubProgram domeeni objektidega. Järjend CircuitBoardPaneli objekte defineerib kasutajaliideses konfigureeritud kujundi. SubProgram domeeni objekti küljes on hulk kirjeldavaid väljasid, mis peaks kasutajale kirjeldama soovitud katse olukorda. SubProgram klassi küljes on ka üks mitmele seos ElectrodeConnection domeeni klassiga. ElectrodeConnection objekt defineerib ära seose kahe elektroodi vahel koos vajalike füüsikaliste parameetritega. Teine peamine domeeniklass kasutaja jaoks on Experiment. Experiment klassil on väljad „id“ ja „description“. Samuti on Experiment domeeni küljes mitu ühele seos BoardConfiguration klassiga, sest katse alguseks on olemas konfigureeritud kujund. Veel on Experiment domeeni klassi küljes üks mitmele seos ExperimentEntry klassiga. ExperimentEntry domeeni klassi küljes on väljad „id“, „name“, „experimentEntryType“ ja „entryId“. Viimased kaks viitavad SubProgram või Pump objektidele. Pump on eraldiseisev seosteta domeeni klass, mille küljes on väljad „id“, „name“ ja „pin“. Andmebaasi skeem on toodud Lisas 2.

4.4. Lähetamine

4.4.1. Tallinna Tehnikaülikooli server

Töö raames andis juhendaja meile ligipääsu Tallinna Tehnikaülikooli halduses olevale Linuxil jooksvale serverile. Antud serverisse oli juba varasematel õppeaastatel lähetatud kapillaarelektroforeesi analüüsi teostav veebirakendus. Meie ülesanne oli sinna varasemaid lähetusi katki tegemata oma rakenduse CI/CD tööle saada. Selle tööle saamiseks oli vaja rakendusse paigaldada NGINX veebiserver ning konfigureerida see tööle nii, et kõik meile eraldatud domeenile (digital-microfluidics.cs.taltech.ee) tehtud päringud suunatakse 7070 pordile, millel meie Java server jookseb.

4.4.2. Pidev integratsioon ja pidev tarne (CI/CD)

Me kasutasime oma projekti lähetamisel serverisse Gitlab'i *Pipeline* tööriista, mis automatiseeris terve lähetamise protsessi. Ehk kohe kui master harusse ehk peaharusse mingi muudatus tekib, käivitub meil serveris skript, mis ehitab rakenduse Java osast uue Dockeri konteineri viimaste muudatustega ning käivitab samuti Dockeris jooksva lokaalse PostgreSQL serveri, kus kõiki kasutaja poolt rakenduses tehtud konfiguratsioone ning andmeid hoitakse.

4.4.3. Docker

Docker on tarkvara, mis võimaldab operatsioonisüsteemi tasemel virtualiseerimist. Docker koosneb konteineritest ehk Docker on kaudselt sarnane virtuaalmasinale, kuid Dockeris ei kasutata igas konteineris oma kernelit, nagu virtuaalmasinas, vaid kõik konteinerid kasutavad sama masina, mille peal nad jooksevad, operatsioonisüsteemi kernelit. See võimaldab korraga jooksutada mitmeid konteinerid, ilma et kasutaja märkaks olulist jõudluse kadu, nagu mitmete virtuaalmasinate jooksutamisel. Dockeri eeliseks on võimekus käivitada sama konteinerit näiteks nii Windows'i kui ka Linux'i masinas, sealhulgas ei ole vaja muretseda operatsioonisüsteemi-põhiste sõltuvuste pärast. Samuti on Dockeri konteineri käivitumine peaaegu kohene ning konteiner võtab tunduvalt vähem ruumi kui virtuaalmasinat kasutades [21] [22].

Dockeri konteiner ehitatakse tõmmisfailist, kus täpsustatakse erinevad sätted, mis failid Dockeri konteinerisse ehitamisel kopeeritakse, mis sõltuvused on selle toimimiseks vajalikud, mis portidel konteineriga suhtlus toimima hakkab ning ka konteineri sisenemispunkt, ehk mis käsku konteineri käivitamisel jooksutatakse. (Lisa 3)

5. Valideerimine

5.1. Tellija testimine ja tagasiside

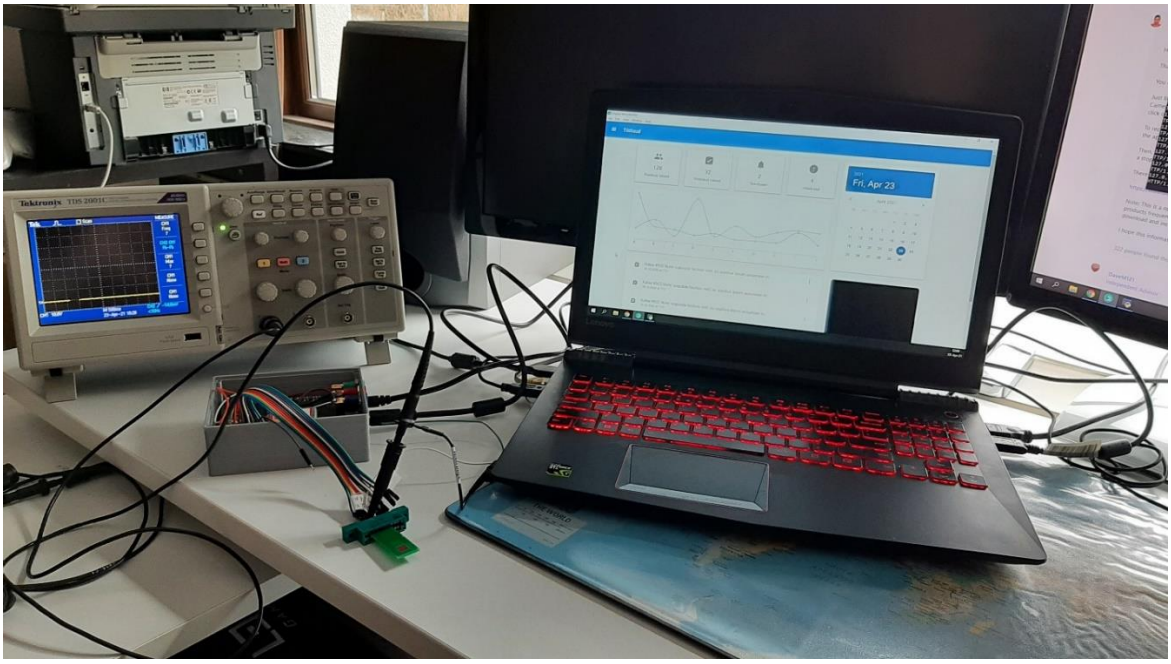
Vajalike nõuete täitmise ja sujuva kasutajakogemuse valideerimiseks sai projekti tellija võimaluse rakendust käivitada oma seadmel. Tellija installeeris rakenduse uusima versiooni, seejärel sai töölauarakendust käivitades hakata konfigureerima elektroodplaate, alamprogramme, eksperimente ja pumpasid. Lisaks sai läbi teha kogu töövoo alates konfigureerimisest kuni eksperimendi käivitamiseni. Semestri viimastel nädalatel sai tellijale tagastatud Arduino moodul, tänu millele sai katsetada ka tilga liikumist elektroodplaadil.

Tänu tellija testimisele sai meeskond tagasisidet iganädalastel koosolekutel, kus lepiti kokku soovitud muudatustes ja täiendustes.

5.2. Riistvaraline testimine

Kogu rakenduse testimiseks oli üles seadistatud kasutaja arvutisse ühendatud Arduino shield-tüüpi moodul, mille väljundite külge oli pandud nelja elektroodiga elektroodplaat. Nii Arduino juhtimise arendamisel kui ka rakenduse tööprotsessi testimisel kasutati multimeetrit ja ostsilloskoopi, millega mõõdeti elektroodplaadi elektroodilt pinget. (Joonis 19)

Ostsilloskoop on mõõtevahend elektriliste signaalide mõõtmiseks ajas [23]. Antud projektis võimaldas see valideerida, kas rakenduses käivitatud eksperimendis määratud elektroodi pinge on tegelikkuses ka see, nagu rakenduses määratud. Seadme kasutamiseks maandati masin Arduino maanduspini külge, seejärel mõõdeti elektroodplaadile ühendamiseks mõeldud juhtmete otstest ostsilloskoobiga pinge.



Joonis 19. Riistvaralise testimise seadistus töölauarakenduse, Arduino mooduli ja ostsilloskoobiga

5.3. Protseduuri lindistus

Rakenduse kasutamise tööprotsessist on valminud ka videolindistus[2], kus valminud rakenduses käivitatakse eksperiment, milles määratakse pinile 3 pinge 50 V kuueks sekundiks ning seejärel määratakse samad parameetrid pinile 5. Ostsilloskoop on ühendatud 5. pini külge ning videolt on näha, et peale rakenduses eksperimendi käivitamist kulub kuus sekundit, kuni elektrood pingestatakse. Elektrood on pingestatud kuus sekundit, peale mida läheb pinge nulli ning rakenduses on ka näha, et eksperiment on lõpetatud.

5.4. Ühik- ja integratsioonitestimine

Java rakendusliidese iga hoidla-teenuse arhitektuurikomponent on kaetud ühiktestidega, milleks on kasutatud JUnit 5 ja Mockitot. Kontrollerkiht on kaetud ka integratsioonitestidega, mida teostati Springi sisseehitatud tööriistadega.

[2] https://www.youtube.com/watch?v=OvvYwDAIf3A

Töölauarakenduse peavaated on kaetud ühiktestidega, kus kontrollitakse komponentide olemasolu ja põhifunktsionaalsuse toimimist virtuaalses DOMis (dokumendi objektimudel). Kasutusel olevateks tehnoloogiateks on React Testing Library ja Jest.

6. Tulemused

Projekti algfaasis osutus peamiseks raskuspunktiks olemasolevast rakendusest ja elektroodplaatide manipuleerimisest aru saamine ning Arduino elektroonikamooduli integreerimine meie arendatavasse rakendusse. Varasema projekti raames arendatud rakenduse autoriga konsulteerides sai kasulikke soovitusi, kuidas plaadiga kõige mõistlikumalt suhtlus üles ehitada ning mida rakenduses tehniliselt teha.

Lisaks tekkisid raskused ülesande testimisega. Kuna kogu süsteem koosneb mitmest eri komponendist ning elektroonikamoodulite eksemplare eksisteeris projekti arenduse hetkel vaid üks, oli vaja tellijaga testimine kooskõlastada, saada riistvara kasutamiseks juhiseid ning üles seadistada kogu kasutatav tark- ja riistvara ja mõõteseadmed. Lisaks pidi valideerima ka Arduino mooduli seadistatud väljunditelt tulevaid pingeid ja sagedusi, et need vastaksid rakenduses määratud väärtustele.

Kokkuvõtteks sai arendatud rakendus, mis muutis võrdlemisi keerulise katse läbiviimise protsessi oluliselt kiiremaks ja mugavamaks. Tellija on oma teadlaste rühmaga rakenduse kasutajavooge testinud ning neile see meeldis. Protsessi käigus meeskond õppis ja sai teadusvaldkonna kohta pidevalt uusi teadmisi. Tagantjärgi vaadates oleks sama tööd uuesti tehes tunduvalt kiiremini valmis saanud, kuna suur osa projektile kulunud ajast läks teemast aru saamisele ja töö kaardistamisele.

7. Kommentaarid

7.1. Serverisse lähetamisega tekkinud probleemid

Esimene serveri lähetamisega tekkinud takistus oli see, et eelnevatel aastatel arendatud veebirakenduse API juba jooksis antud serveris 8080 pordil. Seetõttu pidime oma rakenduse konfiguratsiooni muutma.

Teiseks takistuseks osutus NGINX veebiserveri mitte kasutamine. See tähendab, et eelneval õppeaastal lähetatud lihtne veebileht, kus sai kapillaarelektroforeesi andmetöötluse töölauarakendust alla laadida ning liikuda edasi selle veebirakendusse, oli pandud jooksma Node.js serverina, mis blokeeris 80 porti ja ei lasknud normaalselt NGINX veebiserverit sinna paralleelselt jooksma panna. Seetõttu tuli ka see lähetamise loogika natuke ümber teha, et kõik ilusti toimima jääks.

7.2. Probleemi püstitus

Projekti alguses oli keeruline aru saada, mida tellija meilt täpselt ootab. Üheks selle põhjuseks oli projekti spetsiifilisus, nimelt tuli kõigepealt arusaada, mis asi on digitaalmikrofluidika. Kui see sai selgeks, tuli saada aru, kuidas toimib tellija praegune töövoog, et saaks seda muuta paremaks.

Teiseks põjuseks oli projekti suurus. Kui tellija esitas esialgsed projekti ootused, siis oli raske mõista, et kas need on üldse projekti määratud mahu skoobis. Kõik need probleemid tegid projekti plaani paikapaneku keeruliseks.

7.3. Probleemi mitmetahulisus

Ülesande üheks raskuspunktiks kujunes probleemi mitmetahulisus. Kuna antud ülesande probleem puudutas nii tarkvaralist kui ka riistvaralist poolt, tuli lahenduse leidmiseks omandada põhjalikke teadmisi lisaks tarkvarale ka elektroonika valdkonnas, seda eriti Arduino mooduli toimimise ning võimekuse kohta. Õnneks oli tiimis Arduinoga töötamisega varasem kokkupuude olemas ning sellest oli palju kasu.

7.4. Suhtlus Arduino mooduliga

Lõplik suhtlusmeetod Arduino mooduliga sai kindlaks määratud alles siis, kui kogu moodul sai projektitiimi kätte füüsiliseks riistvaraliseks testimiseks. Selle jooksul selgus tõsiasi, et tunduvalt mõistlikum on kasutada varasemalt implementeeritud Arduino koodi, kui et hakata projekti raames sama tööd uuesti tegema.

Varasemalt on implementeeritud Arduino kood, kustkaudu juhitakse võimendit ja seekaudu suudetakse elektroode kõrgema pingega pingestada. Kuigi esialgse töövoo testimiseks ja näitlikustamiseks sobis PyFirmata hästi, siis plaadi kõrgepingemooduli juhtimiseks oleks olnud ümber vaja teha kogu eelnevalt tehtud HV507 võimendit juhtiv kood.

Neil põhjustel sai lõpuks otsustatud, et Pythoni ja Arduino vaheline suhtlus teostatakse läbi serial pordi, saates sõne kujul käske Arduino moodulile. Arduinol olev kood jäi enamjaolt samasuguseks, sinna arendati pumpade juhtimise võimekus veel lisaks.

8. Edasised sammud

Meie poolt arendadud töölauarakendus pole veel kindlasti täielikult valmis ning ideaalne. Puudu on ka osa funktsionaalsusest, mida tellija algselt enne arendustööd soovis, kuid ei mahtunud ajaliste kitsenduste tõttu skoopi. Samuti on aspekte, mida hetkel olemasolevas lahenduses oleks võinud paremini teha.

8.1. Erinevad kasutajaprofiilid teadlastele

Hetkeseisuga jagavad kõik rakendust kasutavad teadlased omavahel rakendusesiseselt loodud konfiguratsioone. See tähendab seda, et kui ühes arvutis tehakse mingi konfiguratsioon valmis, siis on see kõigile, kes töölauarakendust kasutavad, automaatselt kättesaadav. Samuti on neil ka võimekus kõiki varem tehtud ja andmebaasi salvestatud konfiguratsioone ning eksperimente redigeerida ning kustutada. See ei olnud tellijale esialgu suureks probleemiks ja otsustasime selle nii jätta. Tulevikus on kindlasti võimalik juurde arendada kasutajate autentimine ning rakenduse kasutajale ainult tema kontoga tehtud konfiguratsioonide ning eksperimentide kuvamine.

8.2. Kontaktnurga arvutamise funktsionaalsus

Tellija soovis, et me integreeriks kontaktnurga arvutamise programmi tehtavasse rakendusse. Klient kasutas veetilga kontaktnurga arvutamiseks ImageJ [1] rakendust kontaktnurga mooduliga. Esmalt analüüsides tundus, et seda rakendust on võimalik käivitada ka koodiga. Hakates seda lahendust teostama, tuli välja, et seda siiski ei ole võimalik niimoodi teha, kuna meie rakendus kasutab uuemat Javat ja sellega ei olnud võimalik kontaktnurga arvutamist käivitada. ImageJ saaks kuidagi kindlasti integreerida, aga tol hetkel jäi ta meie plaanist välja.

8.3. Kapillaarelektroforeesi analüüsi töölauarakenduse integreerimine

Tellija üks esialgsetest soovidest oli ka eelneval õppeaastal arendatud kapillaarelektroforeesi töölauarakenduse integreerimine meie rakendusse. Kuna tegemist oli JavaFX raamistiku abil kirjutatud töölauarakendusega, siis selle integreerimine meie rakendusse lihtne ei ole. Me

peaksime kogu rakenduse loogika ning selle omapärad selgeks tegema ning need nullist Javascripti maailma ümber kirjutama. Küll aga on see ideaalne ülesanne, mida järgmine aasta näiteks „Tarkvaraarenduse meeskonnaprojekti“ aine raames tudengid teha võiksid.

Kasutatud kirjandus

[1] „ImageJ,“ [Võrgumaterjal]. Available: https://imagej.nih.gov/ij/. [Kasutatud 15 mai 2021].

[2] „Digital Microfluidics,“ [Võrgumaterjal]. Available: https://www.annualreviews.org/doi/full/10.1146/annurev-anchem-062011-143028#_i2. [Kasutatud 14 mai 2021].

[3] R. Bondarev ja M. Ivanova, „Kapillaarelektroforeesi andmetöötluse töölaua- ja veebirakendus,“ Tallinna Tehnikaülikool, Tallinn, 2020.

[4] „Wikipedia,“ [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Flask_(web_framework). [Kasutatud 14 Mai 2021].

[5] „Wikipedia,“ [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Arduino. [Kasutatud 14 Mai 2021].

[6] „Arduino,“ [Võrgumaterjal]. Available: https://www.arduino.cc/en/reference/firmata. [Kasutatud 14 Mai 2021].

[7] „PySerial,“ [Võrgumaterjal]. Available: https://pyserial.readthedocs.io/en/latest/pyserial.html. [Kasutatud 14 Mai 2021].

[8] „Spring Framework,“ [Võrgumaterjal]. Available: https://spring.io/projects/spring-framework. [Kasutatud 11 mai 2021].

[9] „Gradle,“ [Võrgumaterjal]. Available: https://docs.gradle.org/current/userguide/what_is_gradle.html. [Kasutatud 11 mai 2021].

[10] Electron, „Electron Documentation,“ [Võrgumaterjal]. Available: https://www.electronjs.org/docs/tutorial/support#supported-platforms. [Kasutatud 13 mai 2021].

[11] „Rakenduse Paigaldamine,“ 30 aprill 2021. [Võrgumaterjal]. Available: https://gitlab.cs.ttu.ee/water-analyzer/digital-microfluidics/-/wikis/Dokumentatsioon/Rakenduse-Paigaldamine. [Kasutatud 13 mai 2021].

[12] „Electron,“ [Võrgumaterjal]. Available: https://www.electronjs.org/. [Kasutatud 13 mai 2021].

[13] „Chromium,“ [Võrgumaterjal]. Available: https://www.chromium.org/Home. [Kasutatud 13 mai 2021].

[14] E. Roth, „The 10 Best Chromium Browser Alternatives Better Than Chrome,“ MakeUseOf, 9 juuni 2020. [Võrgumaterjal]. Available: https://www.makeuseof.com/tag/alternative-chromium-browsers/. [Kasutatud 13 mai 2021].

[15] „Quick Start Guide,“ Electron, [Võrgumaterjal]. Available: https://www.electronjs.org/docs/tutorial/quick-start. [Kasutatud 13 mai 2021].

[16] „React,“ [Võrgumaterjal]. Available: https://reactjs.org/. [Kasutatud 13 mai 2021].

[17] „Material-UI,“ [Võrgumaterjal]. Available: https://material-ui.com/. [Kasutatud 2021 mai 13].

[18] „Socket.IO,“ [Võrgumaterjal]. Available: https://socket.io/docs/v4.

[19] L. Mölder, „Digitaalse mikrofluidika platvormi arendamine,“ Tallinna Tehnikaülikool, Tallinn, 2017.

[20] „Arduino,“ [Võrgumaterjal]. Available: https://www.arduino.cc/en/Main/arduinoBoardMega/. [Kasutatud 14 Mai 2021].

[21] „What is a Container?,“ Docker, [Võrgumaterjal]. Available: https://www.docker.com/resources/what-container. [Kasutatud 14 mai 2021].

[22] „Docker,“ [Võrgumaterjal]. Available: https://www.docker.com/. [Kasutatud 14 mai 2021].

[23] „Wikipedia,“ [Võrgumaterjal]. Available: https://en.wikipedia.org/wiki/Oscilloscope. [Kasutatud 14 Mai 2021].

[24] „Electrophoresis,“ [Võrgumaterjal]. Available: https://www.genome.gov/genetics-glossary/Electrophoresis. [Kasutatud 15 mai 2021].

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[3]

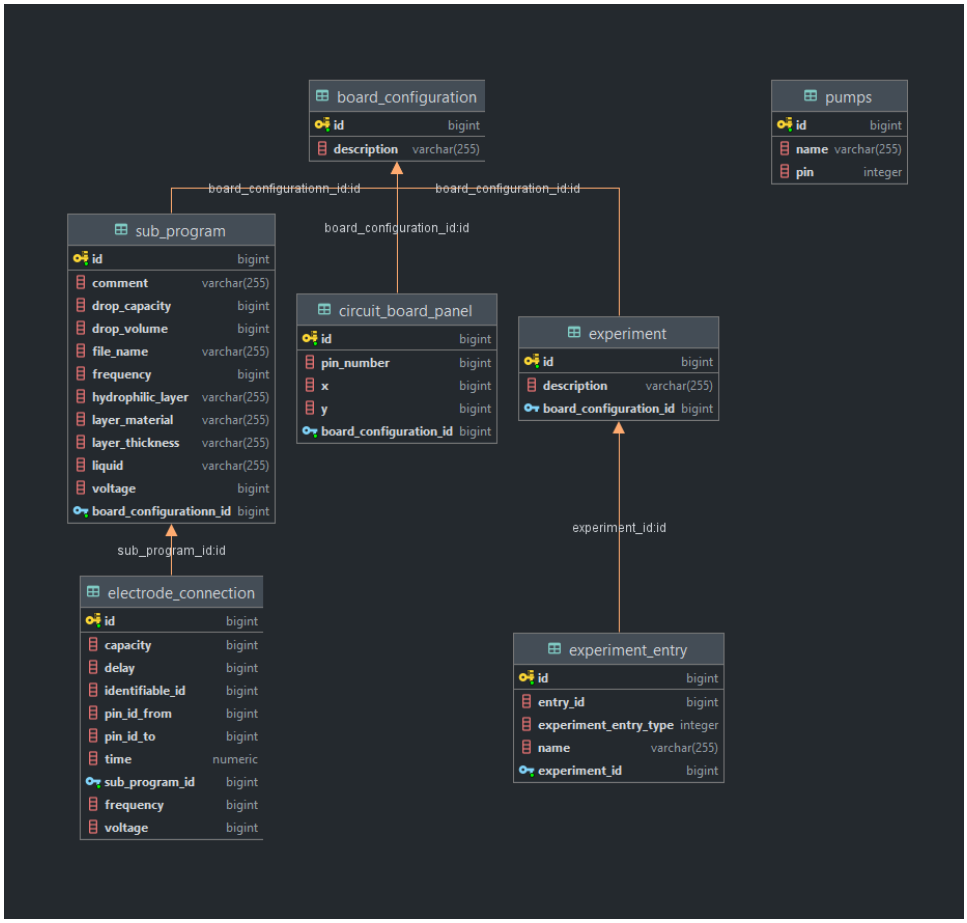
Meie, Raul Altmäe, Robert Altmäe, Karl-Joosep Karp, Rihard Liiva,

1. Anname Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Digitaalse mikrofluidika töölauarakendus", mille juhendajadeks on Evelin Halling ja Jelena Gorbatšova
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Oleme teadlikud, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autoritele.
3. Kinnitame, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

18.05.2021

3 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 - Andmebaasi struktuur



Joonis 20. Andmebaasi struktuur

Lisa 3 – Kasutatud Dockeri tõmmisfaili kood Java rakenduse konteineri ehitamiseks

```
FROM openjdk:11
COPY build/libs/*.jar /app.jar
EXPOSE 7070 8080
ENTRYPOINT ["java","-jar","/app.jar"]
```