TALLINNA TEHNIKAÜLIKOOL
INSENERITEADUSKOND
Virumaa kolledž

Ida-Virumaa Keskhaigla laboratooriumite andurite andmete visualiseerimine ja analüüs R abil

Visualization and analysis of Ida-Virumaa central hospital laboratory sensor data using R

TELEMAATIKA JA ARUKATE SÜSTEEMIDE ÕPPEKAVA LÕPUTÖÖ

Üliõpilane: Sofja Botsarova
Üliõpilaskood: 183584 EDTR
Juhendaja: Natalja Maksimova, lektor

Kohtla-Järve 2022

AUTORIDEKLARATSIOON

Olen koostanud lõputöö iseseisvalt.

Lõputöö alusel ei ole varem kutse- või teaduskraadi või inseneridiplomit taotletud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

"...." ..... 20.....

Autor: .....

/ allkiri /

Töö vastab rakenduskõrgharidusõppe lõputööle/magistritööle esitatud nõuetele

"...." ..... 20.....

Juhendaja: .....

/ allkiri /

Kaitsmisele lubatud

"...." ..... 20.....

Kaitsmiskomisjoni esimees .....

/ nimi ja allkiri /

LIHTLITSENTS LÕPUTÖÖ ÜLDSUSELE KÄTTESAADAVAKS TEGEMISEKS JA REPRODUTSEERIMISEKS

Mina Sofja Botsarova (sünnikuupäev: 07.10.1999)

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose „Ida-Virumaa Keskhaigla laboratooriumite andurite andmete visualiseerimine, analüüs ja prognoosimine R abil ”, mille juhendaja on Natalja Maksimova,

1.1. reprodutseerimiseks säilitamise ja elektroonilise avaldamise eesmärgil, sealhulgas 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. Olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. Kinnitan, et lihtlitsentsi andmisega ei rikuta kolmandate isikute intellektuaalomandi ega isikuandmete kaitse seadusest ja teistest õigusaktidest tulenevaid õigusi.

TalTech Inseneriteaduskond Virumaa kolledž

LÕPUTÖÖ ÜLESANNE

Üliõpilane: Sofja Botšarova, 183584EDJR

Õppekava, peaeriala: EDTR17/18 – Telemaatika ja arukad süsteemid, Telemaatika tarkvara

Juhendaja(d): Natalja Maksimova, lektor, natalja.maksimova@taltech.ee

Konsultant: Sergei Mihhailov, laboriteenistuse kvaliteedijuht, Ida-Virumaa Keskhaigla, Sergei.Mihhailov@ivkh.ee

Lõputöö teema:

(eesti keeles) Ida-Virumaa Keskhaigla laboratooriumite andurite andmete visualiseerimine ja analüüs R abil

(inglise keeles) Visualization and analysis of Ida-Virumaa central hospital laboratory sensor data using R

Lõputöö põhieesmärk:

Shiny rakenduse loomine kondensatsiooni tekkimiste jälgimise jaoks Ida-Virumaa Keskhaigla laboratooriumides.

Lõputöö etapid ja ajakava:

Nr	Ülesande kirjeldus	Tähtaeg
1.	Aegridade ja nende analüüsi meetodite uurimine	10.10.21
2.	R Shiny paketi õppematerjalide läbikäimine ja programmi uurimine	01.11.21
3.	"Kastepunkt" kondenseerumiseks mõiste uurimine	07.11.21
4.	Teha temperatuuri ja niiskuse andmete visualiseerimine ja analüüs	17.11.21
5.	Teha kondensaati tekkimise tingimuste analüüs	24.11.21
6.	Tõestada kondensatsiooni juhtude olemasolu varem kogutud andmetel ja arvutada juhtude arvu.	27.11.21
7.	Programmeerida teiste vigade leidmist, visualiseerida vigu graafiliselt	11.12.21
8.	Luua rakendust temperaturi, niiskuse ja vigade seireks	20.12.21

9.	Lõputöö vormistamine	28.12.21

Töö keel: eesti keel **Lõputöö esitamise tähtaeg:** "11" jaanuari 2022a

Üliõpilane: Sofja Botsarova "....."..... 2021a
/allkiri/

Juhendaja: Natalja Maksimova "....."..... 2021a
/allkiri/

Konsultant: Sergei Mihhailov "....."..... 2021a
/allkiri/

Programmijuht: Žanna Gratšjova "....."..... 2021a
/allkiri/

SISUKORD

EESSÕNA

Käesoleva lõputöö teema on valitud Ida-Virumaa Keskhaigla ja kolledži pakutud ülesannete hulgast. Seda teemat valida ajendas see, et Virumaa kolledž arendab aktiivselt andmeanalüüsi ja masinõpe ning tegeleb selle suuna rakendamisega erinevates valdkondades.

Lõputöö üheks tulemuseks oli rakenduse loomine, mis analüüsib ja visualiseerib laboritesse paigaldatud anduritelt saadud andmetest tulenevaid vigu: temperatuuri või niiskuse tõus ja langus alla lubatud piiri, andmete kadu, kondenseerumine ja võimatud andmed. Lõputöö koostamisel kasutati tarkvaraarenduskeskkonda RStudio ning Shiny teeki visualiseerimiseks ja rakenduse loomiseks. Statistilise andmetöötluse ja graafika puhul kasutati programmeerimiskeelt R.

Võtmesõnad: R programmeerimiskeel, aegrida, andurid, visualiseerimine, RShiny rakendus, lõputöö.

SISSEJUHATUS

Käesoleval ajal paljude praktiliste probleemide lahendamisel on oluline roll andmete töötlemise ja analüüsi protsessil, et saada vajalikku teavet ning tuvastada saadud andmekogumitest mustreid. Andmete analüüs on tihedalt seotud masinõppega. See on protsess, mille käigus uuritakse üksikasjalikult suurandmeid ja saadakse neist sisulist infot, kasutades algoritme ennustamaks teadaolevatel andmetel põhinevaid tundmatuid andmeid. Näiteid masinõppe kasutamisest igapäevaelus on palju. Selleks on näiteks veebilehtedel ja rakendustes erinevate reklaamide soovitused, mis valitakse selle järgi, milliseid veebilehti külastate ja milliseid päringuid otsingusse tipite.

Käesolev lõputöö on seotud temperatuuri- ja niiskusanduritelt saadud andmete analüüsi ja visualiseerimisega. Andmete visualiseerimine on nende analüüsimisel oluline komponent, kuna suure infohulga läbivaatamisel on võimalus eksida ning see on ka ajamahukas.

Töö põhieesmärgiks on RShiny rakenduse loomine kondensatsiooni tekkimise jälgimiseks Ida-Virumaa Keskhaigla laboratooriumides. Kondensaadi tekkimine kujutab endast suurt ohtu, kuna selle tulemusel võivad näiteks testitulemused olla ekslikud ja põhjustada seadmete rikkeid.

Andmete analüüs ja visualiseerimine toimub R keskkonnas. Selle analüüsi eesmärgiks on vigade leidmine ja jälgimissüsteemi loomine. Haiglas on temperatuuri- ja niiskusandurid, mille andmed kantakse vastavasse andmebaasi. Esimene viga võib olla tingitud mikrokliima rikkumisest. Laborites on väga oluline jälgida temperatuuri ja niiskuse näitajaid, need peavad vastama standarditele, et töötajatel oleks neis ruumides mugavad töötingimused. Ruumi mikrokliima rikkumine võib negatiivselt mõjutada töötajate tervist. Teiseks veaks on andmekadu, kus näiteks vigastatud juhtmed võivad põhjustada andmekatkestusi. Ja kolmas viga on seotud kastepunkti temperatuuriga. Kondensatsiooni tekkimine on laboriruumides väga tõsine olukord, kuna liigniiskus võib mõjutada katsete/proovide tulemusi. Andmeanalüüsi käigus oli tuvastatud ka neljas viga: absoluutselt võimatud andmed (näiteks suhteline niiskus rohkem kui 3000%).

1. Lõputöö eesmärgi saavutamiseks püstitati järgmised ülesanded:
2. Koostada temperatuuri ja niiskuse andmete visualiseerimine ja analüüs
3. Vigade fikseerimine
4. Koostada kondensaadi tekkimise tingimuste analüüs

5. Tõestada kondensatsiooni juhtude mõju varem kogutud andmete puhul ja arvutada juhtude arv.
6. Tuvastada saadud andmete hulgast nelja tüüpi vigu
7. Koostada seiresüsteem temperatuuri ja niiskuse jälgimiseks.

Lõputöö on jagatud 4 peatükiks: Probleemi uurimine ja vahendite valik, Andmed, Vigade tuvastamine R-keeles ja Shiny rakenduse loomine.

1. PROBLEEMI UURIMINE JA VAHENDITE VALIK

Laboratoorium on ruum, kus viiakse läbi erinevaid uuringud ja katsed. Nendes ruumides on väga oluline mikrokliima jälgimine vältimaks instrumentide kahjustamist ja tagamaks uuringute puhtuseks vajalikud soodsad tingimused. Oluline on hea ventilatsiooni tagamine töökohal paigaldades selleks kasvõi pidevalt töötav konditsioneer, kunas katkestused kliimaseadme töös põhjustaksid niiskuse kondenseerumist. Niiskus ja kõrge temperatuur soodustavad omakorda sageli seente kasvu, mis võivad hävitada optilisi pindu [1]. Õhuniiskuse kaheks peamiseks näitajaks on niiskusesisaldus ja suhteline õhuniiskus. Niiskusesisaldus on veeauru massi ja kuiva õhu massi suhe ning suhteliseks niiskuseks nimetatakse suurust, mis võrdub õhus tegelikult oleva ning antud temperatuuril ja rõhul õhus maksimaalselt võimaliku ehk küllastava veeauru tiheduse suhtega. Ruumi suhteline õhuniiskus sõltub peamiselt välisõhu parameetritest (välisniiskus) ja ventilatsiooniõhu vahetusest ruumis. Madal suhteline õhuniiskus võib kuivatada inimese limaskesti ja põhjustada ärritust. Liiga kõrge suhteline õhuniiskus omakorda soodustab mikroobide, bakterite ja hallituse kasvu [2]. Mida kõrgem on suhteline õhuniiskus ruumis ühel temperatuuril, seda kõrgem temperatuur õhust kondenseerub. Mikrokliima muutumist võivad mõjutada paljud tegurid, näiteks avatud külmkapp või töötajate pikaajaline viibimine laboris. Mikrokliima jälgimiseks Ida-Virumaa Keskhaigla laborites on paigaldatud temperatuuri- ja niiskusandurid. Andurite andmed laekuvad SQL-i andmebaasi iga 5/15 minuti järel. Ruumide mikrokliima muutuste tuvastamiseks on tarvis programmi, mis analüüsib saadud andmeid ja teavitab töötajaid reaalajas võimaliku probleemi tekkimisest.

Ida-Virumaa Keskhaiglas tuvastati kolme tüüpi vigu:

1. Temperatuuri või niiskuse tõus ja langus alla lubatud piiri

2. Andmete kadu

3. Kastepunkti temperatuur (olukorrad, kui õhuniiskus on järsult tõusnud ja temperatuur on alanenud, mis võib põhjustada kondensaadi tekkimise).

Käesoleva lõputöö käigus tuvastati vea neljas tüüp, kolme avastamine aga oli seatud ülesandeks; viimane - neljas viga oli tuvastatud andmete analüüsi ja visualiseerimise käigus, ning selle tulemusena loodi rakendus kastepunkti esinemise detailseks jälgimiseks.

1.1 Kastepunkt ja kondensatsiooni juhud

Kastepunkt on temperatuur, mille juures õhus olev veeaur kondenseerub kasteks. Seda mõjutavad kaks näitajat – õhutemperatuur ja suhteline õhuniiskus. Suhtelise õhuniiskuse suurenemisega tõuseb kastepunkti näitaja, lähenedes temperatuuri näitajale. Kastepunkti kiireks arvutamiseks kasutatakse tabelit (vt Joonis 1.1). [3]

Tabel näitab, et kui õhutemperatuur on 20 kraadi ja suhteline õhuniiskus 40%, siis kastepunkti temperatuur on umbes +6.00 kraadi; kui suhteline õhuniiskus on sama temperatuuri puhul 50% või 60%, siis on kastepunkti temperatuur 9.3 ja vastavalt 12 kraadi. Kastepunkt ei saa ületada õhutemperatuuri. 100% niiskuse korral on kastepunkt võrdne õhutemperatuuriga. Kastepunkt on minimaalne temperatuur, mille juures hakkab tekkima kondensaat. Kondensaat võib esineda tilkade või udu või härmatise kujul. [4]

°C	Suhteline õhuniiskus						
	30%	35%	40%	45%	50%	55%	60%
+18	0,2	2,3	4,2	5,9	7,4	8,8	10,1
+19	1,0	3,2	5,1	6,8	8,3	9,8	11,1
+20	1,9	4,1	6,0	7,7	9,3	10,7	12,0
+21	2,8	5,0	6,9	8,6	10,2	11,6	12,9
+22	3,6	5,9	7,8	9,5	11,1	12,5	13,9
+23	4,5	6,7	8,7	10,4	12,0	13,5	14,8
+24	5,4	7,6	9,6	11,3	12,9	14,4	15,8
+25	6,2	8,5	10,5	12,2	13,9	15,3	16,7

Joonis 1.1 Kastepunkti arvutamise tabel

Kasutades valemit on võimalik arvutada kastepunkti täpset väärtust.

Kondensaadi temperatuuri on võimalik arvutada järgmise valemi (1.1) abil [4]:

$$T_p = \frac{b*f(T,Rh)}{a-f(T,Rh)} \quad (1.1)$$

$$f(T,Rh) = \frac{a*T}{b+T} + In(\frac{Rh}{100})$$

Kus,

- Tp – kastepunkti temperatuur, °C;
- a (konstantne) = 17,27;
- в (konstantne) = 237,7;
- T – õhutemperatuur, °C;
- Rh – suhteline õhuniiskus, %;
- ln – naturaallogaritm

Selle valemi viga on vahemikus ± 0,4 ° C:

0 °C < T < 60 °C;

0,01 < Rh < 1,00;

0 °C < Tp < 50 °C;

Kondensaadi temperatuuri saab määrata ainult siis, kui on teada õhu temperatuur ja suhteline niiskus. Lõputöö koostamisel on rakendati vaid neid kahte näitajat. Seetõttu on töö visualiseerimine ja analüüs koostatud nende kahe mõiste kaudu koos kastepunktiga. Lisaks sellele oleks mõistlik viia läbi eksperiment, kus katseliselt fikseerida kondenseerumise tingimused ja jälgida neid graafikutelt. Nende katsetulemuste põhjal oleks võimalik kirjeldada täiendavat strateegiat kondenseerumisjuhtumite arvu väljaarvutamiseks. Hetkel kasutatakse järgmist strateegiat: kui on juhtumeid, kus õhutemperatuur langeb järsult kastepunkti temperatuurini ja õhuniiskus hakkab tõusma, siis vaatleja võib eeldada, et see võib põhjustada kondensaadi tekkimist.

Kastepunkt kirjeldab täies ulatuses õhu niiskusesisaldust. Kastepunkt näitab, millise temperatuuri puhul vastab õhuniiskus 100% niiskusele ja millisel temperatuuril algab kondenseerumine. Kondensaadiprotsess seisneb selles, et niiskuse maksimaalne hulk, mida õhk võib gaasi kujul sisaldada, sõltub peamiselt selle temperatuurist. See tähendab, et mida madalam on õhutemperatuur, seda vähem võib õhk sisaldada gaasilisel kujul olevat vett ja vastupidi. Põhilised kondensaadi tekitajateks on kõrge õhuniiskus ja halb ventilatsioon. See tähendab, et õhutemperatuuri langemisel kastepunkti väärtusteni ja alla selle tekib kondensaat. Kastepunktini jõudmist iseloomustab õhu suhtelise niiskuse tõus kuni 100%.

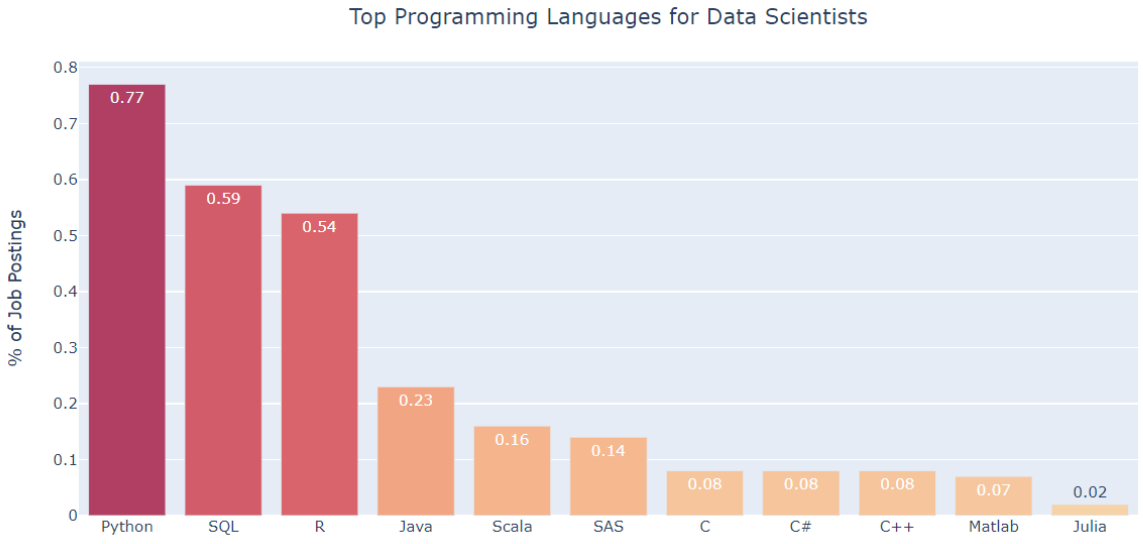
1.2 Miks R?

R on programmeerimiskeel statistiliseks andmetöötluseks ja graafikaga töötamiseks. See keel hõlmab sellist tüüpi andmete visualiseerimist, nagu näiteks aegridade graafikud ja diagrammide vahemikud. Nimetatud keele peamiseks eeliseks on see, et R on loodud spetsiaalselt andmete analüüsiks, mistõttu on kõik süntaksikonstruktsioonid üsna mahukad ja arusaadavad. Keele loomisest alates on R-i kasutatud peamiselt akadeemilistes ja teaduskeskkondades. Kuid ajapikku hakkas keel suurtes ettevõtetes populaarsust koguma, näiteks:

- Facebook - R-i kasutatakse andmete uurimuslikuks analüüsiks, eksperimentaalseks analüüsiks, suurandmete visualiseerimiseks, personalihalduseks ning olekuvärskenduste ja profiilipiltidega seotud kasutajate käitumise analüüsiks.
- Google – kasutatakse reklaami efektiivsuse tõstmiseks, majandusprognoosiks ja suurandmete statistiliseks modelleerimiseks.
- Twitter – kasutatakse andmete visualiseerimiseks ja semantiliseks klasterdamiseks. Twitter´is loodi klienditeeninduse parendamiseks ka avatud lähtekoodiga anomaaliate ja rikkumiste tuvastamise pakette.

- Microsoft - soetati Revolution R ning seda kasutatakse erinevatel eesmärkidel. Microsoft Azure kasutab R-i mõistmaks kasutajate käitumist, visualiseerida infrastruktuuri kasutusandmete visualiseerimiseks ja ebatavalisi sisselogimiste tuvastamiseks.
- Uber – kasutatakse staatilise analüüsi jaoks.
- Airbnb - R-i kasutatakse andmeteaduse skaleerimiseks. Samuti ka varasemate külaliste hinnangute põhjal ümberbroneerimishindade ennustamiseks.
- IBM - investeeritakse andmeteaduse tarvis mõeldud R programmeerimiskeelde ning samaaegselt on R konsortsiumi liige. Kasutati R-i IBM Watsoni loomiseks. See on kognitiivsete andmetöötlusete avatud tehnoloogiaplatvorm, mis esindab uut ajastut andmetöötluses, kus süsteemid „mõistavad" välismaailma inimestega analoogselt: tunnete, õppimise ja kogemuse kaudu.
- ANZ (Australia and New Zealand Banking Group) – kasutatakse krediidiriskide analüüsimiseks. [5]

R oli 2021. aastal üks nõutumaid programmeerimiskeeli andmeteaduse valdkonnas. Selle valdkonna populaarsemate keelte edetabelis on Python, SQL ja R programmeerimiskeelte esikolmikus (vt Joonis 1.2). [6]



Joonis 1.2 Programmeerimiskeeled andmeteadlastele

R-i peamised eelised:

- kvaliteetsed paketid asjakohaste graafikute ja visuaalide loomiseks;
- suur hulk statistiliste meetodite raamatukogusid;
- mugav arenduskeskkond - RStudio;

- võimalus programmeerida matemaatiliste meetodite rakendamist vektor- ja maatriksarvutuste abil. [7]

1.3 Keskkond ja paketid

RStudio on integreeritud arenduskeskkond R programmeerimiskeele jaoks. Selline keskkond sisaldab konsooli; koodi esiletõstmise võimalus, mis tõstab märksõnad ja muutujad esile erinevates värvides [8]. Samuti on tööriistad diagrammide, ajaloo ja tööruumi haldamiseks. RStudio põhieesmärk on luua tasuta avatud lähtekoodiga tarkvara andmetöötluseks, teadusuuringuteks ja tehniliseks kommunikatsiooniks.

R-i on võimalik kasutada pea kõikide andmeanalüüside jaoks, kuna selles programmeerimiskeeles on palju asjakohaseid pakette. Lõputöös kasutati peamiselt 11 paketti:

- Ggplot2, dygraphs, plotly – paketid andmete visualiseerimiseks, diagrammide koostamiseks.
- Dplyr – andmetega manipuleerimise grammatika rakendamine. Mugav tööriist objektidega töötamiseks.
- Slider – üldeesmärgiga „libiseva akna" funktsiooni perekond, erinevate „libistamisoperatsioonide" tarvis.
- Lubridate – lihtsustab tööd kuupäeva ja kellaajaga.
- Stringr – lihtsustab tööd ridadega.
- Zoo, xts – aegridadega tööks.
- Openxlsx – lihtsustab Exceli failide loomist.
- Shiny – interaktiivse visualiseerimisega veebirakenduste loomiseks.

Shiny on pakett, mis on loodud R-il põhinevate interaktiivsete veebirakenduste loomiseks [9]. Shiny ühendab R-i arvutusvõimsuse kaasaegse Interneti interaktiivsusega. Shiny rakendus koosneb kolmest komponendist: kasutajaliidesega skript, serveri skript ja ShinyApp. Kasutajaliidese (*user interface, UI*) skript asub lähtefailis *ui.R*, sisaldab kasutajaliidese elementide määratlusi ja juhib rakenduse välimust. Teine kohustuslik fail *server.R* sisaldab skripti, mis määrab, kuidas server töötab. Kolmas komponent, ShinyApp, kutsub rakenduse koostamiseks välja kasutajaliidese ja serveri funktsioonid. Hästi üles ehitatud ja hästi silutud rakendust on võimalik teiste kasutajatega jagada kolmel viisil:

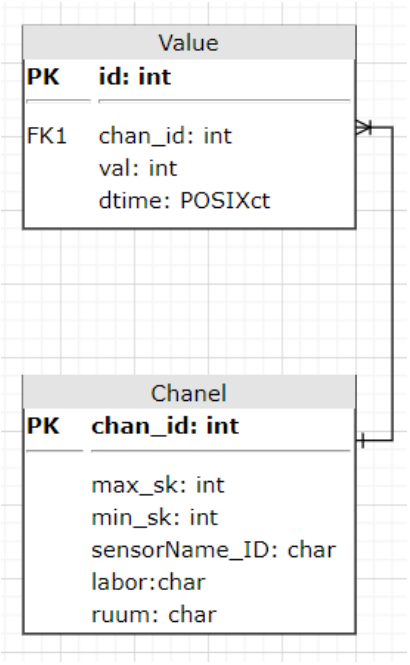
- koodi avaldamine (gist),
- jagada ZIP-kataloogi rakendusfailidega,

- veebi põhjal, kui registreerida shinyapps.io serveris. Interaktiivse Shiny Dashboard paigaldamise ja vaatamise jaoks
- konvertida eraldi R-paketiks.

Loodud rakendus avaldati lingi (https://github.com/SofjaBotcharova/app) kaudu.

2. ANDMED

Ida-Virumaa Keskhaiglas on olemas oma temperatuuri jälgimissüsteem. Andurite mõõdikute säilitamiseks kasutatakse MS SQL andmebaasi, millest oli käesoleva uurimuse läbiviimiseks andmete väike osa aasta ja poolaasta eest esitatud csv formaadis. Töö eesmärgi ja püstitatud ülesannete saavutamiseks kasutati DHT22 andurite andmeid, kuna need mõõdavad temperatuuri ja niiskuse väärtusi. Andmed laekuvad neljast laboratooriumist: kaks laboratooriumi (D3030, D3011) asuvad vereteenistuse korpuses ja kaks laboratooriumi (G2021B, G2021C) asuvad hematoloogia korpuses. Iga laboratooriumi kohta on koostatud kaks tabelit – üks tabel temperatuuri ja teine tabel niiskuse andmetega. Veel on koostatud üks tabel andurite kirjeldustega. Skeemil (vt Joonis 2.1) on kujutatud seos tabelite vahel. Tabel *Value* asendab skeemil andmetega tabeleid, veerud on samad.



Joonis 2.1 Olem seos diagramm

Temperatuuri ja niiskuse andmetega tabelites on järgmised andmed:

- id – mõõtmise tunnusnumber,
- chan.id - andmete mõõtmiseks kasutatava anduri tunnusnumber,
- val – saadud temperatuuri või niiskuse väärtus,
- dtime – andmete vastuvõtmise aeg.

Andurite kerjeldustega tabelis on:

- chan.id - andmete mõõtmiseks kasutatava anduri tunnusnumber,
- max_sk – maksimaalne väärtus,

- min_sk – minimaalne väärtus,
- sensorName_ID – anduri täisnimetus,
- labor – korpuse nimetus,
- ruum – labori nimetus.

Id ja *chan.id* muutujate tüübid on *integer*, mida kasutatakse täisarvude esitamiseks. *Val* näitaja tüüp on *numeric*, mis sisaldab nii täisarve kui ka murdarve. Ning aja muutujaks kasutatakse klassi *POSIXct*, mis tähistab aega sekundites, kui palju on möödunud 1. jaanuarist 1970.

Kuna vaadeldud väärtused tekkisid konkreetse aja möödudes, siis neid on nimetatud aegreaks. Selles peatükis kirjeldatakse RStudio's kasutatud pakette, aegrea teooriat ja saadud mõõtmiste analüüsi.

2.1 Töö aegridadega RStudio´s

Põhipaketid ja funktsioonid RStudio-s aegridadega töötamiseks: *zoo*, *xts*, *ts*, *tsibble*. *ts* saab käsitleda ainult rangelt regulaarseid aegridu ning ei aktsepteeri rohkem kui kahest märgist koosnevat kuupäevavektorit. Seetõttu oli *zoo* pakett mõeldud regulaarsete ja ebaregulaarsete aegridade ning erinevate kuupäevavormingute käsitlemiseks. *Xts* pakett on *zoo* paketi laiendus, mis töötab erinevate kuupäevavormingutega ja salvestab mugavalt maatriksvormingus aegridu. *Xts*-, *zoo*- ja *ts*- paketid ei sobi hästi ajamuutuja ebaregulaarse salvestamisega andmekogumitele, mitme erinevat tüüpi muutujaga andmekogumitele või mitme rühmitusmuutujaga andmekogumitele. Selliste andmetega on lihtsam töötada kasutades paketti *tsibble*.

Klassi *tsibble* objektide loomiseks kasutatakse funktsioon *as_tsibble()*, millel on järgmised argumendid:

- x — objekt andmetega;
- index — ajatemplitega muutuja;
- key — üks või mitu rühmitus- või võtmemuutujat, mis määratlevad kordumatult iga tabelisse salvestatud aegrea;
- regular — loogikaargument, mis näitab tabelisse salvestatud juhtumite loendamise regulaarsust;
- validate — loogikaargument, mis võimaldab kontrollida iga vaatluse unikaalsust kombineerides *index* ja *key* muutujate väärtusi;
- .drop — loogikaargument, mis võimaldab tabelist välja jätta "tühjad" aegread.

Tasub mainimist, et kõige levinum meetod aegridadega töötamiseks R-is on *xts* pakett. Seda paketti kasutades on võimalik ka aegridade visualiseerimine.

Interaktiivsete graafikute loomiseks kasutatakse paketti *dygraphs*. *Dygraphs* on avatud lähtekoodiga JavaScript-i teek interaktiivsete aegridade diagrammide loomiseks. Skaleeritavus võimaldab soovi korral pilti suurendada või vähendada. Sihtides kursorit graafikul mõnele punktile, näeb kasutaja selle täpseid koordinaate ja saab teada vastavat väärtust. *Dygraphs* on võimalik kasutada R-konsoolis, R Markdowni dokumentides ja Shiny rakendustes.

Dygraphs paketi peamised eelised on:

- Oskab töötada tohutute andmemahtudega;
- kokkusobivus enamiku brauseritega;
- ehitab automaatselt *xts* aegridade objekte;
- suur hulk interaktiivseid funktsioone;
- sobib hästi R Markdowni dokumentidele ja Shiny veebirakendusele lisamiseks.

2.2 Aegrida

Aegrida on erinevatel ajahetkedel saadud statistiliste andmete rida. Üksiktunnust fikseeritakse paljukordselt teatud ajavahemike järel, mis on sageli võrdse pikkusega (aasta, tund, päev). Iga aegrea väärtust nimetatakse aegrea tasemeks [10]. Peamine tunnus, mis eristab aegrida tavapärasest andmevalimisest, on iga taseme jaoks määratud mõõtmisaeg või mõõtmise järjekorranumber. Aegrida saab jaotada järgmisteks komponentideks:

- Trend – iseloomustab andmete pikaajalist trendi (langus või tõus).
- Sesoonne muutus - lühiajalised perioodilised muutused fikseeritud sagedusega ja tsükliline komponent - pikaajalised tsüklilised võnked (vähemalt 2 aastat), ei oma fikseeritud sagedust.
- Ebaregulaarne komponent – juhuslikud tegurid.

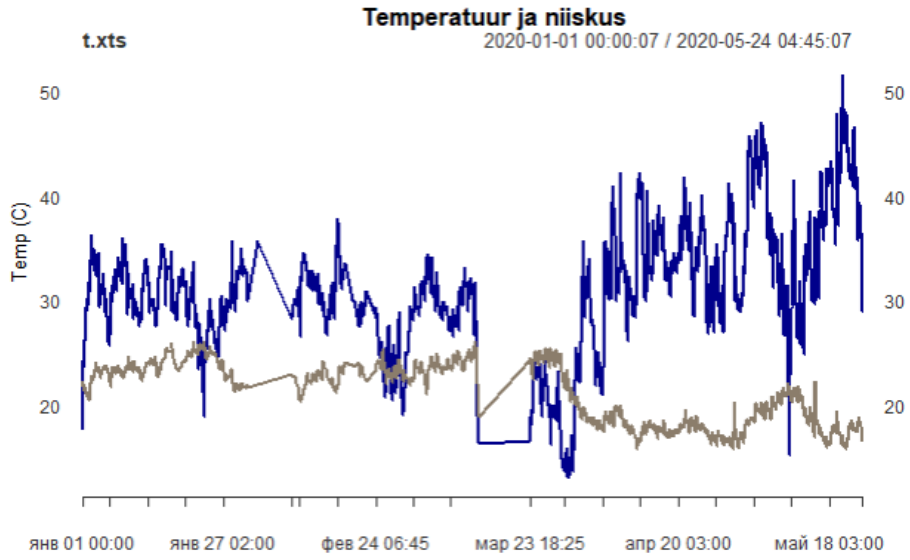
Aegread tekivad mõne näitaja mõõtmise tulemusena. Kui on ainult üks vaadeldav muutuja, nimetatakse aegrida ühemõõtmeliseks. Kui samaaegselt vaadeldakse mitut muutujat, on tegemist mitmemõõtmelise aegreaga.

2.3 Aegridade analüüs

Aegridade analüüsimeetodid erinevad arvuliste andmete analüüsimeetoditest. Aegridade analüüsimisel on olulised mitte ainult aegridade statistilised karakteristikud, vaid tuleb arvestada ka mõõtmiste seost ajaga. Aegridade analüüsi põhieesmärk on

aru saada, milliste komponentide mõjul aegrida väärtus kujuneb ning koostada selle väärtuste prognoos tulevikuks.

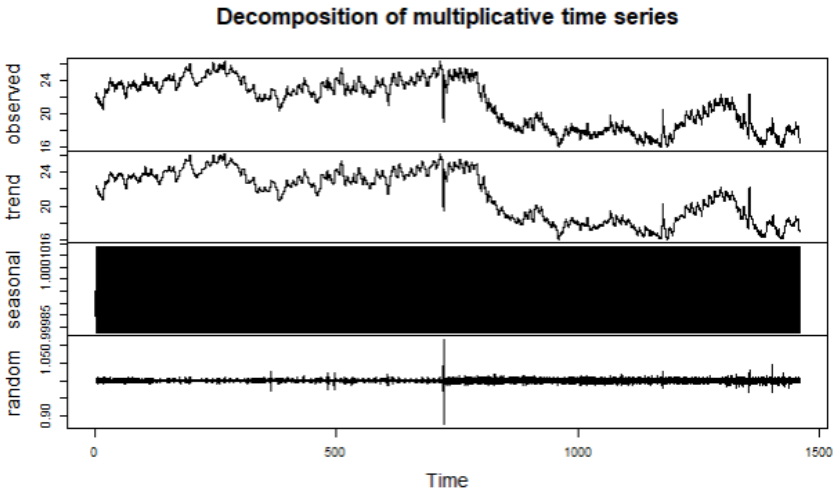
Joonisel 2.2 on esitatud ühe labori (D3030) temperatuuri ja niiskuse graafikud.



Joonis 2.2 Aegrida visualisatsioon (D3030)

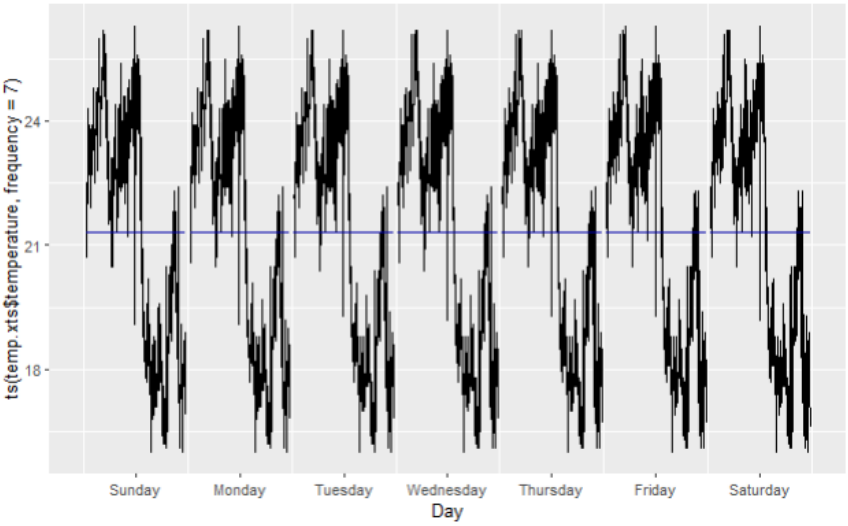
Graafikutele tuginedes on võimalik eeldada, et sellel aegreal puudub ilmselge sesoonsus ja trend. See seletub asjaoluga, et laboris tagatakse kindlad temperatuur ja niiskus. Seda on R-s võimalk kontrollida nelja komponendi abil: (vt Joonis 2.3)

- Observed - tegelik andmegraafik
- Trend - on sesoonsusest ja juhuslikest mõjudest puhastatud aegrida, mis näitab keskmist või pikaajalist arengusuunda ja mida on võimalik jälgida eri ajalõikudel.
- Seasonal - sesoonne komponent kujutab endast kõikumisi, mida on võimalik jälgida aasta jooksul (kuude või kvartalite kaupa) ja mis korduvad aastast aastasse. Sesoonsust võivad põhjustada looduslikud faktorid, haldustingimused, sotsiaalsed ja kultuuritraditsioonid.
- Random - seletamatu osa andmetest, juhuslik komponent ehk ebaregulaarne või müra. Kõik, mis jäi pärast trendi ja sesoonsuse lahutamist.



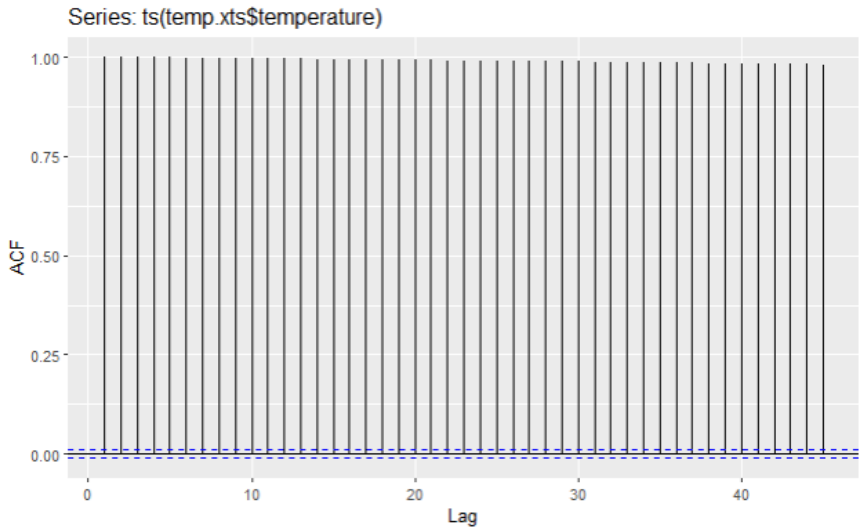
Joonis 2.3 Multiplikatiivne muudel (D3030)

Seda diagrammi uurides (vt Joonis 2.3) on täheldada, et aegridadel pole ootuspäraselt trendi. Andmete juhuslikus komponendis on ainult ühes kohas järsk hüpe, kuid seda võis põhjustada näiteks kütte väljalülitamine. Kuna pilt sesoonsust pole selge, on võimalus seda visualiseerida teiste graafikute abil. Joonisel 2.4 on esitatud rühmitatud mingi kindla sagedusega andmed, kus sinine joon tähendab selle perioodi osa keskmist. Kuna need keskmised on samad, siis sesoonsus puudub ka käesolevates andmetes.



Joonis 2.4 Sessoonsuse visualisatsioon (D3030)

On ka teine võimalus aegridade andmete jälgimiseks, näiteks võrreldes eelnevat vaatlust järgneva vaatlusega, mida nimetatakse autokorrelatsiooniks. Autokorrelatsioon on korrelatsioon ühe ja sama tunnuse erinevate väärtuste vahel, mis on järjestatud. Autokorrelatsiooni graafikut nimetatakse korrelogrammiks [11] (vt Joonis 2.5).



Joonis 2.5 Autoregression (D3030)

Kui autokorrelatsioon ületab sinist joont, mis võrdub ±2/√aegrea pikkusega, siis väärtus sõltub eelmisest väärtusest. Antud andmete puhul on mõõtmete vahel sõltuvusseos. Kui aegreas on trend, siis on autokorrelatsiooni funktsioonid positiivsed (> 0). Andmete sesoonset muutust on võimalik lugeda ka korrelogrammilt, autokorrelatsioonifunktsiooni väärtuse tõusuga kaasneb ka langus ja vastupidi. Esitatud korrelogrammide (vt Joonis 2.4 ja Joonis 2.5) järgi võib kindlalt väita, et sellel aegreal puudub trend ja sesoonsus. ACF (Autokorrelatsiooni-funktsioon) graafik on skalaarne, ilma kukkumiste ja tõusudeta. Sarnane aegridade analüüs oli tehtud teistest ruumidest laekunud andmete põhjal, mille tulemused on sarnased.

Saadud andmete põhjal tuvastati võimatult kõrged niiskuse väärtused laborites D3011 ja G2021B (vt Joonis 2.6) . Sel põhjusel oli tehtud otsus luua võimatute andmete tarvis neljandat tüüpi viga.

Tabel 2-1 Andmete analüüsi tabel

	D3011	D3030	G2021B	G2021C
kesk T (°C)	20.64	21.33	24.11	23.66
kesk H (%)	35.22	31.17	31.48	32.53
min T (°C)	5.80	16.0	9.80	15.20
min H (%)	10.60	13.20	11.90	7.70
max T (°C)	26.70	26.30	28.90	27.70
max H (%)	3294.60	51.70	3304	72.70
mõõtmise arv	16343	35000	35000	35000

3. VIGADE LEIDMINE R-KEELES

RStudio's oli kirjutatud R-keeles kood, et leida saadud andmete põhjal kolme tüüpi vigu. Andmed on neljast laborist, kus iga labori kohta on koostatud kaks eraldi tabelit, temperatuuri ja niiskuse näitude kogumise aja kohta. Haigla ruumides on fikseeritud lubatud vahemikud, mille puhul loetakse ruumide sisekliima töötamiseks mugavaks. Viimased vahemikud erinevad ruumi kasutamise tüübist. Esimese tüübi viga on temperatuuri ja niiskuse miinimum- ja maksimumväärtuste ületamine. See viga peaks sisse lülituma alles kolmanda näidu puhul, kuna lühiajalised muutused võivad tähendada, et näiteks külmkapp on lahti või et aknad on avatud. Teise tüübi vead tähendavad sidekatkestusi. Erinevate laborite andmed laekuvad alalise ajavahega 5 või 15 minutit. Rikke tõttu mõne aja osas andmed ei laeku, seetõttu on tarvis tuvastada ja arvutada tekkimiste arvu. Kolmanda tüübi viga tähendab kastepunkti määramist. Kastepunkti on tarvis tuvastamaks, kas kondensaat on tekkinud või mitte.

3.1 Esimese tüüpi vea arvutamine ja tulemus

Esimese tüübi vea koodid on esitatud joonisel 3.1.

```
filter_t <- function(x){
  all(x >= temperature_top_limit | x <= temperature_low_limit)
}
filter_h <- function(x){
  all(x >= humidity_top_limit | x <= humidity_low_limit)
}

data <- data %>%
  mutate(t_err_1 = slide_dbl(temperature, filter_t, .before = 2,.complete = T)) %>%
  mutate(h_err_1 = slide_dbl(humidity, filter_h, .before = 2,.complete = T))
summary(data)
```

Joonis 3.1 Esimese vea leidmine

Funktsioonides kontrollitakse tingimust, kas temperatuur või niiskus ületavad alguses seatud piirid ning tagastatakse loogilised väärtused *true* või *false*. Tellija poolt nõutud tingimuste kohaselt loetakse piiri ületamise veaks olukorda, kui seda juhtub vähemalt 2 korda. Selleks loodi funktsioon *slide_dbl* parameetriga *before* = 2, mis fikseerib viga kolmandal korral, kui see esines enne juba 2 korda. Funktsioon töötab kogu veeru aknaga, mille pikkus on 3 lahtrit, laskudes iga kord ühe võrra madalamale.

3.2 II tüüpi vea arvutamine ja tulemus

Koodi esimeses versioonis avastati teist tüüpi viga pärast kõigi andmete töötlemist. Kuna aga eesmärgiks on vigade tuvastamine reaalajas, rakendati teist meetodit (vt Joonis 3.2). Esimene võimalus oli tuvastada mõõtmiste algus- ja lõpuaeg, luua funktsioon, mis määrab, kui tihti andmebaasi kandeid tehakse, luua aegrida algusest lõpuni koos funktsiooni sammuga ning paigutada olemasolevad väärtused aegrea lähimates ajapunktides. Seetõttu on andmelünk nähtav kui NA. See meetod töötas

õigesti, kuid tsükli tõttu väga aeglaselt (1,4 – 1,6 minutit).

```
data_H_try=data.frame(time=data$humidity_time,value=data$humidity)
data_H_try$time=as.POSIXct(data_H_try$time)
new_data_H=data_H_try[1,]

for (i in 1:(nrow(data_H_try)-1)){
  T_timeEnd=data_H_try$time[i+1]
  T_timeStart=data_H_try$time[i]
  time_diff=difftime(T_timeEnd, T_timeStart, units='sec')
  n=floor(as.numeric(time_diff/sec_diff_time))
  if (n>=1){
    while (n>0){
      new_row=data.frame(time=data_H_try$time[i]+seconds(sec_diff_time),value=NA)
      new_data_H=rbind(new_data_H,new_row)
      n=n-1
    }
  } else {}
  new_data_H=rbind(new_data_H,data_H_try[i+1,])
}
head(new_data_H)
str(new_data_H)
```

1565	24.0	26.5	2020-01-17 07:00:05.859	2020-01-17 07:00:05.846
1566	*NA*	*NA*	2020-01-17 07:15:35.859	2020-01-17 07:15:35.846
1567	*NA*	*NA*	2020-01-17 07:15:35.859	2020-01-17 07:15:35.846
1568	24.0	27.0	2020-01-17 07:45:06.220	2020-01-17 07:45:06.207

Joonis 3.2 Teise vea leidmine 1

Joonisel 3.3 on esitatud teine võimalus, kus ühendati kaks tabelit - tabel, kus aeg läheb sammuga algusest lõpptulemuseni nullanduri näiduga ning tabel koos andurite näitudega ajas. Kuid selle meetodi puhul läks osa andmeid kaotsi.

```
time_start <- data$temperature_time[1]
time_end <-data$temperature_time[nrow(data)]
sec_diff_time <- round(data$temperature_time[3]-data$temperature_time[2])
time_seq <- seq(from=time_start,to=time_end,by=sec_diff_time)

df2 <- data.frame(t=rep(NA, length(time_seq)),time=time_seq)

t <- data.frame(value=data$temperature)
df1<- cbind(t,time=data$temperature_time)

# round time to min
df2$time <- round.POSIXt(df2$time,units = "mins")
df1$time <- round.POSIXt(df1$time,units = "mins")

#merge tables
df_t_h_with_NA <- merge(x = df1, y = df2, by = "time",all.y = TRUE)
```

1565	*2020-01-17 07:00:00*	24.0	*NA*
1566	*2020-01-17 07:15:00*	*NA*	*NA*
1567	*2020-01-17 07:30:00*	*NA*	*NA*
1568	*2020-01-17 07:45:00*	24.0	*NA*

Joonis 3.3 Teise vea leidmine 2

Ja kolmas võimalus (vt Joonis 3.4), mida valiti selle vea tuvastamiseks, oli puuduvate andmetega ridade ja nende arvu leidmine. Nende arv sai lisatud üldtabelisse. Peab märkima, et teist tüüpi viga väljastatakse, kui näit oli suurem kui null.

```r
data <- data %>%
  mutate(time_diff_t = difftime(time_t, lag(time_t),units = "sec")) %>%
  mutate(time_diff_h = difftime(time_h, lag(time_h),units = "sec")) %>%
  mutate(n_2nd_t=floor(as.numeric(time_diff_t/sec_diff_time))) %>%
  mutate(n_2nd_h=floor(as.numeric(time_diff_h/sec_diff_time))) %>%
  mutate_at(c("time_diff_t","n_2nd_t","time_diff_h","n_2nd_h"), funs(lead), n = 1 )
sum(data$n_2nd_t,na.rm = T) #Количество пропусков данных по температуре
sum(data$n_2nd_h,na.rm = T) #Количество пропусков данных по влажности
```

2020-01-17 07:00:05.859	2020-01-17 07:00:05.846	2700.360 secs	2700.360 secs	2	2
2020-01-17 07:45:06.220	2020-01-17 07:45:06.207	902.710 secs	902.693 secs	0	0

Joonis 3.4 Teise vea leidmine 3

Funktsiooni *Sys.time* abil arvutati, et lõplik versioon töötab kuni 10 korda kiiremini kui *for*-tsükkel ja 2 korda kiiremini kui tabelite ühendamine.

3.3 III tüüpi vea arvutamine ja tulemus

Kolmas viga on kastepunkti viga. Kastepunkt on temperatuur, mille puhul õhus olev veeaur kondenseerub kasteks. Kondensaat tekib pindadele, mille temperatuur on kastepunkti temperatuurist madalam või sellega võrdne. Kondensaat laboris kujutab endast suurt ohtu, kuna liigne niiskus võib analüüsi tulemusi negatiivselt mõjutada. Kastepunkti arvutamiseks valemi 1.1 järgi oli loodud funktsioon nimega ROSA (vt Joonis 3.5).

```r
ROSA <- function(t,
                 h,
                 a=17.27,
                 b=237.7,
                 c=100,
                 na.rm = FALSE)+
  {round((b*((a*t)/(b+t)+log(h/c)))/(a-((a*t)/(b+t)+log(h/c))),digits = 2)}

data <- data %>%
  mutate(t_rosa = ifelse((Fail_T == 0 & Fail_H == 0),
                         ROSA(t=temperature,h=humidity), NA))
```

Joonis 3.5 Kolmanda vea leidmine

Iga temperatuuri ja niiskuse paari jaoks arvutatakse oma kastepunkt, seejuures rakendatakse tingimust, et valeandmete puhul kastepunkti ei arvestata aga väljendata NA väärtust.

3.4 Kõikide tulemuste ühendamine

Andmeid analüüsides selgus, et mõnikord tekivad absoluutselt võimatud andmed - näiteks suhteline õhuniiskus 3000 protsenti. Need otsustati lisada neljanda veatüübina (vt Joonis 3.6).

```r
```{r}
data <- data %>%
  mutate(Fail_T = ifelse((temperature >= temperature_top_filter |
                            temperature <= temperature_low_filter), 1, 0)) %>%
  mutate(Fail_H = ifelse((humidity >= 100 |
                            humidity <= 0), 1, 0))
```

Joonis 3.6 Neljanda vea leidmine

Lõpuks oli loodud tabel koos kõikide vigadega (joonis 3.6).

1. veerg (temperature) tähistab temperatuuri väärtusi,

2. veerg (humidity) tähistab niiskuse väärtusi,

3. veerg (time_t) tähistab temperatuuri väärtuste saamise aega,

4. verg (time_h) tähistab niiskuse väärtuste saamise aega,

5. veerg (time_diff_t) tähistab ajavahet praeguse ja järgmise näidu vahel temperatuuri puhul,

6. veerg (time_diff_h) tähistab ajavahet praeguse ja järgmise näidu vahel niiskuse puhul,

7. veerg (n_2nd_t) tähistab teist tüüpi viga temperatuuri puhul,

8. veerg (n_2nd_h) tähistab teist tüüpi viga niiskuse puhul,

9. veerg (Fail_T) tähistab neljandat tüüpi viga temperatuuri puhul,

10. veerg (Fail_H) tähistab neljandat tüüpi viga niiskuse puhul,

11. veerg (t_rosa) tähistab kastepunkti,

12. veerg (t_err_1) tähistab esimest tüüpi viga temperatuuri puhul,

13. veerg (t_err_2) tähistab esimesest tüüpi viga niiskuse puhul,

14. veerg (all_error_for_temperature) tähistab vigade kirjeldusi temperatuuri puhul,

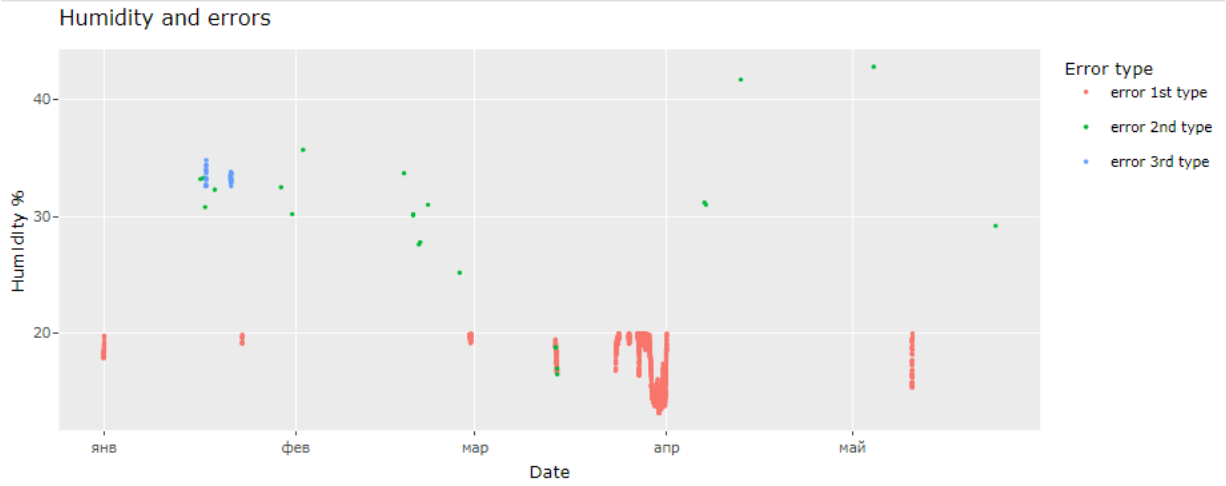
15. veerg (all_error_for_humidity) tähistab vigade kirjeldusi niiskuse puhul.

	temperature	humidity	time_t	time_h	time_diff_t	time_diff_h	n_2nd_t
1	23.4	14.4	2020-01-01 00:00:06	2020-01-01 00:00:06	899.259999990463	899.256000041962	0
2	23.5	14.3	2020-01-01 00:15:05	2020-01-01 00:15:05	900.145999908447	900.166999816895	0
3	23.5	14.3	2020-01-01 00:30:06	2020-01-01 00:30:06	900.09700012207	900.077000141144	0
4	23.5	14.5	2020-01-01 00:45:06	2020-01-01 00:45:06	900.19000005722	900.193000078201	0

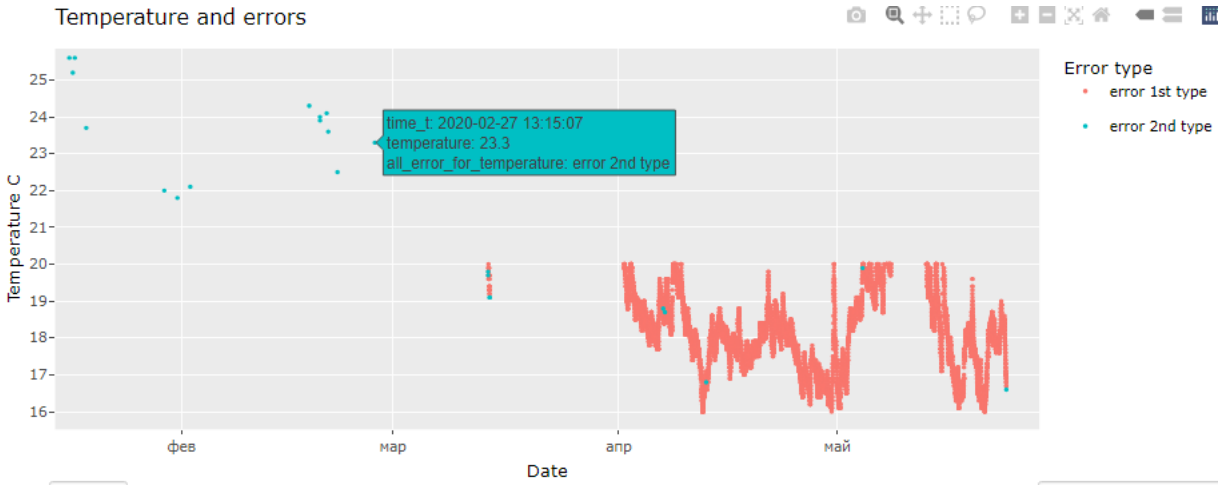
n_2nd_h	Fail_T	Fail_H	t_rosa	t_err_1	h_err_1	all_error_for_temperature	all_error_for_humidity
0	0	0	-5.25	*NA*	*NA*	clear data	clear data
0	0	0	-5.26	*NA*	*NA*	clear data	clear data
0	0	0	-5.26	0	1	clear data	error 1st type
0	0	0	-5.08	0	1	clear data	error 1st type

Joonis 3.6 Vigade tabel
```

Tabeli põhjal koostati kolme tüübi vigadega temperatuuri ja niiskuse kohta eraldi graafikud (vt Joonis 3.7 ja Joonis 3.8).



Joonis 3.7 Labori D3030 vigade visuaal niiskuse jaoks



Joonis 3.8 Labori D3030 vigade visuaal temperatuuri jaoks

Kolmas viga on toodud vaid niiskuse graafikul vältimaks kordusi ja saamaks vigade arvu korrektselt.

3.5 Vigade leidmise tulemus

Labor D3030:

D3030 (vereteenistuse korpus) ajaperiood on 2020-01-01 00:00:07.077 - 24.05.2020 4:45:07. Selle perioodi jooksul tehti 35000 salvestust.

Vead temperatuurianduri puhul: 1. tüübi viga on 37,6% ja 2. tüübi viga 0,07%.

Vead niiskuse anduri puhul: 1. tüübi viga on 4,6%, 2. tüüpi viga 0,07% ja 3.tüüpi viga puudub

Labor D3011:

D3011 (vereteenistus) ajaperiood on 01.01.2020 0:00:12 - 31.12.2020 23:45:14. Selle perioodi jooksul tehtud 16343 salvestust.

Vead tempeatuuri anduri puhul: 1. tüübi viga on 30% ja 2. tüübi viga 0,07%.

Vead niiskuse anduri puhul: 1. tüübi viga on 6,5%, 2. tüübi viga 0,07%, 3.tüübi viga puudub ja 4.tüübi viga on 0,22%.

Labor G2021C:

G2021C (hematoloogia) ajaperiood on 01.01.2020 0:00:06 - 30.12.2020 15:30:06. Selle perioodi jooksul tehtud 35000 salvestust.

Vead tempeatuurianduri puhul: 1. tüübi viga on 1,2% ja 2. tüübi viga 0,03%.

Vead niiskuse anduri puhul: 1. tüübi viga on 13%, 2. tüübi viga 0,02% ja 3.tüübi viga on 4,4%.

Labor G2021B:

G2021B (hematoloogia) ajaperiood on 01.01.2020 0:00:06 - 30.12.2020 15:30:08. Selle perioodi jooksul on 35000 salvestust.

Vead tempeatuurianduri puhul: 1. tüübi viga on 0,11% ja 2. tüübi viga 0,03%.

Vead niiskuse anduri puhul: 1. tüübi viga on 10%, 2. tüübi viga 0,02%, 3.tüübi viga on 3,6% ja 4.tüübi viga on 0,04%.

Tabelis 3.9 on toodud kõik vead ja keskmine vigade arv.

Tabel 3.1 Üldine vigade tabel

Vea tüüp	D3011	D3030	G2021B	G2021C	Keskmine
viga nr 1 - T (%)	30	37.6	0.11	1.2	15
viga nr 1 - H (%)	6.5	4.6	10	13	9
viga nr 2 - T (%)	0.07	0.07	0.03	0.03	0.04
viga nr 2 - H (%)	0.07	0.07	0.02	0.02	0.04
viga nr 3 (%)	0.14	0.12	3.6	4.4	2.4
viga nr 4 - H (%)	0.22	-	0.04	-	0.1

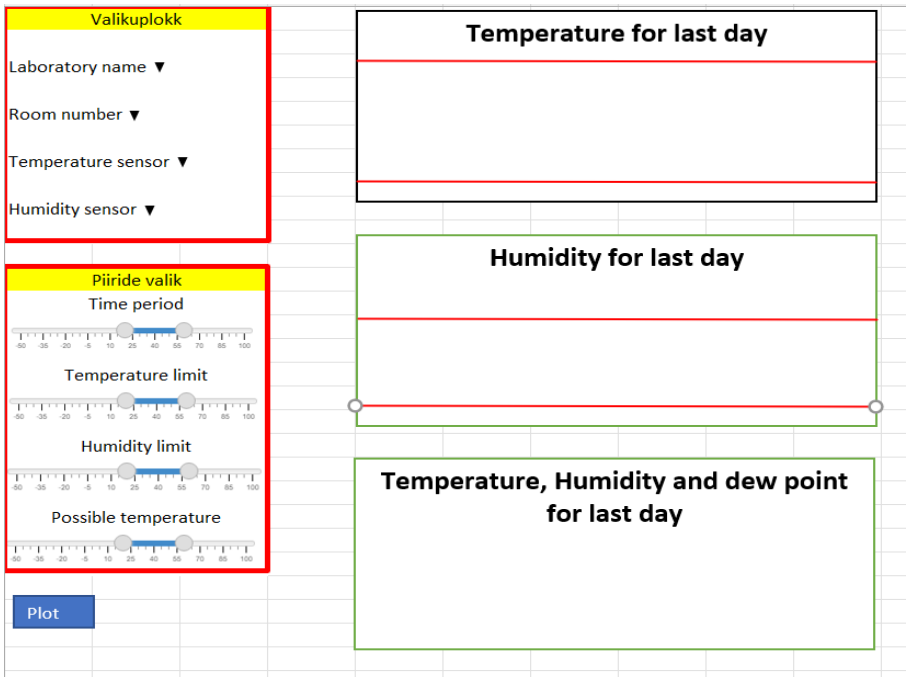
20-kraadise pinnatemperatuuri puhul kolmandat viga laborites D3030 ja D3011 ei esinenud. Tabeli põhjal on näha, et kõige sagedamini esines esimest tüüpi viga (lubatud väärtuste ületamine) ja kõige harvem tekkis teist tüüpi viga (side katkemine).

4. SHINY RAKENDUSE LOOMINE

Kirjutatud koodi põhjal loodi RStudios rakendus Shiny paketi abil. See rakendus on vajalik laborite temperatuuri- ja niiskusanduritelt saadud andmete jälgimiseks. Rakendus peaks täitma järgmisi funktsioone:

- Kõik sama tüübi graafikud peavad ekraanile mahtuma
- Kuva viimase päeva praegused väärtused
- Võimalus valida erinevad andurid
- Kuva igat tüüpi vead valitud perioodi jooksul
- Statistika igat tüüpi vigade arvu kohta kuvatakse valitud perioodi kohta

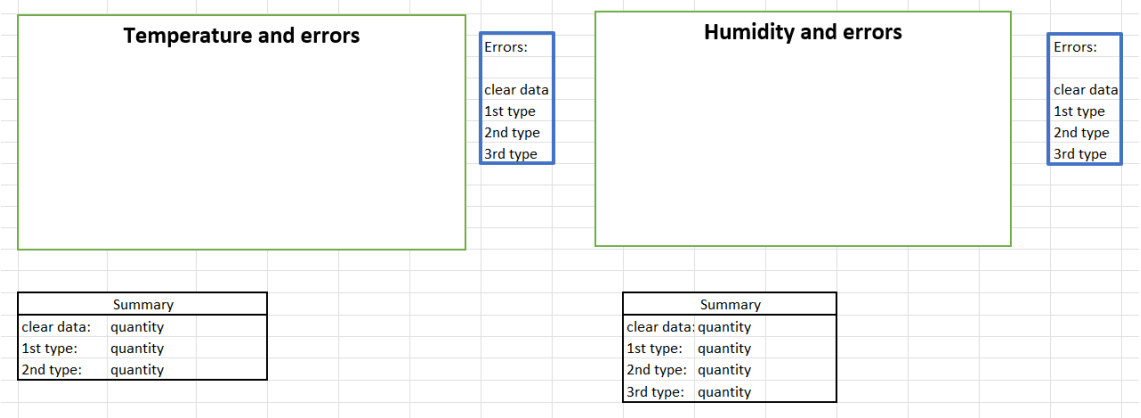
Rakendus koosneb kahest leheküljest. Esimesel leheküljel (vt Joonis 4.1) on labori ja anduri valik, piiride valik ja samuti 3 graafikut viimase päeva kohta: temperatuurigraafik, niiskuse graafik ja üldgraafik koos kastepunkti temperatuuriga.



Joonis 4.1 Rakenduse skeem I

Rakenduse teisel lehel (vt Joonis 4.2) on niiskuse ja temperatuuri graafikud igat tüüpi vigadega. Samuti on iga graafiku juurde lisatud vigade kokkuvõte, kuhu on kirja pandud vigade liigid ja nende arv.

Kõik rakenduses olevad diagrammid on interaktiivsed. Kui liigutada kursorit graafiku peale, sisi on võimalik näha muutuja kirjeldust, selle väärtust ja antud punkti salvestusaega.



Joonis 4.2 Rakenduse skeem II

4.1 Kasutajaliides

Esiteks loodi kasutajaliides (*user interface, UI*) - tarkvara ja riistvara kogum, mis tagab kasutaja suhtluse arvutiga [12]. Rakenduse nimeks oli valitud "Temperature and humidity monitoring system" (vt Joonis 4.3). Liidese elemendid hõivavad 4/12 lehe laiusest.

```
ui <- fluidPage(

   # Application title
   titlePanel("Temperature and humidity monitoring system"),
```

Joonis 4.3 Skript 1

Edasi lisati esimese lehekülje nimetus ning väljad labori tüübi, ruumi numbri, temperatuurianduri ja niiskusanduri valikuks (vt Joonis 4.4).

```
tabsetPanel(
  tabPanel("Monitoring system setups",
           fluidRow(
             column(4,

                    selectInput("lab", "Laboratory name", choices = unique(loger_data$labor)),

                    selectInput("room", "Room number", choices = NULL),

                    selectInput("device_t", "Temperature sensor", choices = NULL),

                    selectInput("device_h", "Humidity sensor", choices = NULL),
```

Joonis 4.4 Skript 2

Selleks, et muuta andmeid graafikute kuvamiseks, lisati uuendusnupud. Seejärel lisati ajapiirangu liugurid (time period), temperatuuri ja niiskuse piirangud (temperature limit, humidity limit) ning võimalikud temperatuuri liugurid (possible temperature) (vt Joonis 4.5). Võimaliku temperatuuri liugur oli lisatud graafiku y-telje minimaalse ja maksimaalse väärtuse määratlemiseks. Andmete analüüsimisel leiti ühekordsed võimatud temperatuuri ja niiskuse väärtused, mis ületasid võimalikke väärtusi kümneid kordi ja muutsid graafiku loetamatuks. Suhtelise õhuniiskuse jaoks määrati koodiks väärtused 0 kuni 100% ja temperatuuri jaoks otsustati lisada graafiku piiride määratlemiseks liugur.

```
p("Update data first, then update timeline"),
p("Update data everytime you changed inputs"),
actionButton("update", "Update!"),
p("Don't forget update timeline once"),
actionButton("time_periond", "Update timeline once!"),

sliderInput("time",
            "Time period (sec)",
            value = c(as.POSIXct("2020-01-01 00:00:07"),
                      as.POSIXct("2021-01-01 00:00:07")),
            min = as.POSIXct("2020-01-01 00:00:07"),
            max = as.POSIXct("2021-01-01 00:00:07"),
            timeFormat = "%Y-%m-%d %H:%M:%S"),

sliderInput("t_limit", "Temperature limit (C)", value = c(20, 60), min = -50, max = 100),
sliderInput("h_limit", "Humidity limit (%)", value = c(20, 80), min = 0, max = 100),
sliderInput("t_poss", "Possible temperature (C)", value = c(-10, 100), min = -50, max = 200),
```

Joonis 4.5 Skript 3

Samuti olid veel lisatud väljad pinnatemperatuuri sisestamiseks (vt Joonis 4.6). Kastepunkti mõistet uurides selgus, et saamaks teada, kas pindadele on tekkinud kondensaat, ei piisa kastepunkti temperatuuri teadmisest, tarvis on teada ka pinnatemperatuuri. Kuna saadud andmed sisaldavad ainult suhtelise niiskuse ja temperatuuri näitusid, otsustati lisada pinnatemperatuuri kirje. Samuti kuvati ühel graafikul niiskuse, temperatuuri ja kastepunkti näidud, et näha õhutemperatuuri järsu languse ja niiskuse tõusu hetki. Kuna sellistel hetkedel võib tekkida kondensaat.

```
numericInput("rosa", "Surface temperature (C)", value = 18, min = -50, max = 200),
```

Joonis 4.6 Skript 4

Esimesest leheküljest 8/12 hõivavad temperatuuri, niiskuse graafikud ja kastepunkti temperatuuri ühendusgraafik (vt Joonis 4.7).

```
column(8,
       dygraphOutput("dygraph_t"),
       dygraphOutput("dygraph_h"),
       dygraphOutput("dygraph_r")
)
```

Joonis 4.7 Skript 5

Teisel lehel kuvatakse kahte graafikut: vead temperatuuri ja vead niiskuse kohta. Samuti kuvatakse iga graafiku all vigade kokkuvõtte tabel (vt Joonis 4.8).

```
tabPanel("Plots for errors",
         fluidRow(
           column(6,
                  plotlyOutput("plot_t")
           ),
           column(6,
                  plotlyOutput("plot_h")
           )
         ),
         fluidRow(
           dataTableOutput("SUMMARY_out")
           )
       )
```

Joonis 4.8 Skript 6

4.2 Serveri skript

Esiteks võetakse liidesest andmed (temperatuuri alumine ja ülemine piir, niiskuse alumine ja ülemine piir, aja parem- ja vasakpiir, võimaliku temperatuuri alumine ja ülemine piir, kastepunkti pinnatemperatuur ja ajavahe) (vt Joonis 4.9).

```
server <- function(input, output) {
  t_low_limit <- eventReactive(input$update, {
    min(input$t_limit)
  })
  t_top_limit <- eventReactive(input$update, {
    max(input$t_limit)
  })
  h_low_limit <- eventReactive(input$update, {
    min(input$h_limit)
  })
  h_top_limit <- eventReactive(input$update, {
    max(input$h_limit)
```

Joonis 4.9 Skript 7

Järgmisena on tarvis aktiveerida varem kirjeldatud funktsioon vigade arvutamiseks. Oluline on, et nupu *Update* vajutamisel laetaks funktsiooni eelnevalt määratud liidese parameetrid (vt Joonis 4.10).

```
DATA <- eventReactive(input$update,{

   data_update <- errors(
     temperature=T_sensor(),
     humidity=H_sensor(),
     temperature_low_limit = t_low_limit(),
     temperature_top_limit = t_top_limit(),
     humidity_low_limit = h_low_limit(),
     humidity_top_limit = h_top_limit(),
     temperature_low_filter = t_low_filter(),
     temperature_top_filter = t_top_filter(),
     surface_t = t_rosa(),
     sec_diff_time = diff_time())
})
```

Joonis 4.10 Skript 8

Graafiku andmed filtreeritakse määratud ajaintervalli järgi. Graafikuid kuvatakse peale nupule *Plot* klõpsamist ja uuendatakse peale *Update* nupu klõpsamist (vt Joonis 4.11).

```
DATA_plot <- reactive({
  input$plot_
  DATA_time <- DATA() %>%
    filter(time_t >= as.POSIXct(time_limit_left()) &
             time_t <= as.POSIXct(time_limit_right()),
           .preserve=T)
})
```

Joonis 4.11 Skript 9

Järgmisena häälestati labori ja ruuminumbri valikut ning ruumide ja andurite muutujad. Samuti koostati graafikud temperatuuri ja niiskuse vigadega, mis kuvatakse nupule *Plot* vajutamisel ja uuendatakse nupu *Update* vajutamisel (vt Joonis 4.12).

```
observeEvent(input$plot_,{
  output$plot_t <- renderPlotly({
    ggplot(DATA_plot(), aes(x = time_t, y = temperature)) +
      geom_point(aes(colour = all_error_for_temperature), size = 0.5) +
      xlab("Date")+ ylab("Temperature C")+
      scale_y_continuous(breaks = seq(0, 33, 1))+
      guides(colour=guide_legend("Error type"))+
      labs(title = "Temperature and errors")
  })
})
```

Joonis 4.12 Skript 10

Veagraafikute jaoks on lisatud kokkuvõtlik tabel. Tabel kuvatakse, kui klõpsata nuppu *Plot* ja see koosneb veatüübi nimetusest ja selle numbrist. Seda samuti värskendatakse nuppu *Update* vajutades (vt Joonis 4.13).

```
DATA_SUMMARY <- reactive({
  input$plot_
  SUMMARY=data.frame(Temperature=as.factor(DATA_plot()$all_error_for_temperature),
                     Humidity=as.factor(DATA_plot()$all_error_for_humidity))
  summary(SUMMARY)
})

observeEvent(input$plot_,{
  output$SUMMARY_out <- renderDataTable(DATA_SUMMARY()
  )
})
```

Joonis 4.13 Skript 11

Samuti lisati viimase päeva temperatuuri ja niiskuse graafikud koos piirmääradega ning üldine temperatuuri, niiskuse ja kastepunkti temperatuuri graafik (vt Joonis 4.14).

```
DATA_dygraph_t <- reactive({
  input$plot_
  xts_t <- DATA() %>%
    filter(time_t >= (as.POSIXct(max(time_t))-days(1)) &
             time_t <= as.POSIXct(max(time_t)),.preserve = T)
  zoo_t <- zoo(x=xts_t$temperature, order.by=xts_t$time_t)
  zoo_t
})
observeEvent(input$plot_,{
  output$dygraph_t <- renderDygraph(

    dygraph(DATA_dygraph_t(),main = "Temperature for last day",
            xlab = "Time",ylab = "Temperature C") %>%
      dySeries("V1", stepPlot = F, color = "red",label = "Temperature") %>%
      dyLimit(limit = t_top_limit(), "Temperature max limit",
              strokePattern = "solid", color = "red")%>%
      dyLimit(limit = t_low_limit(), "Temperature min limit",
              strokePattern = "solid", color = "red")
  )
})
```

Joonis 4.14 Skript 12

Valmisrakenduse pildid on toodud lisades (Lisa 1 - Lisa 5).

KOKKUVÕTE

Lõputöö tulemusena loodi laborite seiresüsteem, mis võimaldab vaadata viimase päeva graafikuid ning saada teada rikkumistest. Rakenduse loomiseks kasutati 2020. aasta andmeid, kuid faktiliselt see on koostatud reaalajas töötamiseks. Rakendus võimaldab kontrollida temperatuuri ja õhuniiskuse lubatud väärtuste ületamist ja langust vältimaks häireid katsete/uuringute läbiviimisel ning tagada vere ja muude analüüside korrektne säilitamine. Samuti võimaldab rakendus tagada, et töötajad tunneksid end neis laborites mugavalt. Loodud seiresüsteem võimaldab teada saada andurite või juhtmete riketest, kuna andmete kadumise korral kuvatakse viga. Rakendus võimaldab kontrollida ka kondensaadi tekkimist pindadel.

Lõputöö koostati Ida-Virumaa Keskhaigla poolt püstitatud ülesande järgi. Probleem seisnes selles, et haiglas on mitmeid ruume, kus on kehtestatud tehnilised nõuded temperatuuri ja niiskuse osas, kuid puudub üldine süsteem kriitiliste juhtumite jälgimiseks. Vastavalt haigla ülesandele peab seiresüsteem teatama kolme tüüpi vigadest: lubatud väärtuste rikkumine, andmete kadu ja kondensaadi tekkimine.

Koostatud tööd võib pidada edukaks, kuna antud seiresüsteem vastab kõikidele seatud nõuetele ja lõputöö käigus tuvastati kõik vead. Samas aga selgus, et saadud andmete (temperatuur ja suhteline õhuniiskus) põhjal pole võimalik avastada kondensaadi tekkimise juhtumeid, kuna selleks on vaja teada ka pinnatemperatuuri. Selle probleemi lahendamiseks kasutatakse rakenduses kastepunkti temperatuuri mõistet ja pinnatemperatuuri käsitsi sisestamist.

Selle rakenduse loomiseks kasutas autor rakenduse RStudio keskkonda, programmeerimiskeelt R, paketti Shiny. Uuris kondensaadi, kastepunkti, mikrokliima ja aegridade analüüsi mõisteid.

Töö käigus selgus, et igas ruumis esines igat tüüpi vigu. Kõige rohkem rikkumisi seostati lubatud väärtuste rikkumisega (keskmiselt 13% andmevigadest) ja kõige vähem vigu oli andmete kadumise tõttu (keskmiselt 0.05% andmevigadest).

SUMMARY

As a result of graduation work was create monitoring system for laboratories, which allows you to watch the graphs for the last day and know about the occurrence of violations. To create the application, were used tables with data for a certain period of time but it is aimed at working in real time. The application helps to learn about exceeding the permissible values of temperature and humidity, in order to avoid disruption in the conduct of experiments and to ensure the correct storage of blood and other analyzes, as well as to ensure that employees feel comfortable in these laboratories. The created monitoring system allows to find out about the breakdown of sensors or wires, since an error is displayed when data is lost. The application also allows to check the occurrence of condensation on surfaces.

Graduation work was created on the instructions of the Ida-Viiruma Central Hospital. The problem was that the hospital has many rooms with technical requirements for temperature and humidity, and there was no overall system for monitoring critical cases. According to the instructions of the hospital, the monitoring system must report three types of errors: violation of permissible values, loss of data and the occurrence of condensation.

The work can be called successful because since this monitoring system meets all the specified requirements and as a result of the graduation work all errors were found, however, it turned out that based on the data obtained (temperature and relative humidity) it is impossible to know about the occurrence of condensation, since this requires also know the surface temperature. To solve this problem, the application uses the concept of dew point temperature and manually entering surface temperature.

To create this application, author used the RStudio environment, the R programming language, the Shiny package to create the application. Concepts of condensate, dew point, microclimate and time series analysis were explored.

As a result of the graduation work, it became clear that errors of all types occurred in each room. The largest number of violations were associated with violations of permissible values (on average, 13% of data errors), and the least number of errors were associated with data loss.

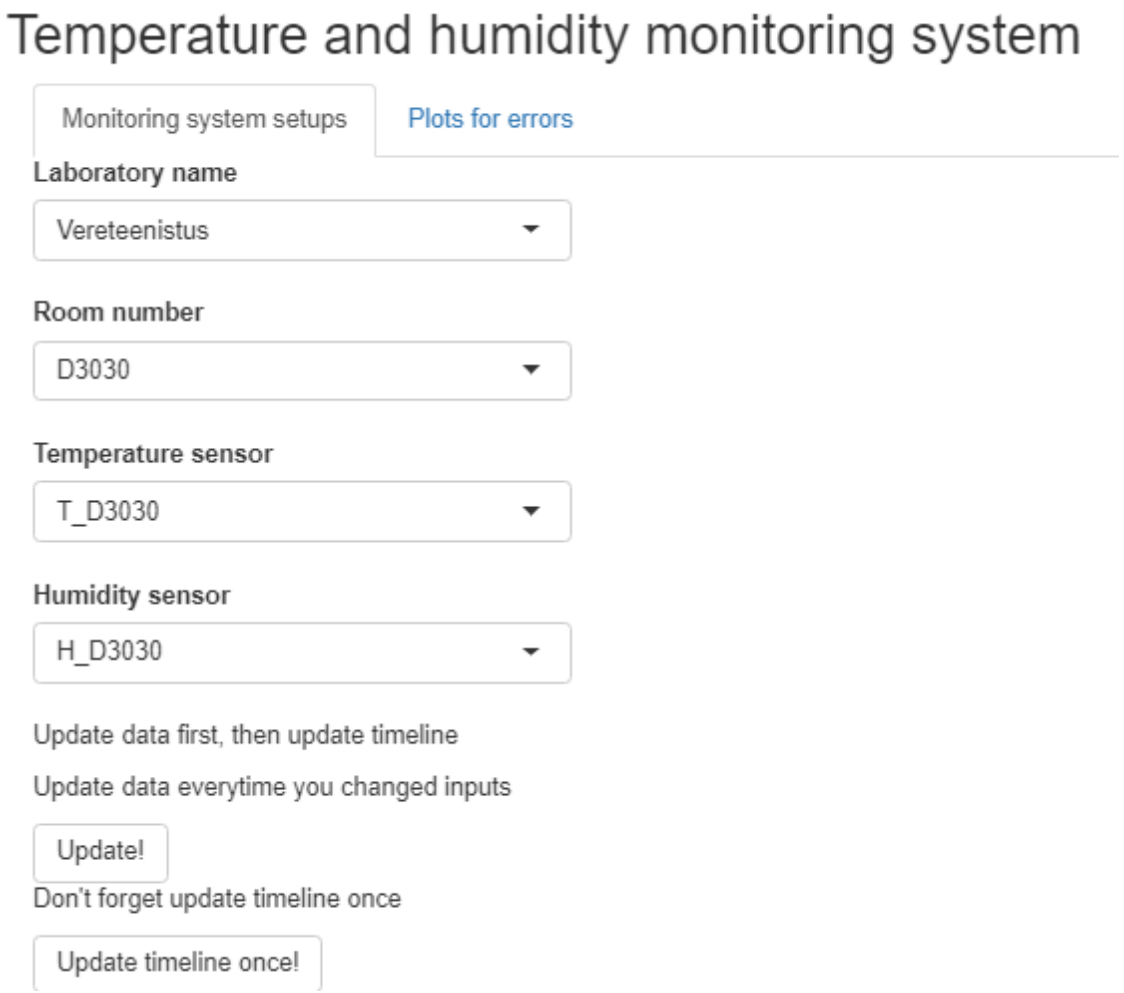
KASUTATUD KIRJANDUSE LOETELU

1. World health organization Geneva, Maintenance and repair of laboratory, diagnostic imaging, and hospital equipment, 1994. [Online] https://apps.who.int/iris/bitstream/handle/10665/139697/5225035841.pdf;sequence=1 , lk 43 (17.11.21)
2. Rene Uuspõld, ÄRIHOONETE ENERGIATÕHUSUSE JA KASUTAJASOBIVUSE ANALÜÜS, 2015. [Online] https://digikogu.taltech.ee/et/Item/db5f0cb7-f598-4ebe-b27b-c75aa29389c6 , lk 14 (20.11.21)
3. Компания «VseOkna», Kastepunkt, 2019. [Online] https://vseokna.by/terminy/tochka-rosy (20.11.21)
4. Игорь Свидерский, ОБРАЗОВАНИЕ КОНДЕНСАТА. ФИЗИЧЕСКАЯ СУТЬ ПРОЦЕССА, 2021. [Online] https://acadomia.ru/articles/inzhenernye-kommunikatsii/formation-of-condensate/ (30.11.21)
5. Deepanshu Bhalla, List of Companies using R, 2017. [Online] https://www.datasciencecentral.com/profiles/blogs/list-of-companies-using-r (05.12.21)
6. Terence Shin, The Most In-Demand Skills for Data Scientists in 2021, 2021. [Online] https://www.kdnuggets.com/2021/04/most-demand-skills-data-scientists.html (06.12.21)
7. Peeter Tinits, Mis on R? Mis on RStudio?, 2020. [Online] https://peetertinits.github.io/gitbooks/tekstid_R_2020/mis-on-r-mis-on-rstudio.html (06.12.21)
8. Shiny, 2020. [Online] https://shiny.rstudio.com/ (08.12.21)
9. Tähtis on R-stuudio funktsioonide kasutamise alustamine [Online] https://et.education-wiki.com/3693990-important-on-how-to-getting-started-with-r-studio-functions--software--programming (09.12.21)
10. Raul Kangro, Tartu, 2016. [Online] https://courses.ms.ut.ee/MTMS.01.023/2016_fall/uploads/Main/aegread.pdf (09.12.21)
11. Ako Saugo, TalTech, kursus Ökonomeetria, doktoriõpe, 2013. [Online] https://www.sauga.pri.ee/portfoolio/OkonomeetriaLoengStatsionaarsedAegread.pdf (15.12.21)
12. Indeed Editorial Team, What Is a User Interface? (Definition, Types and Examples), 2021 [Online] https://www.indeed.com/career-advice/career-

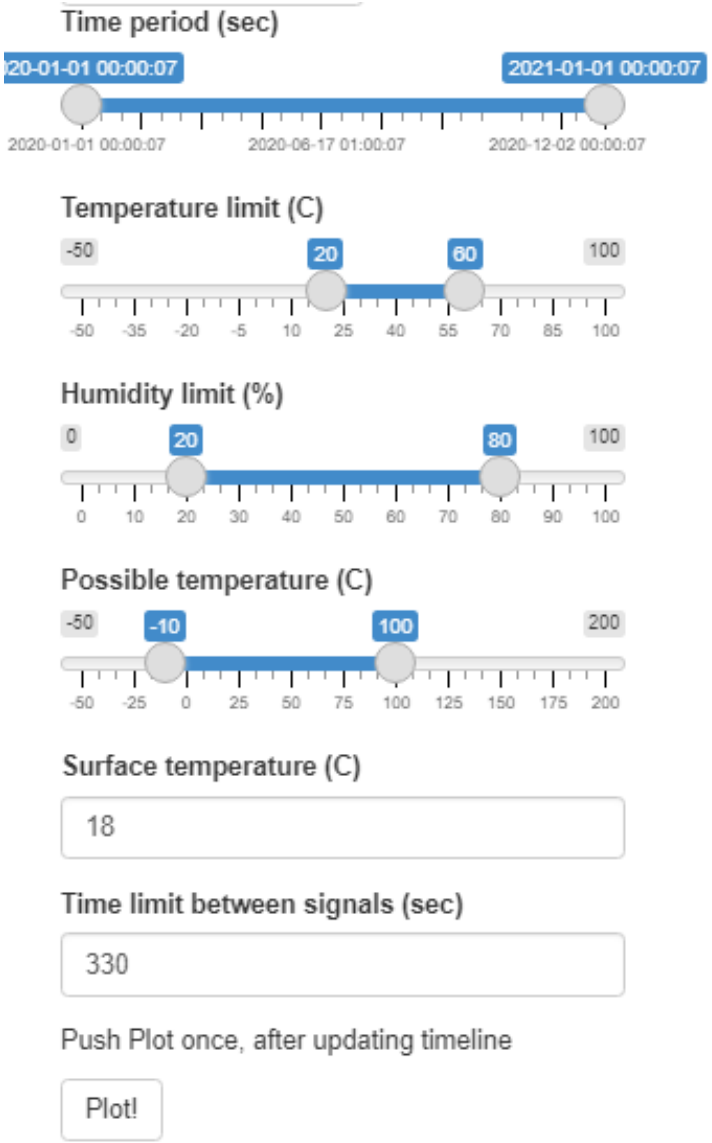
development/user-interface (15.12.21)

LISAD

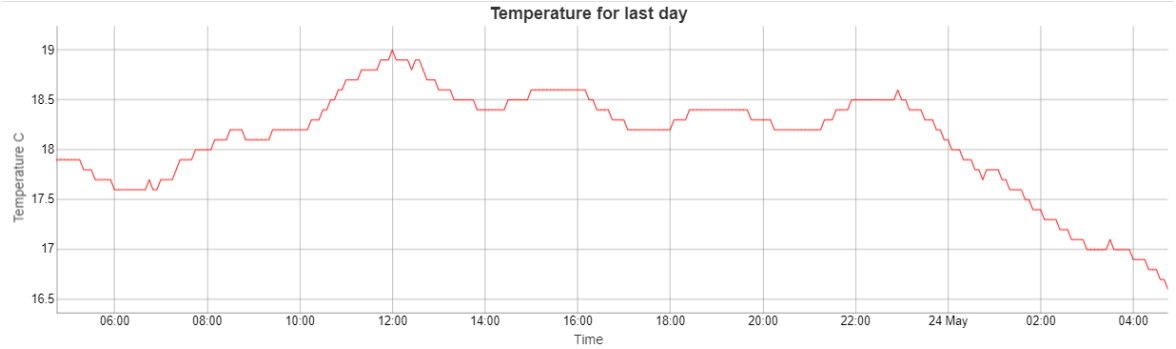
Lisa 1 Rakenduse esimene leht 1

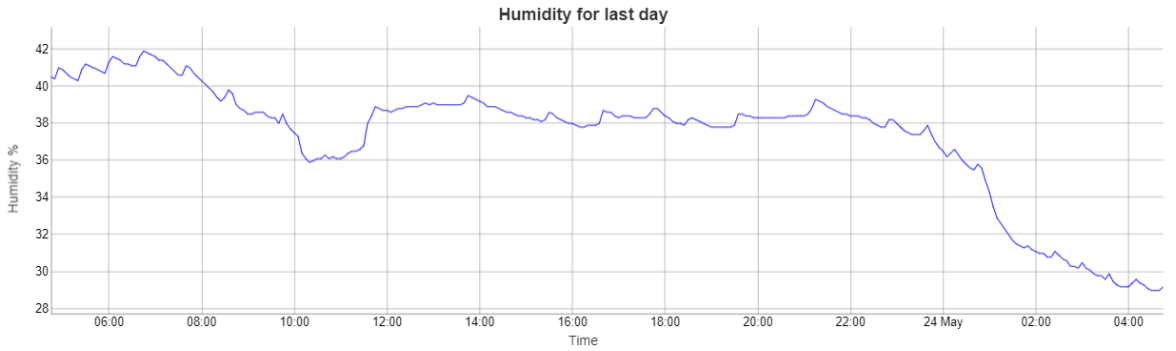


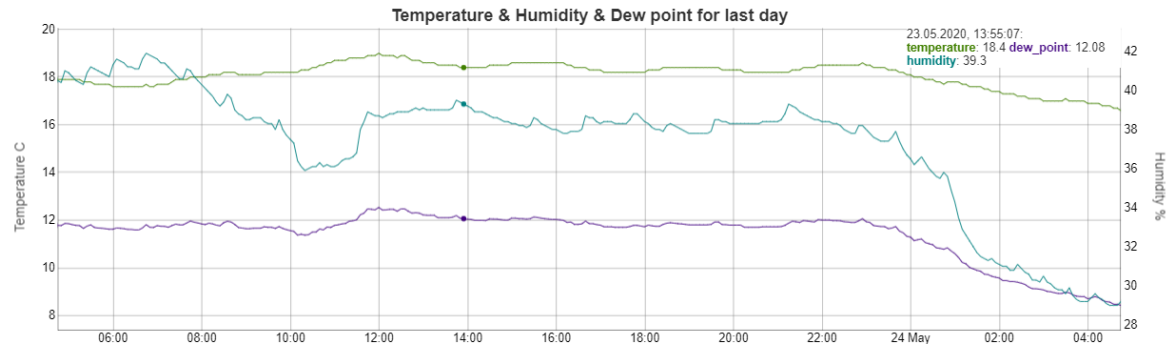
Lisa 2 Rakenduse esimene leht 2



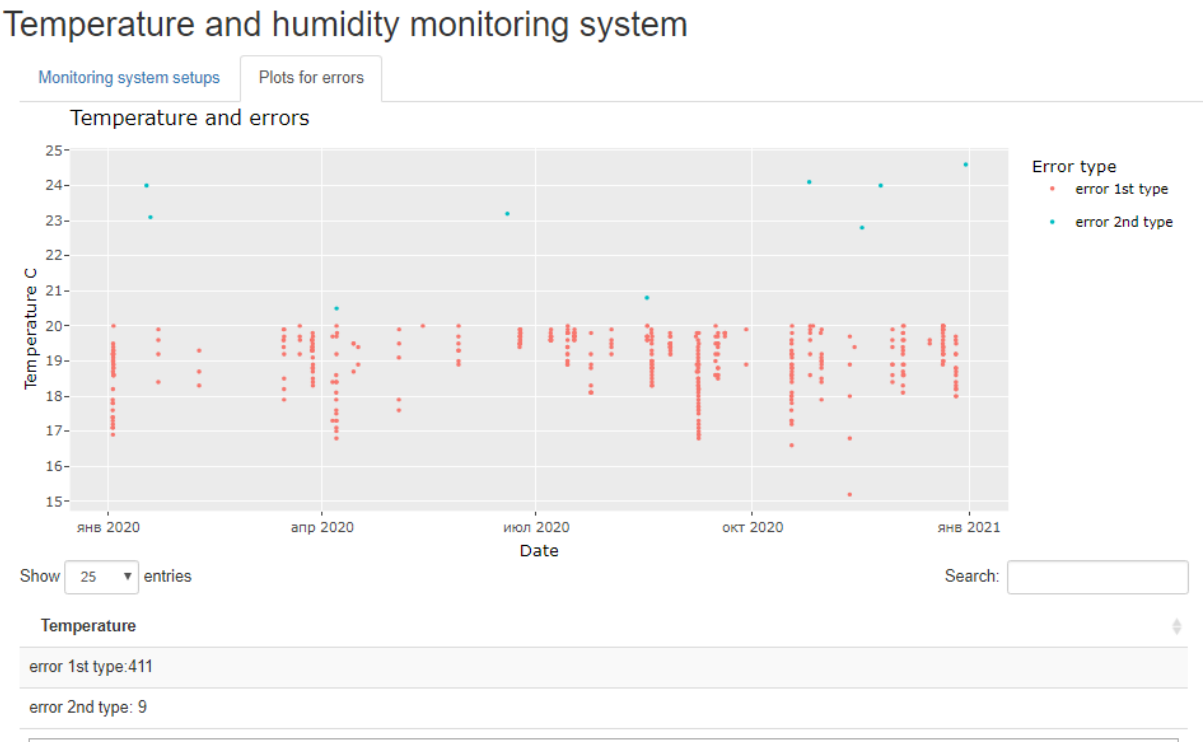
Lisa 3 Rakenduse esimene leht 3



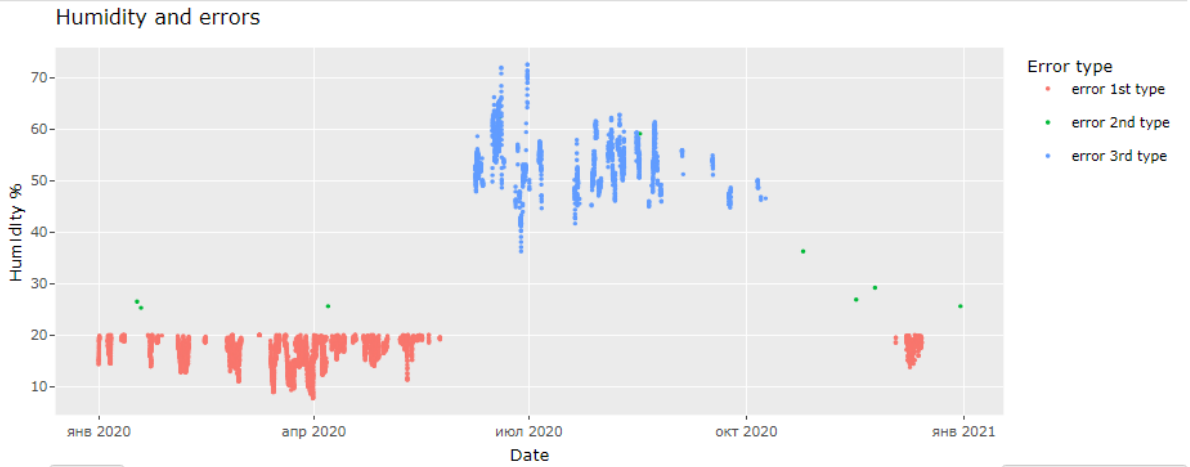




Lisa 4 Rakenduse teine leht 1



Lisa 5 Rakenduse teine leht 2



Show 25 entries Search:

Humidity
error 1st type:4537
error 2nd type: 9
error 3rd type:1534