TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Vjatšeslav Koloskov 193959IADB

Veebirakenduse arendus COVID-19 statistiliste andmete vaatamiseks

Bakalaureusetöö

Juhendaja: German Mumma
MSc

Tallinn 2022

Autorideklaratsioon

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

Autor: Vjatšeslav Koloskov

16.05.2022

Annotatsioon

Käesoleva bakalaurusetöö peamine eesmärk on luua eestikeelne veebirakendus, mis näitab koroonaviiruse haiguse levimise statistikat maailmas. Loodav rakendus peaks pakkuma kasutajatele suur koroonaviiruse levimisega seotud andmete arv.

Hetkel olevaid eestikeelseid veebirakendusi koroonaviiruse statistika vaatamiseks on väike arv ja neid on raske leida.

Lahenduse elluviimiseks selgitatakse läbi analüüsi loodava rakenduse skoop. Arendusprotsessi käigus kirjeldatakse, millist metoodikat ja milliseid tehnoloogiaid kasutatakse veebirakenduse loomiseks.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 27 leheküljel, 5 peatükki, 18 joonist, 0 tabelit.

Abstract

Web Application Development for Viewing COVID-19 Statistical Data

The main goal of this final thesis is to create a web application in Estonian that shows statistical data on the spread of coronavirus disease in the world. The application to be developed should provide users with a large amount of data related to the spread of the coronavirus. This data should be located on graphs and charts for easier understanding and perception of information.

There are currently a small number of Estonian web applications for viewing coronavirus statistics and they are difficult to find.

To implement the solution, the scope of the application created through the analysis is explained. During the development process, the methodology and technologies used to create the web application are described.

The thesis is in Estonian and contains 27 pages of text, 5 chapters, 18 figures, 0 tables.

Lühendite ja mõistete sõnastik

API	*Application programming interface* - rakendusliides
COVID-19	Akronüüm ingliskeelsest nimetusest *coronavirus disease* 2019
CSS	*Cascading Style Sheets* - keel veebilehtede kujundamiseks ja esitamiseks
Eesrakendus	Kliendi rakenduse osa
HTML	*HyperText Markup Language* - keel disainitud dokumentide *v*eebilehitsejas kuvamiseks
Hook	React uude, mis võimaldab kasutada React-i funktsioone ilma klasse kirjutamiseta.
ISO 3166-1 alpha-2	Kahetäheline riigikoodide süsteem
JSON	*JavaScript Object Notation* - andmeformaat
ms	Ühe tuhandiku sekundi lühend
Tag	HTML silt. See on veebilehe põhiline struktuuriüksus
URI	*Uniform Resource Identifier* ehk ühtne ressursiidentifikaator, sõne infoallika üheseks määramiseks veebis

Sisukord

Jooniste loetelu

1 Sissejuhatus

Üks meie aja suurimaid probleeme on koroonaviirus. Koroonaviiruse leviku kohta veebirakenduse otsides võib inimene sattuda võõrkeelsetele veebilehtedele või veebilehtedele, kus on palju reklaamplakateid, mis halvendab rakenduse kogumuljet.

Paraku on internet informatsioonist üle kuhjatud, mis teeb kindla asja leidmise raskeks. Lisaks sellele on raske leida ühtset tervikliku rakendust eesti keeles, millel on arusaadav struktuur. Tihtipeale peab kasutaja otsima juppide kaupa infot ning nendest panema kokku ühtse pildi Koronaviiruse leviku kohta.

Käesoleva lõputöö eesmärk on luua veebirakendust, mis näitab koroonaviiruse haiguse levimise statistikat maailmas. Eesmärgiks on ka leida viise, kuidas teha optimaalseid API (*Application programming interface*) päringuid ja käsitleda saadud andmeid. Rakendus sobib otsimiseks nii maailma kui ka konkreetse riigi kohta COVID-19 levimise statistikat.

Veebirakendus on loodud API arhitektuuri kasutades. Rakenduse graafiline liides on maksimaalselt kasutajale arusaadav ja lihtne.

Bakalaureusetöö on jaotatud neljaks suuremaks sisupeatükiks. Esimeses neist on kirjeldatud sügavamalt lahendatavat probleemi ennast, miks on sarnaseid rakendusi tänapäeva ühiskonnas vaja. Veel on kirjutatud ka tausta, lähtetingimusi ning metoodikat. Teises peatükis on keskendatud analüüsile. Igal rakenduse loomise etapil on olemas eraldi vahend, mille abil on võimalik lihtsalt selle etapi ellu viia. Selles peatükis kirjeldatakse nende etappide vahendi valikut ja analüüsitakse alternatiive. Analüüsiti ka API kasutamist ning analoogseid rakendusi. Kolmandas peatükis on käsitletud lahenduse loomist, sh nõudeid lahendusele, veebirakenduse üldist struktuuri ja ülesehitust ning lahenduse realiseerimise faasi koodinäidetega. Viimases peatükis on kirjeldatud edasiarenduse võimalused.

2 Probleemi püsitus ja projekti eesmärk

Selles peatükis kirjeldatakse lähemalt lõputöös käsitletavat probleemi ning selle püstitust koos sõnastatakse projekti ja diplomitöö eesmärgid.

2.1 Taust ja probleem

Koroonaviiruse haigus või COVID-19 on tänapäevane põhiteema, mis puudutab kõiki meie maailmas. Statistika aitab jälgida tänaseid COVID-19 leviku trende. Informeeritus probleemist aitab inimestel saada ettevaatlikumaks ja oma tervist rohkem hoida.

Otsides veebirakenduse Internetis, mis näitaks koroonaviiruse leviku statistikat maailmas, antud lõputöö autor märkas, et paljud veebirakendused COVID-19 statistikaga ei sisalda üsna palju andmeid või need andmed on väga ühised, ei sisalda graafikuid ja diagramme, mis aitaksid kasutajatel statistikat selgelt mõista. Juhtub nii, et veebirakendus sisaldab palju reklaamplakateid, mis halvendab rakenduse kogumuljet ja kasutajasõbralikkust või lihtsalt see veebirakendus on võõrkeeles.

2.2 Projekti eesmärk

Käesoleva töö eesmärk on analüüsida erinevaid võimalusi antud probleemi lahendamiseks, näiteks leida viise, kuidas teha optimaalseid API päringuid ja käsitleda saadud andmeid.

Teiseks eesmärgiks on luua veebirakendus eesti keeles, mis näitab koroonaviiruse haiguse levimise statistikat nii maailmas, kui ka detailsema statistikat ühe konkreetse maailma riigi kohta. Veebirakendus peab kiiresti töötama ja peab olema kasutajasõbralik.

2.3 Metoodika

Seejärel võrreldakse erinevaid raamistikke ja programmeerimiskeeli ja valitakse välja raamistikud ja keel, millega hakatakse eesrakendust arendama.

Põhjendatakse ka tööriistade ja arenduskeskkondade valikut.

Töö praktilises osas kirjeldatakse töö arhitektuuri üldiselt. Kirjeldatakse mida kasutati tulemuse saamiseks ja kuidas saadud tulemus töötab ning välja näeb.

3 Analüüs

Antud lõputöö praktiliseks väljundiks on luua kasutajasõbralik veebirakendus. Töötavas lahenduses peab olema võimalik vaadata maailma kui ka ühe riigi kohta koroonaviiruse leviku statistikat.

Veebirakendus peab olema tehtud kasutades API. See tähendab, et veebirakendusel on ainult klientpoolne liide, sest kõik statistika andmeid võtakse erinevatest API-dest ja veebirakendus ei vaja serveripoolset liidet, sest ei ole andmebaasi, millega on vaja suhelda. Veebirakenduse kasutaja aga näeb eesrakendus mida nimetatakse ka *frontend*.

Tänapäevases maailmas on loodud palju erinevaid programmeerimiskeeli ja vahendeid, mida kasutatakse veebirakenduste loomiseks. Valiku tegemisel lõputöö autor võttis arvesse nii vehendite populaarsust, kui ka selle vahendi funktsionaalsust ja eeliseid teiste kandidaatide ees.

3.1 Frontend ehk eesrakendus

Frontend (edaspidi eesrakendus) on veebirakenduste avalik osa, millega kasutaja saab suhelda ja otse ühendust võtta. Eesrakendus sisaldab nii funktsionaalsete ülesannete kuvamist, kliendi poolel teostatavat kasutajaliidest kui ka kasutajapäringute töötlemist. Tegelikult eesrakendus on kõik, mida kasutaja veebilehte avades näeb.

Eesrakendus on kõik, mida brauser saab lugeda, kuvada ja/või käivitada. See tähendab, et see on HTML (*HyperText Markup Language)*, CSS (*Cascading Style Sheets)* ja JavaScript. HTML ütleb brauserile, mis on lehe sisu, näiteks: pealkiri, lõik, loend, loendiüksus. CSS annab brauserile teada, kuidas kuvada elemente, näiteks "20 piksli taane pärast esimest lõiku" või "kogu kehaelemendi tekst peab olema tumehall ja kirjutatud Verdana fondiga". JavaScript annab brauserile teada, kuidas kerget programmeerimiskeelt kasutades teatud interaktsioonidele reageerida. [1]

3.1.2 Javascript

JavaScript (sageli lühendatud JS-ks) on kerge, tõlgendatav, esmaklassiliste funktsioonidega objektorienteeritud keel, mida tuntakse kõige paremini veebilehtede skriptikeelena, kuid seda kasutatakse ka paljudes mitte-brauserikeskkondades. See on prototüübipõhine, mitme paradigmaga skriptikeel, mis on dünaamiline ja toetab objektorienteeritud, kohustuslikke ja funktsionaalseid programmeerimisstiile. [2]

JavaScript on mõnda aega olnud üks populaarsemaid veebiarenduskeeli. Juba 9 aastat järjest on JavaScript säilitanud oma tugipunkti kõige laialdasemalt kasutatava programmeerimiskeelena. [3]

JavaScripti korralik aspekt on see, et seda toetavad paljud veebilehitsejad, nagu Google Chrome, Internet Explorer, Firefox, Safari, Opera jne. Seejärel kliendid saavad probleemideta pääseda ja kasutada veebirakendusi mis tahes Interneti-brauseris.

Põhinedes nendel punktidel oli valitud Javascript lõputöö projekti arendamise keeleks.

3.1.3 React.js

Lõputöö veebirakenduse arenduse keel on Javascript ja sellepärast peab olema valitud raamistik veebirakenduse arendamiseks, mis kasutab Javascript. Tänapäevases maailmas kolm kõige populaarsemat raamistikut on React, Angular ja Vue.js.

React kasutab JSX. JSX on spetsiaalne süntaks, mis näeb välja nagu HTML, teisendab Reacti API-kutsed ja lõpuks renderdab HTML-i. [4]

Traditsioonilised raamistikud, nagu Angular ja Vue.js, tugevdavad HTML-i. Nad kasutavad HTML-i sees JavaScripti. Nad on loonud HTML-i atribuudid, mis annavad sellele lisavõimalusi. [4]

Selle lähenemisviisi peamine probleem on see, et programmeerija peab õppima need uued HTML-i atribuudid või alati vaatama ametlikku dokumentatsiooni. [4]

React eristub teistest raamistikutest oma virtuaalse dokumendiobjekti mudeliga (DOM), mis pakub suurepäraseid funktsioone. See on ideaalne raamistik neile, kes ootavad suurt liiklust ja vajavad selle haldamiseks kindlat platvormi. [5]

Põhinedes eeltodud punktidel oli valitud React lõputöö projekti arendamise raamistikuks.

3.2 API

Application programming interface ehk API võimaldab ettevõtetel avada oma rakenduste andmed ja funktsioonid välistele kolmandatest osapooltest arendajatele, äripartneritele ja ettevõtte siseosakondadele. See võimaldab teenustel ja toodetel omavahel suhelda ning dokumenteeritud liidese kaudu üksteise andmeid ja funktsioone kasutada. Arendajad ei pea teadma, kuidas API rakendatakse; nad kasutavad lihtsalt liidest teiste toodete ja teenustega suhtlemiseks. API kasutamine on viimase kümnendi jooksul kasvanud niivõrd, et paljud tänapäeval populaarseimad veebirakendused poleks API-deta võimalikud. [17]

API on määratletud reeglite kogum, mis selgitab, kuidas arvutid või rakendused omavahel suhtlevad. API-d asuvad rakenduse ja veebiserveri vahel, toimides vahekihina, mis töötleb andmeedastust süsteemide vahel. [17]

API töötab järgmiselt:

1. Eesrakendus teeb päringut andmete saamiseks. Seda päringut töödeldakse rakendusest veebiserverisse API URI (*Uniform Resource Identifier)* kaudu.
2. Pärast tõelise päringu saamist teeb API päringu välisele programmile või veebiserverile.
3. Server saadab API-le vastuse koos nõutud andmetega.
4. API edastab andmed algselt taotlevale rakendusele.

3.3 Kasutatud tööriistad

Selles peatükis kirjeldatakse täpsemalt milliseid tööriistu on kasutatud antud lõputöö praktilise osa teostamisel.

3.3.1 Visual Studio Code

Visual Studio Code on koodiredaktor, mis on ümber määratletud ja optimeeritud kaasaegsete veebi- ja pilverakenduste loomiseks ja silumiseks.

Tõsise koodi kirjutamise jaoks raamistikul on olemas tööriistad, mis võimaldavad koodi paremini mõista kui lihtsalt tekstiplokid. Intuitiivsed kiirklahvid, lihtne kohandamine ja kogukonna kaasatud klaviatuuri otseteed võimaldavad kasutajal koodis hõlpsasti navigeerida. [6]

3.3.1 API valik

Enne veebirakenduse tegemise alustamiseks tuli valida mille API andmeid see hakkab kasutama.

API on programmeerimiskoodi komplekt, mis võimaldab andmeedastust ühe ja teise tarkvaratoote vahel. [11]

API-de valik on väga suur erinevates piirkondades. API-d võivad olla nii tasulised, kui ka tasuta. Lõputöö veebirakenduse arendamiseks oli otsustatud kasutada ainult tasuta API-d.

Eelistavalt API-del peaks olema ligipääs võimalikult suurele andmete hulgale selleks, et väheneda päringute arvu, sest iga päring serverile ning serveri vastuse ootamine võtab aega ja aeglustab veebirakenduse töö kiirust.

Disease.sh [14] on tasuta API, mis hangib andmeid mitmest kohast kõige täpsemate andmete saamiseks. Kogu protsess on automatiseeritud, nii et API kasutaja saab alati värskemad andmed. API vastuseid esitatakse JSON-vormingus (*JavaScript Object Notation)*, mis teeb integreerimise kergem ja usaldusväärsem.

See API sisaldab peaaegu kogu olulist info COVID-19 levikust. Siin võib saada nii ajaloolised kui ka praegused statistilised andmed iga konkreetse riigi kohta.

COVID19 API [15] on tasuta API, mis sisaldab andmeid, mida uuendatakse mitu korda päevas. Selle API abil on võimalik saada andmeid iga maailma riigi kohta ühe päringuga JSON-vormingus. See on väga kasulik lehe loomisel, mis sisaldab COVID-19 levikuga seotud üldised statisltilised andmed.

M Media COVID-19 [16] on tasuta API, mille lõpp-punktid ajalooliste ja peaaegu reaalajas andmete käsitlemiseks. Keskmine API reageerimisaeg on 70–250 ms. Selle API suureks plussiks on see, et siin on võimalik saada vaktsineeritud inimeste number iga maailma riigi kohta.

3.4 API kasutamine

API-st andmete saamiseks peab rakendus sellele API-le päringu tegema. Veebirakenduse loomisel oli kasutatud *fetch()* meetod. *Fetch* API pakub HTTP-päringute ja vastustega töötlemiseks JavaScripti liidest. Samuti pakub see globaalset *fetch()* meetodit, mis annab võimaluse andmeid asünkroonselt saada lihtsalt ja loogiliselt. [12]

Reacti *hooks* on korduvkasutatavad funktsioonid. Kui rakenduses on komponendi loogika, mida peab kasutama mitu komponenti, on võimalik panna selle loogika *hook*-I sisse. Omatehtud *hook* reeglitena peab algama sõnaga "*use*", näiteks: *useFetch*.

Joonisel 1 on *useFetch hook*-i näide. See omatehtud *hook* on kasutatud igal lehel, kus toimuvad päringud erinevatele API-dele. Funktisoon võtab parameetrina viite, mis lihtsustab selle funktsiooni kasutamist rakenduse erinevates osades.

```
export default function useFetch(link) {
    const [data, setData] = useState(null);
    const [loading, setLoading] = useState(true);
    const [error, setError] = useState(null);
    useEffect(() => {
        fetch(link)
        .then(res => res.json())
        .then(res => setData(res))
        .catch(err => {
            setError(err);
          })
        .finally(() => {
            setLoading(false);
        });
    }, [])
    return {data, loading, error}}
```

Joonis 1. *Hook useFetch* näide.

Joonisel 2 on näide, kuidas omatehtud *useFetch hook* on kasutatud koodis, mis tagastab järgmiseid parameetreid (Joonis 14):

- *data* – API-st saadud andmed objektina;
- *loading* – parameeter, mis sisaldab infot päringu oleku kohta (kas andmed on juba saadud või mitte);
- *error* – parameeter, mis sisaldab päringu ajal saadud viga, kui see on olemas.

```
const {data, loading, error}  =
useFetch("https://disease.sh/v3/covid-19/all");
```

Joonis 2. useFetch() hook-i kasutamise näide.

Pärast päringu tegemist API-le *data* parameeter omandab kõik andmed JSON-vormingus, mis mingi konkreetne API saatis päringule vastuseks tagasi. Joonisel 3 on näidatud mis sisaldab *data* parameeter pärast Disease.sh API-le päringu tegemist. Siin on *data* parameeter on kompleksne objekt, mis sisaldab lihtobjekte, nagu: *updated*, *cases*, *todayCases* ja nii edasi.

Joonis 3. *Data* parameeter JSON-vormingus pärast Disease.sh API-le päringu tegemist.

Pärast andmete saamist API-st, neid andmeid kasutatakse otseselt koodis. Joonisel 4 on näidatud, et lihtobjekte *data* parameetrist kasutatakse otseselt, küsides nende väärtusi, näiteks tehtud teste arvu saamiseks kasutatakse *data?.tests*. *ToLocaleString* meetod teeb arvud loetavamaks.

```
<ul>
      <li> Juhtumeid kokku: <b>{data?.cases.toLocaleString()} </b> </li>
      <li> Juhtumeid täna: <b>{data?.todayCases.toLocaleString()} </b>
</li>
      <li> Surnud kokku: <b>{data?.deaths.toLocaleString()} </b> </li>
      <li> Surnud täna: <b>{data?.todayDeaths.toLocaleString()} </b>
</li>
      <li> Tervenenud kokku: <b>{data?.recovered.toLocaleString()} </b>
</li>
      <li> Tervenenud täna: <b>{data?.todayRecovered.toLocaleString()}
      </b> </li>
      <li> Tehtud teste kokku: <b>{data?.tests.toLocaleString()} </b>
</li>
</ul>
```

Joonis 4. *Data* parameetri andmete kasutuse näide.

Joonisel 5 on näidatud kõigi eelmiste toimingute üldine tulemus veebilehel.

Juhtumeid kokku: **520,487,792**
Juhtumeid täna: **209,091**
Surnud kokku: **6,286,939**
Surnud täna: **404**
Tervenenud kokku: **475,081,651**
Tervenenud täna: **127,466**
Tehtud teste kokku: **6,330,002,608**

Joonis 5. Statistika bloki näide pärast andmete saamist ja töötlemist.

3.4 Analoogsed lahendused

Tänapäevaks on loodud mitmesuguseid COVID-19 leviku statistika vaatamiseks veebirakendusi inglise keeles, aga ainult väike hulk neist on eesti keeles.

Selles peatükis analüüsitakse eestikeelsaid veebirakendusi koronaviiruse statistiliste andmete vaatamiseks.

Koroona.ut.ee [18] on Tartu Ülikooli teadlaste arendatud koroonaviiruse analüüsimiseks veeebirakendus. Selle rakenduse suureks plussiks on väga detailne statistika Eesti kohta. Veebilehel võib vaadata Eesti koroonaviiruse leviku kaarti, joondiagrammi, kus on COVID-19 haigusjuhtumid Eestis, testide statistikat maakonniti ja kuude vaates. Need andmed asuvad joondiagrammidel ja tabelites.

Koroonaviiruse leviku statistika maailma riikides on ainult positiivselt testitud juhtumite kohta. See tähendab, et selle veebirakenduse kasutaja ei või ammendavalt tutvuda COVID-19 leviku olukorraga maailmas ja see on veebirakenduse puudus.

Terviseamet.ee veebirakendusel on olemas leht, mis on nimetatud "Koroonaviiruse andmestik" [19]. Sellel lehel asuvad detailsed statistilised andmed Eesti kohta. Selle veebirakenduse puuduseks on täielik maailma riikidega seotud statistika puudumine ning andmed Eesti kohta selle veebilehe vaatamise ajal (14.05.2022) uuendati 5 päeva tagasi (09.05.2022).

Lõputöö käigus loodud veebirakendus elimineerib alternatiivide juures välja toodud vigu ning on kasutajale arusaadav ja kerge, sest eeltoodud veebirakendused pakuvad detailset, aga ülekoormatud andmeid.

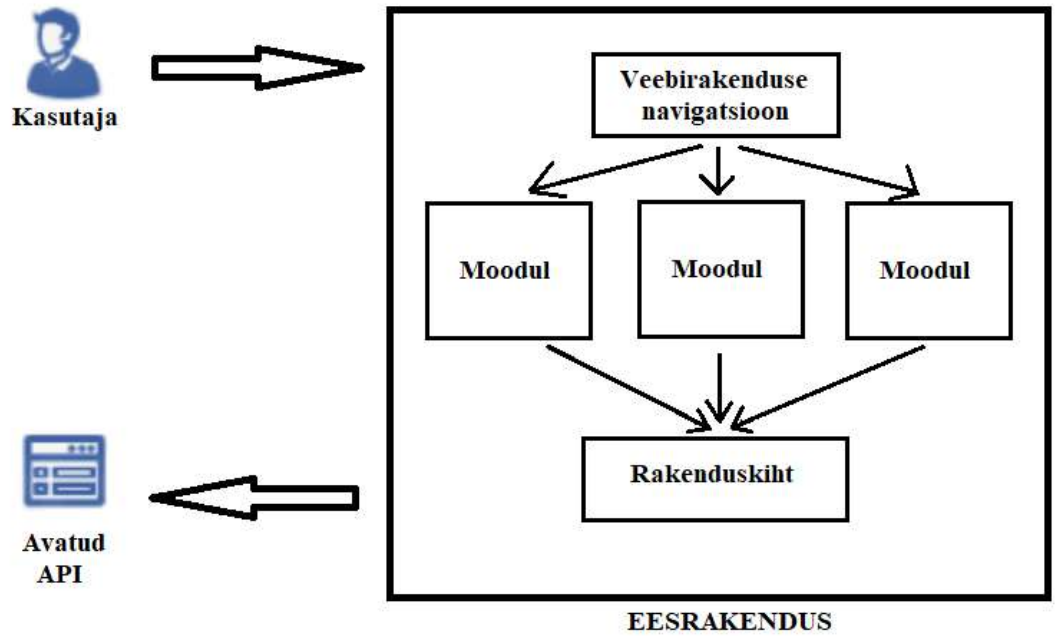
4 Loodud rakendus

Antud peatükis kirjeldatakse veebirakenduse arhitektuuri ning kuidas veebirakendus oli loodud ja kuidas see praegu töötab. Tutvustatakse põhjalikumalt veebirakenduse ülesehitust ja vaateid. Kirjeldatakse ja kuidas veebirakendus suhtleb API-dega ja käsitleb API-dest andmeid ning kuidas graafikud ja diagrammid oli loodud. Tuuakse välja põhjalikumaks arusaamiseks nii koodinäited, kui ka pildid, mis illustreerivad, kuidas veebirakendus näeb välja.

4.1 Arhitektuur

Loodud veebirakendus kujutab endast eesrakenduse. Andmete kättesaamiseks suhtleb eesrakendus tasuta API-dega (Joonis 6).

Veebirakenduse kasutaja veebilehe külastamisel peab kasutama veebilehe navigatsiooni selleks, et näha ja kasutada rinevaid veebirakenduse lehti. Rakenduse navigatsioon kasutab erinaevaid mooduleid, mis kujutavad veebilehe infot ja suhtlevad rakenduskihiga. Rakenduskiht suhtleb API-dega andmete saamiseks ja edastab neid moodulitele, mis töötlevad ja näitavad neid kasutajatele veebilehel.



Joonis 6. Projekti arhitektuur.

4.2 Rakenduse navigatsioon

Navigatsioon veebirakenduses on võimalus selle lehtede vahel navigeerida. Mida lihtsam ja selgem on navigeerimissüsteem, seda parem on külastajatele, sest see võimaldab neil leida vajalikku infot ja kiiresti veebirakenduse lehtedes navigeerida. [7]

Navigatsioonita pole võimalik külastada veebirakenduse lehti peale esilehti. Veebirakenduse navigatsiooni loomiseks oli kasutatud navigatsiooni kogu nimetatud *React Router*.

React Router on täisfunktsionaalne kliendi- ja serveripoolne marsruutimisteek Reactile, JavaScripti teegile kasutajaliideste loomiseks. See töötab kõikjal, kus React töötab: nii veebis, kui ka serveris. [8] Sellega imporditakse eesrakendusse individuaalsed komponendid, mis lihtsustavad liikumise veebirakenduse lehtede vahel ja muudavad seda mugavamaks ja kiiremaks.

Navigatsioon veebirakenduse lehtede vahel on võimalik, kui React komponendid on Routes ja Route siltide vahele pakitud (Joonis 7). Tag (edaspidi HTML silt) *Link* annab võimaluse

navigatsiooni linki klikkides laadida ainult see komponent eesrakenduses, mida küsitakse. See tähendab, et tänu sellele ei raisata ressursse lehe täielikule uuesti laadimisele.

```
export default function App() {
    return (
      <> <header>
        <div className="links-container">
        <Link to="/" className="link">Esileht</Link>
        <Link to="/Compare" className="link">Võrdle</Link>
        <Link to="/CountryStats" className="link">Riik</Link>
        <Link to="/WorldStats" className="link">Maailm</Link>
        </div>
      </header>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/Compare" element={<Compare />} />
        <Route path="/CountryStats" element={<Page />} />
        <Route path="/WorldStats" element={<World />} />
      </Routes> </>
    );
}
```

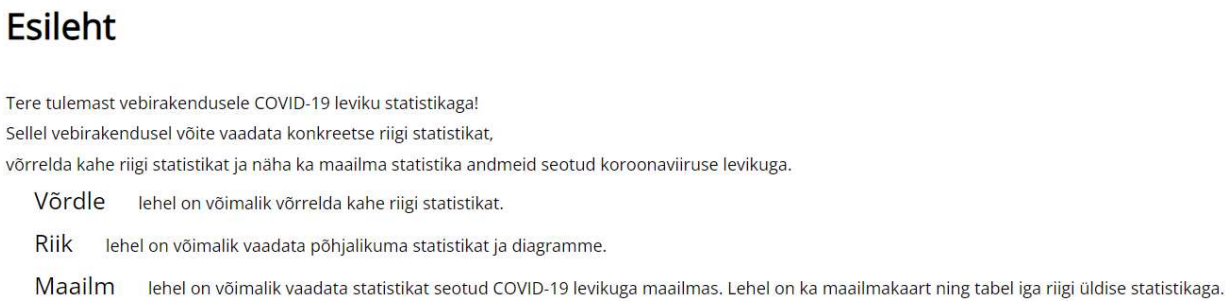
Joonis 7. App funktsiooni kood App.js failis.

Kuna veebirakendus peab olema maksimaalselt lihtne ja arusaadav, otsustati disainida lihtsat navigatsiooni lahtri kasutades HTML ja CSS (Joonis 8).

Joonis 8. Navigatsiooni lahter veebirakenduses.

4.3 Rakenduse avaleht

Veebirakenduse avaleht on esimene leht, kuhu kasutaja satub. Siin asub lühike veebirakenduse kirjeldus ning iga veebirakenduse lehe kirjeldus, ehk mis andmeid võib seal leida ja vaadata (Joonis 9).



Joonis 9. Veebirakenduse esilehel asuv info.

4.4 Maailma statistika

Maailma statistika lehel veebirakenduses asub summaarne maailma statistika, mis paikneb maailmakaardil, sektordiagrammil ja tabelis.

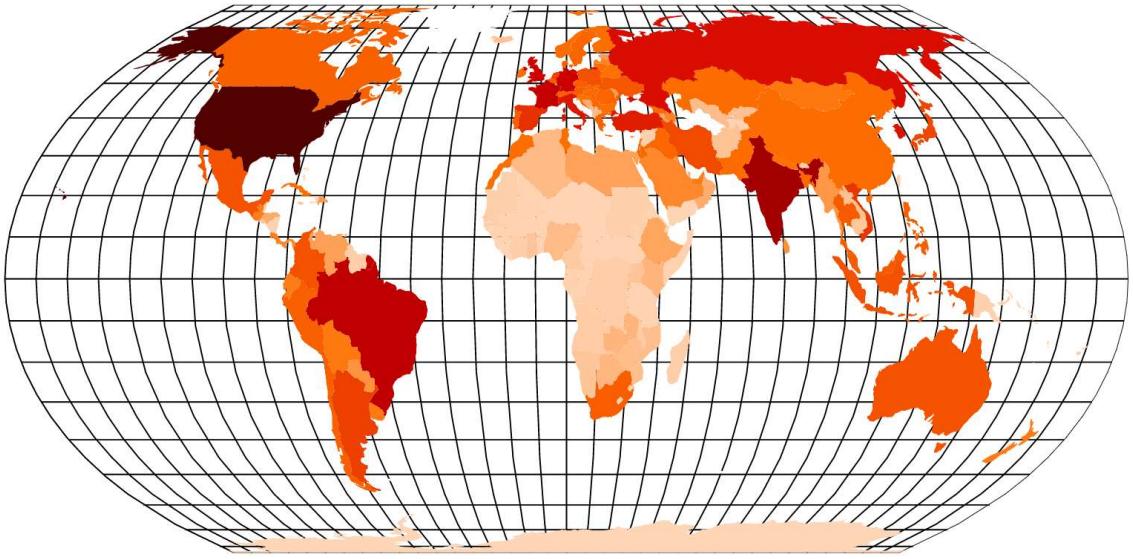
Andmete visualiseerimine on graafiline viis suurte andmete hulga selgelt ja kiireks esitamiseks. Info kogumise ja analüüsimise käigus asetatakse see visuaalsesse konteksti ja näidatakse näiteks kaardi või graafiku, animatsiooni või diagrammi kujul. Nii antakse pildi abil sõnum edasi. Visualiseerimine muudab inimaju jaoks andmete mõistmise lihtsamaks, mistõttu on kergeem tuvastada seaduspärasusi ja tendentse. [9]

Kartogramm on kaart, millele on värviliselt või muul viisil trükitud teave. Kartogramm sobib sellele lehele suurepäraselt, sest peab kujutama koroonaviiruse leviku andmeid iga riigi kohta. Tänu sellele kartogrammile on lihtne saada üldist ettekujutust COVID-19 levikust maailmas.

Kartogrammi moodustamiseks oli kasutatud *React Simple Maps*. *React Simple Maps* kujutab endast komponentide komplekti, mida saab kombineerida, et luua markerite ja märkustega SVG-kaarte.[10]

React Simple Maps annab võimaluse kasutada HTML sildid Reactis, nagu: *ComposableMap*, *Geographies*, *Geography*, *Sphere*, *Graticule*. *ComposableMap* on põhisilt, mille vahel teised sildid peavad olema pakitud. *Geographies* ja *Geography* sildid võtavad statistika andmed vastu ja kannavad need värvide abil kaardile. *Sphere* ja *Graticule* sildid moodustavad võrgu kaardi tagaplaanil (Joonis 10).

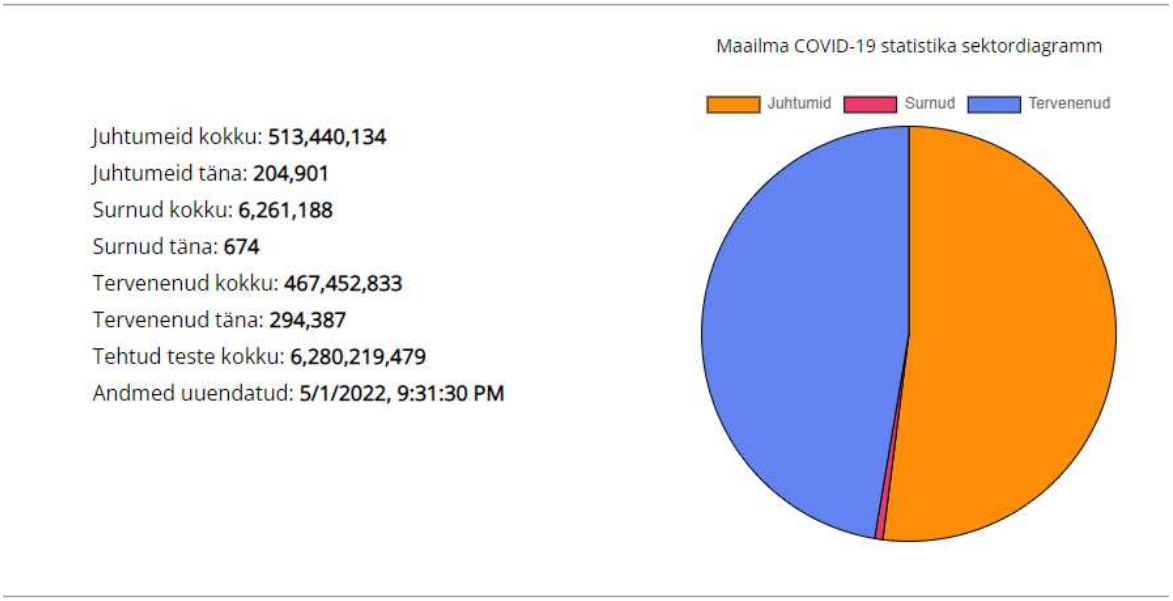
React Simple Maps annab võimaluse kasutada HTML sildid Reactis, nagu: *ComposableMap*, *Geographies*, *Geography*, *Sphere*, *Graticule*. *ComposableMap* on põhisilt, mille vahel teised sildid peavad olema pakitud. *Geographies* ja *Geography* sildid võtavad statistika andmed vastu ja kannavad need värvide abil kaardile. *Sphere* ja *Graticule* sildid moodustavad võrgu kaardi tagaplaanil (Joonis 10).



Joonis 10. Kartogramm Maailma lehel.

Kartogrammi all asub üldine maailma statistika (Joonis 11). Siin on juhtumite, surnud ja tervenenud inimeste arv ja sektordiagramm, mis näitab näitlikult seose nende kolme näitajate vahel.

Sektorgiagrammi ehitades oli kasutatud *Chart.js* [13]. *Chart.js* diagrammid renderdatakse kasutaja pakutavatel *canvas* elementides. *Canvas* on HTML element (<*canvas*>) mida kasutatakse veebilehele graafika joonistamiseks. *Canvas* elementi toetavad kõik brauserid ja seega *Chart.js* on kasulik ja lihtne kasutada oma veebirakendustes diagrammide loomisel.



Joonis 11. Üldine maailma statistika Maailma lehel.

Maailma lehe all asub suur tabel, kus asub iga maailma riigi üldine statistika. Riigid on järjestatud kahanevas järjekorras juhtumite arvu järgi (Joonis 12). Tabel sisaldab statistikat enam kui 190 riigist. Siin on välja arvutatud ka haigusjuhtude ja surmade arvu protsent kogu maailmast iga riigi kohta.

Maailma lehe näidiseid võib leida lisast (Lisa 2).

Koht	Riik	Riigikood	Juhtumite arv	% Maailmast	Surmade arv	% Maailmast
1	United States of America	US	81,349,065	15.886%	993,712	15.949%
2	India	IN	43,079,188	8.413%	523,843	8.408%
3	Brazil	BR	30,448,236	5.946%	663,736	10.653%
4	France	FR	28,835,895	5.631%	146,967	2.359%
5	Germany	DE	24,809,785	4.845%	135,461	2.174%
6	United Kingdom	GB	22,213,972	4.338%	175,552	2.818%

Joonis 12. Tabel Maailma lehel.

4.5 Konkreetse riigi statistika

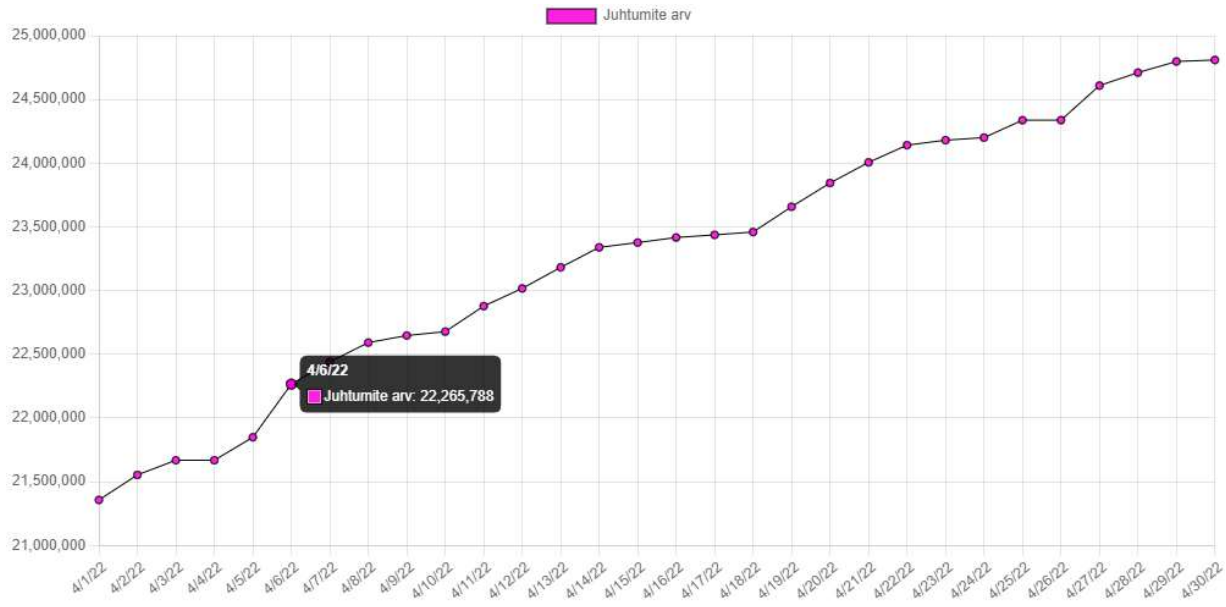
Sellel lehel asuvad põhjalikumad statistika andmed võrreldes Maailma lehega.

Esimesena asub lühike info sellest lehest ja sisendrida, kuhu kasutaja peab kirjutama ühe riigi nimi inglise keeles sisse (Joonis 13). Riigi nimi sisestamiseks võib kasutada ka *ISO 3166-1 alpha-2* koodi, näiteks võib sisestada "EE" ja ilmub lehel andmed seotud COVID-19 levikuga Eesti kohta. Pärast riigi nimi sisestamist ja "Otsi" nupu vajutamist ilmuvad valitud riigi kohta joondiagrammid ja statistika.

Joonis 13. Riigi lehel asuv sisendrida.

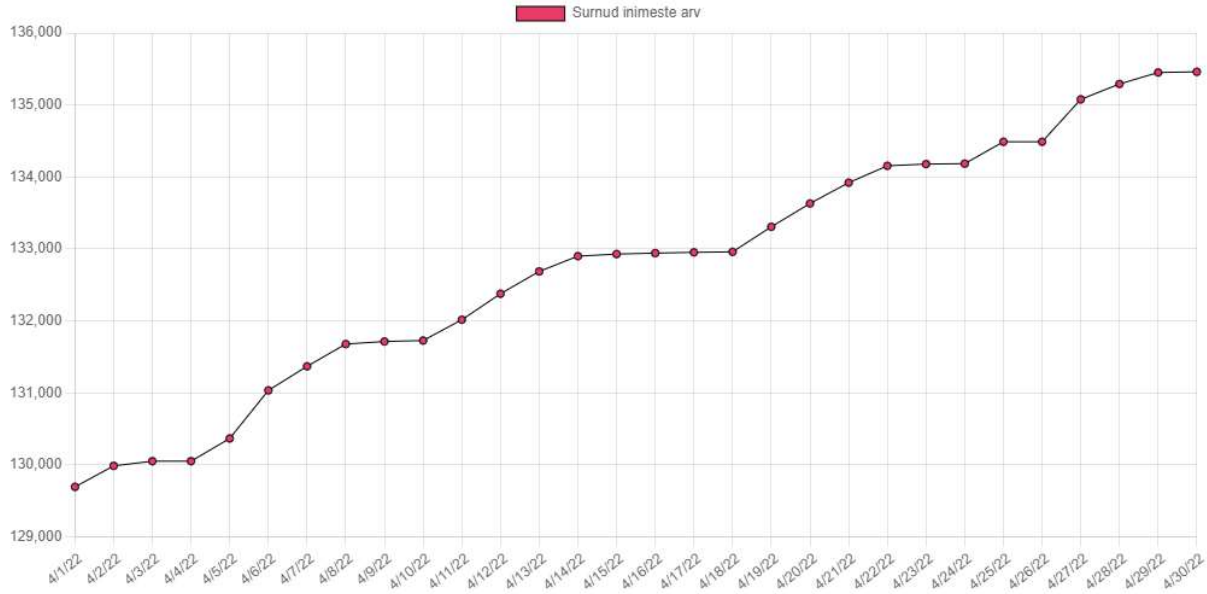
Joondiagramm juhtumite arvuga näitab kuidas konkreetses riigis juhtumite arv suureneb iga päevaga (Joonis 14). See diagramm oli ka tehtud *React.js* abil. Sellel graafikul asuvad viimase 30 päeva andmed.

Kuupäevad asuvad graafiku x-koordinaadil ja juhtumite hulk asub graafiku y-koordinaadil (Joonis 14). Lõputöö autor otsustas teha joondiagrammi ainult viimase 30 päeva põhjal, muidu osutub see liiga hõivatuks ja ühe konkreetse kuupäeva andmeid on raske jälgida.



Joonis 14. Joondiagramm Riik lehel Saksamaa COVID-19 juhtumite arvuga.

Joondiagrammi surmade andmetega loomiseks oli kasutatud sama tehnoloogia ja reeglid nagu eelmises diagrammis (Joonis 15).

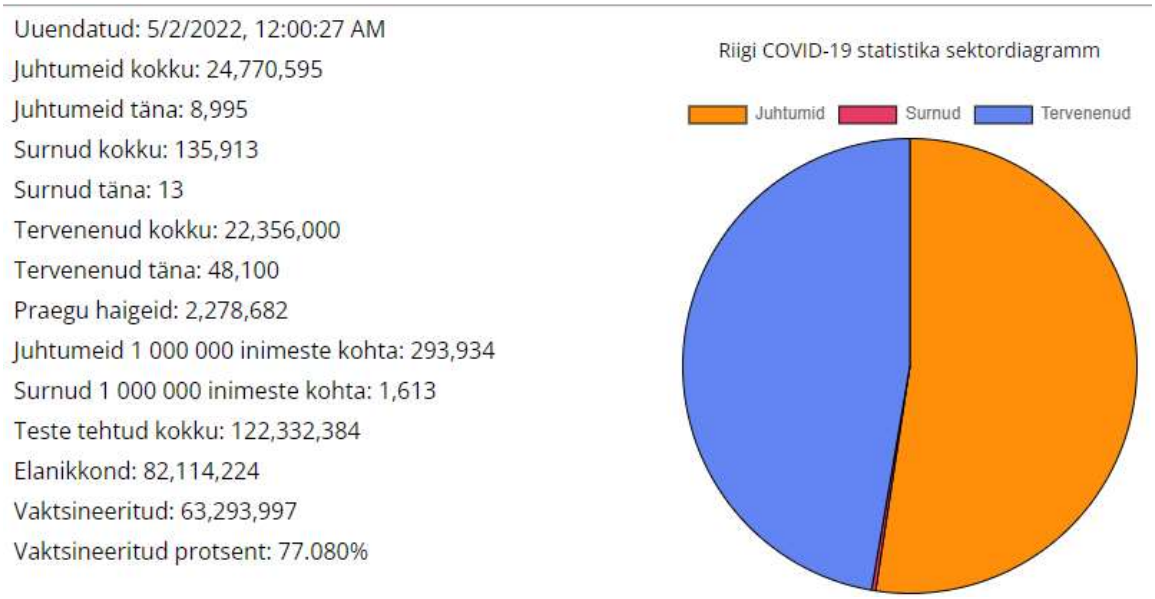


Joonis 15. Joondiagramm Riik lehel Saksamaa COVID-19 surma andmetega.

Joondiagrammide all asub statistikaga seotud blokk (Joonis 16). Siin on COVID-19 juhtumite,

surmade ja tervenenud inimeste arv täna ja kokku. Veel tulevad andmed nagu: praegu haige inimeste arv, tehtud teste arv kokku, elanikkond, vaktsineeeritud inimeste arv ja vaktsineeritud inimeste protsent elanikkonnast (Joonis 16).

Riigi lehe näidiseid võib leida lisast (Lisa 4).



Joonis 16. Saksamaa statistika Riik lehel.

4.6 Statistika võrdlemise leht

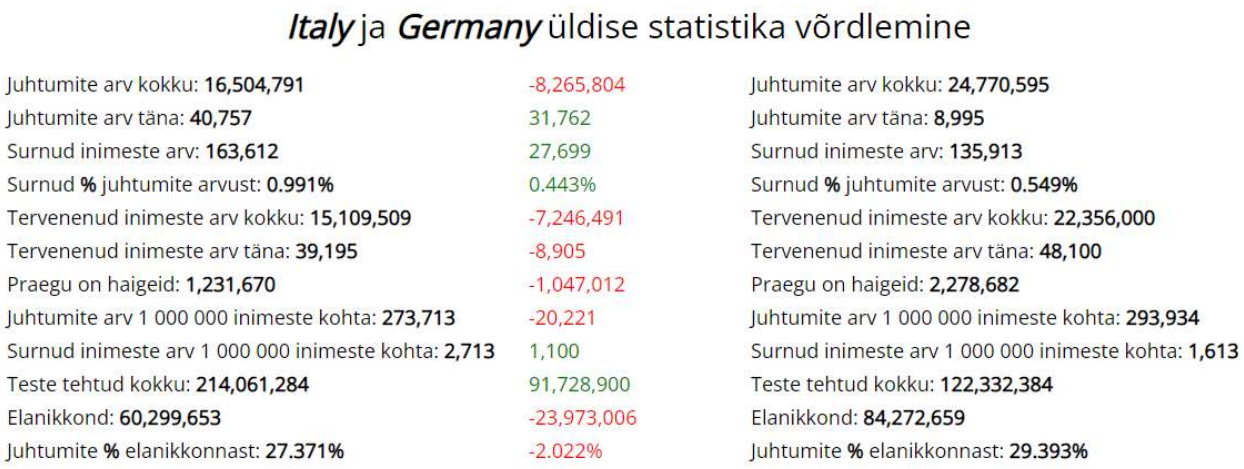
Sellel lehel asub kahe riigi statistika võrdlemine.

Nagu Riik lehel, Võrdle lehel ka esimesena asub lühike info sellest lehest ja sisendrida, kuhu kasutaja peab kirjutama kahe riigi nimed inglise keeles sisse (Joonis 17).

Joonis 17. Võrdle lehel asuv sisendrida.

Kui kahe riigi nimed on sisestatud ja “Otsi” nupp on vajutatud, ilmuvad nende statistika andmed (Joonis 18).

Kui kasutaja sisestab otsingureasse, näiteks “Italy Germany” või “IT DE”, kasutades *ISO 3166 alfa-2* standardi, ilmub Itaalia COVID-19 leviku andmed esimeses veerus ja Saksamaa statistilised andmed kolmandas veerus (Joonis 18).



Joonis 18. Itaalia ja Saksamaa COVID-19 leviku andmed Võrdle lehel.

Teises veerus, mis asub esimese ja kolmanda vahel, on paigatud arvud, mis on saadud esimese ja kolmanda veergude iga rida võrdlemisel. Arvude värv näitab, kas number esimesest veerust on suurem või väiksem kui sama number kolmandast veerust. Joonisel 18 on kujutatud see sõltuvus, näiteks: Itaalias on tervenenud inimeste arv kokku 7 246 491 vähem kui Saksamaal.

Võrdle lehe näidiseid võib leida lisast (Lisa 3).

5 Edasiarenduse võimalused

Käesoleva lõputöö raames sai valmis programmeeritud veebirakendus, aga see rakendus pole lõplik toode, mille edasiarendamiseks on palju teha. Selles peatükis tuuakse välja mõned võimalikud edasiarenduse suunad.

5.1 Laiendamine teistele keeltele

Praegu loodud veebirakendus on eesti keeles. Üks võimalik edasiarendus on veebirakenduse tõlkimine, näiteks vene ja inglise keelde. Tänu sellele oleks võimalik rakendust kasutada ka mitte-eestikeelsetel kasutajatel.

See täiendus tõstaks informeeritust COVID-19 levikust mitte ainult Eestis, vaid kogu maailmas

5.2 Kehtivate piirangute vaatamine

Kehtivate piirangute vaatamise võimalus oleks väga kasulik inimestele, kes reisivad palju või lihtalt tahavad teisi riike külastada.

Antud täienduse jaoks võiks luua uue lehe, kus on võimalik ühe riigi nimi sisestada ja pärast saada kõik praegu kehtivad piirangud selles riigis. Kethivate piirangute nimikirja punktid võivad olla järgmised:

- kas tohib riigi siseneda või mitte;
- kas koroonatest on kohustuslik riigi külastamisel või mitte;
- kas maskide kandmine on kohustuslik või mitte;
- õppeasutused on avatud või suletud;
- poed ja kaubanduskeskused töötavad või mitte;
- saunad ja ujulad on avatud või suletud;
- kas avalikud üritused ja koosolekud on lubatud või mitte.

5.3 Nõuanne leht

Kolmandaks veebirakenduse edasiarenduse võimaluseks on nõuanne lehe loomine. Sellel lehel peavad asuma erinevad nõuanded ja soovitused, kuidas end ja oma lähedaid inimesi kaitsta koroonaviiruse eest.

Veel lehel võivad asuda COVID-19 peamiste sümptomite nimikiri, info sellest, kuidas koroonaviirus levib ja kuidas saada vaktsineerima iga maailma riigis.

Kokkuvõte

Käesoleva bakalaureusetöö põhieesmärgiks oli luua veebirakendus, mis suudaks näidata koroonaviiruse haiguse levimise statistikat nii maailmas, kui ka detailsema statistikat ühe konkreetse maailma riigi kohta.

Lõputöös anti põhjalik ülevaade teema probleemist ning jätkati projekti eesmärkidega tutvumist.

Rakenduse vahendite valikul viidi läbi põhjalik analüüs mitme alternatiivse võimaluse vahel. Valiku tegemisel oli peamiselt keskendatud selle sobivusele antud eesmärk ära täita. Veebirakenduse loomiseks oli keeleks eelistatud võtta JavaScipt, sest JavaScript on mõnda aega olnud üks populaarsemaid veebiarenduskeeli ning seda keeli toetavad kõige populaarsemad veebilehitsejad. Raamistike seast osutus valikuks React.js. See raamistik sobib neile, kes ootavad suurt liiklust oma veebilehel ja vajavad selle haldamiseks kindlat platvormi.

Lõputöö käigus valminud veebirakendus aitab inimestel rohkem teada sada COVID-19 levikust maailmas, suureendab informeeritust probleemist ning aitab inimestel saada ettevaatlikumaks ja oma tervist rohkem hoida.

Töö tulemusena on valmis veebirakendus, mille külastamisel saab kasutaja vaadata detailsed COVID-19 leviku andmed ühe konkreetse riigi kohta, mis asuvad joondiagrammidel ja sektordiagrammidel. Veel kasutajad võivad võrrelda kahe riigi koroonaviiruse leviku üldiseid andmeid omavahel. Veebirakenduse kasutajad võivad ka vaadata üldise koroonaviiruse leviku statistikat maailmas, mis asub maailma kartogrammil, sektordiagrammil ja suures tabelis, mis sisaldab statistikat enam kui 190 riigist.

Veebirakenduse loomisel lõputöö autor üritas statistilisi andmeid maksimaalset visualiseerida. Andmete visualiseerimine on viis suurte andmete hulga selgelt ja kiireks esitamiseks ning visualiseerimine muudab inimaju jaoks andmete mõistmise lihtsamaks, mistõttu on kergem tuvastada seaduspärasusi ja tendentse.

Loodud veebirakendusel samuti on ka edasi arendamise võimalusi, mis viiksid lahenduse teistest

alternatiivsetest lahendustest veelgi rohkem ette. Peamiseks edasiarenduseks on kehtivate piirangute vaatamine. Selline võimalus oleks väga kasulik inimestele, kes reisivad palju või lihtalt tahavad teisi riike külastada. Samuti ka nõuanne leht, mis näitab koroonaviiruse sümptomite nimikiri ning erinevad koroonaviiruse eest kaitsmise nõuanded ja soovitused.

Kasutatud kirjandus

[1] Maya Ystinova, " *Простыми словами о «фронтенде» и «бэкенде»: что это такое и как они взаимодействуют* " 13. aprill 2017 [Võrgumaterjal]. Saadaval: https://tproger.ru/translations/frontend-backend-interaction/. [Kasutatud 22. aprill 2022].

[2] "About JavaScript" [Võrgumaterjal]. Saadaval: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript. [Kasutatus 22. aprill 2022].

[3] Yuliia Fedyk, "The Best Web Application Development Languages in 2022" Saadaval: https://inveritasoft.com/blog/the-best-web-application-development-languages. [Kasutatud 22. aprill 2022].

[4] Dsa Suraj Surve, "Why You Should Use React.js For Web Development" 18 veebruar 2021 [Võrgumaterjal]. Saadaval: https://www.freecodecamp.org/news/why-use-react-for-web-development/. [Kasutatud 22. aprill 2022].

[5] "List of 10 Best Front end Frameworks to Use For Web Development" 24. november 2021 [Võrgumaterjal]. Saadaval: https://www.monocubed.com/blog/best-front-end-frameworks/.
[Kasutatus 22. Aprill 2022].

[6] "Why did we build Visual Studio Code?" 30. märts 2022 [Võrgumaterjal]. Saadaval: https://code.visualstudio.com/docs/editor/whyvscode. [Kasutatud 23. aprill 2022]

[7] " *Навигация сайта?*" 7. oktoober 2017 [Võrgumaterjal]. Saadaval: https://semantica.in/blog/navigacziya-sajta.html [Kasutatud 24.aprill 2022]

[8] "Introduction" [Võrgumaterjal]. Saadaval: https://reactrouter.com/docs/en/v6/getting-started/tutorial [Kasutatud 24.aprill 2022]

[9] "*Преимущества визуализации данных*" 4. Detsember 2019 [Võrgumaterjal]. Saadaval: https://sukhari.com.ua/vizualizatsiya-dannyih-komu-zachem-i-pochemu.html [Kasutatud 24. aprill 2022]

[10] "Usage" [Võrgumaterjal]. Saadaval: https://github.com/zcreativelabs/react-simple-maps. [Kasutatud 24. aprill 2022]

[11] “What is API: Definition, Types, Specifications, Documentation” 28. Juuli 2021 [Võrgumaterjal]. Saadaval: https://www.altexsoft.com/blog/engineering/what-is-api-definition-types-specifications-documentation/. [Kasutatud 25. aprill 2022].

[12] “Using the Fetch API” [Võrgumaterjal]. Saadaval: https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch. [Kasutatud 25. aprill 2022]

[13] “Chart.js” [Võrgumaterjal]. Saadaval: https://www.chartjs.org/docs/latest/. [Kasutatud 10. mai 2022]

[14] “disease.sh - Open Disease Data API” [Võrgumaterjal]. Saadaval: https://disease.sh/. [Kasutatud 10. mai 2022]

[15] “COVID 19 API” [Võrgumaterjal]. Saadaval: https://covid19api.com/. [Kasutatud 10. mai 2022]

[16] “Covid-19-API” [Võrgumaterjal]. Saadaval: https://github.com/M-Media-Group/Covid-19-API. [Kasutatud 10. mai 2022]

[17] “*Application Programming Interface (API)”* 19. august 2020 [Võrgumaterjal]. Saadaval: https://www.ibm.com/cloud/learn/api. [Kasutatud 11. mai 2022]

[18] “Tartu Ülikooli teadlaste arendatud koroonaviiruse analüüsivahendid” [Võrgumaterjal]. Saadaval: https://koroona.ut.ee/. [Kasutataud 12. mai 2022]

[19] “Koroonaviiruse andmestik” [Võrgumaterjal]. Saadaval: https://www.terviseamet.ee/et/koroonaviirus/koroonaviiruse-andmestik. [Kasutatud 12. mai 2022]

Lisa 1 - Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

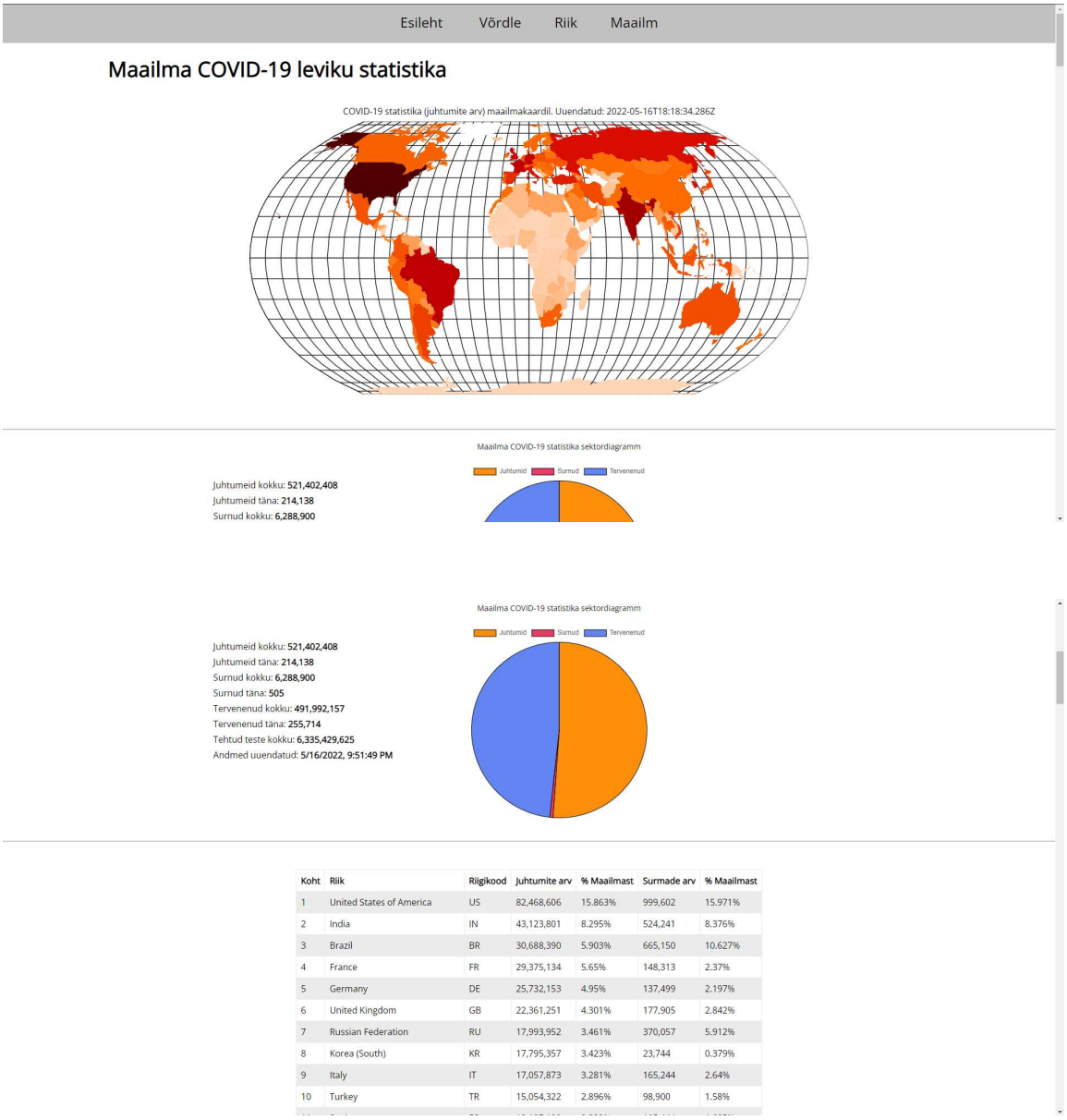
Mina, Vjatšeslav Koloskov

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose " Veebirakenduse arendus COVID-19 statistiliste andmete vaatamiseks", mille juhendaja on German Mumma
   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. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

16.05.2022

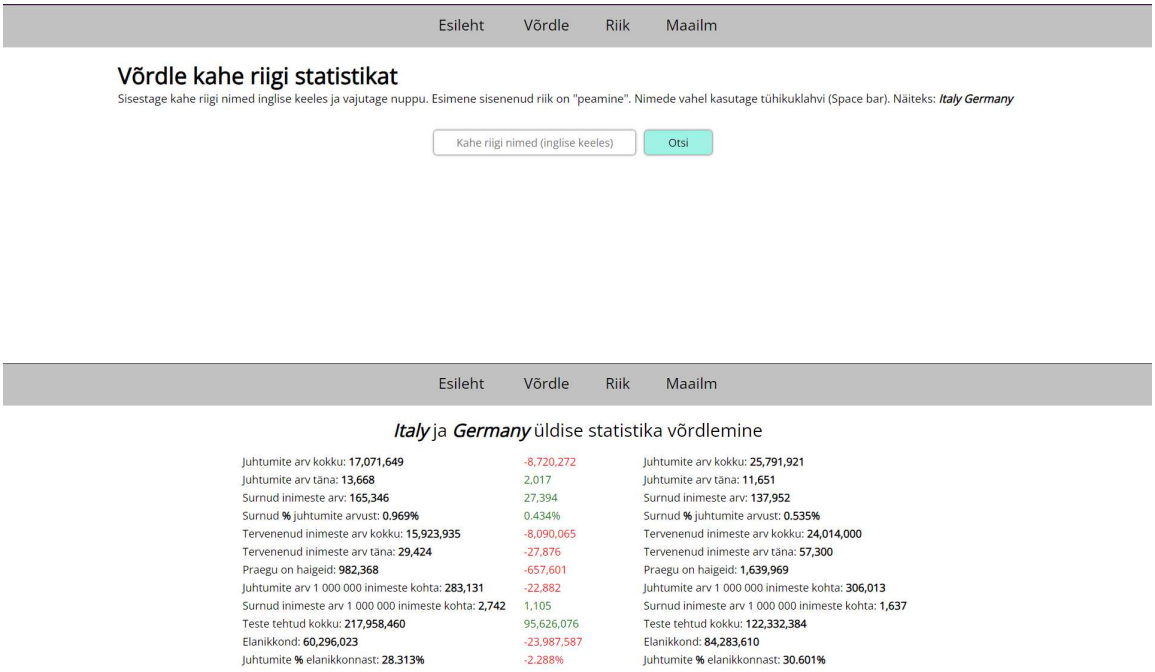
[1] Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või 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 - Maailma lehe näide



Koht	Riik	Riigikood	Juhtumite arv	% Maailmast	Surmade arv	% Maailmast
1	United States of America	US	82,468,606	15.863%	999,602	15.971%
2	India	IN	43,123,801	8.295%	524,241	8.376%
3	Brazil	BR	30,688,390	5.903%	665,150	10.627%
4	France	FR	29,375,134	5.65%	148,313	2.37%
5	Germany	DE	25,732,153	4.95%	137,499	2.197%
6	United Kingdom	GB	22,361,251	4.301%	177,905	2.842%
7	Russian Federation	RU	17,993,952	3.461%	370,057	5.912%
8	Korea (South)	KR	17,795,357	3.423%	23,744	0.379%
9	Italy	IT	17,057,873	3.281%	165,244	2.64%
10	Turkey	TR	15,054,322	2.896%	98,900	1.58%

Lisa 3 - Võrdle lehe näide



Lisa 4 - Riik lehe näide