Tartu Ülikool
Arvutiteaduse instituut
Informaatika õppekava

Mattias Sokk

Veebirakenduse loomine informaatikaviktoriinile Kobras

Bakalaureusetöö (9 EAP)

Juhendaja: Lidia Feklistova, MSc

Tartu 2022

Veebirakenduse loomine informaatikaviktoriinile Kobras

Lühikokkuvõte: Informaatika- ja arvutialaseid oskusi on nüüdisühiskonnas üha rohkem tarvis. Alates 2006. aastast korraldatakse Eestis iga-aastaselt informaatikaviktoriini Kobras, mille eesmärk on õpilastes tekitada huvi informaatika vastu. Viktoriini tarbeks koostatakse aastas 90 erinevat ülesannet, mis on saadaval nii eesti kui vene keeles. Väljaspool viktoriini ei leia valminud ülesanded nii palju kasutust kui võiksid, sest senisel viktoriini veebilehel on raske varasemaid ülesandeid otsida ja puudub võimalus nende interaktiivseks lahendamiseks. Bakalaureusetööna valmis lahendus, mis kasutab pideva integratsiooni ja tarne võimalusi, et luua staatiline veebirakendus, kus saab andmeid mugavalt filtreerida ning varasemaid ülesandeid interaktiivselt lahendada. Valminud lahendus on automaatne ja kaotab viktoriini korraldajatelt vajaduse veebilehte käsitsi uuendada. Töö autor loodab, et töö tulemusena jõuavad huvitavad informaatikaülesanded rohkemate noorteni ning nende huvi ja teadmised informaatika valdkonnas kasvavad.

Võtmesõnad: Informaatikaviktoriin Kobras, staatiline veebirakendus, pidev integratsioon ja tarne, veebiarendus, JavaScript, Vue.js

CERCS: P175 Informaatika, süsteemiteooria

Creating a web application for challenge on informatics Kobras

Abstract: Informatics and computer skills are increasingly needed in today's society. In Estonia since 2006, a challenge on informatics called "Kobras" is held annually. The mission of Kobras is to introduce and promote informatics among pupils. The challenge is held in two languages and each set needs approximately 90 different questions to be composed for each year. Pupils and teachers could use the previous questions more often after the challenge, but there is no way to filter the questions and solve them interactively on the current website. This thesis describes a technical solution that was implemented to overcome these problems. The solution uses continuous integration and delivery capabilities to create a static web application that allows users to filter data and solve previous questions. The author of this thesis hopes that interesting informatics questions will reach more young people, and their interest and knowledge towards informatics will increase.

Keywords: Informatics challenge Kobras, static web application, continuous integration and deployment, web development, JavaScript, Vue.js

CERCS: P175 Informatics, systems theory

Sisukord

Sissejuhatus

Tänapäeval on IKT-sektoris pidev töötajate puudus ja tööandjad üritavad värvata endale uusi töötajaid pakkudes neile Eesti keskmisest tunduvalt kõrgemaid töötasusid ning üha paremaid töötingimusi. Eesti Ettevõtluse Arendamise Sihtasutuse 2019. aasta raporti andmetel kasvab IKT-sektor ligikaudu seitse korda kiiremini kui ülejäänud majandusvaldkonnad [1]. Seega on spetsialistide nappus lähitulevikus tõenäoliselt veelgi enam süvenev probleem. IKT-sektori töötajate kasvava nõudluse rahuldamiseks tutvustatakse antud valdkonda üha enam juba üldhariduskoolide õppekavades ja äratatakse õpilastes huvi IKT-valdkonna teemalisi võistlusi korraldades. Üks sellistest võistlustest on informaatikaviktoriin Kobras, mis toimus Eestis esmakordselt juba 2006. aastal.

Eestis peetava viktoriini korraldajad on loonud viktoriini tutvustamiseks ning varasemate ülesannete ja tulemuste kuvamiseks eraldi veebilehe [2]. Lehe ülesehitus ei võimalda aga mugavalt varasemalt loodud ülesandeid otsida, puudub varasemate ülesannete interaktiivne lahendamise võimalus ning kasutajakogemuse disaini saaks parendada. Ühtlasi on korraldajate sõnul praeguse veebilehe sisu uuendamine tülikas tegevus, sest uusi ülesandeid ja varasemaid tulemusi peab korduvalt mitmes kohas käsitsi uuendama ning ümber kirjutama.

Bakalaureusetöö eesmärk on luua viktoriinile uus veebirakendus, mis võimaldab külastajatel lugeda informatsiooni viktoriini kohta, vaadata ja lahendada varasemaid ülesandeid ning näha varasemate viktoriinide tulemusi. Uue rakenduse ülesehitus peab olema selge, pakkuma lehe külastajatele paremat kasutuskogemust ning viktoriini korraldajate jaoks võimaldama hõlpsamat lehe muutmist ja haldamist.

Bakalaureusetöö esimeses osas tutvustatakse informaatikaviktoriini Kobras ja selle praegust korraldust. Teises osas tutvustatakse uue veebirakenduse loomise protsessi – seatud nõudeid uuele rakendusele, uue lahenduse arhitektuuri ning antakse ülevaade kasutatud tehnoloogiatest. Kolmandas osas analüüsitakse valminud lahendust – tutvustatakse lõppkasutajatelt saadud tagasisidet ning kontrollitakse, kas lõpptulemus vastab seatud nõuetele. Neljandas osas on esitatud ettepanekud võimalikeks edasiarendusteks.

1. Informaatikaviktoriin Kobras

Eestis peetav viktoriin Kobras on osa rahvusvaheliselt toimuvast informaatikaviktoriinist Bebras (“kobras” leedu keeles). Bebras ülesannete koostamisesse panustavad ühiselt viktoriinisarjaga seotud asutused 53 erinevast riigist ning riiklikel viktoriinidel tõlgitakse koostatud ülesanded kohalikku keelde [3]. Ülesanded Bebras viktoriinidel (sh Kobras) vastavad üldiselt kindlatele kriteeriumitele, kus koostatud ülesanded peavad olema [4]:

1) seotud informaatikaga,
2) ülevaatlikud ja lihtsasti mõistetavad,
3) mitte väga mahukad (lahendatavad kolme minutiga),
4) lühidalt ja lihtsalt esitatud,
5) lahendatavad ilma spetsiifilise tarkvara, paberi ja kirjutusvahendita,
6) sõltumatud spetsiifilistest süsteemidest,
7) huvitavad ja/või naljakad.

Viktoriinis käsitletakse üldjuhul noortele huvitavasse konteksti pandud ülesandeid, mis on otseselt informaatikaga seotud ja äratavad õpilaste huvi informaatika vastu.

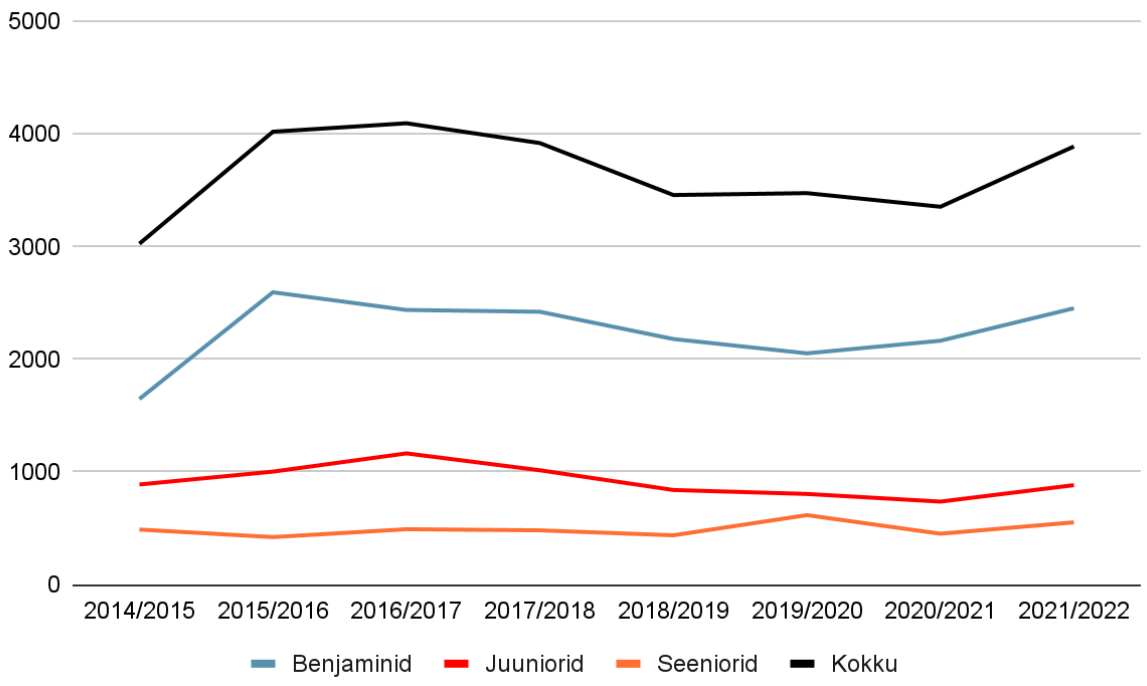
1.1. Viktoriini taust

Rahvusvahelist viktoriinisarja Bebras korraldab Eestis iga aastaselt Tartu Ülikool ning Eestis korraldatava viktoriini nimi on Kobras [3]. Kobras on suunatud kõigile 6. kuni 12. klassi õpilastele [2]. Viktoriini korraldatakse kolmes vanuserühmas – benjaminid (6.-8. klass), juuniorid (9.-10. klass) ja seeniorid (11.-12. klass) ning toimub kahes voorus. Esimene voor toimub internetipõhiselt Teaduskooli viktoriinikeskkonnas ning selle parimad osalejad kutsutakse teise vooru ja IT-teemalisele huvipäevale Tartusse. Koroonaviiruse leviku tõttu oli kahel viimasel aastal viktoriini teine voor avatud veebis kõikidele huvilistele ning huvipäev jäeti ära.

Tartu Ülikooli Teaduskooli andmetel küsitakse viktoriinil osalejatelt informaatikaalaseid küsimusi, mis ei eelda otseseid programmeerimisoskusi [5]. Ülesanded võivad olla arvutialase loogika, arvutus- ja sidetehnika ajaloo, informatsiooni mõistmise ja tõlgendamise ning muu taolise kohta [2]. Enamasti on ülesannete juurde lisatud teksti selgitavad või illustreerivad pildid. Viktoriinile omaselt arvestatakse hindamisel ainult vastust ning lahenduskäiku ei ole vaja esitada. Ülesande vastus on esitatav enamasti valikvastuste seast valimise, täisarvu või lühikese teksti kirjutamisel. Varasemate aastate viktoriinide ülesandeid

ja vastuseid vaadates esineb ka selliseid ülesandeid, kus mõnel on mitu õiget vastusevarianti ning leidub ka ülesandeid, mille puhul oodatakse vastavusse seadmist, lohistamist vms. Korraldajate sõnul on varasemate voorude vastusetüüpide rohkus tingitud sellest, et enne 2020. aastat oli viktoriini läbiviimiseks kasutusel õpikeskkond Moodle, mis pakkus küsimustele rohkem võimalikke vastusetüüpe kui praegu kasutusel olev Teaduskooli viktoriinikeskkond.

Viktoriini veebilehel on kirjas, et Eestis on viktoriini korraldatud alates 2006. aastast ning lehel on andmed osavõtmise aktiivsuse kohta saadaval alates 2014. aastast [2]. Varasemaid tulemusi vaadates selgub, et viimase kuue aasta jooksul on kogu osalejate arv olnud püsivalt vahemikus 3000 kuni 4000 õpilast (vt joonis 1) ning kõige enim osalejaid on olnud benjaminide vanuserühmas. Veebilehel avaldatud esimese vooru tulemustest selgub, et teise vooru ja infopäevale pääses koroonaviirusele eelnenud ajal iga-aastaselt ligikaudu 65 osalejat.



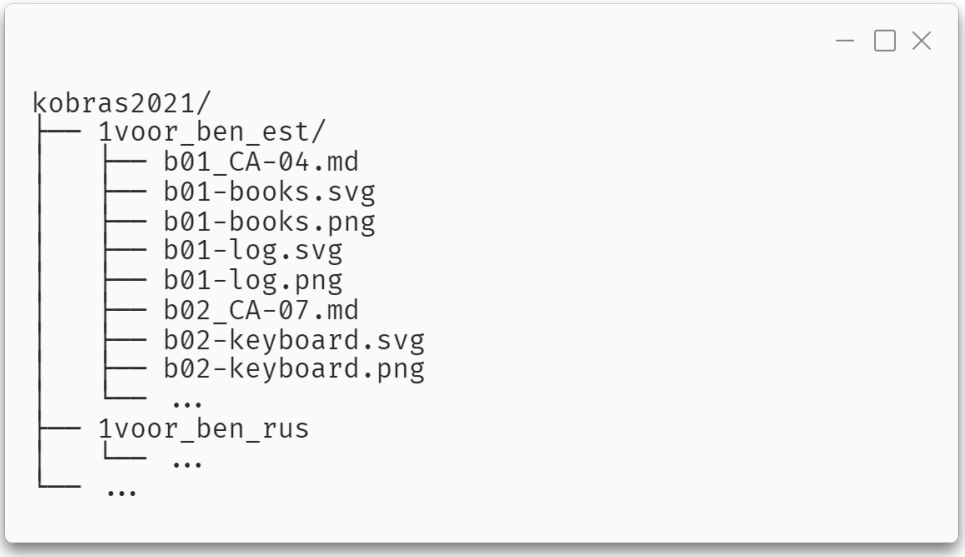
Joonis 1. Viktoriini Kobras erinevate vanusegruppide esimese vooru osavõtuaktiivsus aastatel 2014-2021 [2].

1.2. Ülesannete haldus

Rahvusvahelise organisatsiooni Bebras info kohaselt tegelevad kohaliku viktoriini iga-aastase korraldamisega inimesed Tartu Ülikoolist [3]. Alates 2017/18. õppeaastast on Eestis uute viktoriini ülesannete loomiseks ja teistest riikidest pärit ülesannete tõlkimiseks kasutatud Eesti Keeleressursside Keskuse majutatud GitLabi repositooriumit. Nende aastate jooksul on viktoriiniülesannete valmimisse panustanud kokku kuus liiget. Repositooriumi muutmisajaloo järgi on neist aktiivsemad olnud vaid kaks. Korraldajate sõnul valiti ülesannete loomiseks just GitLabi repositoorium, sest see võimaldab uusi ülesandeid ette valmistada meeskonnatööna, säilitab seejuures kõikide muudatuste ajaloo ning lisaks ülesannetele saab repositooriumis hoida kõiki pildifaile ning muid abistavaid lisafaile.

Igas viktoriinivoorus esitatakse igale vanuserühmale 15 ülesannet. Kuna viktoriini korraldatakse kolmes vanuserühmas, valmistavad korraldajad igaks vooruks ette ligikaudu 45 ülesannet. Leidub aga ka ülesandeid, mida esitatakse ühes voorus mitmele vanuserühmale. See tähendab, et ühe õppeaasta jooksul tekib juurde suurusjärgus 90 ülesannet. Kõik need on saadaval nii eesti keeles kui ka vene keelde tõlgituna. See tähendab omakorda, et ühe õppeaasta viktoriini repositooriumisse lisatakse kokku umbes 180 viktoriini ülesannet sisaldavat faili. Lisaks asuvad repositooriumis iga ülesande kohta ka kõik sellega seotud pildifailid ning tihtipeale ka pildifailide lähtefailid (lihtsamini muudetavas formaadis) ja muud lisafailid.

Paljude erinevate failide hõlpsamaks haldamiseks on 2017/18. aastast viktoriini korraldajatel paigas kindel süsteem, kuidas uusi ülesandeid luuakse ning nendega seotud faile repositooriumisse paigutatakse. Eesti Keeleressursside Keskuse majutatud GitLabis on loodud viktoriini jaoks eraldi grupp “Kobras” ning iga aasta luuakse grupi liikmetele uus repositoorium, kus vastava aasta faile hallatakse. Ülesanded paigutatakse voorude, vanuserühmade ja keele järgi eraldi kaustadesse (vt joonis 2). Igasse sellisesse kausta paigutatakse ülesanded failinimega, kus vanuserühma tähistab eesliide (vastavalt “b” – benjamin, “j” – juunior, “s” – seenior), sellele järgneb ülesande number (nt “01”) ning päritoluriigi kood (nt “CA” – Kanada). Ülesanded on paigutatud *.md* laiendiga Markdown failidesse. Kõik ülesandega seotud pildifailid asuvad samas kaustas ning algavad sama eesliidesega (nt “b01-”). Failinimedes ei kajastu ülesande keel ning seega vastab nii eesti- kui venekeelsete ülesannete struktuur samale süsteemile.



Joonis 2. Viktoriini repositooriumi struktuur.

Ülesannete tekstide vormistamiseks kasutatakse viktoriini GitLabi repositooriumides Markdown vormingut. Markdown on lihtteksti süntaks, mis võimaldab lähteteksti mugavalt lugeda ja kirjutada nii lihttekstina kui teisendada seda automaatselt veebis kuvatavasse HTML (hüperteksti märgistuskeel, ingl *HyperText Markup Language*) vormingusse [6]. Markdown failide muutmine ja vaatamine on GitLabi Giti-põhisesse versioonihaldus-teenusesse sisse ehitatud ning seega on selles formaadis faile lihtne kirjutada, sest saab kohe näha veebis kuvatavat vormindatud tulemust [7].

Ülesannete Markdown failid on vormistatud kindla struktuuriga – ülesanne koosneb alati pealkirjast, kirjeldusest, küsimusest ning vastusest. Viimastel aastatel on lisatud enamikele ülesannetele veel vastuse selgitus ja informaatikaga seotust kirjeldav tekstiplokk (vt joonis 3). Tulevikus on viktoriini korraldajatel plaanis kõikidele ülesannetele juurde lisada ka nende kategooria.

```
# 11. Linnad

Kaarel teeb geograafiatunniks referaati ja tahab sinna lisada joonise Eesti linnade
rahvaarvude kohta. Ta kaalub joonisele mitut kujundust, aga tegi ühe variandi juures vea.

## Küsimus

Milline järgmistest joonistest **EI ESITA** ülejäänutega samu andmeid?

[Raadionupud]

A. ![](b11-a.png)
B. ![](b11-b.png)

C. ![](b11-c.png)
D. ![](b11-d.png)

## Vastus

Õige vastus on: D.

## Vastuse selgitus

Kõigil teistel joonistel on Tartu rahvaarv Narva omast suurem (mis on ka faktiliselt õige),
aga joonisel D on see vastupidi.

## See on informaatika

Sama informatsiooni on sageli võimalik kujutada mitmel erineval moel. Aga isegi kui andmetes
endis vigu ei ole, on mõned esitused paremini aru saadavad kui teised. Näiteks rahvaarvu järgi
järjestatud jooniselt A on kohe näha, et Narva on Eesti suuruselt kolmas linn, samas nimede
järgi järjestatud jooniselt B on selle väljalugemine märksa tülikam.

Muidugi, kui joonise eesmärk ongi näidata mingi suuruse muutumist ajas, nagu allpool on tehtud
Tartu rahvaarvuga erinevatel aastatel, on just tulpade järjestamine nende kõrguse järgi vähem
ülevaatlik.

![](b11-x1.png) ![](b11-x2.png)

Näidetes kasutatud andmed pärinevad [Wikipediast]
(https://et.wikipedia.org/wiki/Eesti_linnad#Eesti_linnade_loend).


## Kategooriad

- Informatsiooni mõistmine
```

Joonis 3. Näide ülesande vormistusest Markdown vorminguga (2020/21. aasta esimese vooru benjaminide vanuserühma 11. ülesanne).

Esimese vooru puhul esitatakse valminud viktoriinid õpilastele Tartu Ülikooli Teaduskooli viktoriinikeskkonnas https://viktoriinid.ee. Seal peavad õpilased viktoriini lahendamiseks konto registreerima või olemasoleva kontoga sisse logima, et viktoriinist osa võtta. Peale osalemist ja viktoriini lõpptähtaega saavad nad keskkonnas oma tulemusi vaadata. Teaduskooli viktoriinikeskkond on kasutusel 2020. aastast ja seega on seal saadaval ainult viimaste aastate ülesanded.

Peale viktoriinivooru toimumist genereeritakse käsitsi kõikidest ülesannete Markdown failidest vastavad HTML failid, paigutatakse need vanuserühma alusel ühistesse failidesse ning laetakse seejärel viktoriini veebilehele. Sarnaselt toimitakse ka tulemustega – CSV-formaadis õpilaste tulemused teisendatakse HTML-tabeliks ning need laetakse vanuserühmiti veebilehele üles. Niimoodi toimides on iga viktoriinivooru iga vanuserühma ülesanded ja tulemused kättesaadavad eraldi alamlehel.

1.3. Probleemid praeguse lahendusega

Iga viktoriini vooru jaoks koostatakse nii eesti kui vene keeles kokku ligikaudu 90 ülesannet. Seni on Markdowni formaadis informaatika alaseid ülesandeid kogunenud ligikaudu 450, nende seas on praegusel veebilehel aga raske orienteeruda. Ülesanded on veebilehel küll kõik kättesaadavad, kuid on grupeeritud aasta, vooru ja vanuserühma alusel 15 ülesandest koosnevatesse komplektidesse (vt joonis 4). Vastustega ja vastusteta ülesanded on paigutatud alamlehtedele eraldi ning seega on raske ülesandeid lehel lahendada. Samuti puudub lehel üldine struktuur ning navigeerimisel peab kasutama veebibrauseri "tagasi" nuppu.

Joonis 4. Ekraanitõmmis Informaatikaviktoriin Kobras praegusest veebilehest.

Peale lõppkasutajate pakub praegune lahendus probleeme ka viktoriini korraldajatele. Nende sõnul on veebilehe käsitsi uuendamine tüütu protsess, sest iga vooru vanuserühma kohta peab Markdown failidest genereerima HTML failid ja need käsitsi veebiserverisse kuvamiseks üles laadima. Praegu kasutatakse selleks lihtsat käsujada, mis kasutab Markdownist HTMLi genereerimiseks Pandoci konvertimise teeki. Viktoriini repositooriumeid uurides selgub, et varasemalt on väiksemaid parandusi Markdowni ja HTMLi failides tehtud ka käsitsi.

2. Uue veebirakenduse loomine

Bakalaureusetöö käigus loodi veebirakendus, mis võimaldab tutvuda viktoriini viimase üldinfoga, sirvida ja filtreerida varasemaid ülesandeid ning vaadata juba toimunud viktoriinivoorude tulemusi. Selles peatükis kirjeldatakse uue veebirakenduse loomise üksikasju – nõuete seadmist rakendusele, rakenduse arhitektuuri valimist ja arendamise käiku.

2.1. Nõuded rakendusele

Senise viktoriini tehnilise lahenduse analüüsimisel ilmnes mitu probleemi, millele oli vaja leida lahendus. Selleks, et uus lahendus vastaks nii lõppkasutajate (õpetajate ja õpilaste) kui korraldajate ootustele, viis autor läbi uue rakenduse nõudeid kaardistava diskussiooni selle korraldajatega. Korraldajad olid saanud juba õpetajatelt asjakohast tagasisidet ning olid teadlikumad, mida uus veebirakendus võiks võimaldada. Koostöös korraldajatega seati uuele rakendusele nii funktsionaalsed kui mittefunktsionaalsed nõuded, et uue rakenduse arendamine oleks selgemini piiritletud.

2.1.1. Funktsionaalsed nõuded

Funktsionaalsed nõuded kirjeldavad kõiki funktsionaalsusi, mida rakendus peab suutma teha. Kogu rakenduse töö taandub funktsionaalsete nõuete täitmisele [8:41]. Käesoleva töö raames valmiva rakenduse funktsionaalsed nõuded on järgnevad.

Rakendus peab võimaldama:

1) veebilehel ülesannete automaatset uuendamist Markdowni lähtefailide põhjal, et korraldajad ei peaks enam käsitsi veebilehe jaoks HTML faile genereerima ning neid veebiserverisse tõstma;
2) viktoriini üldinfo lihtsat muutmist;
3) ülesannete filtreerimist aasta, vooru, vanuserühma, pealkirja, riigi ja kategooria alusel, et oleks võimalik üles leida huvipakkuvaid ülesandeid;
4) igat ülesannet lahendada ja õiget vastust kuvada, et õpilased saaksid lehel valida endale huvipakkuvad ülesanded ning neid lahendada;
5) iga ülesande puhul vaadata selle vastuse selgitust ning ülesande seotust informaatikaga (kui sellised tekstiplokid lähtefailis olemas on);
6) kuvada kõiki filtreeritud ülesandeid õigete vastustega, et peale toimunud viktoriinivooru saaks õpilane tutvuda õigete vastustega;

7) võimaldama vaadata tulemusi ning võimaldama tulemuste filtreerimist aasta, vooru ja vanuserühma põhjal;
8) tulemuste filtreerimist kooli ja osaleja nime põhjal, et õpetajad saaksid näha kõiki oma kooli õpilaste tulemusi ning õpilased otsida enda tulemust.

2.1.2. Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded kirjeldavad tunnuseid või väärtusi, mida rakendus peab pakkuma. Need toetavad tihtipeale funktsionaalseid nõudeid ning käsitlevad näiteks rakenduse kasutatavust, kvaliteeti, kiirust, kulusid jpm. [8:43] Loodava rakenduse mittefunktsionaalsed nõuded on järgnevad:

1) rakenduse kasutamine peab olema lõppkasutajale võimalikult lihtne ja intuitiivne;
2) rakendus peab võimaldama sisu sirvida nii eesti kui vene keeles;
3) viktoriinide ülesanded ei tohi olla kättesaadavad enne viktoriini toimumist ning uues veebirakenduses peavad saama muuta infot ainult korraldamisega tegelevad inimesed;
4) uue veebirakenduse haldus- ja majutuskulud ei tohi ületada senise veebirakenduse halduskulusid;
5) rakendus peab olema kirjutatud vabavaralisele tarkvarale, millel on piisavalt suur kasutajaskond ka lähitulevikus ning seega võimalik leida edaspidist tuge;
6) rakenduse funktsionaalsused peavad toimima probleemideta kõikide järgnevate brauseritega alates versioonidest: Chrome 100.0.4896, Firefox 98.0.2, Safari (Mac) 15.0, Safari (iOS) 15.4 ja Edge 99.0.1150.30;
7) rakenduse funktsionaalsused peavad toimima probleemideta erineva kuvasuhte ja resolutsiooniga seadmetel – lauaarvutitel, tahvelarvutitel ja nutitelefonidel;
8) rakenduse esialgne laadimisaeg ning alamlehtede vahetamise ja andmete filtreerimise järel laadimiseks kuluv aeg ei tohi ületada 3 sekundit.

2.2. Uue rakenduse ülesehitus

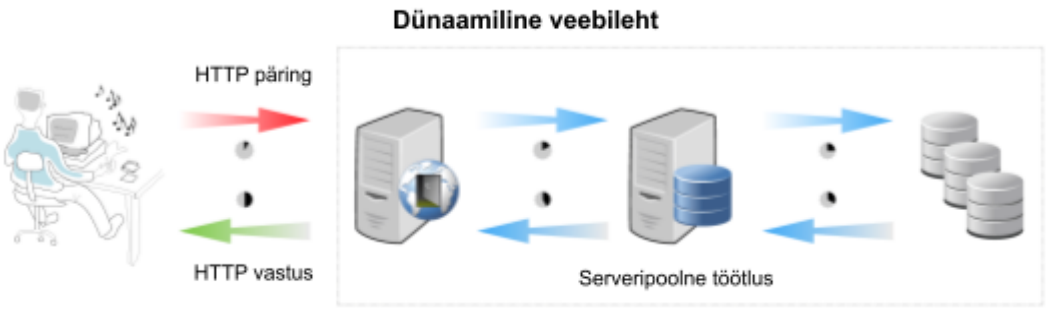
Enne uue rakenduse ise ehitamist on mõistlik tutvuda, kas leidub lahendus, mis vastab eelnevalt kirjeldatud nõuetele ja suudab lahendada varasemalt kirjeldatud probleemid. Sobiva valmislahenduse puudumisel tuleb tutvuda veebirakenduste üldvõimalustega ning panna paika loodava rakenduse arhitektuur.

2.2.1. Olemasolevad võimalused

Praegu eksisteerib juba palju rakendusi viktoriinide korraldamiseks, kuid enamasti ei võimalda need varasemate viktoriinivoorude ülesandeid ükshaaval kuvada, filtreerida ning oma veebilehele paigutada. Viktoriine võimaldavad luua rakendused nagu SurveyMonkey, TypeForm, FlexiQuiz jpt. Selliste rakenduste puhul puudub aga üldiselt võimalus kõikide viktoriinivoorude varasemaid ülesandeid korraga vaadata ning filtreerida neist välja huvipakkuvaid. Ühtlasi vajavad paljud sellised rakendused eraldi kasutajakontot, mis muudab rakenduse kasutamise tülikamaks. Samuti muutuvad paljud rakendused tasulisteks, kui ülesandeid on soov ära peita või neid mitme kontoga hallata. Lisaks kõigele eelnevale puuduvad enamikel sellistel lahendustel rakendusliidesed, mis võimaldaks ülesandeid programmaatiliselt lisada või neid oma lehel kuvada. Kuna töö autoril ei õnnestunud leida ühtegi rakendust, mis vastaks peatükis 2.1. toodud nõuetele, otsustas töö autor rakenduse ise ehitada.

2.2.2. Dünaamilised veebirakendused

Dünaamilised veebirakendused on süsteemid, mis kuvavad serveripoolse töötluse toel kasutajale veebilehel dünaamiliselt muutuvat sisu. Selliste rakenduste tööpõhimõte (vt joonis 5) on järgmine: veebiserver võtab kasutajatelt vastu HTTP päringud, edastab need töötlemiseks serveripoolsele loogikale, mis teeb enamasti veel lisaks andmebaasipäringuid ning tagastab seejärel kasutajale serveripoolse loogika koostatud tulemuse [9].

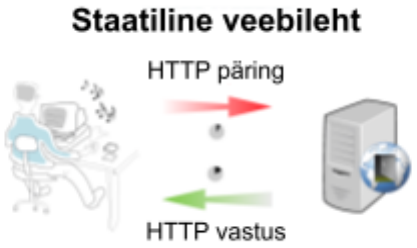


Joonis 5. Dünaamilise veebilehe tööpõhimõte [9].

Tänapäeval on enamik funktsionaalsetest veebilehtedest dünaamilised, et salvestada andmebaasidesse erinevate kasutajate tehtud tegevusi ning kasutada seda infot muude funktsionaalsuste täitmisel. Dünaamiliste veebirakenduste laadimine on tavaliselt aeglasem, sest lehekülg on seotud andmebaaside ja serveripoolse loogikaga, millest info pärimine ja töötlemine võtab lisaaega [10].

2.2.3. Staatilised veebirakendused

Staatilised veebirakendused on süsteemid, mis toimivad järgneva tööpõhimõtte (vt joonis 6) järgi: kasutaja saadab veebibrauseri kaudu veebiserverile HTTP päringu ja server saadab kasutajale tagasi juba eelnevalt serverisse salvestatud faili [9]. Staatilisel veebilehel puudub serveripoolne töötlus ja loogika ning seetõttu, on staatiliste lehtede laadimine üldiselt kiirem ning võimalusi vigade või turvanõrkuste tekkimiseks on vähem [10].



Joonis 6. Staatilise veebilehe tööpõhimõte [9].

Kuna staatiliste lehtede majutamine ei vaja palju arvutusjõudlust ning on üldiselt soodne, siis on mitmeid võimalusi staatiliste lehtede tasuta majutamiseks. Tasuta majutust pakuvad näiteks GitHub, GitLab, Netlify, Surge.sh jpt.

2.2.4. Loodava veebirakenduse arhitektuur

Uue veebirakenduse nõuetele tuginedes ei ole vaja rakenduse lõppkasutajate andmeid koguda. Rakenduse põhiline eesmärk on kuvada viktoriini üldinfot, varasemaid viktoriiniülesandeid ja tulemusi. Lehel kuvatavaid andmeid saavad muuta ainult viktoriini korraldajad. Seni on andmete kirjutamiseks ja muutmiseks kasutatud GitLabi repositooriumit.

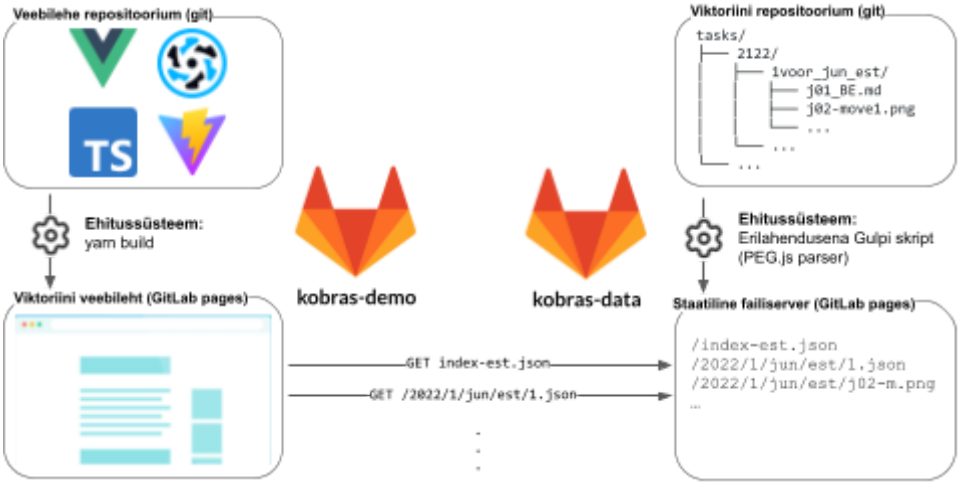
Esialgu tekkis käesoleva töö autoril mõte luua dünaamiline veebirakendus koos andmebaasiga, kuhu kõik varasemad ülesanded ja tulemused paigutada. Korraldajad aga olid rahul ülesannete haldamisega versioonihaldustarkvaras Git. Nende soov oli seda ka edaspidi kasutada, sest see oli korraldajatele juba selge, seal on hea ülevaade versiooniajaloost (kes ja millal on muudatusi teinud), tagatud on andmete säilimine tehniliste rikete korral ning

võimaldatud on failidele lihtne ligipääsude haldamine. Andmebaasi kasutamisel oleks antud lahendus nõudnud, et kõiki Gitis toimunud muutusi peaks dubleerima andmebaasi.

Töö autoril tekkis küsimus, kas versioonihaldustarkvaras talletatud andmeid ei saaks otse veebirakenduses kasutada. Peale rakendusele määratud nõuete analüüsimist ning dünaamiliste ja staatiliste veebirakenduste võimalustega tutvumist osutus rakenduse jaoks valituks staatilise rakenduse tööpõhimõtte kasutamine. Nimelt võimaldab GitLab iga repositooriumi kohta avada staatilise veebilehe, kus on võimalik kuvada repositooriumist pärit andmeid. See võimekus võimaldas teisendada ülesannete ja tulemuste lähtefailid repositooriumi muutustel ning avalikustada need staatilisel lehel.

Seatud nõuete järgi peab uus rakendus võimaldama ülesannete ja tulemuste filtreerimist erinevate tunnuste alusel. Seega ei piisa ainult lähtefailide sobivale kujule parsimisest ning veebilehele paigutamisest. Kuna filtreeritavaid tunnuseid on palju, siis ei ole mõeldav genereerida ka igale võimalikule filtrite valikule vastavat lehekülge.

Tulemuse saavutamiseks saab aga staatilises rakenduses kasutada andmete filtreerimiseks JavaScripti. Vältimaks kõikide andmete korraga veebibrauserisse laadimist peab need indekseerima ning võimaldama nende laadimist ükshaaval. Sellise lahenduse implementeerimiseks on mõistlik eraldada veebilehe lähtekood ning viktoriini andmed teineteisest ja kuvada infot lehele läbi veebipäringute teise repositooriumi lehelt (vt joonis 7).



Joonis 7. Loodava rakenduse arhitektuur.

Töö autor struktureeris uue rakenduse lähtekoodi selliselt, et veebirakenduse ehituseks vajaminev kood on ühes ning sellele kuvatavad andmed ja parsimise loogika teises repositooriumis.

2.3. Andmete parsimine ja avalikustamine

Vastavalt 1.2. peatükis kirjeldatule on ülesanded ja muud viktoriini jaoks loodud andmed repositooriumis küll hästi struktureeritud, kuid siiski mitte masinloetaval kujul. Andmete veebilehel kuvamiseks ning interaktiivsete ülesannete loomiseks on vaja Markdown vormingus ülesanded muuta programmides kergemini loetavale kujule. Ülesannete struktuuri poolest oli autori arvates mõistlik andmed paigutada hierarhilisse andmestruktuuri. Seda võimaldas kõige paremini JSON (JavaScripti olemimärgisüsteem, ingl *JavaScript Object Notation*) andmevahetusvorming. JSON on JavaScriptis kasutatav tekstipõhine formaat, kuhu on andmed paigutatud JavaScripti olemi süntaksiga [11].

Viktoriini ülesannete struktuur on korrapäraselt kirja pandud, kuid samas on see keeleliselt üsna mitmekesine. Tavalise keeletöötluse abil on ülesandeid raske selgelt masinloetavale kujule viia ja selleks peaks kirjutama mahukaid programmilõike. Autor leidis, et parem alternatiiv sellise probleemi lahendamiseks on kasutada parseri generaatorit. Parseri generaator on programm, mis võimaldab etteantud grammatika põhjal luua parseri [12]. Parser ise on programm, mis suudab mingile keelele vastava lähteteksti struktureerida masinloetavale kujule, et seda saaks edasi töödelda [13]. Käesoleva töö jaoks valiti PEG.js parseri generaator, mis loob kirjutatud grammatikast parseri JavaScriptis. Grammatikas kirjeldati kuus erineva vastusetüübiga küsimust – täisarv, täisarvu vahemik, tekst, raadionupud, märkeruudud ja vastavus. Loodud grammatika suudab parsida ülesandeid nii eesti kui vene keeles, sest küsimuste üldine struktuur kattub.

Algselt oli töö autoril kavas luua kogu ülesande faili kirjeldav grammatika, kuid tegelikult ei osutunud see otstarbekaks. Grammatika põhjal genereeritud parseriga viidi masinloetavale kujule ainult küsimuse plokk, sest kogu faili parsimine muutis veahalduse keeruliseks. Lisaks ülesande küsimusele pidi eraldama ka vastuse ja kategooriate informatsiooni, kuid seda oli optimaalsem teha tavalise tekstitöötluse ja regulaaravaldistega. Ülejäänud tekstiplokkidest ei olnud vaja infot eraldada ning need jäid algsele kujule. Sellise töö tulemusena teisendati ülesanded Markdown kujult struktureeritud JSON formaati (vt joonis 8).

Joonis 8. Ülesande lähtefail Markdown vormingus (vasakul) ning JSON vormingus sihtfail (paremal).

Kõikide repositooriumis olevate ülesannete parsimine ei olnud siiski nii lihtne. Ülesannete failid olid küll struktureeritud, kuid failide struktuur ei olnud alati piisavalt range. Näiteks leidusid järgnevad probleemid:

- esines ülesandeid, kus vastuse selgitus oli paigutatud eraldi ploki asemel vastusega samasse plokki ning tegi nende automaatse eraldamise keerulisemaks;
- lisaks eelmainitud ülesande tüüpidele oli kirjeldatud samu küsimusetüüpe erinevate nimetustega;
- esines muid harvemaid vigu ülesannete failides (mis loodud grammatikale või parsimise loogikale ei vastanud).

Sellised erinevused olid üsna seaduspäratud ning seega ei olnud otstarbekas neid eraldi keeletöötluse ja grammatika käigus käsitleda, kuna see oleks muutnud loogika liialt segaseks ja pikaks. Töö autor otsustas probleemid kõrvaldada viies kõik sellised lähtefailid üldisemale kujule. Taolisi ülesandefaile oli ligikaudu 80 (eesti ja vene keeles kokku, vead enamasti dubleeritud) ning nende tuvastamiseks kasutati juba loodud parsimisloogikat. Kõik failid, mis parsimise käigus takerdusid vaadati hiljem käsitsi üle ning tehti vajalikud parandused.

Lisaks ülesannetele peab loodav veebirakendus pakkuma ka võimalust vaadata varasemaid tulemusi ning viktoriini üldinfot. Võimaldamaks kogu rakenduse muutmist ühest kohast, oli mõistlik teha ka muude andmete muutmine sarnaselt ülesannete haldusele võimalikuks viktoriini repositooriumis. Esilehe haldamiseks lisati repositooriumisse kaust “web”, kus saab

juba tuttavas Markdown vormingus muuta eraldi esilehe sisu ja paigutust nii eesti kui vene keeles.

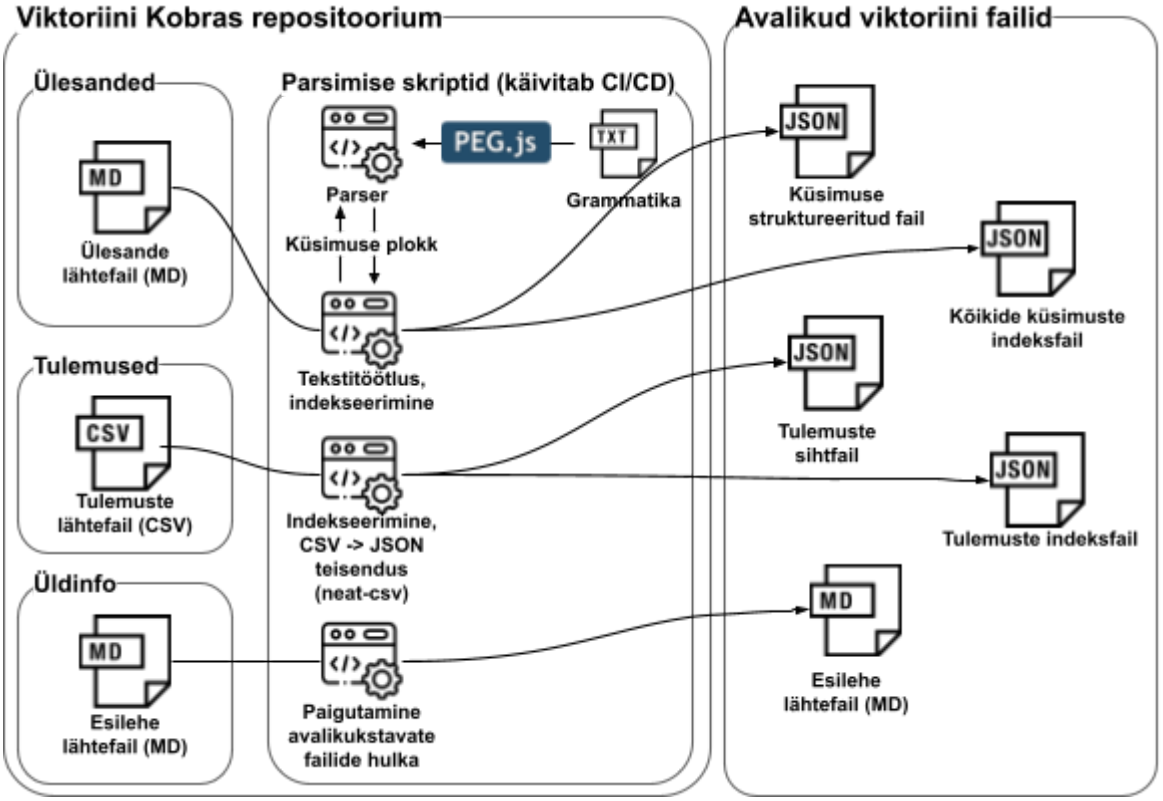
Sarnaselt saab toimida ka tulemuste haldusega – tulemused peab lisama “results” kausta vastava aasta alamkausta koos vooru ja vanuserühma täpsustava failinimega CSV formaadis. Tulemused on algselt pärit Tartu Ülikooli Teaduskooli viktoriinikeskkonnast. Sealt saab iga viktoriinivooru tulemused vanuserühmiti alla laadida tulemusi koondavas tabelis CSV failiformaadis. Seejärel peab tulemuste tabelis käsitsi kontrollima kattuvaid nimesid, vanuserühmi, järgmisse vooru pääsejaid ning tegema vastavate tabeliridade kohta täpsustavad märked. Muudatustega CSV-formaadis faili viktoriini repositooriumisse lisades teisendab programm tulemused neat-csv teegiga JSON formaati, et neid oleks lihtsam veebirakendusse laadida ja seal kuvada.

Selles peatükis kirjeldatu saavutamiseks on vaja lisaks parsimisele ja keeletöötlusele leida üles repositooriumist kõik vajaminevad failid ning nende peal eelnevalt kirjeldatud loogikat rakendada. Selle saavutamiseks võeti kasutusele JavaScripti Gulp.js teek. Gulp.js võimaldab hõlpsasti leida soovitud mustrile vastavad failid ning mitmete lisateekide abil töödelda ja paigutada need failid uutesse kaustadesse [14]. Gulpi ja kõikide lisateekide haldamiseks oli kasutusel Node.js JavaScripti käitussüsteem ja NPM (Node’i paketihaldur, ingl *Node Package Manager*).

Lisaks failide endi parsimisele JSON kujule peab need veel indekseerima. Indekseerimise failid peab looma nii tulemuste kui eesti- ja venekeelsete ülesannete jaoks. Indeksfailidesse saab paigutada iga indekseeritud faili asukoha ning kõik selle faili filtreeritavate väljade väärtused. See võimaldab hiljem veebirakenduses indeksfaili alla laadida, seda filtreerida vastavalt soovitud väljadele ning laadida rakendusse ükshaaval kõik filtritele vastavad andmed. Indekseerimise failid sai samuti loodud JSON formaadis, et neid oleks hiljem veebirakenduses parem käsitleda.

Seni kirjeldatud rakendust on võimalik küll enda arvutis lokaalselt käivitada, kuid repositooriumi muudatuste korral automaatselt uute sihtfailide genereerimiseks ja avalikustamiseks peaks kasutama pideva integratsiooni ja tarne (ingl *continuous integration / continuous deployment*, CI/CD) võimalusi. CI/CD on oma olemuselt käskude ja loogika kogum, mida rakendatakse, et tarkvara lähtefailide muutmisel testida, käivitada ja avalikustada uus tarkvara versioon [15]. CI/CD konfigureerimiseks on palju võimalusi ning antud võimalusi saab edukalt ära kasutada ka viktoriini andmete automaatsel parsimisel ning avalikustamisel. Käesoleva töö raames kasutatakse CI/CD jaoks GitLabi sisseehitatud

võimalusi – repositooriumi failis ".gitlab-ci.yml" on kirjeldatud CI/CD loogika ning käivitatavad käsud [16]. Viktoriini repositooriumis käivitatakse JavaScriptis kirjutatud andmete parsimise ja failide paigutamise loogikat sisaldab (vt joonis 9). Failis töö nimeks "pages" seades avalikustab GitLab vaikekonfiguratsiooni järgi "public" kausta paigutatud failid repositooriumi staatilise lehena. Samas võimaldab CI/CD töö valminud failid saata automaatselt ka mujale GitLabi-välisesse serverisse.



Joonis 9. Andmete parsimise ja avalikustamise protsess.

Viktoriini repositooriumis hallatakse lisaks toimunud viktoriinide andmetele ka tulevasi viktoriiniülesandeid. Need ülesanded ei tohi kindlasti sattuda veebirakendusse enne, kui viktoriin on juba toimunud. Selle tagamiseks on olemas kaks võimalust. Ülesannete parsimisest ja andmete automaatsest avalikustamisest on võimalik pääseda, kui kasutada Giti harusid. Praegu on CI/CD protsess konfigureeritud nii, et see käivituks vaid põhiharu "main" muutumisel. Samuti võimaldab kindlate kaustade/failide avalikustamist ära hoida praegune Gulp.js konfiguratsioon. Selle loogikas on välja jäetud kõik failid ja kaustad, mis algavad punktiga. Seega saab näiteks järgmise viktoriinivooru kausta ette lisada punkti, et see parsimisest ja avalikustamisest kõrvale jätta. Samuti saab failinime punktiga alustades jätta kõrvale nt kindlate abifailide avalikustamise.

2.4. Veebirakenduse arendamine

Veebirakenduse selgema struktuuri tagamiseks oli kasutusele võetud JavaScripti-põhine veebiraamistik Vue.js. Raamistiku kasutamist ajendas 2021. aasta Rebekka Breedise tehtud veebiraamistike võrdlus, kus Vue.js osutus teiste suurema kasutajaskonnaga raamistike seas parimaks valikuks [17].

Vue.js enda kirjelduse põhjal on see JavaScriptile ehitatud veebiraamistik, mis kasutab komponendipõhist arhitektuuri [18]. See tähendab, et rakendus koosneb komponentidest, mida saab korduvalt kasutada ning siduda omakorda teiste komponentidega. Komponendid koosnevad ise struktuuri ja ehitust kirjeldavast HTML keelest, HTMLi disaini kirjeldavast CSS (kaskaadlaadistik, ingl *Cascading Style Sheets*) stiilist ning veebilehele dünaamilisust pakkuvast JavaScripti loogikast.

Veebirakenduse arendamiseks kasutati Vue.js 3 versiooni koos Yarn paketihalduri ning Vite ehitustööriistaga. Vue.js komponendid olid konfigureeritud kasutama Typescripti. Typescript on tüübitud versioon JavaScriptist, mis soodustab keerulisemate objektide kirjeldamist eraldi liideste, tüüpide või klassidena ning hoiab ära tüübitud muutujatesse valet tüüpi andmete sattumise [19]. Tänu TypeScripti rakendamisele oli autoril võimalus tagada koodi parem loetavus ning tagada vähem vigu ja arusaadavam süsteemi disain.

Veebirakenduse arendamiseks kasutati lisaks teeke Vue Router, Mitt ja Marked. Vue Router võimaldab rakenduse erinevatele komponentidele vastavusse panna veebilingi, hallata navigatsiooni rakenduses jne [20]. Viktoriini jaoks loodud veebirakenduses võimaldas antud teek avada veebilingiga näiteks tulemuste vaate, ilma et peaks külastama rakenduse esilehte. Mitt on minimalistlik teek rakenduse kesksete olekute haldamiseks, et saaks hallata ühiseid andmeid erinevate komponentide vahel [21]. Viktoriini veebirakenduses on see kasutusel kasutaja valitud kuvakeele haldamiseks. Marked on minimalistlik Markdowni parser, mis võimaldab Markdowni parsimist HTML-i [22]. Rakenduses kasutatakse seda ülesannete ja muude lähtefailide Markdownist HTML-i konverteerimiseks ning veebilehel kuvamiseks.

Sarnaselt viktoriini andmete repositooriumile paigutati ka uue veebirakenduse lähtekood GitLabi repositooriumisse. Staatilise lehe majutamiseks kasutati GitLab pages funktsionaalsust ning CI/CD abil ehitati failidest kuvatava veebilehe lähtekood, mis avalikustati GitLabis staatilise lehena. GitLabi majutatud lehte on võimalik ühendada senise veebilehe domeeniga. Rakendust saab soovi korral automaatselt paigutada ka GitLabi-välisele serverile.

Loodava veebirakenduse üks peamisi nõudeid oli tagada hea kasutajakogemus. Jakobi seaduse põhjal on lihtsa kasutuse üks alustalasid pakkuda kasutajatele midagi, mis on neile juba tuttav [23:1]. Nii ei pea nad uue rakendusega ümber kohanema ning saavad seda kasutada sarnaselt juba teistele tuttavatele rakendustele. Informaatikaviktoriini veebi-rakenduse loomisel oli üks eesmärk pakkuda intuitiivset kasutust ning seega oli mõistlik kasutada rakenduses kasutajatele juba tuttavaid elemente. Loodud rakenduse puhul on sellised elemendid näiteks lehe ülaosas asuv menüüriba, kaartidena sisu kuvamine ja otsingufiltrite paigutus enne kuvatavaid andmeid. Lehe ülaosas kuvatav menüüriba on veebilehtedel kõige enamlevinud menüü tüüp [24] ning Jakobi seaduse järgi seega esimene koht, kuhu inimesed alamlehe vahetamisel vaataksid [23:1]. Samuti võiks olla kasutajatele juba tuttav kaardipõhine sisu kuvamine, mida kasutatakse näiteks suhtlusvõrgustikus Facebook ning see, et otsingufiltreid paigutatakse enne kuvatavaid andmeid.

Loodav rakendus koosneb ka väiksematest elementidest nagu tekstiväljad, ripploendid, tabelivaated jpm. Selliste elementide loomisel saab kasutada küll puhast HTMLi, kuid autori hinnangul näevad sellised elemendid välja üsna algelised ning ei paku head kasutajakogemust. Parema tulemuse saamiseks võib disainida neid elemente kasutades mõnda põhjalikult testitud ning juba mujal veebis laialt kasutusel olevat disainikeelt. Töö autor valis rakenduse arendamiseks Material Design disainikeele, mis on mõeldud eelkõige rakenduste disainimiseks veebis ja nutiseadmetes [25]. Google'i poolt välja töötatud disainikeel on põhjalikult dokumenteeritud ja läbimõeldud, et tagada selle printsiipide kasutamisel võimalikult hea rakenduste kasutatavus ja ühtne stiil. Material Design printsiibid on populaarsed ning selle põhjal on loodud Vue.js jaoks mitu erinevat komponendipõhist disainiraamistikku [26]. Töös on kasutusel disainiraamistikuna Quasar, mis sisaldab endas Vue.js komponente disainikeeles kirjeldatud elementide loomiseks, tagab selliste elementide lihtsa konfigureerimise ning muudab rakenduste loomise lihtsaks erinevate brauserite ja ekraaniresolutsioonide jaoks [27]. Ühtlasi pakub raamistik enda kirjelduse põhjal head arenduskogemust ning paremaid jõudlusnäitajaid kui teised taolised raamistikud [27].

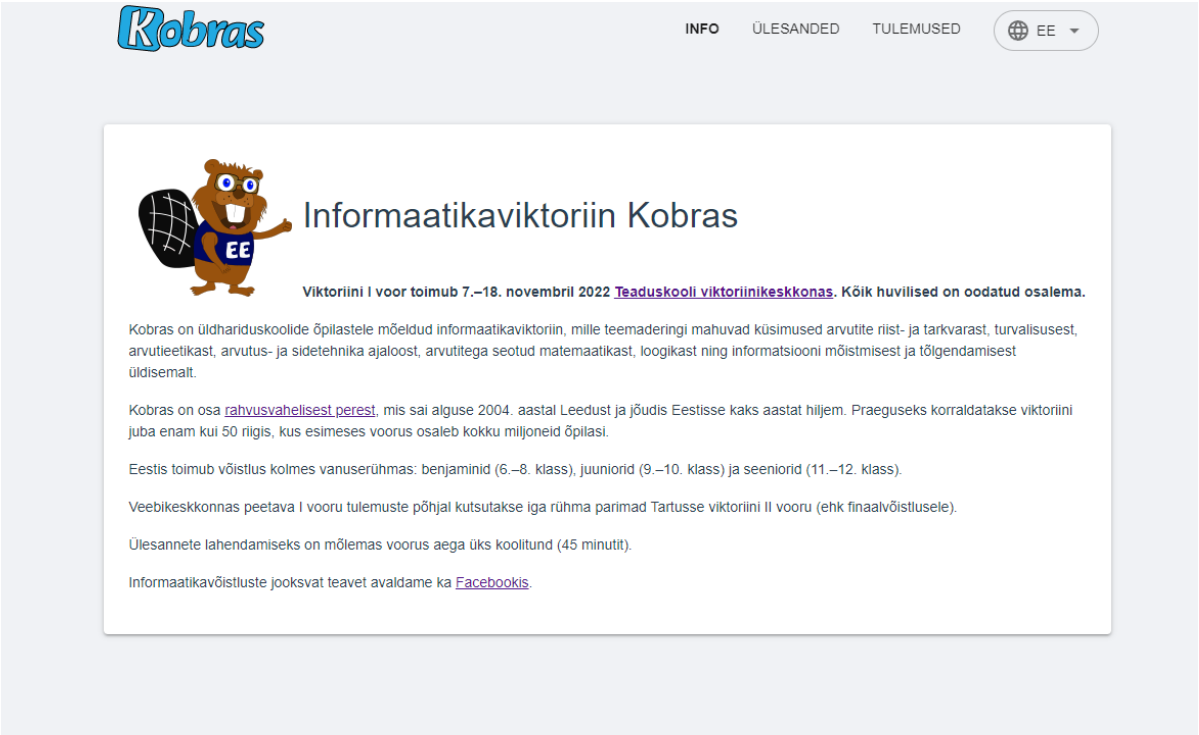
Nõuete järgi peab rakendus olema mugavalt kasutatav erinevate suuruste ja resolutsioonidega ekraanidel. See eeldas tööd menüü, tulemuste tabeli ning üldise lehe paigutuse kallal. Kuna kõik menüülingid ei pruugi mahtuda kitsale ekraanile, lisati väiksemate ekraanide jaoks nupust avanev menüü. Väiksematel ekraanidel tulemuste tabelit vaadates kuvatakse tabeli asemel tabelirea info plokkidesse paigutatuna. Suurema resolutsiooniga ekraanide jaoks koondati lehe sisu rohkem lehe keskossa, et lehekülg ei valguks ekraanil liialt laiali.

3. Valminud veebirakendus

Käesoleva bakalaureusetööna valmis uus veebirakendus informaatikaviktoriinile Kobras. Valminud rakendus on nähtav veebiaadressil https://mattiassokk.gitlab.io/kobras-demo/, lingid lähtekoodile on kättesaadavad lisas 1. Loodud veebirakendus on kahes keeles kättesaadav ja sisaldab mitu erinevat vaadet:

- üldinfo;
- ülesannete vaade;
- tulemuste vaade.

Veebirakendust avades kuvatakse kasutajale esmalt viktoriini üldinfot sisaldav leht (vt joonis 10). See leht sisaldab endas üldist informatsiooni viktoriini ja selle korralduse kohta ning teadaandeid näiteks tulevastest voorudest. Sellele vaatele laetakse andmed vastavalt keelele viktoriini repositooriumi GitLabi lehelt.

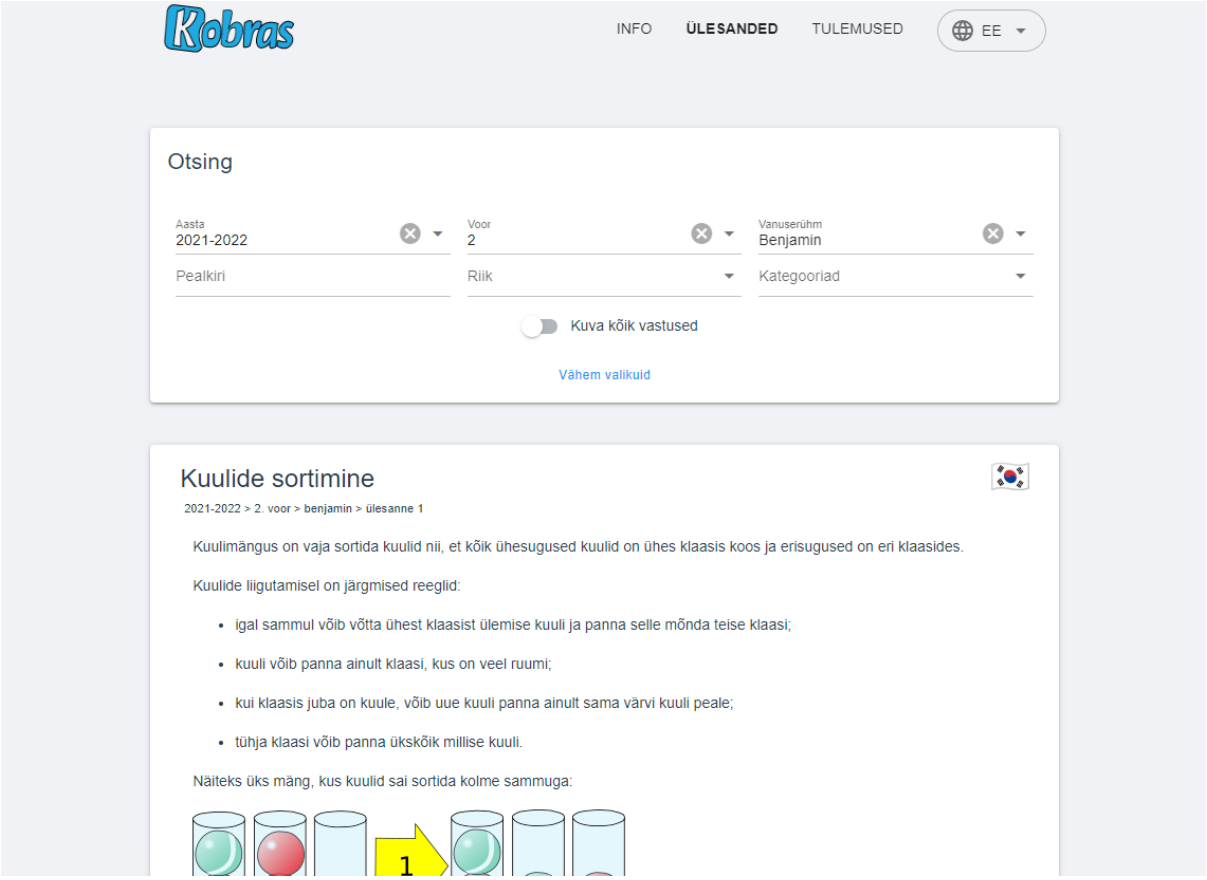


Joonis 10. Kuvatõmmis loodud veebirakenduse esilehest.

Veebirakenduse nõuete järgi on üks tähtsamaid ning funktsionaalsemaid vaateid ülesannete vaade (vt joonis 11). See võimaldab kasutajatel vaadata viktoriini varasemate voorude ülesandeid. Neid on võimalik filtreerida aasta, vooru, vanuserühma, riigi ja kategooriate järgi. Lisaks saab otsida ülesannet ka pealkirja põhjal. Vaikimisi on pealkirja, riigi ja kategooria järgi otsimine peidus, et kasutajat liigsete filtritega mitte segadusse ajada. Samuti on võimalik

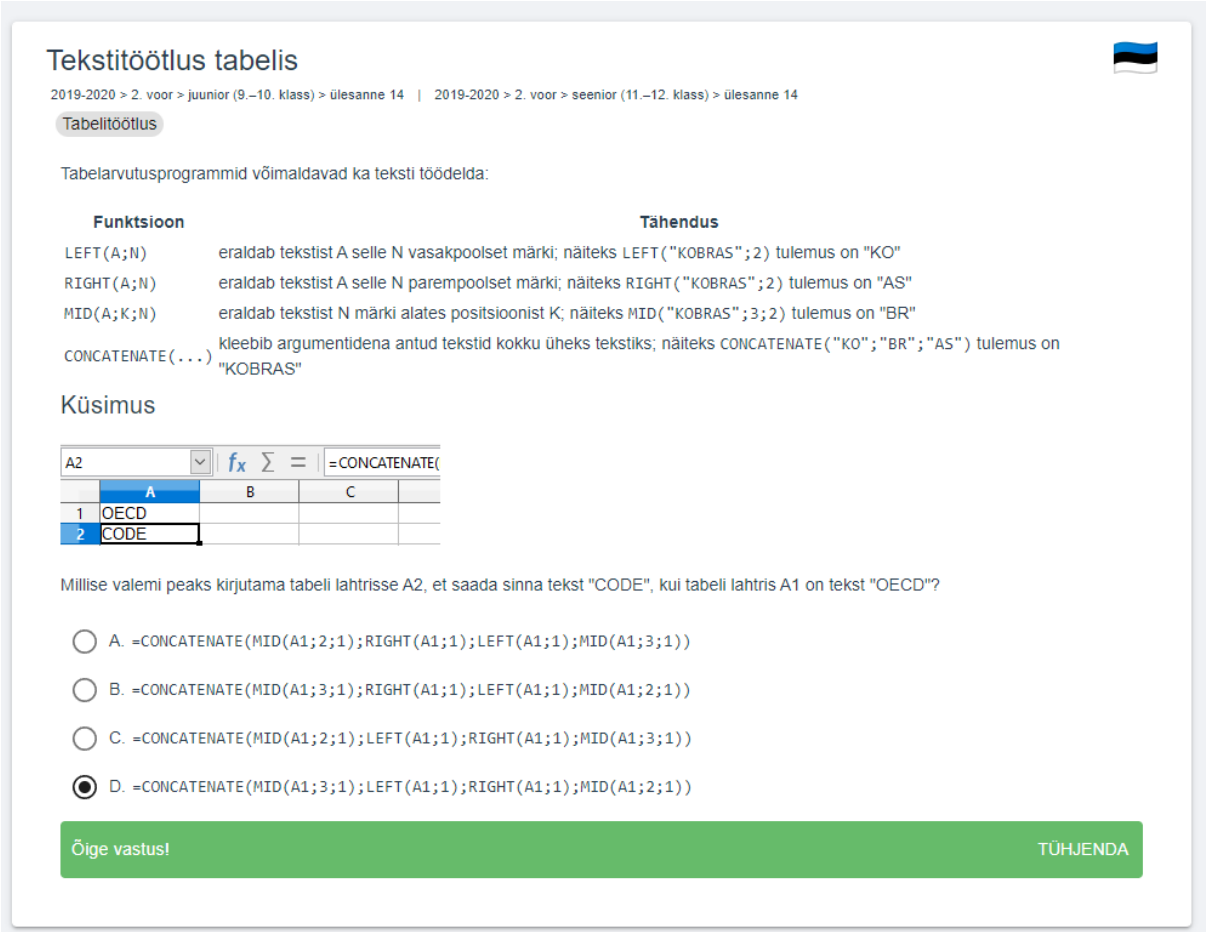
valida, kas ülesanded on näidatud õigete vastustega või vastamata kujul. Selleks peab lülitama filtrite seas sisse kõikide vastuste kuvamise.

Ülesannete lehele sattudes laeb rakenduse külastaja veebibrauser automaatselt ülesannete indeksfaili, mis sisaldab kõiki ülesandeid koos filtreeritavate väljade väärtustega. Ülesannete filtreerimisel valitakse kõik indeksfaili ülesanded vastavalt seatud kriteeriumitele ning laetakse nende ülesannete kogu info eraldi failist. Vaate avamisel on otsingufiltriteks vaikimisi valitud viimane viktoriinivoor ja benjaminide vanuserühm, kuna kõige suurem huvi on tõenäoliselt viimase viktoriinivooru vastu ning kõige enim osalejaid on olnud benjaminide vanuserühmas (vt peatükk 1.1.). Ülesandeid laetakse lehele viie haaval, et vältida veebibrauseri ülekoormamist ja liigsete andmete kuvamist lehel. Lehekülje lõppu kerides laetakse automaatselt järgnevad viis ülesannet ja seda tehakse kuni andmed saavad otsa. Filtritega on võimalik lehele valida palju ülesandeid ning piisavalt kaua lehte kerides tekib ekraanile nool, millega saab lihtsasti tagasi lehe ülaossa naasta ning muuta valitud otsingufiltreid.



Joonis 11. Kuvatõmmis loodud rakenduse ülesannete vaatest.

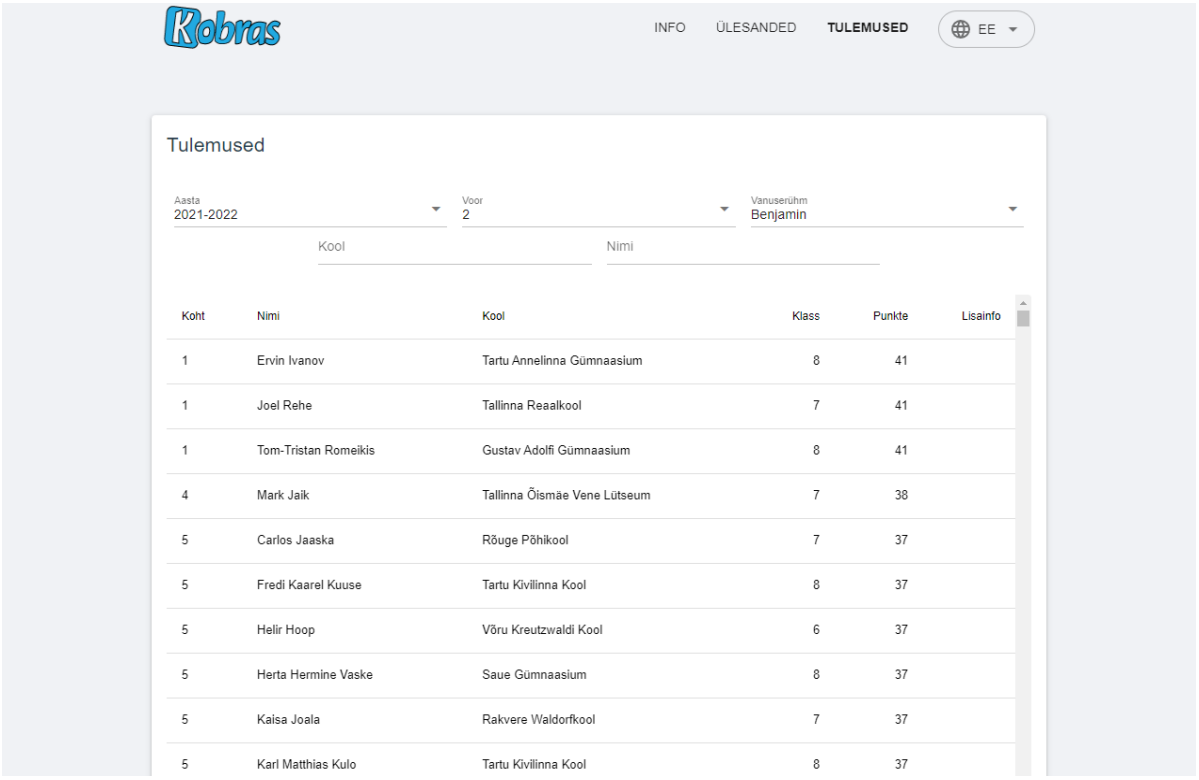
Iga ülesanne on kuvatud eraldi kaardil (vt joonis 12) ning sisaldab pealkirja, ülesande lähteriigi lippu, ülesande esinemise infot (aasta, voor ja vanuserühm), ülesande kategooriat, ülesande kirjeldust, küsimuse sõnastust, plokki vastuse sisestamiseks, õige vastuse selgitust ning kirjeldust, kuidas ülesanne on seotud informaatikaga. Viimased kaks on saadaval vaid siis, kui sellised plokid on ülesande lähtefaili lisatud. Õige vastuse selgitus ja ülesande seotus informaatikaga on vaikimisi peidetud ning neid saab kuvada vastava teksti peale klikkides. Veebirakendust kasutades on võimalik ülesandeid läbi lahendada ja kontrollida oma vastuse õigsust. Vale vastuse valides on võimalik lehele tekkinud nupule vajutamise järel kuvada ülesande õige vastus. Nii õigesti kui valesti vastatud ülesannet on võimalik tühjendada.



Joonis 12. Ekraanitõmmis loodud veebirakenduse ülesannete vaate vastatud küsimusest.

Tulemuste vaates (vt joonis 13) on võimalik rakenduse külastajal valida sobiv aasta, voor ja vanuserühm, et näha vastava vooru tulemusi. Lisaks saab tulemuste seast otsimiseks kasutada tekstivälja kooli ja osaleja nime täpsustamiseks. Vaate avades on vaikimisi valitud viimane viktoriinivoor ning benjamini vanuserühm. Tabelis on tulemused vaikimisi sorteeritud osaleja tulemuse järgi kahanevalt – viktoriinivooru parimad osalejad on tabelis üleval ning vähem

punkte saanud osalejad all. Iga osaleja kohta on täpsustatud tema koht, nimi, kool, klass ja saadud punktide arv. Lisaks on tabelis veerg lisainfo jaoks, kus võivad olla osaleja kohta täpsustavad märked. Näiteks võib olla seal täpsustatud, et osaleja on pääsenud järgmisse vooru või on diskvalifitseeritud, kuna osales mitu korda või lahendas vale vanuserühma ülesandeid.



Joonis 13. Kuvatõmmis loodud veebirakenduse ülesannete vaatest.

Töö käigus selgus lisanõue, et viktoriini varasemad tulemused võivad olla nähtavad ainult viktoriini rakenduses ning otsingumootoritesse ei tohi osalejate info jõuda. Nõude täitmiseks lisas töö autor rakenduse lehele otsingumootorite indekseerimist keelava märke.

4. Valminud lahenduse analüüs

Järgnevas peatükis analüüsib töö autor valminud lahendust ja kontrollib rakenduse vastavust 2.1. peatükis seatud nõuetele. Nõuete kontrollimiseks testitakse rakendust ka lõppkasutajatel.

4.1. Rakenduse testimine

Üks rakenduse põhilisi nõudeid oli tagada sellele hea kasutajakogemuse disain. Tegu on mittefunktsionaalse nõudega, mille kontrollimiseks peab tagasisidet küsima rakenduse kasutajatelt.

4.1.1. Testimise metoodika

Kasutuskogemuse testimiseks otsustas autor läbi viia kvalitatiivse kasutusmugavuse uuringu. Uuringu koostamisel kasutati maailma juhtiva teaduspõhise kasutuskogemust uuriva Nielsen Norman Groupi koostatud juhendeid [28]. Antud juhendite järgi viiakse kasutusmugavuse testimist läbi üldiselt sessioonina, kus testi läbiviija saab kokku testijaga ning laseb tal testitavas rakenduses täita erinevaid ülesandeid. Ülesanded peaksid kirjeldama üldiselt tegelikke situatsioone, mis võivad rakendust kasutades ette tulla. Peale ülesande testijale andmist jälgib läbiviija rakenduses tehtud tegevusi ning otsib kohti, mis võivad testijale arusaamatuks jääda. Peale ülesannete täitmist küsitakse testijalt tagasisidet, mis tema tegevuse vaatlemisel ei pruukinud ilmneda. Kasutusmugavuse testimine aitab leida loodud rakenduse kitsaskohti, leida võimalusi edaspidiseks arenguks ning õppida tundma kasutajate harjumusi ja eelistusi. Uuringusse valitakse inimesi, kes kuuluvad rakenduse tegelike kasutajate sihtrühma.

Käesoleva rakenduse kasutuskogemuse uuring koostati õpetajatele, kelle õpilased osalevad viktoriinis. Testimiseks valiti õpetajad, sest korraldajatel olid nende kontaktandmed, õpetajate motivatsioon uuringus osaleda on tõenäoliselt kõrgem, sealjuures saab nende peal testida rohkem tegelikke kasutusjuhte. Lisaks erinevate kasutusjuhtude testimisele küsiti õpetajatelt ka üldist viktoriini puudutavat infot, et seda saaks kasutada rakenduse edaspidisel arendamisel. Toetudes Norman Nielsen Groupi välja töötatud materjalidele koostas töö autor kava (vt lisa 2), millele tuginedes intervjueeriti õpetajaid Zoomi vahendusel.

4.1.2. Testimise tulemused

Eelnevas alampeatükis kirjeldatud uuring viidi läbi nelja õpetajaga (vt tabel 1). Intervjueeritavad õpetajad andsid informaatika või matemaatika tunde kas põhikooli- või gümnaasiumiastmele. Iga seanss võttis kokku ligikaudu 30 minutit ning uuringu käigus leiti bakalaureusetöö raames arendatud rakenduses nii kitsaskohti kui üldiseid ideid uute funktsionaalsuste arendamiseks.

Tabel 1. Uuringus osalenud õpetajad.

	Õppeastmed	**Õppeained**	**Veebirakenduse keel**
Õpetaja 1	põhikool, gümnaasium	informaatika	eesti
Õpetaja 2	põhikool	matemaatika	eesti
Õpetaja 3	gümnaasium	informaatika	eesti
Õpetaja 4	põhikool, gümnaasium	matemaatika, informaatika	vene

Uuringu alguses küsiti õpetajatelt tagasisidet olemasoleva viktoriini veebilahenduse kohta. Selle käigus kaardistati, kuidas õpetajad on viktoriini senist veebilehte kasutanud. Kolm õpetajat (õpetajad 2, 3 ja 4) ei valmista oma õpilasi eelnevalt viktoriiniks ette. Viktoriini veebilehele suunatakse varasemate aastate ülesandeid lahendama õpilasi, kes on esimeses voorus saanud hea tulemuse ning pääsenud järgmisse vooru. Selgus, et õpetajad kasutavad lehte pigem tunnis käsitletavate teemade või nuputamisülesannete otsimiseks, et pakkuda õpilastele vaheldust. Kolm õpetajat (õpetajad 1, 2 ja 3) nimetas seni kasutusel olnud veebilehe probleemiks, et huvipakkuvaid ülesandeid on raske üles leida.

Uurides õpetajatelt, mis funktsionaalsustest nad enim senise veebilehe puhul puudust tunnevad, mainisid kõik õpetajad, et ülesannete otsimine võiks olla lihtsam. Samuti pakkus kaks õpetajat (õpetajad 2 ja 3) välja, et rakendus võiks olla interaktiivne, sest õpilased ei ole motiveeritud ülesandeid lahendama, kui need on kirjas tavalise tekstina ning vastuseid saab kontrollida eraldi lehel.

Seejärel keskendus uuring uuele rakendusele. Kasutusjuhtude testimisel said õpetajad väga hästi hakkama. Kõikide kasutusjuhtude jaoks tehtavad tegevused sooritati keskmiselt vähem kui 30 sekundi jooksul ning täiendavat juhendamist õpetajad ei vajanud. Õpetaja 2 suutis vajaminevad tegevused korduvalt lõpetada isegi enne, kui töö autor jõudis kasutusjuhu

lõplikult ette lugeda. Esines küll väiksemaid arusaamatusi (näiteks vale aastaarvu valimine), kuid rakenduses endas leiti tegevusteks kõik õiged kohad üles. Kõik õpetajad kasutasid rakendust esimest korda ning seega võib öelda, et see oli nende jaoks piisavalt intuitiivne, et kõik vajalikud tegevused ilma kõrvalise abita mõistliku aja jooksul sooritada.

Kasutusjuhtude testimise pingsamal jälgimisel ilmnes siiski kahel õpetajal väiksemaid arusaamatusi. Kuna kasutusjuhtudes oli mainitud erinevaid klasse, siis võttis õpetajatel 1 ja 3 õige klassi ja vanuserühma seostamine veidi aega. Kuna taoline olukord võib esineda veelgi enam, kui rakendust kasutab õpilane, siis lisati vanuserühma rippvalikutesse vanuserühma järele sellele vastavate klasside vahemik. Veel jäi õpetaja 3 jaoks segaseks ülesannete vaates valitud filtri tühjendamine. Kasutusjuhu simuleerimine nõudis kõikide aastate, voorude ja vanuserühmade tühjendamist, kuid välja tühjendamisega ei osatud seostada välja järel olevat "X" märgiga ikooni. Hiljem selle ikooni avastades mainis õpetaja, et ei osanud seda lihtsalt otsida ning tagantjärele tundub see väga loogiline. Õpetaja 4 testis rakendust väiksema ekraaniresolutsiooniga ning leidis, et ülesannete paigutus võiks olla kompaktsem ja kogu ülesanne võiks mahtuda ekraanile. Probleemi põhjustasid enamasti suured pildid ülesande kirjelduses. Andmete parsimise ja veebilehele paigutamise käigus ei tohiks piltide suurust automaatselt muuta ning piltide suurust võiks arvesse võtta ülesande lähtefailide koostamisel.

Peale kasutusjuhtude testimist õpetajatega suheldes kinnitasid kõik, et uue rakenduse kasutamine tundus intuitiivne ja kõik vajalik oli lihtsasti üles leitav. Õpetaja 2 ütles, et selline testimine andis talle justkui hea kiirkursuse kõikidest võimalustest, mida on uues rakenduses võimalik teha. Kõik testimises osalenud õpetajad olid arvamusel, et uuest rakendusest on palju abi nii neile kui ka teistele õpetajatele, sest ülesanded on lihtsamini üles leitavad ning rakenduse interaktiivsus võimaldab seda paremini kasutada ka õpilastega koos. Õpetajad uskusid, et kasutavad loodud rakendust tulevikus rohkem kui senist viktoriini veebilehte on kasutanud.

Arutledes rakenduse lisavõimaluste üle pakkus õpetaja 2 välja, et lehekülg võiks võimaldada ka kontoga sidumist, et õpetajad saaksid oma õpilaste tegevusi jälgida – see võimaldaks rakendust kasutada näiteks kodutööde jätmiseks. Samuti tunti puudust teha klassisiseseid võistlusi ning lahendada valitud ülesandeid viktoriini vormis. Õpetaja 3 pakkus sellise edasiarenduse välja samas veebirakenduses, õpetaja 2 aga arvas, et sellised viktoriinid võiks olla valmis kujul olemas mõnes Kahooti-taolises reaalajalises viktoriinirakenduses.

4.2. Vastavus seatud nõuetele

Järgnevas alampeatükis antakse ülevaade, kas valminud lahendus vastab peatükis 2.1. seatud nõuetele. Iga nõude puhul kirjeldatakse, mis tehti, et nõue oleks täidetud ning selgitatakse, kas see oli nõude täitmiseks piisav või mitte.

Funktsionaalsed nõuded

1. *Rakendus peab võimaldama veebilehel ülesannete automaatset uuendamist Markdowni lähtefailide põhjal, et korraldajad ei peaks enam käsitsi veebilehe jaoks HTML faile genereerima ning neid veebiserverisse tõstma.*

   Nõue on täidetud, kuna ülesandeid muutes käivitub automaatselt CI/CD töö, mis parsib andmed ning teeb need GitLab pagesi toel internetis nähtavaks. Veebirakenduse kaudu laetakse need andmed avalikust internetiallikast alla ja paigutatakse lehele.

2. *Rakendus peab võimaldama viktoriini üldinfo lihtsat muutmist.*

   Nõue on täidetud, sest viktoriini esilehel kuvatavat üldinfot saab muuta sarnaselt ülesannetele viktoriini repositooriumist failides “home-ee.md” (eesti keel) ja “home-ru.md” (vene keel).

3. *Rakendus peab võimaldama ülesannete filtreerimist aasta, vooru, vanuserühma, pealkirja, riigi ja kategooria alusel, et oleks võimalik üles leida huvipakkuvaid ülesandeid.*

   Nõue on täidetud. Rakenduse ülesannete vaates on võimalik ülesandeid filtreerida kõigi eelnimetatud väljade alusel. Aasta, vooru, vanuserühma, riigi ja kategooriad saab valida ripploendist. Pealkirja on võimalik otsida tekstina. Ülesannete vaates laeb rakenduse külastaja veebibrauser alla kõikide ülesannete indeksfaili ning seejärel filtritele vastavate ülesannete sisu viie kaupa.

4. *Peab võimaldama igat ülesannet lahendada ja ülesannete õigeid vastuseid kuvada, et õpilased saaksid lehel valida endale huvipakkuvad ülesanded ning neid lahendada.*

   Nõue on täidetud, sest ülesannete vaates on võimalik iga ülesande puhul vastuse sisestamise ploki kaudu kontrollida enda vastust. Samuti on võimalik filtrite seast valiku “Kuva kõik vastused” valides näha kõiki ülesandeid juba lahendatud kujul.

5. *Peab võimaldama iga ülesande puhul vaadata selle vastuse selgitust ning ülesande seotust informaatikaga, kui sellised tekstiplokid lähtefailis ülesande kohta olemas on.*

   Nõue on täidetud. Kui ülesande Markdowni failis on lisatud lõik vastuse selgituse kohta ja/või lõik, milles selgitatakse ülesande seotust informaatikaga, siis paigutatakse vastavad tekstilõigud automaatselt ülesandefaili ning tehakse kättesaadavaks veebirakenduse jaoks. Veebirakendus laadib andmed ülesandega koos alla ja paigutab avatava paneelina kaardi lõppu.

6. *Peab võimaldama kuvada kõiki filtreeritud ülesandeid õigete vastustega, et peale toimunud viktoriinivooru saaks õpilane tutvuda õigete vastustega.*

   Nõue on täidetud. Ülesande vaate avades on võimalik otsingufiltrite seas sisse lülitada "Kuva kõik vastused" valik ning vaadata kõiki filtreeritud ülesandeid koos õigete vastustega.

7. *Kuvama tulemusi ning võimaldama nende filtreerimist aasta, vooru ja vanuserühma põhjal.*

   Nõue on täidetud. Tulemuste vaates on võimalik vaadata varasemate viktoriinivoorude tulemusi tabelina valides rippnimekirjas soovitud parameetreid. Tulemuste vaate avades laeb külastaja veebibrauser automaatselt tulemuste indeksfaili ning sarnaselt ülesandele on võimalik valida huvipakkuv viktoriinivoor ja vanuserühm. Tulemuste tabel on vaikimisi sorteeritud punktide põhjal kahanevalt ning parimate tulemustega osalejad on eespool.

8. *Võimaldama tulemuste filtreerimist kooli ja osaleja nime põhjal, et õpetajad saaksid näha kõiki oma kooli tulemusi ning õpilased otsida enda tulemust.*

   Nõue on täidetud. Lisaks õppeaastale, viktoriinivoorule ja vanuserühmale saab soovi korral kuvatud tulemusi filtreerida osaleja nime ja kooli järgi. Otsingufiltrid kontrollivad otsitava fraasi sisalduvust tabelirea vastaval väärtusel.

Mittefunktsionaalsed nõuded

1. *Rakenduse kasutamine peab olema lõppkasutajale võimalikult lihtne ja intuitiivne.*

   Nõue on täidetud. Rakenduse arendamisel lähtus töö autor Jakobi seadusest ning kasutas Google'i välja töötatud ja põhjalikult testitud Material Design disainikeelt. Hiljem õpetajatel läbiviidud kasutusmugavuse uuringu käigus selgus, et rakenduse

põhilisi funktsionaalsusi täitvad kasutusjuhud on lihtsad ning kõik õpetajad suutsid need täita ilma kõrvalise abita mõistliku aja jooksul.

2. *Rakendus peab võimaldama sisu sirvida nii eesti kui vene keeles.*

   Nõue on täidetud. Veebirakenduse menüüsse on lisatud koht kuvakeele valimiseks. Seal saab valida eesti ja vene keele vahel. Keele valides salvestatakse see veebibrauserisse ning alamlehtedel ringi liikudes kuvatakse rakenduse sisu valitud keeles. Samuti laaditakse ülesannete vaates ülesanded vastavalt valitud keelele.

3. *Viktoriinide ülesanded ei tohi olla kättesaadavad enne viktoriini toimumist ning uuel veebilehel peavad saama muuta infot ainult korraldamisega tegelevad inimesed.*

   Nõue on täidetud. Viktoriini repositooriumis saab hallata nii varasemaid kui järgnevate viktoriinivoorude ülesandeid. Vältimaks tulevaste viktoriinivoorude ülesannete automaatset avalikustamist saab luua uue Giti haru või vastava aasta/vooru kaustadele lisada ette punkti.

4. *Uue veebirakenduse haldus- ja majutuskulud ei tohi ületada senise veebirakenduse halduskulusid.*

   Nõue on täidetud. Uus veebirakendus koosneb kahest osast, mis on mõlemad staatilised. Neid saab muuta automaatselt CI/CD käigus ning staatilisi faile majutada tasuta GitLabi, senise veebilehe või mõne kolmanda serveri peal. Staatiliste veebirakenduste majutamiseks on mitu võimalikku tasuta lahendust. Rakendus on väga vähenõudlik.

5. *Rakendus peab olema kirjutatud vabavaralisele tarkvarale, millel on hinnanguliselt suur kasutajaskond ka lähitulevikus ning seega võimalik leida edaspidist tuge.*

   Nõue on täidetud. Rakenduse arendamisel on valitud programmeerimiskeelteks JavaScript ja TypeScript. Mõlemal keelel on suur ja aktiivne kasutajaskond. Lisaks programmeerimiskeeltele on kasutatud üldtuntud raamistikke ja teeke nagu Vue.js, Gulp.js jms, mille kohta leidub internetist palju materjali.

6. *Rakenduse funktsionaalsused peavad toimima probleemideta kõikide järgnevate brauseritega alates versioonidest: Chrome 100.0.4896, Firefox 98.0.2, Safari (Mac) 15.0, Safari (iOS) 15.4 ja Edge 99.0.1150.30.*

   Nõue on täidetud. Veebirakenduse arendamiseks võeti kasutusele Quasar disainiraamistik, mis võimaldab kasutada Vue.js rakendustes valmiskomponente,

mida on testitud erinevatel veebibrauseritel. Kontrollimiseks katsetas töö autor ka ise loodud veebirakendust nõuetes väljatoodud veebibrauserite versioonidel ning ei avastanud ühtegi veebibrauserist tulenevat probleemi.

7. *Rakenduse funktsionaalsused peavad toimima probleemideta erineva kuvasuhte ja resolutsiooniga seadmetel – lauaarvutitel, tahvelarvutitel ja nutitelefonidel.*

   Nõue on täidetud. Rakenduse arendamisel kasutati Material Design disainikeelt ja Quasari disainiraamistikku, et tagada üksikute elementide hea kasutatavus erinevate kuvasuhete ja resolutsiooniga ekraanidel. Lisaks loodi kitsamatel ekraanidel toimiv menüü, eriline tabelivaade ning paigutatud suuremate ekraanide jaoks sisu rohkem keskele. Kasutusmugavuse uuringu käigus selgus, et väiksema resolutsiooniga ekraanidel on pikki ja suuri pilte sisaldavaid ülesandeid raske lahendada, sest kogu ülesande materjal ei mahtunud ekraanile ära. Seda peab võtma arvesse ülesannete pildifailide loomisel või nende Markdown failidesse paigutamisel.

8. *Rakenduse esialgne laadimisaeg ning alamlehtede vahetamise ja andmete filtreerimise järel laadimiseks kuluv aeg ei tohi ületada 3 sekundit.*

   Nõue on täidetud. Rakenduse loomiseks on kasutatud raamistikku ja teeke, mis on ise kergekaalulised ning kust kasutamata jäänud üleliigne kood on eemaldatud (ingl *tree-shaking*). Nii ei pea kasutaja rakenduse avamiseks taustal liialt palju või liialt mahukaid faile alla laadima ning lehe laadimiskiirus püsib kiire. Nõude täitmise testimiseks kasutas töö autor Google Chrome'i veebibrauserisse sisseehitatud tööriista Lighthouse. Seal rakenduse kiirust testides selgus, et selle esialgne laadimine võtab ligikaudu ühe sekundi ning edaspidi võtavad rakenduses tehtud vaate vahetamised ja filtrite muutmised alla ühe sekundi aega. Oluline on märkida, veebirakenduse kiirust mõjutab kasutaja enda võrguühenduse kiirus, lõppkasutaja ja veebimajutust pakkuva serveri kaugus teineteisest jpm.

5. Võimalikud edasiarendused

Eelmises peatükis toodud valminud veebirakenduse analüüs näitas, et käesoleva bakalaureuseöö raames valminud rakendus toimib ning vastab kõigile töö algul seatud nõuetele. Uut veebirakendust luues ning õpetajaid intervjueerides tekkisid võimalikud edasiarenduse ideed.

Üks edasiarengu suund võiks võimaldada õpetajal luua varasematest ülesannetest komplekte ja neid viktoriinivormis esitada. Nii saaks õpetaja koostada klassile näiteks varasematest ülesannetest ise viktoriini ning lasta enda õpilastel seda lahendada. Selle idee pakkus välja uuringus osalenud õpetaja 3. Võimalused sellise funktsionaalsuse lisamiseks on olemas ning praeguse rakenduse arhitektuur toetab seda.

Eelnevalt väljakäidud idee puhul on oht, et klassisisesel viktoriinil osalemisel kasutatakse ebaausaid võtteid ning otsitakse viktoriini rakendusest õiged vastused. Selle vältimiseks pakkus üks viktoriini korraldajatest välja, et õpetaja saaks ise ülesannete komplekti (näiteks 5 loogikaülesannet, 5 tabelitöötluse ülesannet ja 5 algoritmi ülesannet) koostada ja selle PDF-ina rakendusest eksportida ja printida, et hiljem seda õpilastega kirjalikus vormis lahendada. Ühtlasi võib olla see kasulik tundides, kus kõigile õpilastele ei ole võimalik pakkuda arvuti kasutamise võimalust. Selle idee teostamiseks on samuti kõik eelnevad nõuded täidetud ning piisab ülesannete välja valimise loogika lisamisest ning mõne olemasoleva Markdownist PDF-i konverteerimise võimaluse implementeerimisest.

Õpetajatega suheldes selgus, et Eestis leidub teisigi viktoriine (näiteks Känguru), mille jaoks on loodud küll väga palju huvitavaid materjale, kuid mille haldus on kohmakas. Probleemi lahendamiseks saaks käesoleva bakalaureusetööna valminud rakendust üldistada, dokumenteerida ja edasi arendada nii, et see kohanduks rohkematele viktoriinidele. Seni valminud lahendus on rakendatav ka teistele viktoriinidele, kuid eeldab samasugusel kujul ülesannete haldust ning mõningaid viktoriini korraldusest tulenevaid muutusi veebi-rakenduses.

Lisaks pakkus õpetaja 2 välja, et rakendus võiks pakkuda õpilaste tegevuse jälgimist, et kontrollida, kuidas õpilastel läheb. See võimaldaks õpetajatel jätta ülesandeid näiteks õpilastele iseseisvaks lahendamiseks. Sellist edasiarendust praegune arhitektuur ei toeta, sest taoliste andmete kogumine nõuaks andmebaasi tuge ning dünaamilist veebilahendust. Kui tulevikus on soov pakkuda viktoriini rakenduses rohkem funktsionaalsusi, siis selleks peaks rakenduse arhitektuuri ümber ehitama. Sellises olukorras oleks mõistlik ka ülesanded

liigutada andmebaasi. Käesoleva töö raames on selleks juba üks eeldustest täidetud ning ülesanded on parsitud masinloetavale kujule, mida on lihtsam andmebaasi paigutada.

Kuigi rakendus on piisavalt kiire ja kasutusmugavuse uuringus ei tekkinud kasutajatel etteheiteid, siis saaks selle kiirust veelgi parandada. Veebibrauserid salvestavad üldiselt staatilisi faile vahemällu, et parandada lehe laadimise kiirust. Bakalaureusetöö raames valminud veebirakenduses on vahemälu kasutamine välja lülitatud, sest muudatuste korral võib veebibrauser kuvada rakenduses vanu andmeid. Lehel vahemälu kasutamiseks peaks implementeerima vahemälu haldamiseks lisaloogika.

Käesoleva töö autor plaanib ka peale bakalaureusetöö valmimist veebirakendusele tuge pakkuda ning implementeerida eelnevalt kirjeldatud ideid. Õpetajate tagasisidest ajendatult on lähitulevikus plaanis rakendusele lisada viktoriinide koostamise funktsionaalsus, mis võimaldaks luua nii interaktiivseid kui PDF-failina eksporditavaid viktoriine.

Kokkuvõte

Informaatika- ja arvutialased teadmised on nüüdisühiskonnas üha vajalikumad ning nende teemade tutvustamist alustatakse juba üldhariduskoolides. Lisaks koolitundidele tutvustatakse valdkonda paljudes koolides süvitsi innustatades õpilasi osa võtma informaatikavõistlustest. Eesti üks suurim selline ainevõistlus on informaatikaviktoriin Kobras, mida korraldatakse aastast 2006.

Iga aasta koostatakse viktoriinil esitamiseks ligikaudu 90 erinevat ülesannet, mis on saadaval nii eesti kui vene keeles. Ülesanded avalikustatakse peale viktoriini toimumist ka Kobras veebilehel. Huvipakkuvate ülesannete leidmiseks ja nende filtreerimiseks puudus aga senisel veebilehel võimalus, sest kõik ülesanded olid paigutatud voorude ja vanuserühmade kaupa eraldi alamlehtedele. Ühtlasi puudus võimalus ülesannete interaktiivseks lahendamiseks ning veebilehe käsitsi uuendamine oli korraldajatele tülikas.

Käesoleva bakalaureusetööna valmis uus veebirakendus informaatikaviktoriinile Kobras (https://mattiassokk.gitlab.io/kobras-demo/), mis kõik eelnevalt mainitud probleemid lahendas. Loodi uus staatiline veebirakendus, mis viktoriini haldamiseks kasutusel olevas Git repositooriumis muudatuste korral andmed tekstikujult struktueeritud vormi parsib ja teeb uues veebirakenduses kättesaadavaks. Lisati võimalus kõiki varasemaid viktoriini ülesandeid filtreerida ning rakenduses interaktiivselt lahendada. Valminud rakendust testiti õpetajate seas ning selgitati välja, et loodud veebirakendus vastab kasutusmugavuse nõuetele ning kõik huvipakkuv info on vähese vaevaga ja intuitiivselt leitav. Õpetajate hinnangul leiab valminud veebirakendus rohkem kasutust kui senine viktoriini veebileht – seega võib arvata, et viktoriini jaoks loodavad ülesanded jõuavad edaspidi rohkemate õpilasteni ning töö eesmärk on edukalt täidetud.

Valminud lahendus tõestab, et piiratud nõuetega rakenduste loomiseks saab edukalt kasutada staatilise veebirakenduse arhitektuuri, mis ei võimalda kasutada andmebaase. Staatilised veebirakendused pakuvad dünaamiliste veebirakenduste ees mitmeid eeliseid ning tasuta majutusvõimalused võimaldavad veebirakendust hallata igakuiste teenustasudeta. Autori poolt valitud ja realiseeritud arhitektuur toetab veebirakenduse edasiarendamise võimalusi, mis muudaks rakenduse õpetajate ja õpilaste jaoks veelgi atraktiivsemaks.

Viidatud kirjandus

[1] Ettevõtluse Arendamise Sihtasutus (2019). IKT-raport. https://eas.ee/wp-content/uploads/2019/01/IKT-raport.pdf (13.11.2021).

[2] Kobrase koduleheküIg. https://kobras.eio.ee/ (14.11.2021).

[3] Structure of the Bebras Community. https://bebras.org/community.html (14.11.2021).

[4] What is a Bebras task. https://bebras.org/goodtask.html (14.11.2021).

[5] Tartu Ülikooli teaduskool. Kobras. https://teaduskool.ut.ee/et/ainevoistlused/kobras (09.12.2021).

[6] Gruber J. Markdown Introduction. https://daringfireball.net/projects/markdown/ (03.03.2022).

[7] GitLab Flavored Markdown (GLFM). https://docs.gitlab.com/ee/user/markdown.html (03.03.2022).

[8] Hay D. C. Requirements Analysis: From Business Views to Architecture. USA: Pearson Education Inc. Publishing, 2003.

[9] Ramos, M 2016. SSGs Part 1: A Static vs Dynamic Website. https://about.gitlab.com/blog/2016/06/03/ssg-overview-gitlab-pages-part-1-dynamic-x-static/ (07.03.2022).

[10] Tomasis, R 2021. Static vs Dynamic Websites: The Differences, Advantages and Which to Use. https://www.wix.com/blog/2021/11/static-vs-dynamic-website/ (07.03.2022).

[11] Working with JSON. https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON (10.03.2022).

[12] Parser Generator for JavaScript. https://pegjs.org/documentation (10.03.2022).

[13] Andmekaitse ja infoturbe leksikon. https://akit.cyber.ee/ (10.03.2022).

[14] Gulp: a toolkit to automate & enhance your workflow. https://github.com/gulpjs/gulp (17.03.2022).

[15] What is CI/CD? https://www.redhat.com/en/topics/devops/what-is-ci-cd (17.03.2022).

[16] GitLab CI/CD. https://docs.gitlab.com/ee/ci/ (17.03.2022).

[17] Breedis, R. JavaScripti kasutajaliidese raamistike võrdlus. TÜ arvutiteaduse instituudi bakalaureusetöö, 2021. https://comserv.cs.ut.ee/ati_thesis/datasheet.php?id=72077 (23.03.2022).

[18] Vue.js Introduction. https://vuejs.org/guide/introduction.html (23.03.2022).

[19] Typescript: JavaScript with syntax for types. https://www.typescriptlang.org/ (26.03.2022).

[20] Vue Router Introduction. https://router.vuejs.org/introduction.html (30.03.2022).

[21] Mitt: Tiny 200 Byte Functional Event Emitter. https://github.com/developit/mitt (30.03.2022).

[22] Marked Documentation. https://marked.js.org/ (03.04.2022).

[23] Yablonski J. Laws of UX: Using Psychology to Design Better Products & Services. USA: O'Reilly Media, Inc., 2020.

[24] Fitzgerald, A., 2021. The Essential Guide to Website Navigation. https://blog.hubspot.com/website/main-website-navigation-ht (27.03.2022).

[25] Material Design Introduction. https://material.io/design/introduction (24.03.2022).

[26] Pattakos A, 2022. Vue UI Component Libraries. https://athemes.com/collections/vue-ui-component-libraries/ (24.03.2022).

[27] Introduction to Quasar. https://quasar.dev/introduction-to-quasar (24.03.2022).

[28] Moran K, 2021. Qualitative Usability Testing: Study Guide. https://www.nngroup.com/articles/qual-usability-testing-study-guide/ (15.04.2022).

Lisad

1. Rakenduse lähtekood

Käesoleva bakalaureusetööna valminud veebirakenduse lähtekood on jaotatud kahte GitLab repositooriumisse.

1. https://gitlab.com/mattiassokk/kobras-demo
   aadressil asub veebirakenduse enda lähtekoodi sisaldav repostoorium, mis on ehitatud kasutades Vue.js JavaScript raamistikku.
2. https://gitlab.com/mattiassokk/kobras-data
   aadressil asub varasemate ülesannete, tulemuste ja andmete repositooriumi koopia. See sisaldab parsimise loogikat, mis käivitub CI/CD töö abil automaatselt ning parsib ja teeb kättesaadavaks repositooriumist seatud tingimustele vastavad failid.

2. Loodud veebirakenduse testimise kava õpetajatel

1. **Sissejuhatus.**
    a. Enda tutvustus
    b. Lõputöö tutvustus
    c. Tänamine osalemise eest
2. **Küsimused testija kohta, tema kokkupuude viktoriini ja selle senise veebilahendusega.**
    a. Testija enda lühitutvustus
    b. Kui kaua Te juba õpilastega Kobrase viktoriinil osalenud olete? Mis on Teie senine kogemus?
    c. Kas valmistate õpilasi viktoriini jaoks tundides ette või tutvustate selleks võimalusi? Kuidas ettevalmistus välja näeb? Kas lahendate ka varasemaid viktoriinil esinenud ülesandeid?
    d. Mis te praegusest Kobrase veebilahendusest arvate? Kas satute lehele tihti või pigem ei kasuta seda lehte üldse õppetöös? (Praegune veebileht: https://kobras.eio.ee/)
    e. Kui tõenäoline on, et kasutaksite lehte tihemini, kui see pakuks rohkem funktsionaalsusi? Millistest funktsionaalsustest olete tundnud enim puudust?
3. **Annan ülevaate testimisest.**
    a. Küsin, et testija jagaks ekraani.
    b. Seletan testimise eesmärki.
    c. Annan napisõnaliselt ülevaate uuest veebirakenduset ning annan selle lingi.
4. **Viin testimise läbi. Lasen õpetajal järgnevaid kasutusjuhte simuleerida.**
    a. Üks Teie 10. klassi õpilane pääses sel õppeaastal viktoriini teise vooru ja tuli Teie juurde sooviga, et seletaksite talle ülesannet pealkirjaga “Puude langetamine”. Otsige see ülesanne üles.
    b. Üks Teie 8. klassi õpilastest pääses viktoriini lõppvooru ja tahab seal hea tulemuse saada, kuid tunneb end nõrgalt graafiteooriaga seotud ülesannetes. Leidke graafiteooriaga seotud ülesandeid.
    c. Soovite 7. klassi informaatikatunnis enda ekraani jagades koos klassiga 2020/21. aasta ülesandeid lahendada, et aktiivsematele kaasamõtlejatele lisapunkte anda. Peale iga ülesannet selgitate, kuidas ülesannet lahendada ning tutvustate, kuidas see ülesanne on informaatikaga seotud.

d. 2021/22. aasta teine voor just lõppes ja soovite vaadata kõiki selle vooru ülesandeid õigete vastustega.
e. 2019/20. aasta esimene voor just lõppes ja soovite vaadata, kes Teie õpilastest said edasi teise vooru.
f. 2019/20. aasta teine voor just lõppes ja Teie ***X*** klassi õpilane ***Eesnimi Perenimi*** soovib teada, mitmenda koha ja mitu punkti ta sai.
g. Soovite näha millal toimub viktoriini järgmine voor.

5. **Lasen testijal veidi rakendust vabalt kasutada ja tagasisidet anda.**
6. **Esitan uut rakendust puudutavad üldisemad küsimused.**
    a. Kui intuitiivne uue rakenduse kasutamine Teie hinnangul oli?
    b. Kas simuleeritud kasutusjuhud tundusid reaalsed või pigem selliseid situatsioone tegelikkuses ei ole?
    c. Kas uus rakendus jätab parema mulje kui senine? Kui tõenäoliselt leiaks uus rakendus rohkem kasutust Teie informaatikatunnis?
    d. Kui tõenäoliselt soovitaksite uut rakendust informaatikast huvi tundvatele õpilastele (kes valmistub Kobrase viktoriiniks)?
    e. Kas Teil endal on veel ideid või ettepanekuid?
7. **Lõpetussõnad, tänan veelkord testimises osalemast.**

3. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, **Mattias Sokk**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose

   Veebirakenduse loomine informaatikaviktoriinile Kobras,
   mille juhendaja on **Lidia Feklistova**,

   reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada digitaalarhiivi DSpace kuni autoriõiguse kehtivuse lõppemiseni;

2. annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi DSpace kaudu Creative Commonsi litsentsiga CC BY NC ND 3.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni;

3. olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile;

4. kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

Mattias Sokk
10.05.2022