Tartu Ülikool
Arvutiteaduse instituut
Informaatika õppekava

Tobias Reiter

Testide teisendamine tekstidokumendist struktuursele kujule

Bakalaureusetöö (9 EAP)

Juhendaja:
Jaak Vilo, PhD

Tartu 2024

Testide teisendamine tekstidokumendist struktuursele kujule

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärk on veebipõhise rakenduse loomine Tartu Ülikoolile. Valikvastustega testide kasutus on e-õppes laialdane, kuid nende formaatide mitmekesisus ja spetsiifilised rakendused teevad raskeks nende koosloome ning eri keskkondade vahel taaskasutuse. Loodav süsteem võimaldab üles laadida Microsoft Wordi faili, mida saab ühiselt luua keskkondades Google Drive või MS Online. Peale üleslaadimist tagastatakse valitud väljundformaadis tulem, mille saab vastavasse õpikeskkonda – Tartu Ülikooli puhul Moodle'isse või Courserasse – laadida. Käesolev bakalaureusetöö on suuremas osas praktiline. Teoreetiline osa annab ülevaate, miks on rakendus vajalik ja analüüsib kahe õpikeskkonna testide ekspordi ja impordi vahelisi erinevusi. Edasi kirjeldatakse rakenduse töökäiku ja arengut ning tuuakse välja põhilised tööpõhimõtted ja arhitektuursed valikud. Teoreetilises osas selgitatakse tehtud rakenduse edasiarendamise võimalusi ja missuguseid tehnoloogiaid süsteemi loomiseks kasutati.

Võtmesõnad:

Veebirakendus, Moodle, Coursera, teisendamine, Microsoft Word, Google Docs, Javascript, Next.js, Java, Spring

CERCS:

P175 Informaatika, süsteemiteooria

Converting tests from text document to structural shape

Abstract:

The main aim of this Bachelor's thesis is to create a web application for the University of Tartu. The use of multiple choice tests is widespread in e-learning, but the variety of these formats and their specific applications make it difficult for them to co-create and re-use across different environments. The system allows you to upload a Microsoft Word file that can be created jointly in Google Drive or MS Online environments. After uploading, the result will be returned in the selected output format, which can be uploaded to the relevant study environment - Moodle or Coursera, in the case of the University of Tartu. This Bachelor's thesis is primarily practical. The theoretical section explains why the application is necessary and the technologies used to create the system. The process and development of the application will be described further, and the basic operating principles and architectural

choices will be outlined. The theoretical part also analyzes the export and import of the two learning environment tests and explains the possibilities for further application development.

Keywords:

Web application, Moodle, Coursera, converting, Word, Google Docs, Javascript, Next.js, Java, Spring

CERCS: P175 Informatics, systems theory

Sisukord

Sissejuhatus

Õpikeskkondade kasutamine on viimastel aastatel jõudsalt kasvanud, pakkudes õppijatele ja õpetajatele üha enam võimalusi muuta õppeprotsess mitmekesisemaks ja efektiivsemaks. Tartu Ülikool ja teised õppeasutused on pannud suurt rõhku digitaalsetele vahenditele, et muuta õppeprotsess kättesaadavamaks ning mugavamaks. Siiski eksisteerivad tehnilised piirangud, näiteks testide loomise aja- ja töömahukus, õpetajatevaheline koostöö testide loomisel ja võimalus tõsta küsimusi ühest keskkonnast teise. Nimetatud kitsaskohad vajavad uuenduslikke lahendusi, et säästa õppejõudude väärtuslikku aega ning muuta õppematerjal kvaliteetsemaks.

Bakalaureusetöö eesmärk on luua veebirakendus, mis võimaldab Tartu Ülikooli õppejõududel kiiremini ja mugavamalt hallata ning jagada teste Moodle'i ja Coursera platvormide vahel. Töös keskendutakse veebirakenduse loomisele, milles on võimalik üles laadida Microsoft Wordi faile, mida saab luua koostöiselt Google Drive või MS Online keskkondades. Peale üleslaadimist tagastatakse valitud väljundformaadis tulem, mille saab vastavasse õpikeskkonda laadida. Lahenduse eesmärk on muuta õppejõudude koostöö mugavamaks testide loomisel ning lihtsustada õppetestide loomist ning haldamist.

Veebirakenduse arendamine tugineb Moodle'i ja Coursera õpikeskkondade impordi- ja ekspordivõimaluste põhjalikule uurimisele ning analüüsile. Arenduse protsessis on arvestatud õppejõudude märkusi veebirakenduse osas ning on valitud mugav ja igapäevatöös laialt kasutatud keskkond koostöö tegemiseks. Tehnoloogia valikul oli määravaks teguriks kaasaegsus ning rakenduse edasiarendamise võimalused.

Bakalaureusetöö ühendab teoreetilised teadmised ja praktilise arendustöö. Töö on jaotatud kuueks peatükiks. Esimeses peatükis saab lugeja tuttavaks töös kasutatud terminite ja mõistetega. Teise peatüki rõhk on õppejõudude tööprotsessi toetamisel ning õpetajatepoolsel tagasisidel, mida võiks rakendus sisaldada, sealhulgas selgitavad õppejõud täpsemalt, kuidas on seni testide koostamise töövoog välja näinud. Kolmandas peatükis saab ülevaate Tartu Ülikoolis kasutusel olevast kahest õpikeskkonnast, Moodle'ist ja Courserast. Neljandas peatükis tutvustatakse arvutiteaduse instituudi testide näidisraamistikku, mille õppejõud võtavad aluseks, et hakata teste koostama. Viienda peatüki rõhk on veebirakenduse kirjeldamisel ja analüüsimisel ning kuuendas peatükis on rakenduse arenguvõimalused ja teenusehaldus. Lisadest on võimalik leida küsimused, mida küsiti õppejõududelt, rakenduse kasutusjuhend, Moodle XML-vormeering, Tartu ülikoolile loodud näidismall, testimiseks

kasutatud näidisdokument ning litsents.

1. Mõisted ja terminid

Mõistete kirjapanemiseks on kasutatud https://akit.cyber.ee/.

Parsimine (ingl *parsing*) on protsess, mille käigus analüüsitakse teksti või andmeid vastavalt teatud reeglitele, et muuta need arvutiprogrammi jaoks üheselt mõistetavaks vormiks.

Eesliides (ingl *frontend*) ehk kasutajaliides, mis on nähtav ja töödeldav kasutajale.

Tagasüsteem (ingl *backend*) ehk serveripoolsed tehnoloogiad, mis töötavad taustal ja pole nähtavad rakenduse kasutajale.

Üheleheline rakendus (ingl *Single page application)* ehk veebirakendus, mis laeb kogu lehe sisu esimese korraga ning hiljem laeb ainult väikeseid osasid.

Rakendusliidese (ingl *application programming interface,* API) ehk liides, mis võimaldab erinevatel rakendustel omavahel suhelda.

Serveripoolne laadimine (ingl *server-side rendering*, SSR) ehk sisu genereeritakse serveris ning saadetakse kasutajale eeltöödeldud kujul.

Hüperteksti märgistuskeel (*HyperText markup language*, ka HTML) ehk märgistuskeel, mis on populaarne veebirakenduste struktuuri loomisel.

Kaskaadlaadistik (ingl *Cascading Style Sheets*, ka CSS) ehk stiilikeel, mida kasutatakse kujunduse loomiseks rakendustele.

2. Testide koostamise töövoog

Peatükk annab ülevaate õppejõudude testide koostamise töövoo kohta, tuginedes Tartu Ülikooli õppejõududega peetud kuuele vestlusele, kes kasutavad Moodle'i või Coursera teste oma õppeaines (vt esitatud küsimusi lisa 1). Eesmärk on mõista testide koostamise protsessi kuni käesoleva hetkeni ning uurida viise, kuidas muuta töövoogu efektiivsemaks ja testide sisu hariduslikult väärtuslikumaks. Testide koostamine on muutunud õppeprotsessis aina olulisemaks, mõjutades nii õppeprotsessi ennast kui ka omandatud teadmiste hindamist, tuginedes eelmise aasta Tartu Ülikooli E-õppe statistikale [1]. Statistikast on näha, et alates aastast 2010 on ülikoolis igal aastal e-kursuste arv pidevalt suurenenud. Peatükist saab arusaama, kuidas õppejõud koostavad hetkel teste ning mida nad sooviksid testide koostamise töövoo juures parandada.

Kuue vestluse käigus selgus, et suurem osa õppejõududest alustavad protsessi ühisdokumendis ning enamasti kasutatakse Moodle'i keskkonda, et teste läbi viia. Üks õppejõud avaldas soovi viia enda kursus Coursera keskkonda, kuid see aga tähendaks käsitsi kõikide testide uuesti koostamist ning suurenevat ajakulu. Õppejõudude seas on populaarne teha ühisdokument Google Docs'is, kuna keskkond võimaldab reaalajas informatsiooni jagamist. Peamiselt kasutavad õppejõud testide loomisel valikvastuseid, mitme õige vastusega küsimusi ning avatud küsimusi. Samuti kasutatakse küsimustes ning vastustes pilte. Üks õppejõud mainis õppemängu Kahoot kasutamist, et pakkuda õpilastele võistlusmomenti ja lõbusat vaheldust, et õpilastele pigem teadmisi meenutada, kui et neid kontrollida.

Hetkel on testide koostamine õpikeskkonnapõhine, põhjustades õppejõududele lisatööd. Õpetajad alustavad testide loomise protsessi ühisdokumendi koostamisest, kus genereeritakse ideid ja arutatakse küsimuste väärtust teiste kolleegidega, samuti toimetatakse testiküsimusi. Seejärel tuleb ühel õppejõul kõik küsimused sisestada käsitsi valitud õpikeskkonda, mis nõuab palju aega, kuna tuleb järgida valitud keskkonnanõudeid. Õpetajad ei olnud teadlikud teenustest, mis pakuvad testide konverteerimist, osaliselt ka seetõttu, et turul ei ole selliseid vabavaralisi lahendusi.

Eelmainitu andiski idee arendada lihtne, samas efektiivne, koostööl põhinev dokument ja veebirakendus, mis võimaldaks õppejõududel koostada ja jagada teste mitte ainult ühe, vaid mitme eri õpikeskkonna vahel, hoides algandmed ja info Wordi failides. See vähendab

õpetajate lisatööd avades samaaegselt uusi võimalusi õppematerjalide kvaliteedi ja kättesaadavuse parandamiseks.

Selle lähenemisviisi rakendamiseks loodi eraldi dokumendi mall (vt lähemalt peatükk 4), kus on olemas küsimuste koostamise juhend ja näidisküsimused. Õppejõududega vestlusest selgus, et ühisdokumendiks eelistatakse .docx laiendiga Google Docsi või Microsofti pilvelahendust. Seega tuvastab ka käesoleva töö raames tehtav veebirakendus ainult .docx laiendiga manuseid.

Tuginedes Tartu Ülikooli õppejõudude reaalsetest kogemustest ja väljakutsetest, on selge, et testide koostamise töövoogu on võimalik parandada. Muudatuste rakendamine võib positiivselt mõjutada õppekvaliteeti ja õpilaste ning õpetajate rahulolu, pakkudes tõhusamat ja kaasahaaravamat õpikeskkonda.

3. Õppekeskkondade taust

Käesolev peatükk annab ülevaate teoreetilisest raamistikust, mis toetab hiljem bakalaureusetöös läbiviidavat praktilist osa. Kirjeldatakse kahe peamise õpikeskkonna – Moodle'i ja Coursera – omadusi ja eripärasid. Täpsemalt on lahti kirjutatud Moodle'i ja Coursera küsimuste import- ning eksportvõimalused erinevatesse vormingutesse.

3.1 Moodle'i keskkond

Moodle [2, 3] on avatud lähtekoodiga õppehaldussüsteem, milles on võimalik luua kursusi ning neid kujundada. Kasutusel ülemaailmselt – sellel on üle 390 miljoni kasutaja ja üle 46 miljoni kursuse. Ka Tartu Ülikoolis on kõige laialdasemalt kasutusel just Moodle'i õpikeskkond. Mainitud õppehaldussüsteem pakub mitmeid erinevaid failivorminguid testide importimiseks, täpsemalt neli: 1) GIFT (*General Import Format Template*), 2) Moodle XML (*Extensible Markup Language*), 3) Aiken ning 4) Blackboard. Moodle'i jaoks soovitatakse kasutada Moodle enda loodud XML-vormingut, viidates, et see suudab salvestada kõige rohkem andmeid [4].

3.1.1 GIFT

GIFT formaat võimaldab koostada küsimusi valikvastustega, õige/vale küsimustega, lühivastustega, puuduvate sõnadega, esseedega jne (vt joonis 1).

```
Küsimus? {
    ~%-50%Vale vastus #tagasiside
    ~%50%Õige vastus #tagasiside
    ~%50%Õige vastus #tagasiside
    ~%-50%Vale vastus #tagasiside
    #### Üldine tagasiside
}
```

Joonis 1. GIFT valikvastusega küsimuse näidis.

Mainitud formaat on peamiselt mõeldud ning välja arendatud Moodle'i kogukonnas. GIFT-i on võimalik eksportida ka Moodle'st välja, mis teeb küsimuste töötlemise lihtsamaks. Samuti on käesolevat formaati kasutades võimalik lisada küsimustele pilte ning helisid [5].

3.1.2 Moodle XML

XML-vorming on Moodle'i kogukonna poolt loodud ning kasutusel enamasti vaid Moodle'i keskkonnas. XML-parser on väga tundlik kirjavigadele ning vigasid automaatselt vorming ei tuvasta.

```
<questiontext format="html">
    <text>küsimus?</text>
</questiontext>
<answer format="html" fraction="-100.0">
    <text> Vale vastus</text>
    <feedback>
        <text> Tagasiside</text>
    </feedback>
</answer>
<answer format="html" fraction="-100.0">
    <text> Vale vastus</text>
    <feedback>
        <text> Tagasiside</text>
    </feedback>
</answer>
<answer format="html" fraction="-100.0">
    <text> Vale vastus</text>
    <feedback>
        <text> Tagasiside</text>
    </feedback>
</answer>
<answer format="html" fraction="100.0">
    <text> Õige vastus</text>
    <feedback>
        <text> Tagasiside</text>
    </feedback>
</answer>
<generalfeedback>
    <text>Üldine tagasiside</text>
</generalfeedback>
```

Joonis 2. Moodle XML valikvastusega küsimuse näidis.

Sarnaselt GIFT formaadiga on ka XML-iga (näide XML-vormingust vt joonis 2 ja lisa 3) võimalik koostada küsimusi nii valikvastustega, õige/vale küsimustega, lühivastustega, puuduvate sõnadega, esseedega jne. Samuti lubab see lisada pilte ning helisid. Moodle'i XML-i on võimalik eksportida ka Moodle'ist välja [6].

3.1.3 Aiken

Aikeni formaat on lihtne ja kiiresti õpitav. Formaadi teeb hõlpsaks asjaolu, et küsimusi on võimalik koostada vaid valikvastustega (vt joonis 3).

```
küsimus?
A) Vale vastus
B) Vale vastus
C) Vale vastus
D) Õige vastus
ANSWER: D
```

Joonis 3. Aikeni valikvastusega küsimuse näidis.

Samuti tohib küsimustel olla ainult üks õige vastus [7]. Seejuures võrreldes näiteks varasemalt mainitud formaatidega pole Aikeniga võimalik lisada küsimustele pilte ega helisid. Formaat tuleb salvestada .txt failina ning kasutada UTF-8 (*UCS Transformation Format 8*) kodeeringut [8].

3.1.4 Blackboard

Blackboard on õppehaldussüsteem, kus on samuti võimalik koostada teste ning kursuseid. Moodle toetab küsimuste importfailina Blackboardi dat- või zip-faile. Blackboard toetab mitmesuguseid küsimuste tüüpe, sealhulgas valikvastustega, õige/vale, lünkade täitmisega, lühivastustega ja esseeküsimusi. Formaat toetab helide ja piltide lisamist. Blackboardi eelistatakse teistele formaatidele, kuna see ühildub mitmete testi koostamise programmidega, mis muudavad testi loomise protsessi mugavamaks ja ajasäästlikumaks [9].

3.2 Coursera keskkond

Coursera on ulatuslik avatud veebikursuste pakkuja (*Massive Open Online Courses, MOOC*) ehk keskkond, kus on suurel hulgal tasuta veebipõhiseid kursuseid. Platvorm on tuntud kõrgkvaliteetsete kursuste ja õppematerjalide poolest, mida pakuvad maailma tippülikoolid ja organisatsioonid. Coursera keskkonnas on 113 miljonit kasutajat ning üle 7000 asutuse [10]. See võimaldab ülikoolidel luua kursuseid ning koostada teste materjalide alusel. Küsimuste importimiseks on Courseras erinevaid variante. Küsimusi on võimalik lisada ja eksportida järgnevates formeeringutes: YAML (*Yet Another Markup Language*), Microsoft Word/Google Docs ja QTI. Coursera dokumentatsioon sisaldab küsimuste impordi osas kõige rohkem infot just Microsoft Wordi formeeringute kohta [11].

3.2.1 Microsoft Word või Google Docs

Coursera võimaldab küsimusi importida kasutades Google'i pilvelahendust ning samuti on võimalik Microsoft Word faile otse importida. Formaat annab võimaluse koostada küsimusi

valikvastustega, lünkade täitmisega ja lühivastustega jne. Samuti on Coursera dokumentatsioonis põhjalik juhend, kuidas koostada küsimusi ning mida tuleks nende moodustamisel silmas pidada [11].

3.2.2 YAML

YAML (*Yet Another Markup Language*) on mõeldud kasutajatele, kes tunnevad ennast kindlamalt koodi kirjutades või näiteks robotitele, kes suudavad genereerida küsimusi YAML-formeeringus (vt joonis 4).

```
prompt: küsimus?
options:
  - answer: Õige vastus
    isCorrect: true
    feedback: Tagasiside
  - answer: Vale vastus
    isCorrect: false
    feedback: Tagasiside
  - answer: Vale vastus
    isCorrect: false
    feedback: Tagasiside
```

Joonis 4. YAML valikvastusega küsimuse näidis.

See on loetav nii inimestele kui masinatele ning seetõttu on suurte küsimuste mahtude puhul võimalik kirjutada ka koodirida, mis koostab küsimusi automaatselt, olles võimeline moodustama ka eri tüüpi küsimusi. Samas tasub märkida, et YAML-i suur miinus on ülim tundlikkus. See tähendab, et faili käsitsi muutes võivad kergelt esineda süntaksivead. Selliste vigade üles leidmine võib aga suurtest failidest osutuda aeganõudvaks. Samuti pole piltide ega helide lisamine YAML-failidesse võimalik [12, 13].

3.2.3 QTI formeering

QTI (*Question and Test Interoperability*) on universaalne formeering, mis aitab erinevate õppehaldussüsteemide vahel küsimusi ja teste jagada.

```
2. küsimus?
*a. Õige vastus
@ Tagasiside
b. Vale vastus
@ Tagasiside
```

Joonis 5. QTI valikvastusega küsimuse näidis.

Formaat toetab Courseras valikvastustega, õige/vale ja lünkade täitmine/lühivastustega küsimusi (vt joonis 5). Coursera on formaadi teinud dokumentatsiooniga lihtsaks ning arusaadavaks [14].

3.3 Kahe keskkonna analüüs ja võrdlustabelid

Kaks õpikeskkonda erinevad üksteisest erinevate lähenemiste poolest küsimuste impordile ja ekspordile (vt lähemalt tabel 1). Kõik formeeringud olid Moodle'i ja Coursera keskkonna vahel erinevad.

Tabel 1. Õpikeskkondade import ja eksport.

Õppekeskkond	Google Docs/ Microsoft Word	YAML	Blackboard	GIFT	Moodle XML	Aiken	Coursera QTI
Coursera(eksport)	X	X	-	-	-	-	X
Coursera(import)	X	X	-	-	-	-	X
Moodle(eksport)	-	-	-	X	X	X	-
Moodle(import)	-	-	X	X	X	X	-

Moodle'i impordi ja/või ekspordi jaoks tuleks kasutada nende enda Moodle'i XML-vormingut, kuna see võimaldab koostada erinevat tüüpi küsimusi koos piltide ja helidega. Coursera impordi ja/või ekspordi tarbeks on tugev dokumentatsioon loodud nii Google Docsi kui ka Microsoft Wordi jaoks. Mõlemad formeeringud toetavad piltide lisamist ning erinevaid küsimuste tüüpe.

Tabel 2. Coursera küsimustüüpide võrdlustabel.

Küsimusetüüp	YAML	QTI formeering	Google Docs/ Microsoft Word
Valikvastused	X	X	X
Õige/Vale		X	X
Lühivastused	X	X	X
Lünkade täitmine		X	X
Esseed			X
Pildid			X
Helid			

Saab öelda, et Microsoft Word tagab kõige laialdasema küsimustüüpide toetuse. YAML piirab küsimuste loomist märgatavalt, lubades ainult valikvastuseid ja lühivastuseid ning QTI lisaks eelnevale lubab veel õige/vale küsimusi ja lünkade täitmist (vt lähemalt tabel 2). Coursera pakub põhjalikku dokumentatsiooni Microsoft Wordi formeeringule ning lisavõimalusi edasiarenduseks. Seetõttu otsustati valida just need formeeringud Coursera keskkonna impordi jaoks.

Tabel 3. Moodle küsimustüüpide võrdlustabel.

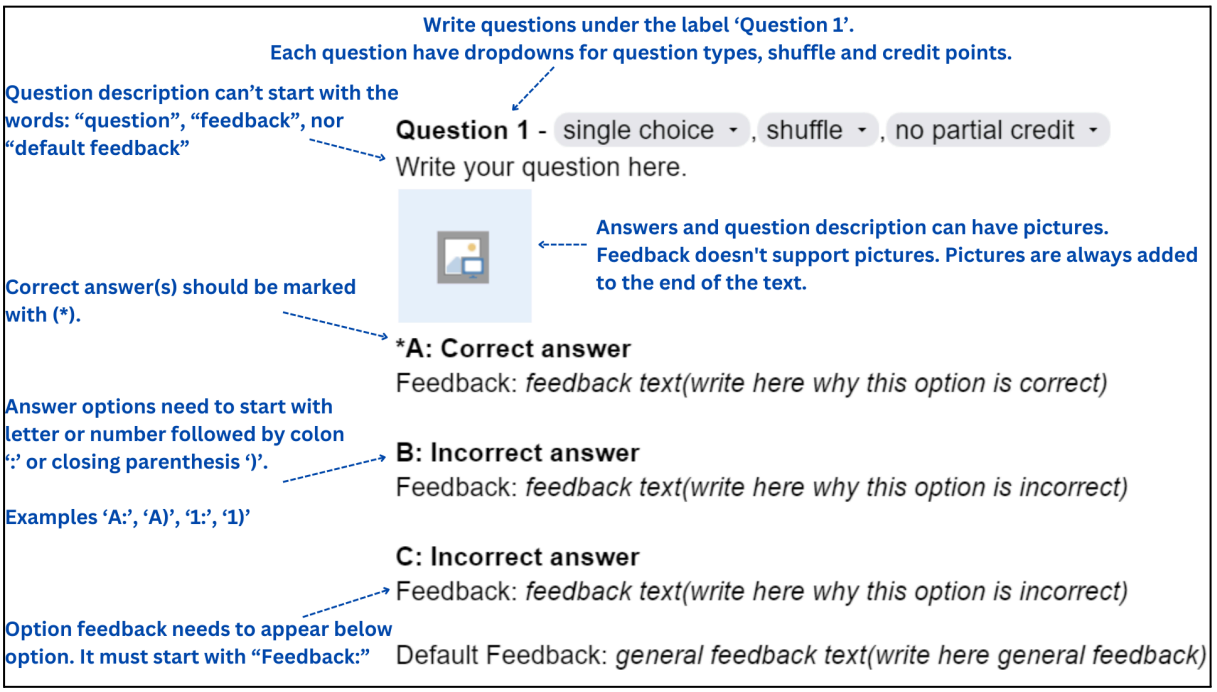
Küsimusetüüp	GIFT	Moodle XML	Aiken	Blackboard
Valikvastused	X	X	X	X
Õige/Vale	X	X		X
Lühivastused	X	X		X
Lünkade täitmine	X	X		X
Esseed	X	X		X
Pildid	X	X		X
Helid	X	X		X

Tabel 3 põhjal võib öelda, et Moodle'i keskkond on paindlikum. Aiken on ainuke formeering, mis ei toeta kõiki küsimustüüpe. Moodle pakub enda poolt välja töötatud formeeringut ning seetõttu osutus Moodle XML antud keskkonna impordi formeeringuks.

4. Tartu Ülikooli jaoks oma malli loomine

Käesolev peatükk kirjeldab Tartu Ülikooli arvutiteaduse instituudi (UniTartuCS) malli vajalikkust ning tutvustab juhendit. UniTartuCS mall on vajalik õppejõudude töö lihtsustamiseks ning koostamise protsessi kiirendamiseks. Dokument annab ette kindla struktuuri, millesse on võimalik koostada küsimusi vastavalt küsimuse tüübile ja lisada juurde valikuid. Antud struktuur on oluline, et arendatud rakendus suudaks aru saada küsimuste tüüpidest ning lisafunktsioonidest.

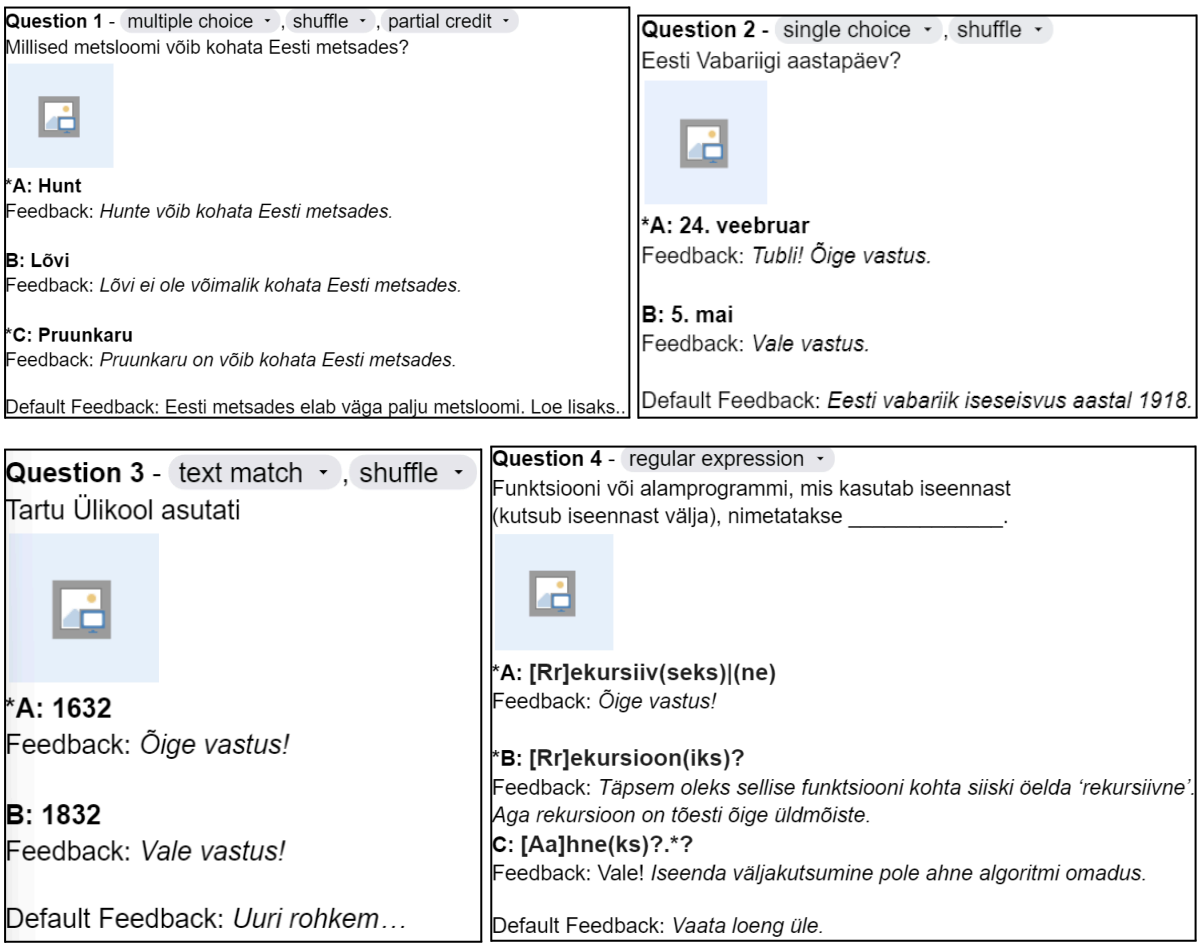
UniTartuCS mall (vt lähemalt lisa 4) koos juhendiga (vt lähemalt lisa 2) loovad selged juhised, kuidas koostada konverteerimiseks valmis olev dokument. Juhendi järgimine aitab ära hoida vigaseid küsimusi ja arusaamatusi. Joonis 6 visualiseerib, milline näeb välja ühe õige vastusega küsimus. Samuti on kirjeldatud reeglid, mida tuleb järgida. Küsimus tuleb antud pildi kontekstis koostada "*Question* 1" kirja alla ning igal küsimusel on olemas rippmenüüd küsimuste tüüpide, järjekorra segamise ja hindamise kohta.



Joonis 6. Küsimuse koostamise õpetus.

Struktuuri koostamisel oli rõhk lihtsusel, et õppejõududele oleks kõik arusaadav (vt lähemalt joonis 7). Küsimuse ja vastuste lõppu on võimalik lisada ka pilte. Küsimust ei tohi alustada sõnadega "*question*", "*feedback*" ja "*default feedback*". Vastusevariantide arvu saab õppejõud ise valida, aga oluline on silmas pidada, et õige vastusevariandi ette oleks lisatud tärn ('*'),

seejärel täht või number, kooloni ":" või sulgeva sulu ")" järele vastusevariant. Vastusest järgmisele reale on võimalik lisada tagasiside rida. Tagasiside ei toeta pilte ning tagasiside lisamine on vabatahtlik. Kui märkusi lisada pole, võib jätta rea tühjaks. Peale vastusevariantide valmimist on võimalik lisada ka üldine tagasiside, kus õppejõud saab anda laiema tagasiside küsimuse kohta. See võimalus on Moodles iga küsimusega võimalik aga Courseras ainult "*Text match*" küsimuse tüübiga. Üldiseks tagasisideks võivad olla seletused või selgitused kuidas jõuti lahenduseni.



Joonis 7. Näidisküsimused Question 1..4. Küsimuse nr 4 on koostanud juhendaja Jaak Vilo.

Kõikidel õppejõududel on võimalus alla laadida UniTartuCS malli (vt lähemalt lisa 4) ja alustada testide loomisega, tehes vajadusel koostööd teiste õppejõududega. Testi valmimise korral tuleb dokument laadida testi konverteerimise rakendusse ja rakendus tagastab Moodle'i või Coursera import formaadis faili.

5. Veebirakenduse kirjeldus ja analüüs

Viies peatükk keskendub veebirakenduse arendusprotsessile, arhitektuurile, kasutatud tehnoloogiatele ja kasutajakogemusele. Eesmärk on anda täpne ülevaade valminud veebirakendusest – miks on eelistatud teatud tehnoloogiaid ja kuidas valikud toetavad lõppeesmärki pakkudes kasutajale võimalikult head kasutajakogemust.

5.1 Ülevaade veebirakendusest

Veebirakenduse eesmärk on pakkuda Moodle´i ja Coursera testide loojatele võimalust koostada teste ühisdokumendis, hiljem need konverteerida veebirakenduses ümber Moodle XML- või Coursera .docx-formeeringusse. Konverteeritud faile on lihtne üles laadida õpikeskkondades, salvestades küsimused küsimustepanka. Peamiseks sihtrühmaks on õppejõud, kes koostavad kursustele teste ja teadmistekontrolle.

Uue rakenduse peamine eelis on see, et süsteem on vabavaraline, olemas on UniTartuCS mall (vt lähemalt peatükk 4) ja koostamise juhend (vt lähemalt lisa 2). Mallis on võimalik koostada hõlpsasti teste, mida on hiljem võimalik konverteerida Moodle'i või Coursera keskkondade jaoks ümber. Ülesandepüstitus tuli juhendaja ettepanekust muuta testide tegemine lihtsamaks ning luua ka mitte-spetsialistile arusaadav lahendus, kus oleks lihtne teha koostööd teiste õppejõududega. Lisaks on oluline, et saaks jagada oma tegevust, kommenteerida või pidada vestlust teiste õppejõududega. Oluline on faili üleslaadimisel kuvatav statistika dokumendi küsimuste ja nende tüüpide kohta ja kasutajale antav tagasiside, toimingu ning konverteerimise õnnestumise osas.

Veebirakenduse eesliides (ingl *frontend*) on minimalistlik ning kasutajasõbralik, päises on nupp dokumentatsioon, kus on rakenduse kasutusjuhend (vt lähemalt lisa 2) ja UniTartuCS mall (vt lähemalt lisa 4). Peamine funktsionaalsus tuleb lehe keskosas, kus on faili lisamise ala, küsimuse detailide avamise nupp ning selle all konverteerimise valikud. Eesliides saadab faili andmed tagasüsteemi (ingl *backend*), kus algab konverteerimise protsess. Enne konverteerimisega alustamist tuleb läbi lugeda kasutusjuhend, soovitatav on kasutada loodud UniTartuCS testide loomise malli, et vältida tõrkeid. Tõrgete mitteesinemise korral tagastatakse kasutajale konverteeritud fail, mida on võimalik importida vastavasse õpikeskkonda.

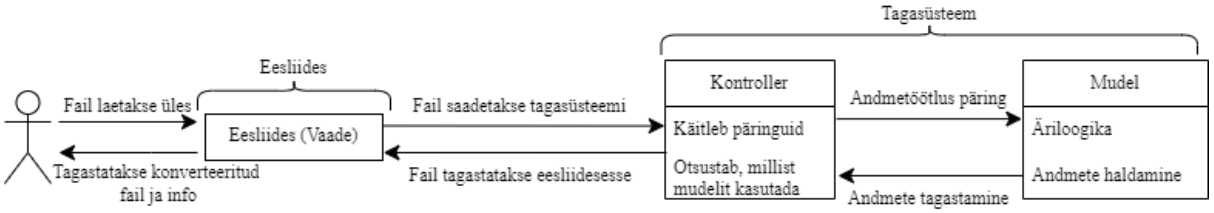
5.2 Veebirakenduse arhitektuur

Veebirakendus kasutab mudel-vaade-kontroller (ingl *Model-View-Controller*, edaspidi ka MVC) arhitektuurimustrit (vt lähemalt joonis 8), mis eristab veebirakenduse kolme erinevat osa:

1) mudel (andmete haldamine ja äriloogika);
2) vaade (esitab andmed kasutajale);
3) kontroller (värskendab vaadet ning vahendab andmeid mudelile).

Selline struktuur aitab tagada rakenduse koodi parema organiseerituse, lihtsustab koodi haldamist ning arendamist [15].

Vaade ehk veebirakenduse eesliides põhineb ühelehelisel rakendusel (ingl *single page application,* edaspidi ka SPA), mis tagab kasutajale kiire ja sujuva veebirakenduse kasutamise kogemuse. SPA arhitektuur võimaldab andmeid dünaamiliselt kuvada ja laadida, mis parandab rakenduse kiirust ning muudab lehe sujuvamaks, kuna lehe sisu ei pea mitu korda laadima.



Joonis 8. Veebirakenduse arhitektuur.

Kasutaja saab eesliidesest valida, millises formaadis soovib faile konverteerida. Valitud failid saadetakse seejärel API kaudu tagasüsteemi, kus toimub failide valideerimine ja konverteerimine. Töödeldud failid koos küsimuste ja vastuste infoga tagastatakse eesliidesele, protsessi illustreerib ka joonis 8. Eesliidese ja tagasüsteemi vaheline suhtlus toimub rakendusliidese (ingl *application programming interface,* ka API) kaudu, mis tagab andmete turvalise ja tõhusa vahetuse ning võimaldab süsteemi komponentide vahel selget kommunikatsiooni.

5.2.1 Tagasüsteemi funktsionaalsus

Tagasüsteem on loodud kasutades Java keelt koos Spring raamistikuga, mis tagab paindliku lahenduse veebirakendusele. Spring raamistik pakub terviklahendust Java põhistele veebirakendustele võimaldades sellega ära kasutada rikkalikke tööriistu, funktsionaalsust ja

kogukonna tuge [16]. Selline terviklahendus on hea viis arendada Java-põhiseid veebirakendusi, toetades süsteemi arendust ja hooldamist.

Tagasüsteemi struktuur koosneb viiest peamisest moodulist. Esimene moodul on kontroller, mis vastutab kasutajate päringute töötlemise eest. Teine moodul on abimoodul, mis toetab teisi mooduleid funktsioonidega nagu küsimuse lisamine ja valideerimine. Kolmas moodul kirjeldab mudeleid, määratledes kõik süsteemi andmetüübid ja andmeobjektid nende omadustega. Neljas moodul koosneb komponentidest, mis sisaldavad spetsiifilisi funktsioone, näiteks küsimustele detailide loomine. Viiendaks mooduliks on teenused, kus asub konverteerimise loogika ja failide töötlemine.

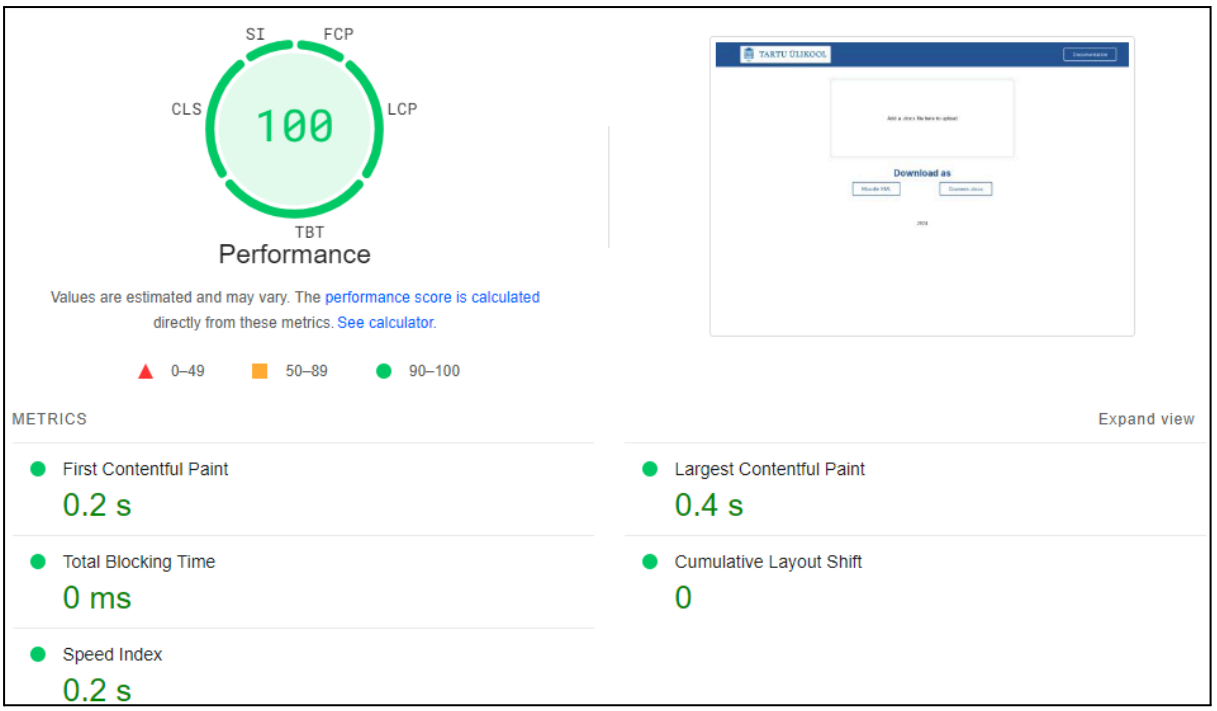
Failitöötluse protsess tagasüsteemis tagab failide täpse ja efektiivse töötluse. Docx failide parsimise jaoks on kasutatud Apache POI teeki [17]. Protsess algab faili vastuvõtmisega eesliideselt, seejärel töödeldakse faili lõikude kaupa, vastavalt etteantud UniTartuCS ühisdokumendi raamile (vt lähemalt lisa 4) ja kasutusjuhendile. Failitöötluse protsess läbib mitu etappi:

1) küsimuse ja küsimuse pildi töötlemine;
2) vastuste ja vastuste piltide töötlemine;
3) tagasiside kontrollimine;
4) üldise tagasiside kontrollimine.

Seejärel algab protsess uuesti järgmise küsimusega, kuni kõik küsimused on töödeldud. Praeguses etapis suudab loodud veebirakendus tõhusalt konverteerida nii ühe kui ka mitme õige vastusega küsimusi, lühikese vastusega küsimusi ja regulaaravaldiste küsimusi. Kui kõik küsimused on edukalt töödeldud ja konverteeritud, siis saadetakse uues formaadis fail tagasi eesliidesesse koos küsimuste informatsiooniga. Kui failis peaks esinema vigu või hoiatusi, antakse sellest kasutajale teada ning veaga küsimust ümber ei töödelda.

5.2.2 Eesliidese funktsionaalsus ja disain

Eesliides on loodud Next.js raamistikuga, mis põhineb Reacti teekil ja võimaldab luua kiireid ja serveri põhiseid (*server-side rendering*, edaspidi ka SSR) veebirakendusi [18]. SSR parandab veebirakenduse jõudlust ja optimeerib laadimisaegasid, mis võib tulevikus rakenduse edasiarendamisel olla oluline (vt lähemalt joonis 9). Veebirakenduse puhul on oluline rõhuda kiirusele ja kasutajakogemusele, mida Next.js pakub tänu oma kaasaegsele arhitektuurile ja lihtsale ülesehitusele.



Joonis 9. Google PageSpeed veebilehe jõudluse ülevaade [19].

Next.js kasutamine võimaldab luua struktuuri, mis toetaks rakenduse edasiarendamist ning haldamist. Eesliidese komponendid kasutavad Reacti teeke. Eelkõige on olulisel kohal taaskasutatavad komponendid nagu näiteks nupud, mis tagavad ühtlase välimuse kogu rakenduses. Veebirakenduse ühtlase disaini jaoks on kasutatud Tailwind CSS (vt lähemalt peatükk 5.4), mis lubab kirjutada otse hüpertekst märgistuskeele (ingl *hyperText markup language,* ka HTML) sisse kujunduskoodi. See kiirendab kujunduse protsessi ning samuti muudab komponendid paremini loetavaks.

Rakenduse põhivaated on:

1) avaleht;
2) kasutusjuhend;
3) päis;
4) jalus.

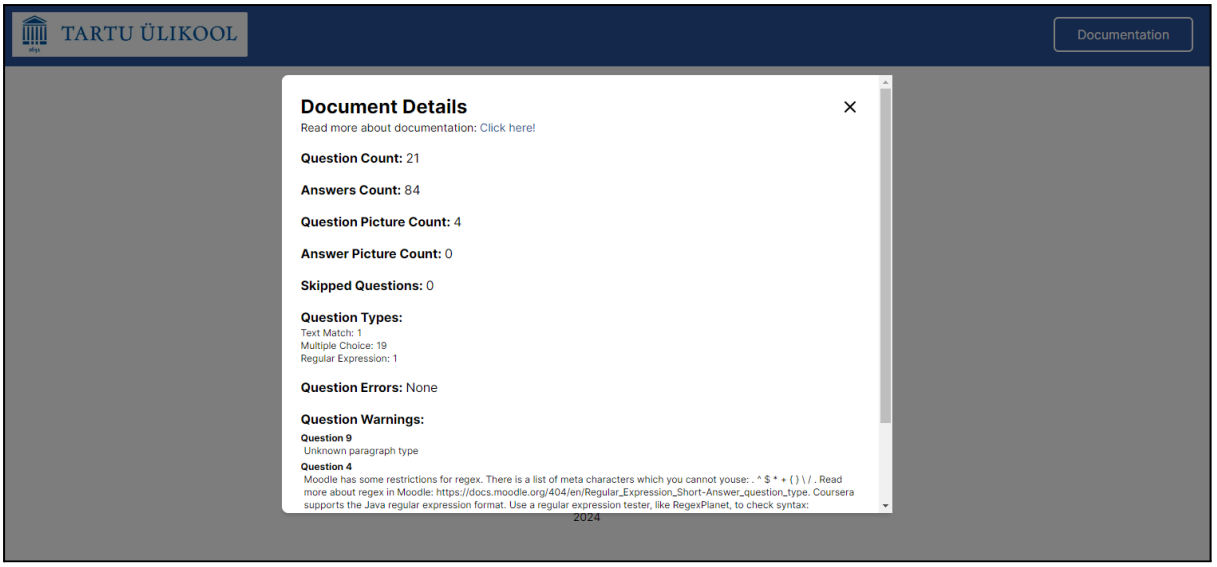
Üks olulisemaid vaateid on avaleht, kuna seal toimub kogu konverteerimise protsess. Avalehel on kasutajal võimalus üles laadida fail konverteerimiseks (vt lähemalt joonis 10). Faili üleslaadimiseks kasutakse react-dropzone moodulit [20]. Kui fail ei vasta nõutud tüübile, annab rakendus sellest teada, muutes ääred punaseks ja teatades, et faili tüüp pole sobiv. Kui aga faili tüüp on õige, siis annab eesliides sellest kasutajale teada, kuvades faili nime kasti keskel ja seejärel on võimalik valida allalaadimise formaat. Vajutades soovitud

allalaadimise formaadi peale, tuleb paremale alla konverteerimise protsessi kohta teade ning kui vigu ei olnud, annab eesliides sellest kasutajale märku.

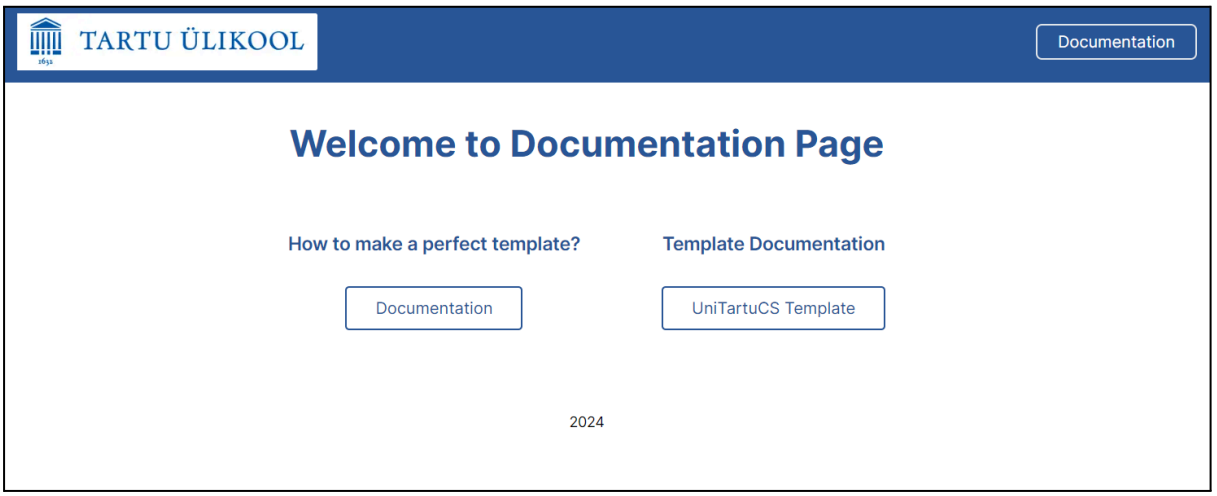
Joonis 10. Rakenduse avalehe vaade.

Positiivse konverteerimise tulemusel avaneb aken faili infoga (vt lähemalt joonis 11). Infoaken sisaldab järgnevat informatsiooni üleslaetud failist:

1) küsimuste arv;
2) vastuste arv;
3) piltide arv küsimustes;
4) piltide arv vastustes;
5) vahelejäetud küsimused;
6) küsimuste tüübid;
7) küsimuste vead;
8) küsimuste hoiatused.



Joonis 11. Rakenduse detailide vaade. Testimiseks kasutatud fail on lisa 5.



Joonis 12. Dokumentatsiooni vaade.

Avalehe päis ja jalus sisaldavad üldist informatsiooni. Päises on nupp dokumentatsiooni lehele (vt lähemalt joonis 12). Dokumentatsiooni lehel on nupud, mis suunavad täpsete juhenditeni, kuidas ühisdokumendis koostada teste ja samuti on olemas näidis UniTartuCS raamistikust, mis hõlbustab dokumendi koostamist.

5.2.3 Veebirakenduse turvalisus

Turvalisus on tänapäeval järjest enam tähelepanu nõudev faktor. Veebirakenduse arendamisel on turvalisus prioriteet, kuna inimesed on hakanud järjest rohkem kasutama internetti [21]. Üks peamisi turvalisuse suurendamise strateegiaid on otsus mitte kasutada andmebaasi, kuhu saaks salvestada kasutajate infot ning konverteerimise ajalugu. See on kriitilise tähtsusega, kuna välistab võimaluse andmelekkeks. Kogu faili töötlusprotsess toimub virtuaalserveris,

kasutaja laeb faili üles eesliideses, seejärel dokument konverteeritakse tagasüsteemis ning tagastatakse kasutajale, ilma et see salvestuks meie süsteemi. See tähendab, et kogu informatsioon jääb kasutaja kontrolli alla ning kolmandatel isikutel ei ole võimalik failile rakenduse kaudu ligi pääseda. Selline lähenemine vähendab küberkurjategijate huvi veebirakenduse vastu, kuna kasutajate andmeid ei saa varastada.

5.3 Rakenduse arendus

Veebirakendus valmis kahe üliõpilase koostööna, bakalaureuse- ja magistritöö raames. Arendamiseks kasutati agiilset lähenemist, tehes palju videokõnesid ning koostades ülesandeid erinevate funktsionaalsuste kohta. Selline lähenemine andis võimaluse personaalseks tööks. Kõik eesmärgid saavutati õigeaegselt. Bakalaureuse tudengi peamine fookus oli eesliidese arendamisel, kasutusjuhendi ning Tartu Ülikooli jaoks malli loomisel ning magistriõpilase Anneli Klamase fookus tagasüsteemi arendamisel. Disain, arhitektuursed valikud ning koodi ülevaatamised ja soovitused sündisid koostöös. Veebirakenduse struktuuri ja arhitektuuriga alustati juba 2023. aasta oktoobris ning disaini prototüüp valmis novembris. Lõplik disain sai paika detsembris ning seejärel alustati veebirakenduse arendamisega 2024. aasta jaanuaris. Prototüüp valmis märtsis, kus nii tagasüsteem kui ka eesliides töötasid ootuspäraselt.

5.4 Kasutatud tehnoloogiad

Tehnoloogiate valikul oli oluliseks aspektiks kaasaegsus ja edasiarendamise võimalused, samuti pandi rõhku optimaalsele jõudlusele, kasutajasõbralikkusele ja hooldusvõimalustele.

Java & Spring raamistik

Java on laialdaselt kasutatud ning tuntud enda turvalisuse, jõudluse ja mitmekülgsuse poolest. Töö autori varasem kokkupuude ja kogemus Javaga muutis selle keele parimaks valikuks serveripoolseks arenduseks. Spring raamistik pakub põhjalikku toetust veebirakenduse arenduseks omades lihtsustatud konfiguratsiooni ja kasulikke funktsioone, nagu rakendusliideste haldamine (ingl *application programming interface,* API) ja turvalisus.

React & Next.js raamistik

React on veebiarenduses laialt kasutuses komponendipõhise arhitektuuri tõttu. See võimaldab luua interaktiivseid kasutajaliideseid [22]. Next.js on Reacti raamistik, mis on mõeldud serveripõhiste rakenduste jaoks. Next.js aitab parandada veebirakenduse otsingumootorite

optimeerimist (ingl *search engine optimization*) ja jõudlust. Samuti pakub Next.js kiiret ja optimeeritud lehekülgedel navigeerimist [23]. Töö autori varasem kokkupuude muutis selle valiku parimaks eesliidese arendamise jaoks.

Tailwind CSS

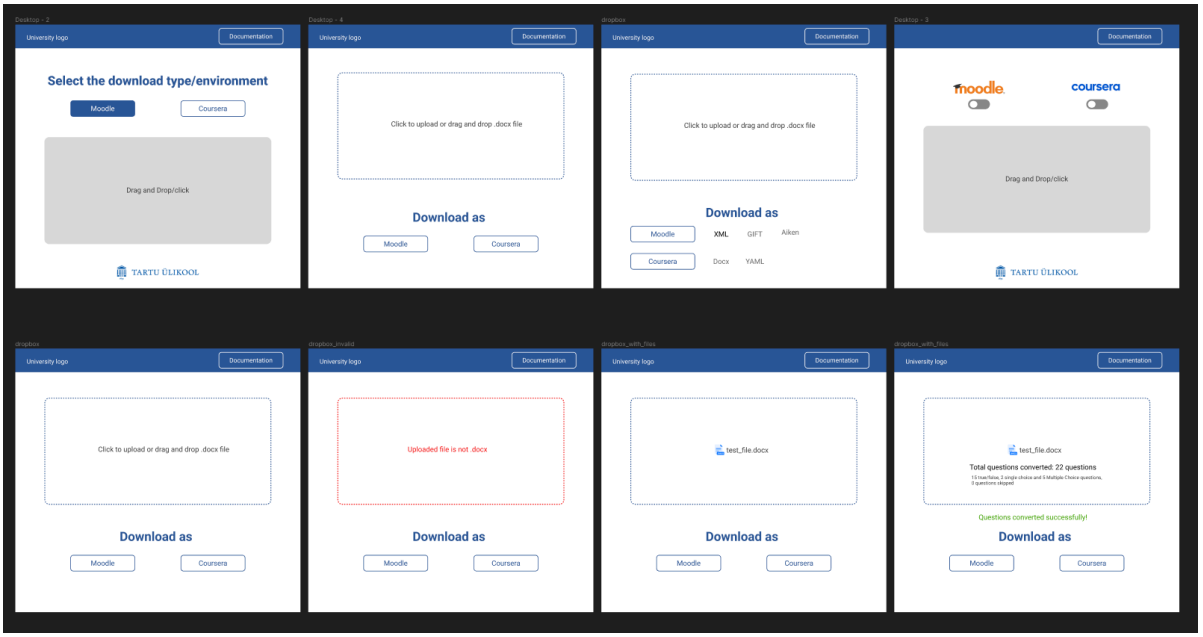
Tailwind CSS raamistiku peamine eelis on võimekus rakendada stiile otse HTML'i sisse, kiirendades arendusprotsessi, parandades loetavust ning tagades ühtlase disaini terves veebirakenduses [24]. Selline paindlikkus ja konfiguratsioonide hõlpsus toetab kiireid disainimuudatusi, ilma eesliidese kvaliteeti mõjutamata. Raamistik on loodud Adam Wathan'i poolt aastal 2017 [25].

Github & Git

Versioneerimiseks kasutati Git'i koos Github platvormiga, kus hoiti ja tehti koostööd koodi haldamisel. Github aitab tõhusalt hallata koodi muudatusi, teha koostööd, jagada ülesandeid ja teostada koodi ülevaatamisi. See muudab kogu arendusprotsessi koostöö sujuvaks, et valmiks kõrge kvaliteediga kood, mida on hiljem kerge edasi arendada.

Figma

Veebirakenduse esimene prototüüp ja disain valmis Figma keskkonnas (vt joonis 13). Tööriist võimaldab meeskonnal mugavalt reaalajas koos töötada, kavandada ja testida erinevaid eesliidese disaine enne arendamise etappi. Nii on võimalik tagada, et valmiv veebirakendus vastaks ootustele ja vajadustele. Figma 1.0 loodi 2015 aastal ning asutajateks olid Evan Wallace ja Dylan Field [26].

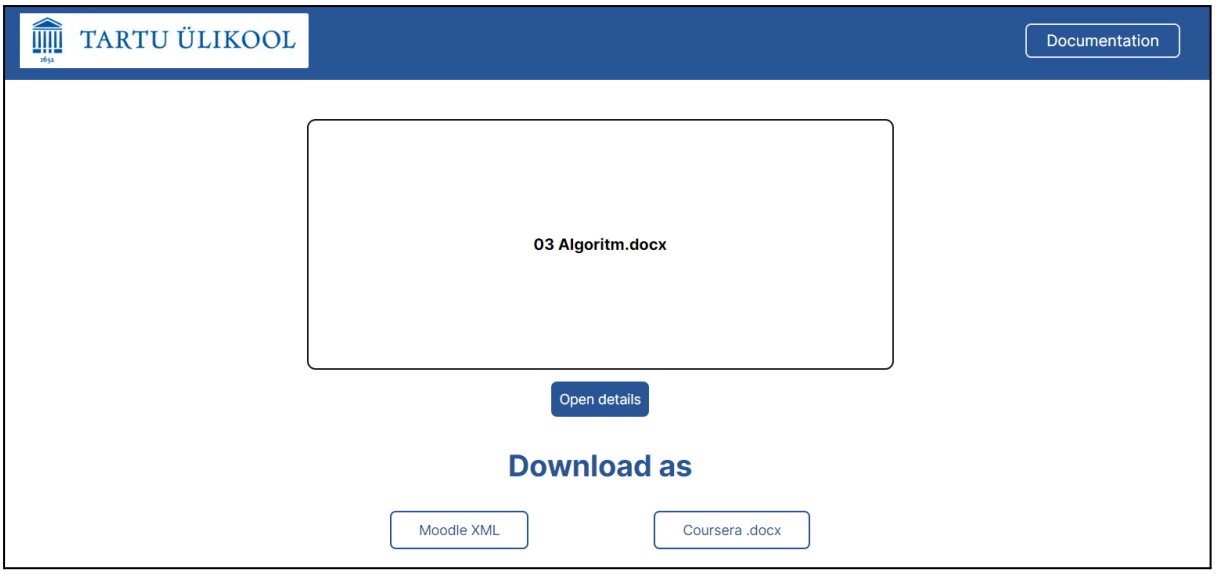


Joonis 13. Figma avalehe disaini esimesed näidised.

Kokkuvõttes tagab valitud tehnoloogiate kasutuselevõtt rakenduse vastupidavuse ja kasutajasõbralikkuse. Samuti tagab valik, et veebirakendus on valmis silmitsi seisma tuleviku väljakutsete ja laienemisvõimalustega.

5.5 Rakenduse kasutamine

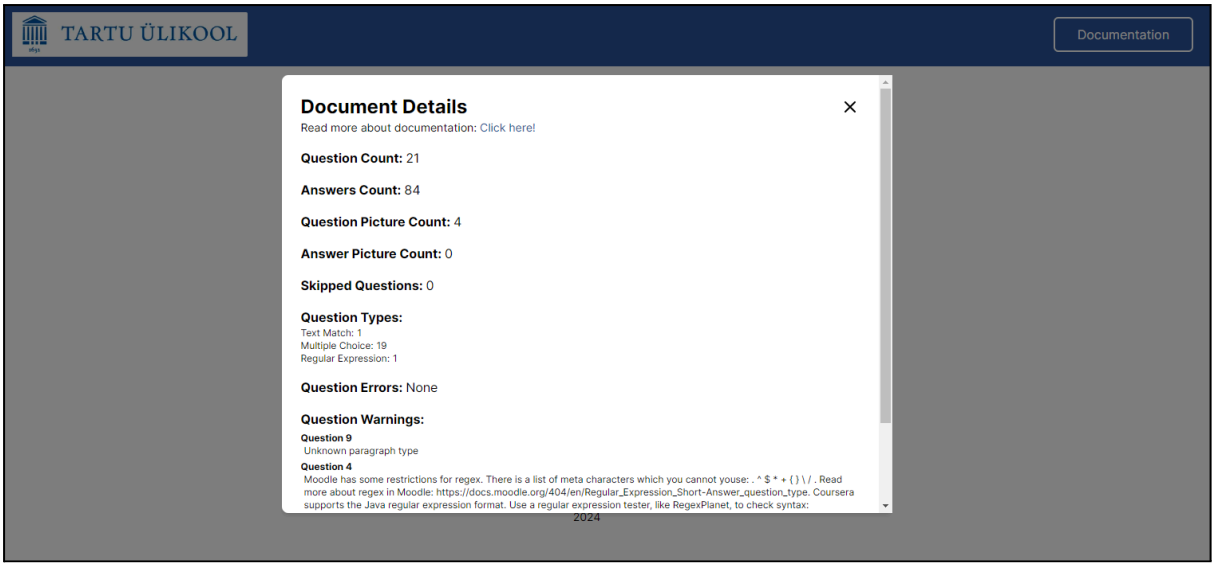
Käesolevas alapeatükis tutvustatakse veebirakenduse kasutamist. Veebirakendusse on võimalik üles laadida ainult .docx laiendiga faile ning soovituslik on aluseks võtta UniTartuCS mall (vt lisa 4) ning lugeda kasutusjuhendit (vt lisa 2). Korrektse faili tüübi lisamisel ilmub kasti sisse faili nimi (vt joonis 14).



Joonis 14. Faili lisamine.

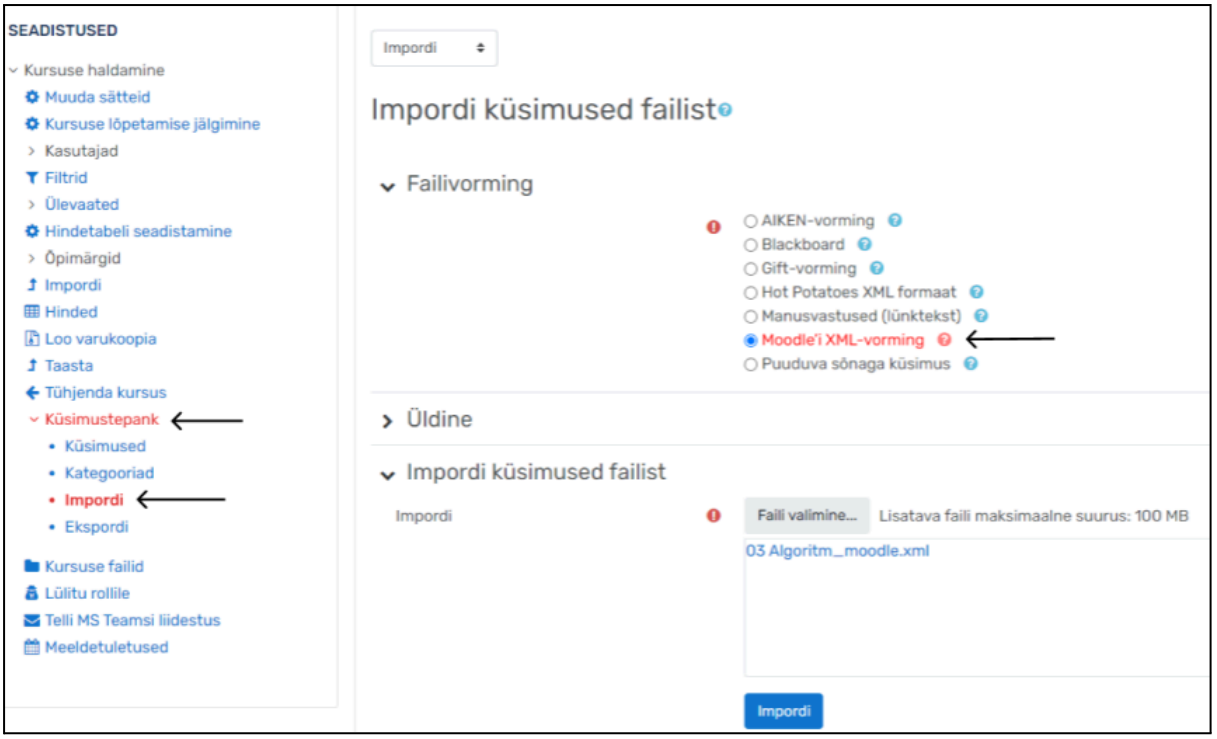
Peale faili üleslaadimist on kasutajal võimalus valida kahe erineva väljundformaadi vahel – Moodle XML ja Coursera .docx. Valitud formaadi konverteerimine algab automaatselt ning konverteerimise protsessi on võimalik jälgida lehe paremale alla tekkivast hüpikaknast. Konverteerimise protsessi etapid jaotuvad kolmeks:

1) **laadimine**: faili üles laadimine;
2) **viga**: faili üles laadimisel tekkis viga;
3) **õnnestumine**: faili üles laadimine õnnestus.



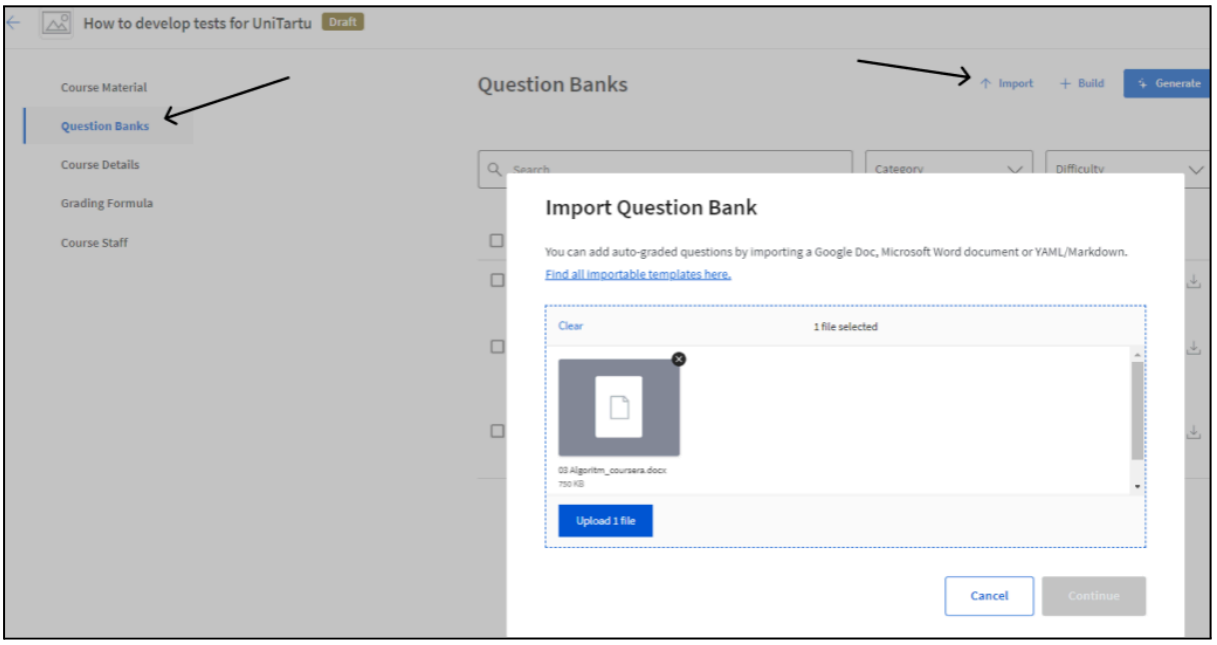
Joonis 15. Küsimuste detailid. Testimiseks kasutatud fail on lisa 5.

Õnnestunud üleslaadimise korral laetakse automaatselt alla konverteeritud fail ning kuvatakse kasutajale konverteeritud faili andmete aken (vt lähemalt joonis 15). Hüpikaken annab võimaluse üle kontrollida, kas kõik küsimused konverteeriti korrektselt ümber ning osutatakse tekkinud vigadele. Ebaõnnestunud laadimise korral faili ei töödelda ning tagastatakse vastav teade. Kui fail ei vasta kasutusjuhendile, siis kuvatakse hüpikaknas vastavad veateated ning hoiatused.



Joonis 16. Küsimustiku baasil loodud faili üles laadimine Moodle keskkonda.

Moodle'i kasutajad saavad konverteeritud faili laadida küsimustepanka. Selleks tuleb valida kursuse seadistuste alt "Küsimustepank" ning vajutades "Impordi" valikule (vt lähemalt joonis 16). Peale sinisele "Impordi" nupule vajutamist avaneb uus vaade, kus kuvatakse kõik failis olnud küsimuste pealkirjad. See annab võimaluse enne küsimustepanka üleslaadimist kontrollida küsimused üle, kui kõik küsimused on olemas võib protsessi jätkata ning küsimused laetakse küsimustepanka.



Joonis 17. Küsimustiku baasil loodud faili üles laadimine Coursera keskkonda.

Coursera puhul toimub küsimuste import sarnaselt. Esmalt tuleb valida kursus, kuhu soovitakse küsimused importida. Seejärel on võimalik valida seadete alt "Küsimustepank" ning vajutades "Import" nupule avaneb järgnev import vaade (vt lähemalt joonis 17). Esmalt tuleb lisada fail ning seejärel ilmub sinine "Upload" nupp, mis viib järgmisesse vaatesse. Järgneval vaatel on kirjas informatsioon küsimuste kohta. Kui protsessis vigu ei esinenud, on võimalik lisada küsimused panka.

6. Võimalikud edasiarendused ja teenusehaldus

Eelnev peatükk “Veebirakenduse kirjeldus ja analüüs” tõestas, et bakalaureusetöö jaoks loodud veebirakendus toimib ning vastab eesmärkidele hõlbustada õpetajate vahelist koostööd ning lihtsustada testide koostamist. Käesolev viimane peatükk kirjeldab tuleviku võimalikke edasiarendusi ning teenuse halduse põhimõtteid.

6.1 Võimalikud edasiarendused

Esimene potentsiaalne edasiarenduse võimalus on jagada teste otse erinevate keskkondade vahel ilma vahekonverteerimiseta. Moodle XML’i oleks võimalik otse konverteerida ka Coursera .docx formaati ja vastupidi. Selle idee käis välja ka üks õppejõududest, kellega vesteldi.

Selle edasiarenduse võimaluse juures tekib juurde turvalisuseprobleeme, kuna veebirakendus peab lubama lisaks .docx failidele ka XML-vormingus faile üles laadida. See tähendaks failide lisavalideerimist, et vältida pahatahtlikku sisu.

Teine võimalik täiendus veebirakendusele on juurde lisada mitmekeelne tugi. See võimaldaks erinevatest kultuuridest ja keeleruumist inimestel mõista rakenduse sisu paremini. Hetkel on veebirakendus inglise keeles ning olemas on eraldi en.json fail, mis annab hea aluse mitmekeelseks toeks. Selle edasiarendusega võib laieneda kasutajate arv.

Kolmas edasiarenduse suund võiks olla lisada uusi õpikeskkondasid, küsimusetüüpe või failiformaate. Samuti võib üks edasiarenduse suund olla Moodle’i ja Coursera küsimuste visuaalne kuvamine pärast konverteerimist. See funktsionaalsus tagab selle, et õppejõud näevad, kuidas küsimused näevad välja lõplikus formaadis, andes niimoodi kindlustunde, et konverteerimine on positiivselt õnnestunud ja küsimused on soovitud kujul.

6.2 Teenusehaldus

Veebirakenduse lähtekood on avalik ning kättesaadav Gitlabis (https://gitlab.cs.ut.ee/klamas/test-converter). Kood viidi eraldi Gitlabi keskkonda, kus Tartu Ülikooli suure jõudlusega arvutuskeskus (*high performance computing center,* HPC) saab rakendust tulevikus hallata. Veebirakenduse jaoks loodi HPC virtuaalmasin jõudlusega 4 virtuaalset keskprotsessori ühikut (*virtual Central Processing Unit*, ka vCPU), 2GB RAM ja 10GB kõvaketta mahtu. Veebirakendusele on võimalik ligipääseda järgnevalt lingilt: http://193.40.155.50/. Rakenduse jaoks loodi Docker konteiner, kus sees jookseb rakenduse

eesliides ning tagasüsteem, tagades nii kahe keskkonna vahelise suhtluse. Docker'i vajadus on seetõttu, et HPC saaks tulevikus rakenduse kolida ümber Kubernetesse, tagades sellega süsteemi skaleeritavuse ja paindlikkuse.

Kokkuvõte

Bakalaureusetöö eesmärk oli arendada veebirakendus Tartu Ülikoolile, mis võimaldab kasutajatel üles laadida koostöiselt arendatud dokumenti, et kiirendada ja lihtsustada testide loomise protsessi. Esimene samm enne arendustöö algust oli uurida ja analüüsida Tartu Ülikoolis kasutatavate õpikeskkondade eksport/import võimalusi ning kitsaskohti testide loomisel. Sellest lähtuvalt koostati veebirakenduse disain ja valiti arendustehnoloogiad: Next.js eesliidese arenduseks, Java Spring tagasüsteemi arenduseks ja Figma disainilahenduste jaoks. Tehnoloogia valikud põhinevad võimekusel toetada kasutajaliidest ja pakkuda skaleeritavat ja turvalist süsteemi, millel on edasiarendamise võimalus.

Selleks, et veebirakenduse arenduse protsess oleks võimalikult efektiivne ja keskenduks kasutajate tegelikele vajadustele tehti põhjalik õpikeskkondade eksport/import analüüs (vt lähemalt peatükk 3) ja koguti õpetajatelt sisend testide koostamise töövoo ja kitsaskohtade kohta (vt lähemalt peatükk 2). Analüüsi põhjal tehti kindlaks arenduse arhitektuur ja kasutatavad failiformaadid, pöörates erilist tähelepanu kasutatavusele, toetatavusele ja rakenduse lihtsale eesliidesele. Lisaks tagab koodi puhtus ja modulaarsus rakenduse edasiarenduse lihtsuse. Näidismall loodi õppejõudude poolt saadud tagasiside põhjal, mis järgib rakenduse kasutusjuhendit demonstreerides rakenduse põhifunktsioone ning pakkudes paindlikkust iseseisvaks dokumendi koostamiseks.

Peale analüüsi liiguti järgmisesse faasi – rakenduse arenduse perioodi. Eesliidese disain loodi Figma keskkonnas (vt lähemalt peatükk 5), kus rõhk oli kasutajasõbralikkusel, lähtudes analüüsist. Seejärel asuti funktsionaalsuste ja tööülesannete kirjapanemise juurde, kasutades selleks Githubi haldussüsteemi (vt lähemalt peatükk 5). Sellega tagati veebirakenduse protsessi paindlikkus ja meeskonnaliikmete vaheline selge kommunikatsioon. Tagasüsteemi arendamisel kasutati Java Spring raamistikku (vt lähemalt peatükk 5) jaotades töö viieks peamiseks mooduliks: kontrollerid tagasid kasutajate päringute töötlemise, abimoodulid pakkusid tugifunktsioone, mudelid kirjeldasid andmeobjekte, komponendid sisaldasid spetsiifilisi funktsioone ja teenused sisaldasid rakenduse konverteerimise loogikat. Selline struktureeritud lähenemine võimaldas luua skaleeritava ja hooldatava süsteemi. Eesliidese arendamisel kasutati Next.js raamistiku (vt lähemalt peatükk 5), jaotades rakenduse vaated neljaks: 1) avaleht, 2) kasutusjuhend, 3) päis ja 4) jalus. Arendustöö lõppedes pandi kirja edasiarendamise võimalused ning selgitati kuidas hakkab välja nägema teenusehaldus (vt lähemalt peatükk 6).

Viidatud kirjandus

[1] Pilt L. E-õppe statistika 2023, https://etu.ut.ee/2024/e-oppe-statistika-2023/ (21.04.2024)

[2] Buhu, A., Buhu, L., 2015. Open Source eLearning Platforms and Software for Online Textile Engineering Learning. *Proceedings of the 11th International Scientific Conference eLearning and Software for Education*, Bucharest, April 23-24, Educational Dimension, 2015, 534-537. https://www.researchgate.net/publication/277163240_OPEN_SOURCE_ELEARNING_PLATFORMS_AND_SOFTWARE_FOR_ONLINE_TEXTILE_ENGINEERING_LEARNING (06.12.2023).

[3] Moodle. https://stats.moodle.org (26.11.2023)

[4] Moodle import questions. https://docs.moodle.org/403/en/Import_questions (26.11.2023)

[5] Moodle GIFT. https://docs.moodle.org/403/en/GIFT_format (26.11.2023)

[6] Moodle XML. https://docs.moodle.org/403/en/Moodle_XML_format (26.11.2023)

[7] Mintii, I. S., Shokaliuk, S. V., Vakaliuk, T. A., Mintii, M. M., Soloviev, V. N., 2019, Import test questions into Moodle LMS. Educational Dimension. 2019. Volume 1. P. 111–124. https://acnsci.org/journal/index.php/ed/article/view/442/456 (06.12.2023).

[8] Moodle Aiken. https://docs.moodle.org/403/en/Aiken_format (27.11.2023)

[9] Blackboard. https://help.blackboard.com/Learn/Instructor/Original/Tests_Pools_Surveys/Orig_Reuse_Questions/Import_or_Export_Tests_Surveys_and_Pools (27.11.2023)

[10] Coursera. https://about.coursera.org/ (27.11.2023)

[11] Coursera. Example Question Types for Importing, https://docs.google.com/document/d/1UxyaVDkGBip8wntBu0uAyAv47nAQhzU9bzIax_ta0wc/edit (27.11.2023)

[12] YAML. https://yaml.org/ (24.04.2024)

[13] Red Hat. What is YAML? https://www.redhat.com/en/topics/automation/what-is-yaml (24.04.2024)

[14] Coursera. Create, Manage, and Use Question Banks in an Assignment, https://www.coursera.support/s/article/360057235331-Create-Manage-and-Use-Question-Banks-in-Staff-Graded-Assignments (24.04.2024)

[15] M. Fowler, Patterns of Enterprise Application Architecture. Boston: Addison-Wesley. 2011

[16] Java Spring. https://spring.io/projects/spring-framework (27.03.2024)

[17] Apache POI. https://poi.apache.org/ (13.05.2024)

[18] Next.js. https://nextjs.org/ (21.04.2024)

[19] PageSpeed Insights. https://pagespeed.web.dev/ (15.05.2024)

[20] React-dropzone. https://react-dropzone.js.org/ (21.04.2024)

[21] Mirjalili, M.; Nowroozi, A.; Alidoosti, M. A survey on a web penetration test. Advances in Computer Science an International Journal 2014, 3, 117–121. https://www.researchgate.net/profile/Alireza-Nowroozi-2/publication/270523617_A_survey_on_web_penetration_test/links/54acdfd00cf2479c2ee8555c/A-survey-on-web-penetration-test.pdf (23.04.2024)

[22] React. https://react.dev/ (29.03.2024)

[23] Next.js and React. https://nextjs.org/learn/react-foundations/what-is-react-and-nextjs (26.04.2024)

[24] Tailwind CSS. https://tailwindcss.com/ (26.04.2024)

[25] Wathan A. Going Full-Time on Tailwind CSS 2018, https://adamwathan.me/going-full-time-on-tailwind-css/ (13.05.2024)

[26] Figma. https://www.figma.com/about/ (13.05.2024)

Lisad

I. Õppejõududele esitatud küsimused

7. märtsil uuriti Tartu Ülikooli õppejõududelt järgnevaid küsimusi testide koostamise töövoo kohta.

Küsimused:

- Kas te kasutate Moodle või Coursera keskkonda testide haldamiseks?
    ”Jah” -> Kuidas haldate teste?
    “Ei” -> Mis põhjusel?

-Kuidas koostate teste?
-Palun kirjeldage, millised on testi koostamise etapid.
-Mis teile selle puhul meeldib ja mis ei meeldi?

-Kas teete testide koostamisel koostööd teiste õppejõududega?
    “Jah” -> Kuidas te koostööd koordineerite ja milliseid vahendeid te kasutate?
    “Ei” -> Kas selline vajadus võib tekkida, kui on olemas võimalused?

-Kas te kasutate Moodle või Coursera testide loomisel küsimustepanka?
-Kuidas te haldate ja valite küsimusi nendest pankadest?

-Kas kasutate testide koostamisel uusi tehnoloogiaid või innovatiivseid lähenemisi?

-Mis seadeid testide loomisel kasutate? (Shuffle, partial credit)

II. Rakenduse kasutusjuhend

Introduction

This document is for learning more about the formats and question types supported for uploading documents into the UniTartuCS quiz converter.

Here you can find UniTartuCS Template: UniTartuCS Template

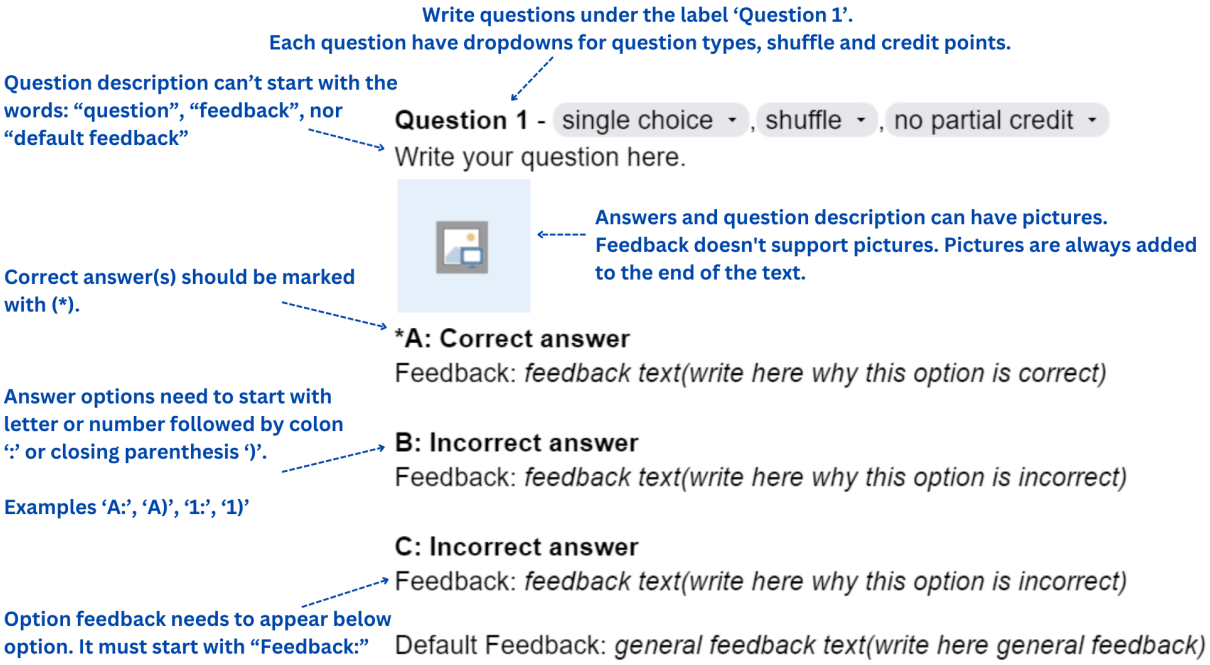
Follow steps on the next page.

2024

STEP 1

Question Formatting

Here is an example of how questions should be formatted.



STEP 2

Question Properties

Our template has default properties for each question.

Example: **Question 1 -** Single choice, no shuffle, partial credit

We have added dropdowns with all possible properties.

Properties you can select:

1. **Question type:**
   Option: "single choice"
   Option: "multiple choice"
   Option: "text match"
   Option: "regular expression"
2. **Shuffle**
   Default: "shuffle"
   Other option: "no shuffle"
3. **Partial Credit**
   Default: "partial credit"
   Other option: "no partial credit"

STEP 3

Question Feedback

You can give feedback after each question whether it is correct or false, also there is a possibility to add default feedback at the end of the question to explain something more specifically.

Feedback doesn't support pictures. Coursera doesn't support default feedback for multi nor single choice questions. Example:

Question 1 - single choice , shuffle , partial credit
What is the chemical symbol for water?

*A: H_2O
Feedback: Correct! H_2O is water.

B: CO_2
Feedback: Incorrect. CO_2 is carbon dioxide.

Default Feedback: Read more about chemical symbols...

Question type	Option-specific feedback	Default feedback
Multiple Choice	YES	YES(Moodle only)
Single Choice	YES	YES(Moodle only)
Text Match	YES	YES
Regular Expression	YES	YES

This table shows where is feedback supported

STEP 3

Question Type Examples

Here you can find all different question types that are supported.

Single Correct Answer

Question 1 - single choice , shuffle
Write your question here.

***A: Correct answer**

Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

C: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

Multiple Correct Answers

Question 1 - multiple choice , shuffle , partial credit
Write your question here.

***A: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

***C: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

Default Feedback: *general feedback text(write here general feedback)*

Text Match

Text match/short answer questions are not case-sensitive.

Question 1 - text match
Write your question here.

***A: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

Regular Expression

Text match/short answer questions are not case-sensitive.

Question 1 - regular expression
Write your question here.

***A: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

STEP 4

Question and Answer Images

Answers and question descriptions can have pictures. Pictures are always added to the end of the text. Example:

Question 1 - single choice, shuffle
Write your question here.

***A: Correct answer**

Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer

Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

III. Moodle XML-vormeering

Järgnev pilt on näide küsimusest ja vastusevariantidest XML-vormeeringus.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<quiz>
    <question type="multichoice">
        <name>
            <text>Question 1 </text>
        </name>
        <questiontext format="html">
            <text>&lt;br&gt;Mis on algoritm? (vali kõik mis peavad paika)</text>
        </questiontext>
        <answer format="html" fraction="33.333333333333336">
            <text> Kirjeldus, mis annab ette tegevusjuhised, tavaliselt mingi probleemi lahenduseks</text>
            <feedback>
                <text>jah</text>
            </feedback>
        </answer>
        <answer format="html" fraction="33.333333333333336">
            <text> Piltjuhised mingi probleemi lahendamiseks.</text>
            <feedback>
                <text>jah, algoritm võib olla esitatud ka piltidena</text>
            </feedback>
        </answer>
        <answer format="html" fraction="33.333333333333336">
            <text> Programmikood, mille eesmärk on olla inimloetav ja mis pole seotud ühegi konkreetse programmeerimiskeelega.</text>
            <feedback>
                <text>see kirjeldab ennekõike pseudokoodi mõistet kuid mahub algoritmi määratluse alla</text>
            </feedback>
        </answer>
        <answer format="html" fraction="-33.333333333333336">
            <text> Programmi sisendi ja väljundi kirjeldus</text>
            <feedback>
                <text>see pole algortim vaid pigem kirjeldus mis on algoritmile vajalik sisend ja oodatav väljund. tegevusjuhis on puudu.</text>
            </feedback>
        </answer>
        <shuffleanswers>1</shuffleanswers>
        <single>false</single>
        <generalfeedback>
            <text>vaata loengus käsitletud mõisted üle!</text>
        </generalfeedback>
    </question>
</quiz>
```

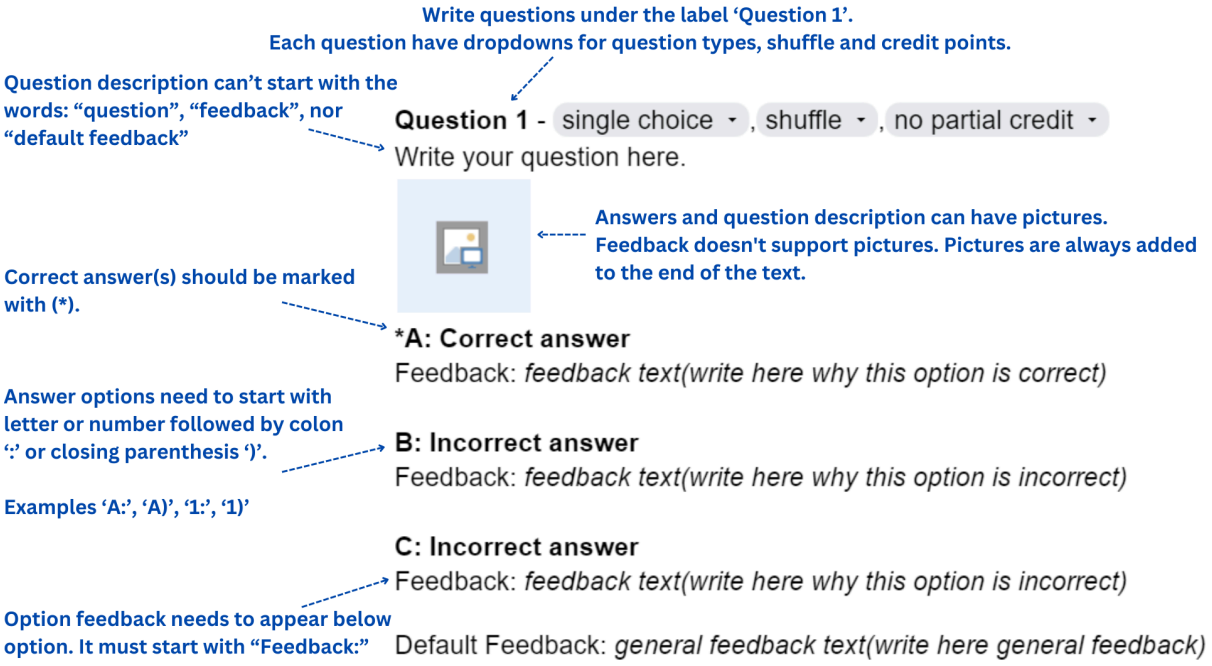
Bakalaureusetöö autori koostatud pilt, küsimuse on koostatud juhendaja Jaak Vilo.

IV. UniTartuCS näidismall

How to create a template?

At the moment the system supports only single choice, multiple choice/checkbox and text match/short answer questions. Also Supports multi paragraph question descriptions.

Below are guidelines on formatting the document so it can be imported successfully. Throughout this document, you can read comments to learn quick tips.



Read more about Question types.

Question 1 - single choice , shuffle , partial credit
Write your question here.

***A: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

B: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

C: Incorrect answer

Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

Question 2 - multiple choice , shuffle , partial credit
Write your question here.

***A: Correct answer**
Feedback: *feedback text(write here why this option is correct)*

***B: Correct answer**
Feedback: *feedback text(write here why this option is incorrect)*

C: Incorrect answer
Feedback: *feedback text(write here why this option is incorrect)*

Default Feedback: *general feedback text(write here general feedback)*

Question 3

V. Testimiseks kasutatud dokument

Digitaalse maailmapildi aine ühe nädala küsimustik, vastutav õppejõud prof. J. Vilo

Question 1 - multiple choice, shuffle, no partial credit

Mis on algoritm? (vali kõik mis peavad paika)

*A: question Kirjeldus, mis annab ette tegevusjuhised, tavaliselt mingi probleemi lahenduseks

*B: Piltjuhised mingi probleemi lahendamiseks.

*C: Programmikood, mille eesmärk on olla inimloetav ja mis pole seotud ühegi konkreetse programmeerimiskeelega.

D: Programmi sisendi ja väljundi kirjeldus

Default Feedback: Vaata loengus käsitletud mõisted üle!

Question 2 - multiple choice, shuffle, partial credit

Mis on plokkskeem? (vali täpseim)

*A: Joonis kindlas notatsioonis, mis kirjeldab algoritmi toimimist.

Feedback: Jah, see on tegevussammude omavahelise järjekorra kirjeldus

B: Piltjuhised mingi probleemi lahendamiseks.

Feedback: Plokkskeemis üldiselt pilte ei kasutata.

C: Programmikood, mille eesmärk on olla inimloetav ja mis pole seotud ühegi konkreetse programmeerimiskeelega.

Feedback: See kirjeldab pigem pseudokoodi mõistet.

D: Juhiste jada, mis on märksõnade abil kirja pandud.

Feedback: See pole siiski kõige täpsem.

Default Feedback: Vaata loengus käsitletud mõisted üle!

Question 3 - multiple choice, shuffle

Mis on pseudokood? (Vali üks täpseim)

*A: Programmi või algoritmi kirjeldus, mille eesmärk on olla inimloetav (näiteks dokumentatsioonis) ja mis pole seotud ühegi konkreetse programmeerimiskeelega.

Feedback: Jah, sisuliselt on need juhised, mis on kirja pandud programmikoodile sarnasel viisil.

B: Kood, mis pole semantiliselt korrektne ja käivitades tööle ei lähe.

Feedback: Pseudokood polegi mõeldud otseselt käivitamiseks.

C: Programmi esitus masinkoodi kujul, mis pole üldjuhul inimesele loetav.

Feedback: Pseudokoodi juures on tähtis, et ta oleks inimloetav.

D: Piltjuhised mingi probleemi lahendamiseks.

Feedback: Pseudokoodis pilte pigem ei kasutata.

Default Feedback: Algoritme proovitakse reeglina kirja panna inimloetaval viisil mis pole tingimata üheski programmeerimiskeeles.

Question 4 - regular expression

Funktsiooni või alamprogrammi, mis kasutab iseennast (kutsub iseennast välja), nimetatakse _____.

*A: [Rr]ekursiiv(seks)|(ne)

Feedback: Väga hea!

*B: [Rr]ekursioon(iks)?

Feedback: Täpsem oleks sellise funktsiooni kohta siiski öelda ‘rekursiivne’. Aga rekursioon on tõesti õige üldmõiste.

C: [Jj]aga ja valitse.*?

Feedback: Peaaegu. Jaga ja valitse öeldakse tavaliselt probleemi lahendamisel sellise lähenemise kohta, kus probleem jagatakse väiksemateks osadeks. Selline

D: [Aa]hne(ks)?.*?

Feedback: Iseenda väljakutsumine pole ahne algoritmi omadus.

Default Feedback: Vaata loengus käsitletud mõisted üle!

Question 5 - multiple choice, easy difficulty

Kas kartulisalati valmistamise retsepti saab lugeda algoritmiks kui kõik sammud on ühemõtteliselt ja piisava täpsusega väljendatud?

*A: Jah

Feedback: Kui sammud poleks ühemõttelised, ei saaks seda retsepti formaalselt algoritmiks pidada :)

B: Ei

Feedback: Retsepti ja algoritmi definitsioonid on tegelikult päris sarnased.

Question 6 - multiple choice, shuffle, hard difficulty

Vaatame järjestamise algoritmi, kus esmalt võetakse üks jada üks element ning võrreldakse kõiki teisi selle vastu. Kõik väiksemad või võrdsed elemendid paigutatakse ette- ja suuremad tahapoole. Seejärel sooritatakse sama protseduur eraldi nii vasaku (väiksemad elemendid) kui parema poolega (suuremad elemendid).

Millise sorteerimisalgoritmi ideed on kirjeldatud?

A: Mullimeetod (bubble sort)

*B: Kiirmeetod (quicksort)

C: Valikmeetod (selection sort)

D: Põimemeetod (merge sort)

Default Feedback: Vaata loengumaterjalid sorteerimismeetodite osas üle

Question 7 - multiple choice, shuffle, hard difficulty

Vaatame järjestamise algoritmi, kus elementide jada jagatakse pooleks ning jagatud osad sorteeritakse eraldi, rekursiivselt. Seejärel põimitakse mõlemad sorditud pooled omakorda kokku ühte järjekorda. Millise sorteerimisalgoritmi ideed on kirjeldatud?

A: Mullimeetod (bubble sort)

B: Kiirmeetod (quicksort)

C: Valikmeetod (selection sort)

*D: Põimemeetod (merge sort)

Default Feedback: Vaata loengumaterjalid sorteerimismeetodite osas üle

Question 8 - multiple choice, shuffle, hard difficulty

Vaatame järjestamise algoritmi, kus kõigepealt leitakse kogu jada väikseim element ning vahetatakse see kõige esimese elemendiga. Väikseim element on nüüd õigel kohal. Sama protseduuri korratakse ülejäänud jadaga leides ülejäänud jadast väikseim ja paigutades see vastava alamjada esimeseks. Nii jätkatakse, kuni kõik elemendid on õige koha leidnud. Millise sorteerimisalgoritmi ideed on kirjeldatud?

A: Mullimeetod (bubble sort)

B: Kiirmeetod (quicksort)

*C: Valikmeetod (selection sort)

D: Põimemeetod (merge sort)

Default Feedback: Vaata loengumaterjalid sorteerimismeetodite osas üle

Question 9 - checkbox, partial credit, shuffle

Mis iseloomustab rekursiooni? (võrreldes iteratsiooniga)

A: Kasutab tavaliselt vähem mälu

*B: Lõpetab enda tegevuse kordamise kui jõuab baasjuhuni

*C: Kasutab tavaliselt veidi rohkem mälu

D: Lõpetab enda tegevuse kordamise kui tsükli tingimus enam ei kehti

Default Feedback: Kui rekursiivne programm kutsub iseennast ja see uuesti iseennast jne… siis on ainus viis sellel peatuda kui ülesanne on piisavalt väike, et enam ei pea ennast rekursiivselt välja kutsuma. See baasjuht tuleb kirjeldada. Rekursiooni käigus peab aga arvuti mälus meeles pidama eelmise soorituse seisu, seetõttu on sageli vaja rohkem mälu kui lihtsa tsükli kasutamisel

Question 10 - checkbox, partial credit, shuffle

Mis iseloomustab iteratsiooni (võrreldes rekursiooniga)

A: Võib jääda lõpmatuseni tegutsema kui ei jõua kunagi baasjuhuni

Feedback: See käib pigem rekursiooni kohta

B: Vajab eraldi funktsiooni defineerimist

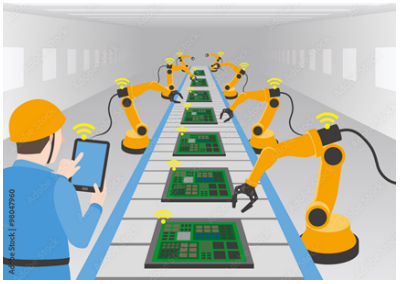
*C: mingit programmi osa korratakse niikaua kui täitub tingimus tsükli peatamiseks või soorittatakse tsükli sees käsk tsüklist väljumiseks (n. break)

*D: Võib jääda lõpmatuseni tegutsema kui tsükli tingimus kehtetuks ei muutu

Default Feedback: Iteratsioon on “tavaline tsükkel” mis kontrollib tsükli täitmisel tingimust (kas järtkata või mitte). Baasjuhtumistr räägitakse rekursiooni võtmes.

Question 11 - checkbox, easy difficulty

Tuntud kiibitootja tehases töötab robot, millel on kaamera ning robotkäsi. See robot suudab pildituvastusalgoritmi abil konveierlindil liikuvaid kiipe eristada ja tõstab neid sobivasse konteinerisse oma kätt juhtiva algoritmi abil. Mida on kirjeldatud?



A: Ideed

Feedback: Ideega siiski tegu ei ole, sest tegemist pole päris abstraktse, vaid konkreetses tehases töötava roboti kirjeldusega.

B: Algoritmi

Feedback: Algoritmi kirjeldus peaks esiteks koosnema sammsammulistest juhistest.

C: Programmi

Feedback: Programmi puhul peaks olema mingite (programmeerimis)keelereeglite ja käskudega kirja pandud, mida robot teeb.

*D: Tehnoloogiat

Feedback: Jah, see on juba valmis töötav tehnoloogia oma konkreetses tegutsemiskeskkonnas

Question 12 - checkbox2, easy difficulty

Mida kirjeldab pilt?

See on graafi (või võrgustiku) G läbikäimise meetod mis käib “laiuti” käbi võrgu alustades ühest tipust, Sisuliselt käiakse läbi naabrid, naabrite naabrid (2 sammu), nende naabrid (3 sammul), jne. Koodis on kirjeldatud ka seda kuidas kasutatakse jada - queue. Sinna pannakse ootele tipud mida on vaja edasi töödelda. Ja sealt võetakse alati kõige kauem seal paiknenud tipp edasiseks tööks…

(kui pilti pole Courseras siis seetõttu et nende tehnoloogia pole töötanud hästi)

```
1  procedure BFS(G, root) is
2      let Q be a queue
3      label root as explored
4      Q.enqueue(root)
5      while Q is not empty do
6          v := Q.dequeue()
7          if v is the goal then
8              return v
9          for all edges from v to w in G.adjacentEdges(v) do
10             if w is not labeled as explored then
11                 label w as explored
12                 Q.enqueue(w)
```

A: Ideed

Feedback: Idee ei ole kõige täpsem vastus, siin on ikkagi konkreetsed algoritmi sammud kirja pandud.

*B: Algoritmi

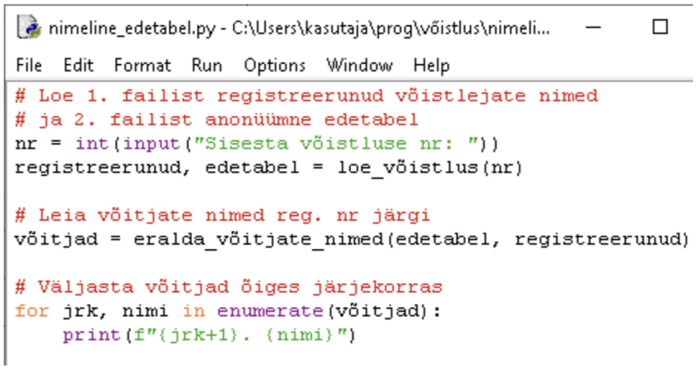
Feedback: Jah, pildil on pseudokood, mis on üks algoritmi kirjapanemise viis, sest täpselt sellisena ei pruugi see algoritm mitte üheski programmeerimiskeeles olla realiseeritav

D: Tehnoloogiat

Feedback: Tegemist pole veel praktilise lahenduse vaid abstraktsema kirjeldusega. Seda tüüpi algoritme saab kasutada näiteks lühima teekonna otsimisel. Näiteks - kui mitu sõpra ja tema sõpra on vaja läbi käia, et jõuda tutvuste kaudu ex-president Barack Obamani.

Question 13 - checkbox, partial credit, easy difficulty

Mida kujutab pilt? Siin on Tekstitoimeti aken kus on programmi kood. (kui pilti pole Courseras siis seetõttu et nende tehnoloogia pole töötanud hästi)



A: Ideed

Feedback: Pigem on juba tegemist mingi idee rakendamisega.

*B: Algoritmi

Feedback: Jah, pildil on kommentaaridega Pythoni programm, mis on ka üks algoritmi kirjapanemise viis.

*C: Tehnoloogiat

Feedback: Jah, ka seda. Lisaks algoritmile on tegemist ka konkreetse Pythonis kirjutatud programmiga.

Question algovaited - checkbox, partial credit, shuffle, medium difficulty

Millised väited algoritmide kohta on tõesed?

*A: Koosneb järjestatud sammude kirjeldusest

*A: Koosneb järjestatud sammude kirjeldusest

*B: Võib kirja panna näiteks programmikoodi või plokkskeemina

C: Tagab töötava lahenduse mingi üldise probleemi lahendamiseks

Feedback: Algoritm ei pruugi lahendada kõiki mingi probleemi erijuhte ning kehv algoritm võib hoopis vigaselt toimida.

*D: Võib sisaldada tsükli abil kirja pandud lõplikku või lõpmatut tegevust

Feedback: Algoritmiga võib kirjeldada korduvaid tegevusi. Reeglina peaks igal sellisel tegevusel olema lõpetamise tingimus. Kuna sisend võib olla lõpmatu ja teiseks me ei pruugi isegi teada kas algoritm iga sisendiga peatub, siis on teoreetiliselt ka lõppematu algoritm võimalik.

Question 15 - checkbox, partial credit, shuffle, medium difficulty

Millised väited algoritmide kohta kehtivad?

A: Algoritm peaks sama sisendiga kirjeldama samas järjekorras sooritatavaid samme

Feedback: Ei pruugi nii olla, sest juhuslik otsustamine, mida järgmiseks teha, on sageli kasulik/vajalik (n. quicksort). Juhsulik algoritm ei käitu alati ühtmoodi.

B: Kirjeldus peaks sisaldama vähemalt ühte tsüklit või ühte hargnevat sammu

Feedback: Algoritm võib olla ka lihtne järjestikuline tegevusjuhis ilma hargnemiste või tsükliteta

C: Algoritmist peab inimene aru saama

Feedback: Nii võiks ju olla. Kuid see, kas inimene saab aru või mitte, ei muuda algoritmi olemust

*D: Algoritm võib olla nii inimese kui arvuti leiutatud või olla inspireeritud isegi loodusest

Feedback: Tõesti võib algoritmi saamise/leiutamise aluseks olla erinevaid lähenemisi.

Question 16 - checkbox, partial credit, shuffle, medium difficulty

Pildil on õpetus ülikooli läbimiseks. (1. Registreeru ainele, 2. ???, 3. Saa EAP-d 4. Korda eelmisi samme). Kas tegemist on algoritmiga?

*A: Kui on, siis väga katkisega…

Feedback: Üldiselt need juhised siiski kõigile algoritmi tingimustele ei vasta, sest samud on ebamäärased.

*B: Ei, 2. samm on ebamäärane

Feedback: Algoritmi sammud peavad olema üheselt mõistetavad.

C: Ei, 3. samm on ebamäärane

Feedback: EAP-de saamist pole küll siin defineeritud, aga põhimõtteliselt saaks selle omaette algoritmina vormistada.

*D: Kuna juhistes on lõpmatu tsükkel siis see ei pruugi olla adekvaatne algoritm

Feedback: Tsüklitel peaks olema mingi lõpetamise või tsüklist väljumise tingimus.

F: Ei, sest pole ühtegi sisendit

Feedback: Algoritmil ei pea olema ühtegi sisendit.

Question 17 - multiple choice, shuffle, medium difficulty

Olgu antud hulk (suurusega n) täisarvu ja sorteerimisalgoritm, mille järgi kulub iga täisarvu õigesse kohta panemiseks sama palju samme kui on esialgses hulgas täisarve. Mis on sellise algoritmi ajaline keerukus?

*A: n**2 (n ruudus) - ruutkeerukus

Feedback: Jah, teisisõnu igal täisarvu kohta tehakse n sammu ehk kokku tehakse n elemendile igaühele kuni n sammu ehk n*n sammu kokku. Samm võib koosnedad mitmest tegevusest,

seda saab tähistada mingi konstandiga c, näiteks c * n * n. Kuid valemit domneerib n*n, seetõttu kutsume seda ruutkeerukuseks.

B: n - lineaarne keerukus

Feedback: N sammu teeb algoritm juba ainuüksi ühe täisarvu õigesse kohta paigutamiseks.

C: 1 (konstantne)

Feedback: Konstantne aeg tähendab et ükskõik kui suur oleks sisend, töötaks programm ikka sama kiiresti. See on veidi optimistlik. Kui muutub sisendiks antavate täisarvude arv, muutub reeglina ka sammude arv, mida algoritm tegema peab.

D: n*log(n) – "n log n keerukus"

Feedback: Liiga optimistlik. Kiiremad sorteerimisalgoritmid sorteerivad jah n * log(n) ajalise keerukusega, kuid siin antud kirjelduses see nii ei olnud.

E: log(n) - logaritmiline keerukus

Feedback: Liiga optimistlik. Logaritmilist seost siin ei ole, sest iga elemendi peale läheb aega n sammu ulatuses.

Question 18 - variation 2, multiple choice, shuffle, medium difficulty

Virtuaalses kaardipakis on 52 kaarti, nende töötlemiseks (näiteks juhuslikus järjekorras kaartide välja võtmiseks) teeb algoritm 52 sammu. Kui jagada kaart pooleks kulub poole vähem aega ja kui võtta kaks pakki siis poole rohkem aega. Mis on selle algoritmi ajaline keerukus?

A: n**2 (n ruudus) - ruutkeerukus

Feedback: Ei, sellise keerukusega algoritm teeks 52 kaardiga 52*52 = 2704 sammu.

*B: n - lineaarne keerukus

Feedback: Jah, teisisõnu teeb algoritm täpselt sama palju samme kui on kaarte pakis.

C: 1 (konstantne)

D: n*log(n))

Feedback: n log n seost siin ei ole

E: log(n)

Feedback: Logaritmilist seost siin ei ole.

Question 19 - multiple choice, shuffle, medium difficulty

Virtuaalses kaardipakis on 52 kaarti,nende töötlemiseks (näiteks juhuslikus järjekorras kaartide välja võtmiseks) teeb algoritm 15 sammu. 26-kaardise paki puhul teeb algoritm samuti 15 sammu. Mis on selle algoritmi eeldatav ajaline keerukus?

A: n**2 (n ruudus) - ruutkeerukus

Feedback: Ei, sellise keerukusega algoritm teeks 52 kaardiga 52*52 = 2704 sammu.

B: n

Feedback: Sellise keerukusega algoritm teeks 26-kaardise paki puhul vähem samme kui kaks korda suurema pakiga.

*C: 1 (konstantne)

Feedback: Jah, teisisõnu, algoritmi keerukus on konstantne. Näiteks kui jagada kaardipaki pealt laiali kolm viie kaardiga mängukätt. Siis pole tähtis, kui suur oli algne pakk.

D: n*log(n)

Feedback: Logaritmilist seost siin ei ole.

E: log(n)

Feedback: Logaritmilist seost siin ei ole.

Question 20 - multiple choice, shuffle, medium difficulty

Algoritmi sooritamise ajaline keerukus kirjeldab:

*A: Seda, seda, kuidas sõltub algoritmi tööaeg sisendi pikkusest.

Feedback:

B: Seda, kuidas sõltub mälu vajadus sisendi pikkusest.

Feedback: Peaaegu. Keerukus võib olla seotud nii aja kui mälumahuga. Kuid üldiselt eristatakse aja ja mälumahu keerukust omavahel. Ka väikese mäluga saab algoritm “kaua” tööd teha. Ajaline keerukus mõõdab “aega” vajalike tehete arvuga.

C: Seda, kui kaua programm minu arvutis konkreetse sisendi peal töötab

Feedback: Algoritmi ajaline keerukus mõõdab pigem, seda, mis juhtuks programmiga üha suuremate (ka lõpmatusse kasvava) sisendiga. Reeglina loetakse sammude arvu - seda, kui kaua see võütab konkreetses arvutis aega, pole alati võimalik lihtsalt teada saada,

*D: Seda, kuidas algoritmi tööaeg muutub sisendi kasvades, kasvõi lõpmatult suureks.

Feedback: Jah, algoritmi ajaline keerukus hindab, mis juhtuks programmiga üha suuremate sisenditega

Default Feedback: vale vastus OI EI!

Question 21 – text match, shuffle, medium difficulty

Eestis elav loom

*A: ilves

Feedback: õige vastus

B: suitsupääsuke

Feedback: peaagu, küll aga tegu on rahvuslinnuga mitte loomaga

*C: hunt

Default Feedback: vale vastus

VI. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Tobias Reiter,

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

   Testide teisendamine tekstidokumendist struktuursele kujule,

   mille juhendaja on prof. Jaak Vilo,

   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 4.0, 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.

Tobias Reiter

15.05.2024