TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

Beata Sillat

“Andmeturve” kursuse näitel automaathindamise realiseerimine ja hinnete automaatne laadimine Moodle’i keskkonda

Bakalaureusetöö (9 EAP)

Juhendaja(d): Alo Peets, MSc

Tartu 2024

“Andmeturve” kursuse näitel automaathindamise realiseerimine ja hinnete automaatne laadimine Moodle’i keskkonda

Lühikokkuvõte:

Bakalaureusetöö eesmärk oli kursuse “LTAT.06.002 Andmeturve” näitel realiseerida automaathindamise protsess ja automaatne hinnete laadimine Moodle’i keskkonda. Töös analüüsitakse meetodeid, mida kasutatakse Tartu Ülikoolis teiste kursuste raames hinnete automaatsesks sisestamiseks Moodle’i keskkonda. Paigaldati Moodle keskkond ja uuriti selle funktisoonaalsust ja arhitektuuri. Töö käigus arendati skripte esimese kahe praktilise töö hindamiseks aines “Andmeturve” ja hinnete ülekandmiseks Moodle’i platvormile. Samuti kirjeldati arendusetapid detailselt lahti. Skriptide arendamisel järgiti GDPRi nõudeid isikuandmete konfidentsiaalsuse ja kaitse tagamiseks. Vaadeldi võimalikku arendatud süsteemi kasutamist hariduskeskkonnas Tartu Ülikooli näitel.

Võtmesõnad:

API, Moodle, GDPR, skript, automaatne ülekandmine, automaatne hindamine, Andmeturve

CERCS:

P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Automated grading and grade uploading in Moodle environment: A "Computer Security" case study

Abstract:

The objective of this Bachelor's thesis was to implement a system for automatic grading and the seamless integration of grades into the Moodle environment, using the "LTAT.06.002 Computer Security" course at the University of Tartu as a case study. The thesis explores the methodologies employed at the university for the automatic uploading of grades within Moodle across various courses. The existing Moodle environment was initially examined to theoretically explore its architecture and structure, subsequently assessing its functionality. Scripts were meticulously developed for evaluating the initial two practical assignments in the "Computer Security" course, as well as for facilitating the efficient upload of grades to the Moodle platform. Each stage of development was exhaustively documented. Moreover, the development adhered to GDPR guidelines to ensure the confidentiality and protection of personal data. The feasibility of deploying this system within the educational framework of the University of Tartu was also thoroughly investigated.

Keywords:

API, Moodle, GDPR, script, automatic uploading, automatic assessment, Computer Security.

CERCS:

P170 Computer science, numerical analysis, systems, control

Sisukord

Sissejuhatus

Tartu Ülikoolis toimuvad enamik kursuseid Moodle'i keskkonnas [1]. Moodle on haridusplatvorm, mida üliõpilased ja õppejõud saavad kasutada erinevatel eesmärkidel, näiteks kursuse tutvustamiseks, materjalide vaatamiseks ja ülesannete täitmiseks [2]. Vaatamata selle süsteemi ulatuslikele võimalustele, jääb üliõpilaste praktiliste tööde kontrollimise protsess töömahukaks ja sageli nõuab õppejõududelt palju aega, eriti suurtel kursustel, kus osalejate arv on üle saja. Probleem, mida käesolevas bakalaureusetöös uuritakse, seisneb vajaduses optimeerida hindamisprotsessi kursuse "Andmeturve (LTAT.06.002)" näitel. Antud kursuse raames kulub õppejõududel praegu ühe praktilise ülesande hindamiseks 200–300 õpilase puhul keskmiselt 10–25 tundi.

Käesoleva töö eesmärk on Moodle'i testkeskkonna paigaldamine serverisse ja skriptide arendamine, mis optimeerivad ja automatiseerivad üliõpilaste tööde hindamise protsesse esimese kahe "Andmeturve" praktilise töö näitel. Lisaks sellele töö käigus arendatakse süsteem, mis automaatselt kannab üle hindeid Moodle'isse, kasutades Moodle'i API-d platvormiga suhtlemiseks, minimaalse kasutaja osalusega. Arendatud skriptid kasutavad ainult Moodle'i süsteemi poolt genereeritud identifikaatoreid ning ei töötle otseselt üliõpilaste isikuandmeid.

Töö on jagatud neljaks suuremaks peatükiks. Esimeses osas kirjeldatakse tööga seotud mõisted ja terminid. Teises osas antakse ülevaade Moodle'i platvormi arhitektuurist ja funktsionaalsusest, praeguste hindamismeetodite analüüsist ning andmekaitse põhimõtetest. Kolmandas osas analüüsitakse meetodeid hinnete automaatseks ülekandmiseks Moodle'isse teistest süsteemidest, mida juba kasutatakse Tartu Ülikoolis. Neljandas osas vaadeldakse skriptide loomist ülesannete automaatseks kontrollimiseks ja hinnete automaatseks ülekandmiseks Moodle'i platvormile. Samuti kirjeldatakse võimalikke edasisiarengusvõimalusi. Töö lõpus on olemas lisa, kuhu lisati link GitHubi repositooriumile, kus need skriptid on saadaval. Töö koostamisel oli kasutatud ChatGPT tekstiroboti abi, et saada tagasisidet töö sisukorra ja sisupeatükkide kavandite struktureerimise kohta ning peatükkide põhiteksti keelekasutuse korrektsuse osas. Saadud tagasiside põhjal viimistleti töö teksti ning parandati keelevigasid.

1. Mõisted ja terminid

API (ingl *Application Programming Interface)* on reeglistik, mille alusel arvuti operatsioonisüsteem või rakendusprogramm toimib, see võimaldab rakendusprogrammil kasutada süsteemi teenuseid erinevate protokollide ja tarkvararakenduste vahendite abil [3].

Assotsiatiivne massiiv (ingl *associative array*) on andmestruktuur, mis salvestab võti-väärtus paarid [4].

GDPR (ingl *General Data Protection Regulation*) on Euroopa Liidu üldmäärus, mis käsitleb isikuandmete töötlemisel füüsiliste isikute kaitset [4].

HTTP (ingl *HyperText Transfer Protocol*) on protokoll mida kasutatakse veebis HTML-dokumentide vahetamiseks [3].

JSON (ingl *JavaScript Object Notation*) on lihtne andmevahetusformaat, mis põhineb JavaScripti alamhulgal ning on lihtne lugemiseks ja kirjutamiseks inimese jaoks [4].

LAMP on lühend, mis tähistab vabavara komplekti, kuhu tavaliselt kuuluvad järgmised komponendid:

- L – Linux (operatsioonisüsteem);
- A – Apache (veebiserver);
- M – MySQL (andmebaas);
- P – Perl, PHP ja/või Python (skripti keeled) [3].

LTI (ingl *Learning Tools Interoperability*) on standard, mis võimaldab õpetööriistadel, mida pakuvad erinevad platvormid, integreerida õppehaldussüsteemidega, võimaldades õpetajatel ja õppijatel kasutada erinevaid õppetööriistu ühes keskkonnas [5].

Lähtekood (ingl *source code*) on kood, mis on esitatud sellisel kujul, et see sobib sisendiks assemblerile, kompilaatorile või muule translaatorile [4].

MySQL on relatsioonbaasihaldur, mille lähtekood on avatud ja mille on välja töötanud Rootsi firma MySQL AB. See kasutab struktureeritud päringukeelt ning MySQL erineb kiiruse, töökindluse ja paindlikkuse poolest, mis on sisuhalduse jaoks oluline [3].

Pistikprogramm (ingl *plugin*) on lisarakendus, mis laiendab olemasoleva programmi funktsionaalsust ja on lihtsalt installitav [4].

Protokoll (ingl *protocol*) on formaalne määratlus andmevahetuse vormingutele ja reeglitele, eriti andmeside kontekstis. See võib olla teostatud nii riist- kui ka tarkvaras ning määrab suhtluse süntaksi, semantika ja sünkroniseerimise [4].

Tõend ehk rakendusepääse (ingl *token*) on ühene identifikaator, mis on mõeldud juurdepääsu taotleva rakenduse autentimiseks [4].

URL (ingl *Uniform Resource Locator*) on unikaalne aadress, mis on määratud igale internetis asuvale veebidokumendile või ressursile [3].

Virtuaalmasin (ingl *Virtual Machine*) on tarkvara, mis võimaldab käivitada erinevaid operatsioonisüsteemi samaaegset ühel arvutil, kus iga operatsioonisüsteem käivitab oma programme [3].

2. Teoreetiline taust

Järgnevates peatükkides vaadeldakse teoreetilisi aspekte, mis on vajalikud Moodle'i automaatse hinnete ülekandmiseks süsteemi mõistmiseks ja arendamiseks. Selgitatakse välja isikuandmete kaitse seaduste tähtsus, analüüsitakse praeguse hindamissüsteemi probleemi ning uuritakse Moodle'i arhitektuuri.

2.1 GDPR'i tähtsus

Euroopa Liidu isikuandmete kaitse üldmäärus, tuntud ka kui GDPR ehk General Data Protection Regulation, mis jõustus 25. mail 2018, on oluline standard konfidentsiaalsuse ja andmekaitse tagamisel [6]. Uus seadusandlus tugevdab andmekaitset Euroopa Liidus, et lahendada uusi privaatsuse valdkonna väljakutseid, mis tekivad seoses digitaalsete tehnoloogiate arenguga [7].

GDPR reguleerib isikuandmete kogumist, säilitamist ja töötlemist [8]. Isikuandmete hulka kuuluvad kõik andmed, mis võivad olla seotud konkreetse füüsilise isikuga: täisnimi, isikukood, telefoninumbrid, IP-aadressid, e-posti aadressid, asukohateave, sünniajad, samuti muu teave, mis on seotud isiku geneetilise, majandusliku, kultuurilise või sotsiaalse identiteediga [7]. Andmeid, mis ei sisalda selliseid identifikaatoreid, peetakse tavaliselt anonüümseteks ja ei kuulu GDPR reguleerimisalasse [7]. Üliõpilaste hinded on isikuandmed, sest need on seotud konkreetse üliõpilase nimega hindetabelis. Õppejõul on vaja üliõpilase nime ja hindeid töödelda oma igapäevatöös õppetöö läbiviimise eesmärgil ehk on tegu põhjendatud töötlemisega. Samas õppejõud peab olema hoolsad, et ei levita ega publitseeri hindeid teistele üliõpilastele ega muudele kõrvalistele isikutele.

Üks GDPR'i olulistest põhimõtetest on andmete minimeerimise (ingl *data minimization*) põhimõte, mis nõuab, et isikuandmeid kogutaks ja töödeldaks ainult vajalikus ulatuses ning otstarbekalt [8]. See tähendab, et ülikool peaks koguma ainult neid andmeid, mis on vajalikud konkreetse ülesande või teenuse täitmiseks ning mitte rohkem. Andmete kaitsmiseks ja privaatsuse tagamiseks kasutatakse sageli pseudonümiseerimise meetodit, kus identifitseeritavad parameetrid asendatakse juhuslike identifikaatoritega [8]. Kuigi Moodle kasutab unikaalseid, kuid mitte juhuslikke kasutaja identifikaatoreid, käesolevas töös rakendatakse põhimõtet andmete minimeerimiseks, asendades üliõpilaste isiklikud nimed ja perekonnanimed kasutaja identifikaatoriga. See võimaldab suurendada isikuandmete konfidentsiaalsust, järgides GDPRi nõudeid.

2.2 Probleem praeguse hindamissüsteemiga

Tartu Ülikoolis korraldatakse igal aastal küberturvalisuse kursus "LTAT.06.002 Andmeturve". Kursuse hinde moodustavad 60% praktikumide harjutused, 10% referaat, 5% Moodle kontrolltest ja 25% kirjalik eksam [9]. Aine õppimise käigus üliõpilased lahendavad enamiku ülesandeid VirtualBoxi virtuaalmasinates või VPN vahendusel Tartu Ülikooli sisevõrgus [9]. VirtualBox on avatud lähtekoodiga tarkvara, mis võimaldab x86 ja AMD64 arhitektuuri virtualiseerida [10]. Aine praktilised ülesanded hõlmavad mitmesuguste teemade uurimist, sealhulgas paroolide murdmist, OpenSSL-i ja sertifikaatidega töötamist, võrgu turvalisuse tagamist, krüpteerimist ja palju muud [9].

Hetkel toimub hindamisprotsess nii, et üliõpilased esitavad töid Moodle keskkonnas, pärast mida õpetajad kontrollivad saadud faile ja sisestavad hinded. Isikuandmete kaitse nõuete (GDPR) tõttu ei ole võimalik avalikustada üliõpilaste isiklikke andmeid, nagu nimesid ja perekonnanimesid, ühises tabelis koos hinnetega. Seetõttu üliõpilased saavad Moodle'i platvormil näha ainult oma hinnet. Üliõpilased laadivad üles erinevate vormingutega faile kontrollimiseks, näiteks võivad need olla ekraanipildid või CSV/TXT formaadis failid, mis sisaldavad vastuseid ülesannetele. Failid võivad olla esitatud nii üksikult kui ka kokku pakitud kujul. See tähendab, et õpetaja peab käsitsi läbi vaatama ja kontrollima iga faili eraldi. Eelnev on õppejõududele märgatav ajakulu ning tasub uurida, kas hindamise protsessi oleks võimalik automatiseerida. Tavaliselt osaleb Andmeturve kursusel 200-300 õpilast, mis tähendab, et õpetaja peab hindama suurt hulka faile. Iga üliõpilase ühe praktilise tunni käsitsi kontrollimine võtab keskmiselt 3-5 minutit, mis tähendab, et 200-300 üliõpilase kontrollimine võtab kokku umbes 10 kuni 25 tundi. Kirjeldatud hindamisviisi probleem seisneb selles, et see on ebamugav ja võtab palju aega, nii hindamiseks kui ka hinnete Moodle'i keskkonda ülekandmiseks. Automaatse hindamise ja hinnete automaatse üleslaadimise näited on realiseeritud just sellele kursusele. Automaathindamise protsessi analüüsimiseks ja testimiseks kasutatakse kahe esimese praktikumi näiteid. Antud lahendus optimeerib hindamisprotsessi, ja säästab nii õpetajate kui ka üliõpilaste aega.

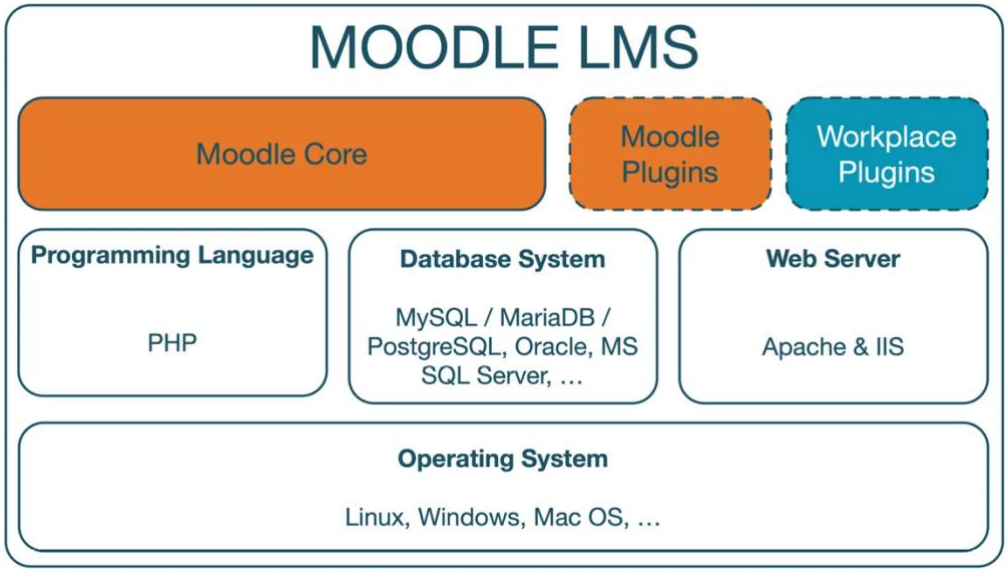
Käesolevas töös arendatakse automatiseeritud süsteem üliõpilaste praktiliste ülesannete hindamiseks ja tulemuste ülekandmiseks Moodle'i platvormile, tagades samal ajal andmete töötlemise ja säilitamise vastavuse GDPR standarditele. Igale Moodle'i õpilasele genereeritakse nende registreerimisel süsteemis automaatselt unikaalne identifikaator numbrilisel kujul, tagades individuaalse tuvastamise ja samas säilitades kasutaja

isikuandmete konfidentsiaalsuse ja turvalisuse. See identifikaator sisaldub CSV-failis, mida kasutatakse hinnete automaatseks ülekandmiseks. Oluline on märkida, et CSV-fail, mis sisaldab kasutajate ID-sid ilma isikliku teabeta, hoitakse ja on kättesaadav ainult kursuse õppejõule. Kõrvalistel isikutel pole sellele failile ligipääsu.

2.3 Moodle'i arhitektuur ja töö põhiprintsiibid

Moodle on avatud lähtekoodiga haridusplatvorm, mille eesmärk on pakkuda õpetajatele, administraatoritele ja õpilastele ühtset, usaldusväärset, turvalist ja integreeritud süsteemi, mida saab kohandada ja kasutada personaalse õpikeskkonna loomiseks [11]. Moodle'i platvorm omab kvaliteetset arhitektuuri, on tasuta ja saadaval kõigile interneti kasutajatele[2].

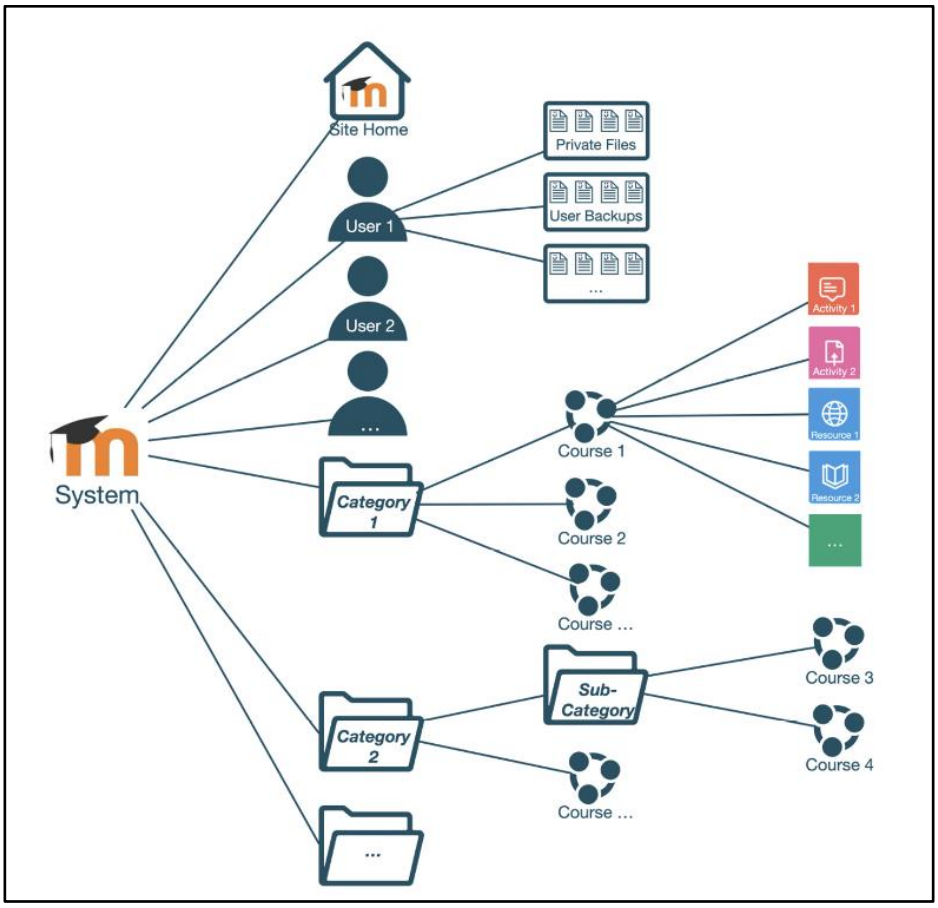
Järgnevalt antakse ülevaade Moodle'i arhitektuurist ja struktuurist põhinedes "Moodle 4 Administration: An administrator's guide to configuring, securing, customizing, and extending Moodle" õpikule [12]. Moodle loodi LAMP-struktuuri põhjal, mis sisaldab Linuxi operatsioonisüsteemi, Apache veebiserverit, MySQL andmebaasi ja PHP programmeerimiskeelt. Joonisel 1 on esitatud üldine Moodle'i arhitektuuri skeem, mis näitab platvormi peamisi komponente ja toetatud süsteeme. Nende hulka kuuluvad Moodle'i tuum (ingl *Moodle core*), programmeerimiskeel (ingl *programming language*), toetatud andmebaasisüsteemid (ingl *database systems*), veebiserverid (ingl *web servers*) ja erinevad operatsioonisüsteemid (ingl *operating systems*).



Joonis 1. Üldine Moodle platvormi arhitektuur [12].

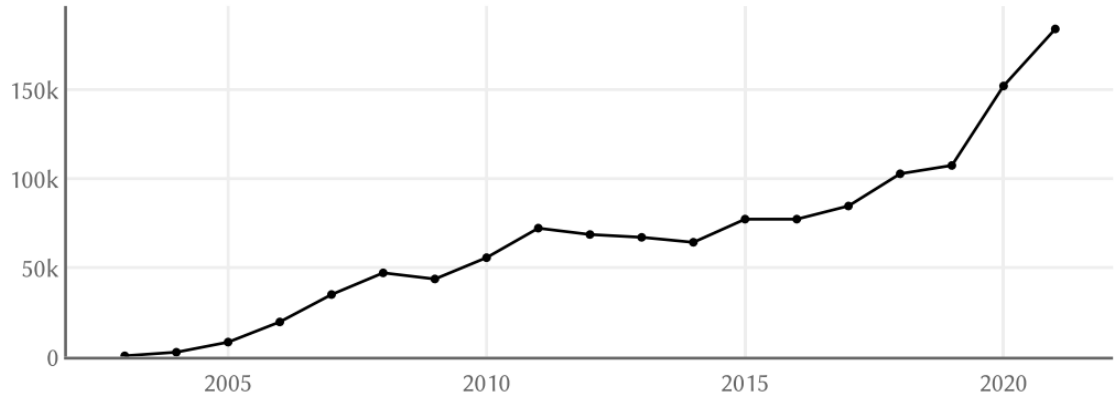
Tänu nende komponentide paindlikkusele ja Moodle'i moodulite struktuurile (sealt ka "M" nimetuses), toetatakse paljusid erinevaid operatsioonisüsteeme, andmebaase ja veebiservereid.

Moodle's iga kursus sisaldab erinevaid õppematerjale ja ülesandeid, mida võib grupeerida alajaotisteks parema korralduse jaoks. Platvormis on igal kasutajal oma isiklik failihoidla, mida saab kasutada isiklike materjalide või kursuste varukoopiate hoidmiseks. Kuigi Moodle süsteemis on ainult üks pealeht, toimib see platvormi keskse sõlmpunktina, pakkudes juurdepääsu kõikidele kursustele ja kasutajafailidele ning lihtsustades navigeerimist [12]. Moodle'i struktuur on esitatud joonisel 2.



Joonis 2. Moodle struktuur [12].

Üle 184 000 veebisaidi kasutab Moodle'it, kus on kokku 260 miljonit kasutajat [13]. Neis veebilehtedes on üle 293 miljoni ressursi ja üle 4 miljardi testiküsimuse (2021. aasta seisuga) [13]. Moodle'i kasutavate veebisaitide arv on pidevalt kasvanud alates tarkvara loomisest 2003. aastal, nagu näidatud joonisel 3.



Joonis 3. Moodle'i lehtede registreeritud arv alates 2003. aastast [13].

Tänu oma paindlikkusele ja skaleeritavusele Moodle'i efektiivselt kasutatakse koolides, ettevõtetes, valitsusasutustes ja ühiskondlikes organisatsioonides [11]. Moodle'i platvorm võimaldab tarkvara paigaldamist isiklikule veebiserverile, võimaldades klientidel kohandada ja muuta õpikeskkonda vastavalt nende vajadustele ja eelistustele [11].

Enamik Tartu Ülikooli kursuseid kasutab Moodle'i õpikeskkonda[1] õppetöö läbiviimiseks. Eriti suurenes Moodle'i kasutus ka kurustele, mis olid veel e-õppe toeta COVID-19 pandeemia ajal. Paljud õppejõud jäid Moodle'i keskkonda kasutama õppetöö läbiviimisel ka pärast kohustusliku distantsõppe lõppu. Kursused võivad sisaldada erinevat teavet selle kohta, millest kursus koosneb, õppematerjalid, üliõpilaste teadmiste kontrollimiseks testid, samuti on olemas võimalus luua Moodle'is kontrolltöid või isegi eksameid [2]. Moodle'i hindamissüsteem võimaldab õppejõududel hinnata töid mitmekülgselt, kasutades erinevaid hindamiskriteeriume ja tagasiside vorme, mis on üks peamisi põhjuseid, miks Moodle'i eelistatakse õpikeskkonnana. Üks paljudest kasulikest funktsioonidest on ka foorumid, kus üliõpilased ja õppejõud saavad omavahel suhelda ja näiteks küsimusi materjalide või testide kohta esitada [2].

[1] https://moodle.ut.ee

2.4 Moodle'i integreerimise võimalused teiste süsteemitega ja API kasutamine

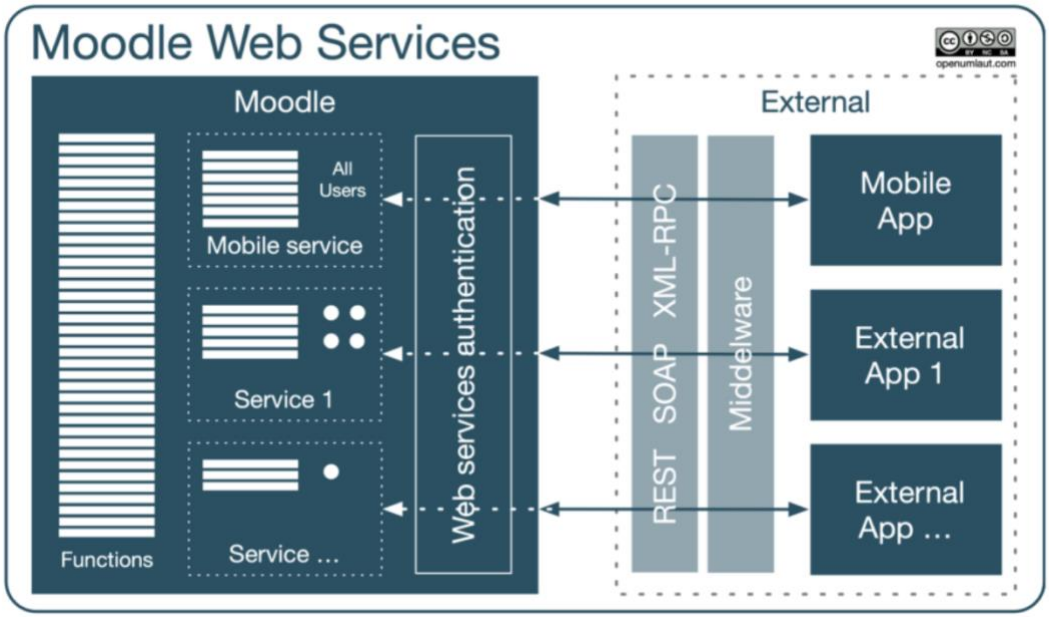
Moodle platvormil on võimalik funktsionaalsust laiendada ja integreerida erinevate programmidega [11]. Moodle'i integreerimine teiste süsteemidega tähendab nende ühendamist üheks tervikuks, et need saaksid omavahel sujuvalt töötada.

Moodle'is on mitu viisi teiste süsteemide ja ressurssidega integreerumiseks. Nende hulka kuuluvad LTI välised tööriistad (ingl *LTI external tools*), erinevad pistikprogrammid ning API platvormiga suhtlemiseks HTTP-päringute kaudu [14]. LTI välised tööriistad on täiendavad rakendused, mida saab integreerida kursusesse (näiteks interaktiivsed ülesanded) [15]. Õpetajad saavad lisada nende ülesannetele lingid otse Moodle'i kursuse lehele ning üliõpilased saavad neid kasutada, väljumata Moodle'ist või sisse logimata teise süsteemi[15].

Käesolevas bakalaureusetöös viiakse hinnete automaatne ülekanne Moodle'i platvormile läbi API, kasutades veebiteenuste funktsioone[2]. Moodle'i platvormi pakutavad veebiserveri funktsioonid moodustavad olulise osa automatiseerimise ja optimeerimise protsessist. Veebiteenused on platvormi funktsionaalsuse laiendamise ja teiste süsteemidega integreerimise mehhanism, mis on visualiseeritud joonisel 4 [12]. Järgnevalt antakse ülevaade sellest teemast põhinedes "Moodle 4 Administration: An administrator's guide to configuring, securing, customizing, and extending Moodle" õpikule [12]. Veebiteenused võimaldavad teistel süsteemidel teha toiminguid Moodle'is ja vastupidi. Veebiteenuste toimimise aluseks on Moodle'is pakutavate funktsioonide kogum ja kasutajate ligipääsu nende funktsioonidele määramine. Näiteks Moodle'is on võimalik kasutada funktsioone, nagu hinnete lisamine ja kursuste andmete hankimine. Kasutades sisseehitatud teenust, näiteks mobiilirakendustele loodud teenust (ingl. *Mobile Service*), on sellele juurdepääs kõigile Moodle kasutajatele. Erinevalt sisseehitatud teenusest, võib kohandatud teenusele ligipääsu anda ainult kindlatele kasutajatele. Turvalisuse tagamiseks peavad välised rakendused läbima Moodle'i veebiteenuste autentimisplugini kaudu autentimise.

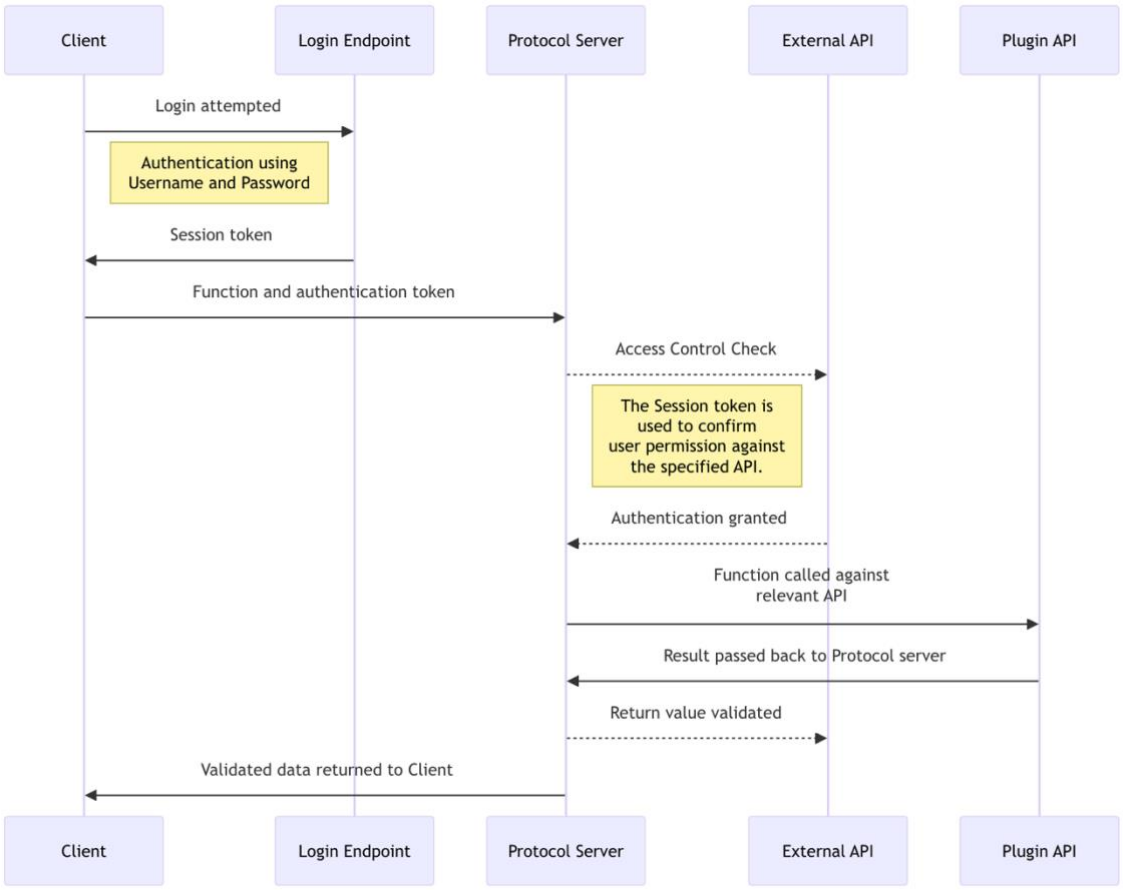
[2] https://docs.moodle.org/dev/Web_service_API_functions

Veebiteenuste toetatud protokollid sisaldavad SOAP-protokolli, RESTi ja XML-RPC-protokolli [12].



Joonis 4. Moodle veebiteenused [12].

Moodle API-ga loodud skripti koostoime illustreerimiseks võib kasutada autentimise protsessi ja tööprotokolli diagrammi, mis on toodud joonisel 5 [14]. Joonisel 5 kujutatud järgnevusskeem (ingl *Sequence Diagram*) illustreerib, kuidas Moodle API-ga suhtlemise protsess toimub. Diagramm sisaldab peamisi elemente nagu klient (ingl *Client*), logimise otspunkt (ingl *Login Endpoint*), protokolli server (ingl *Protocol Server*), ja väline API (ingl *External API*). See visualiseerib, kuidas autentimisprotsess käib alates kasutaja logimisest ja seansi identifikaatori (ingl *session token*) saamisest kuni funktsioonide kutsumiseni ja tulemuste tagastamiseni kliendile. Nagu diagrammil näidatud, läbib klient (antud juhul hinnete laadimise skript) autentimise, kasutades unikaalset tõendit (ingl *token*), mis annab ligipääsu API funktsioonidele. Tõend tagab turvalisuse ja kinnitab päringute autentsuse Moodle'i veebiteenusele, kõrvaldades vajaduse edastada kasutajatunnuseid ja paroole iga päringuga.



Joonis 5. Moodle API autentimise ja protokolli diagramm [14].

Joonisel 5 on näha, kuidas koostoimeprotsess algab tõendi ja funktsiooni nime saatmisega protokolli serverile. Protokolli server viib läbi ligipääsu kontrolli, kinnitades skripti õigused, lähtudes antud tõendist, mis võimaldab juurdepääsu API funktsioonidele. Pärast autentimise kinnitamist saab skript kutsuda vajalikke funktsioone, näiteks *core_course_get_contents* kursuseinfo hankimiseks ja *core_grades_update_grades* hinnete värskendamiseks. Kui API on töötlenud päringud ja täitnud vajalikud funktsioonid, siis saadetakse tulemused tagasi protokolli serverile. Seejärel kontrollitakse andmeid ja edukate päringute korral saadetakse need tagasi kliendile ehk skriptile. See meetod tagab sujuva ja turvalise koostoime välissüsteemide ja Moodle'i vahel, toetades protsesside automatiseerimist ja turvastandardite järgimist.

3. Võrdlev analüüs hinnete ülekandest Moodle'i keskkonda.

Käesolevas peatükis analüüsitakse olemasolevaid meetodeid hinnete ülekandmiseks või lisamiseks Moodle'i platvormile, mida kasutatakse Tartu Ülikooli erinevates kursustes. Analüüsi läbiviimiseks võeti ühendust ülikooli infotehnoloogia osakonnaga, mille käigus koguti informatsiooni kasutatavate meetodite kohta. Moodle'i kursustele saab välistest süsteemidest hindeid üle kanda mitme Tartu Ülikooli õppejõudude loodud arenduse puhul:

1. Hinnete ülekanne süsteemist Lahendus[3]. Lahendus see on ülesannete automaatkontrollisüsteem, mis on loodud selleks, et õpilased saaksid iseseisvalt töötades ülesannete lahendamisel automaatset tagasisidet [16]. See süsteem kasutab kahte veebiteenust: üks õpilaste andmete saamiseks ja teine hinnete edastamiseks. Ülesannete andmeid üldse ei tõmmata, vaid hinde edastamise teenus tekitab ise uue hindeveeru, mille identifikaatorid on määratud IT osakonna poolt. Seega vastavat ülesannet Moodle'is polegi, on ainult hindelahter hindetabelis. Lahenduse puhul on kõik ülesanded ainult Lahenduses ja kogu esitamine ja hindamine käib samuti seal.

Õpilaste andmete tõmbamine ja sünkroonimine Lahendus platvormis toimub üks kord ööpäevas öösel ning seda on võimalik ka käsitsi käivitada Lahendus platvormist. Hinded kantakse üle reaalajas ükshaaval kohe, kui nad ilmuvad, näiteks kui automaatkontroll hindab konkreetse üliõpilase lahenduse ära või kui õpetaja muudab hindeid. Samuti on võimalik kõik olemasolevad hinded korraga üle kanda, mis võib olla vajalik mõne vea või eelneva sünkroniseerimisprobleemi korral.

2. Füüsika instituudis arendati süsteemi kodulaborite tööde hinnete ülekandmiseks Moodle'isse. Süsteemi tehnilised aspektid ja töö detailid on piiratud, kuna süsteemi arendaja ei tööta enam Tartu Ülikoolis. Kuid on teada, et see meetod on seotud CSV-faili kasutamisega, mis sisaldab kõiki vajalikke andmeid ja imporditakse hinnete ülekandmiseks Moodle'sse.

Lisaks eelmainitud süsteemidele on Moodle'is MatLab Graderi ja Panopto Quizi puhul LTI ühendusega hinnete ülekandmine Moodle'i kursusele. MatLab Grader on tööriist

[3] https://lahendus.ut.ee/

kodeerimisülesannete automaatse hindamiseks [17] ja tagasiside andmiseks, samas kui Panopto Quiz võimaldab luua video sisule põhinevaid küsitlusi ja teste [18].

4. Praktiline töö

Järgnevates peatükkides kirjeldatakse välja töötatud skriptide rakendamist hinnete automatiseeritud ülekandmiseks Moodle'i platvormile ja kahte praktikumide kontrollimiseks aines "Andmeturve". Kõikide skriptide lähtekood on kättesaadav GitHubi repositooriumis, mis on kättesaadaval aadressil https://github.com/BeataSl/automatic-grading-uploading (Lisa I). Samuti uuritakse Moodle'i serveri paigaldamise protsessi ning turvalisuse tõendi kasutamist veebiteenuste seadistamiseks.

Skriptide kirjutamiseks valiti programmeerimiskeeleks PHP[4]. PHP on programmeerimiskeel, milles Moodle on arendatud, ja see on integreeritud veebiserverisse [19]. Kui kasutaja küsib Moodle'i lehekülge, tuvastab veebiserver, et see on kirjutatud PHPs ja saadab selle PHPs töötlemiseks [19]. See rõhutab PHP olemasolu tähtsust Moodle'i töös ning miks just see programmeerimiskeel valiti skripti kirjutamiseks.

4.1 Moodle'i serveri paigaldamine ja demokursuse loomine

Töö algusfaasis pöörduti Tartu Ülikooli IT-osakonna poole päringuga, et saada ligipääsu vastava lõputöö tegemiseks nende paltvormil https://moodle.ut.ee/. Saadud vastus oli negatiivne ning soovitati esmalt läbi viia testid laborikeskkonnas kohaliku testmoodle peal, enne kui kaalutakse uuesti ligipääsu andmist. Seejärel algas testimiskeskkonna ettevalmistamine, sealhulgas serveri loomine ja Moodle installimine. Käesolevas töös otsustati paigaldada Moodle'i versioon 4.1 LTS, kuna Tartu Ülikooli IT-osakonna saadud informatsiooni põhjal kasutavad nad just seda versiooni ja ei kavatse lähitulevikus üle minna teisele. See tagab parema ühilduvuse ülikooli praeguse infrastruktuuriga. Samuti sama versiooni kasutamine väldib erinevaid probleeme, mis võiksid tekkida andmete ülekandmisel testkeskkonnast põhikeskkonda.

Moodle'i keskkonna paigaldamise protsessi käigus kasutati juhendit[5], mis asub Moodle'i ametlikul veebisaidil, kus on põhjalikult kirjeldatud kõik kohustuslikud sammud. Veebiserveri paigaldamine ja seadistamine oli üks esimesest sammust. Käesolevas töös

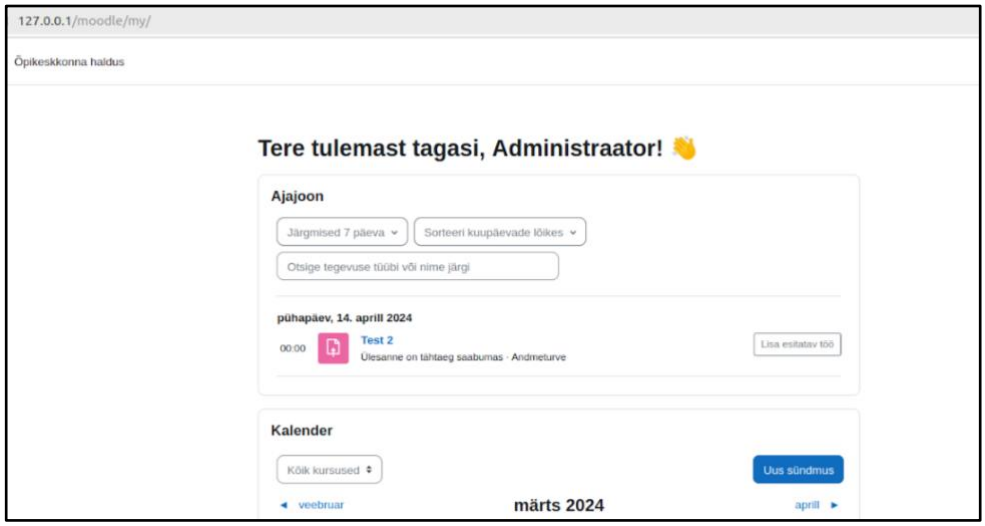
[4] https://www.php.net/

[5] https://docs.moodle.org/401/en/Installing_Moodle

valiti veebiserveriks Apache, mis paigaldati virtuaalmasinale Linux Ubuntu operatsioonisüsteemiga.

Moodle'i ametlikust GitLabi repositooriumist[6] klooniti vajalikud failid Moodle'i platvormi paigaldamiseks, kus on samuti saadaval erinevad platvormi versioonid. Oluline etapp oli Moodle'i failide kaitsmine veebiserveri kirjutamise eest. Failidele ja Moodle'i kataloogidele seadistati õiged ligipääsuõigused vastavalt turvalisuse soovitustele. Seejärel loodi uus andmebaas, mis on vajalik Moodle'i platvormi korrektselt toimimiseks. Andmebaasi peamine eesmärk on hoida kogu platvormi kasutajate, kursuste, õppematerjalide ja muu olulise andmeid. Minimaalne MySQLi versioon, mis on vajalik töö raames kasutatava Moodle'i jaoks, on 5.7 [20]. Uue andmebaasi loomisel kasutati viimast versiooni, mis on 8.0, et tagada ühilduvus Moodle'i süsteemi nõuetega. Pärast seda loodi Moodle'i andmekataloog (*moodledata*), mis on vajalik kõikide Moodle veebilehe failide salvestamiseks.

Viimaseks etapiks oli Moodle'i installeerimise-programmi käivitamine, mis lõi vajalikud andmebaasi tabelid ja seadistas uue veebilehe. Pärast edukat installimist loodi ka administraatori konto ja Moodle'i platvorm oli saadaval lingil http://127.0.0.1/moodle, vaata joonis 6.



Joonis 6. Moodle'i pealeht administraatori vaates.

[6] https://github.com/moodle/moodle.git

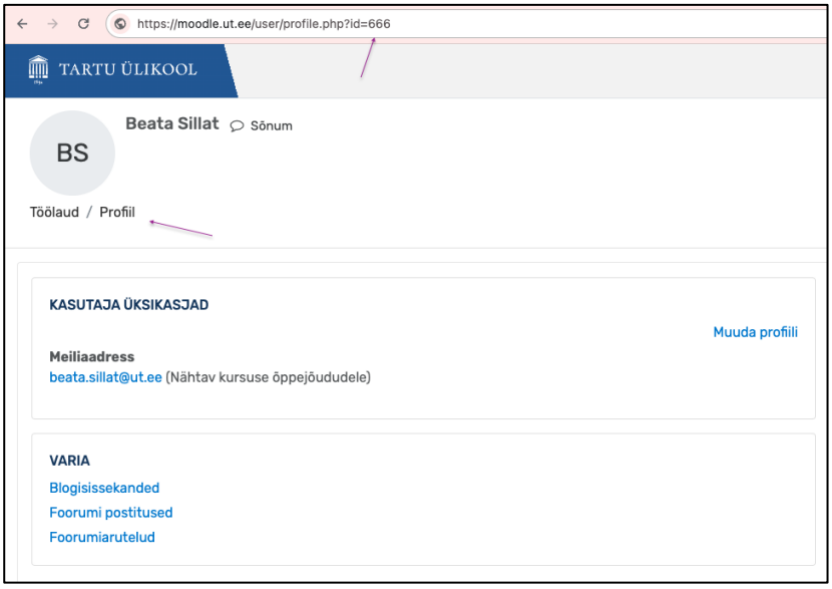
Pärast Moodle’i platvormi paigaldamist loodi üliõpilaste testkontod. Samuti loodi demokursus ja lisati need sinna, mis võimaldab testida arendatud süsteemi testkeskkonnas, kus osalevad üliõpilased. Demokursusel lisati kaks hinnatavat tegevust: "Praktikum 1" ja "Praktikum 2".

4.2 Andmeturve praktikumide automaatse hindamise skriptide arendamine

Käesolevas töös otsustati arendada skripte ülesannete automaatseks kontrollimiseks, mida üliõpilased teevad “Andmeturve” esimese kahe praktikumi raames. Nendes praktikumides tuletavad meelde üliõpilased Linuxi käsurea kasutamist ja uurivad uusi teemasid, mille järel nad täidavad ülesandeid ja esitavad õpetajale kontrollimiseks.

Esimese ja teise praktikumide skripti eesmärk on genereerida CSV-formaadis fail hinnetega, mis sisaldab vajalikke andmeid: unikaalne kasutaja identifikaator, praktikumi nimetus ja saadud hinne. Seda faili tulemustega kasutatakse Moodle’i põhiskriptis hinnete ülekandmiseks.

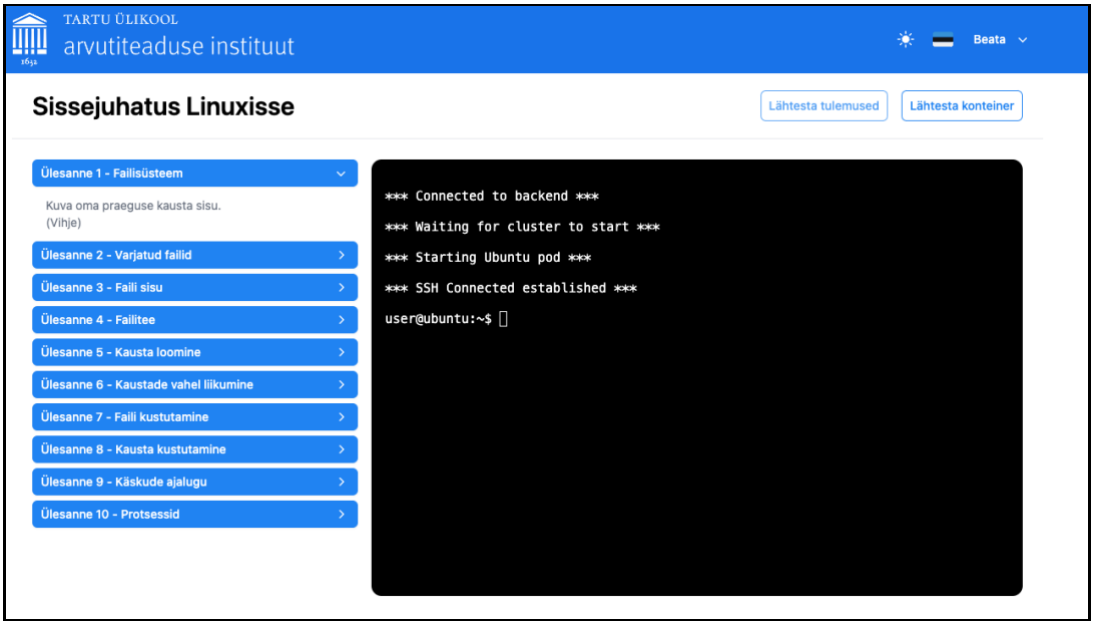
Praktikumide lahenduse protsessis peavad üliõpilased kasutama oma identifikaatorit, mida saab leida Moodle’i "Profiil" aknas. Selleks tuleb minna oma profiili ja vajutada nuppu "Profiil", pärast mida on võimalik leida oma identifikaatorit aadressiribal, nagu on näidatud joonisel 7.



Joonis 7. Identifikaatori leidmine aadressiribal.

4.2.1 Praktikumi 1 ülesannete ülevaade ja skripti analüüs

Praktikumi 1 - "Sissejuhatus töökeskkonda" raames ette valmistavad üliõpilased töökeskkonna järgnevateks praktikumideks, paigaldades Linuxi operatsioonisüsteemi virtuaalmasinasse. Linuxis harjutavad üliõpilased erinevaid põhilisi terminalikäske. Pärast Linuxi terminalikäske harjutamist peavad õpilased täitma hindamisülesande Linuxi käskude harjutamise keskkonnas[7], mis on näidatud joonisel 8. Peale ülesannete täitmist selles keskkonnas salvestatakse õpilaste tulemused serverisse, kust õpetaja saab need alla laadida oma arvutisse CSV-failina, mis sisaldab õpilaste identifikaatoreid ja nende hindeid.



Joonis 8. Linuxi käskude harjutamise keskkond.

Moodle'i automaatse hinnete ülekande põhiskripti kasutamiseks tuleb kõik üliõpilaste failid tulemustega ühendada ühte faili. Lisaks tuleb sellesse faili lisada kolmas parameeter - ülesande nimi. See on vajalik selleks, et iga hinne oleks seotud vastava ülesannega Moodle's.

Skripti käivitamiseks on vaja määrata õige tee kausta, kus asuvad kõik allalaaditud üliõpilaste tööd. Pärast selle sammu täitmist skript alustab oma põhitööd tsükliga, läbides iga faili kaustas. Kõik faili read salvestatakse tühjas massiivis *$lines*, seejärel teise tsükli

[7] http://lingid.ee/kasurida

abil eraldatakse kasutaja identifikaator *$userid* ja hinne *$grade*. Joonisel 9 on kujutatud kaks neid tsüklit.

```
foreach (scandir($folder_path) as $filename) {
    $file_path = $folder_path . '/' . $filename;

    if ($filename === '.' || $filename === '..') {
        continue;
    }

    $studentFileLines = file($file_path, FILE_IGNORE_NEW_LINES);

    array_shift($studentFileLines);

    foreach ($studentFileLines as $line) {
        $row = str_getcsv($line);
        $userid = $row[0];
        $assign_name = 'Praktikum 1';
        $grade = $row[1];

        fputcsv($result_csv_file, [$userid, $assign_name, $grade]);
    }
}
```

Joonis 9. Koodiblokk skriptis failide töötlemise tsüklitega.

Tulemusena luuakse fail nimega "resultPraktikum1.csv", kuhu salvestatakse õpilaste tulemused. Joonisel 10 on visualiseeritud kuidas võib välja näha see fail. Fail sisaldab kolme veergu: kasutaja identifikaator (userid), ülesande nimi(assignName) ja hinne(grade).

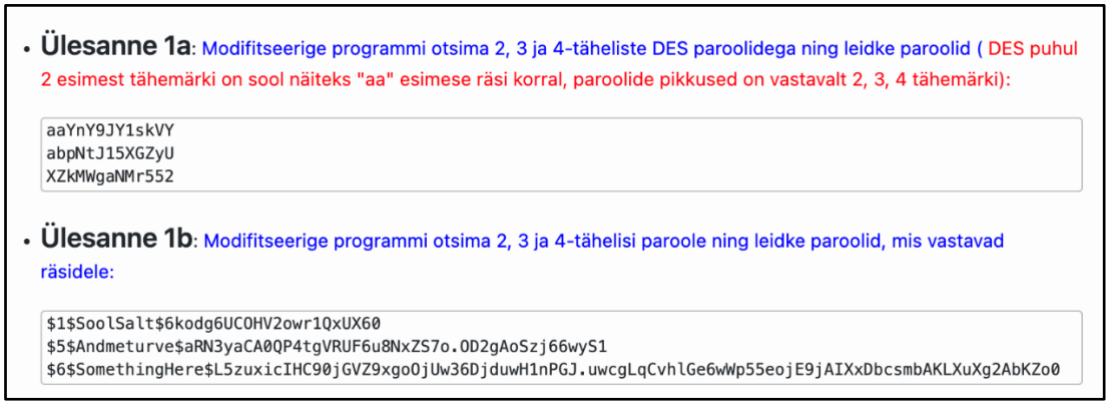
	A	B	C
1	userid	assignName	grade
2	9	Praktikum 1	2
3	8	Praktikum 1	0
4	7	Praktikum 1	0.5
5	6	Praktikum 1	2

Joonis 10. Näide "resultPraktikum1.csv" faili sisust.

4.2.2 Praktikumi 2 ülesannete ülevaade ja skripti analüüs

Praktikumi 2 - "Paroolide murdmine" käigus üliõpilased uurivad, kuidas arvutisüsteemid salvestavad paroole, kuidas neid saab murda ning kuidas paroole turvaliselt kasutada. Lisaks uuritakse erinevaid krüptograafilisi räsifunktsioone.

Üks esimestest ülesannetest on muuta ette antud programmi paroolide otsimiseks ja salvestada leitud paroolid TXT-faili. Ülesanne on esitatud joonisel 11.



Joonis 11. Praktikumi 2 ülesanded 1a ja 1b [9].

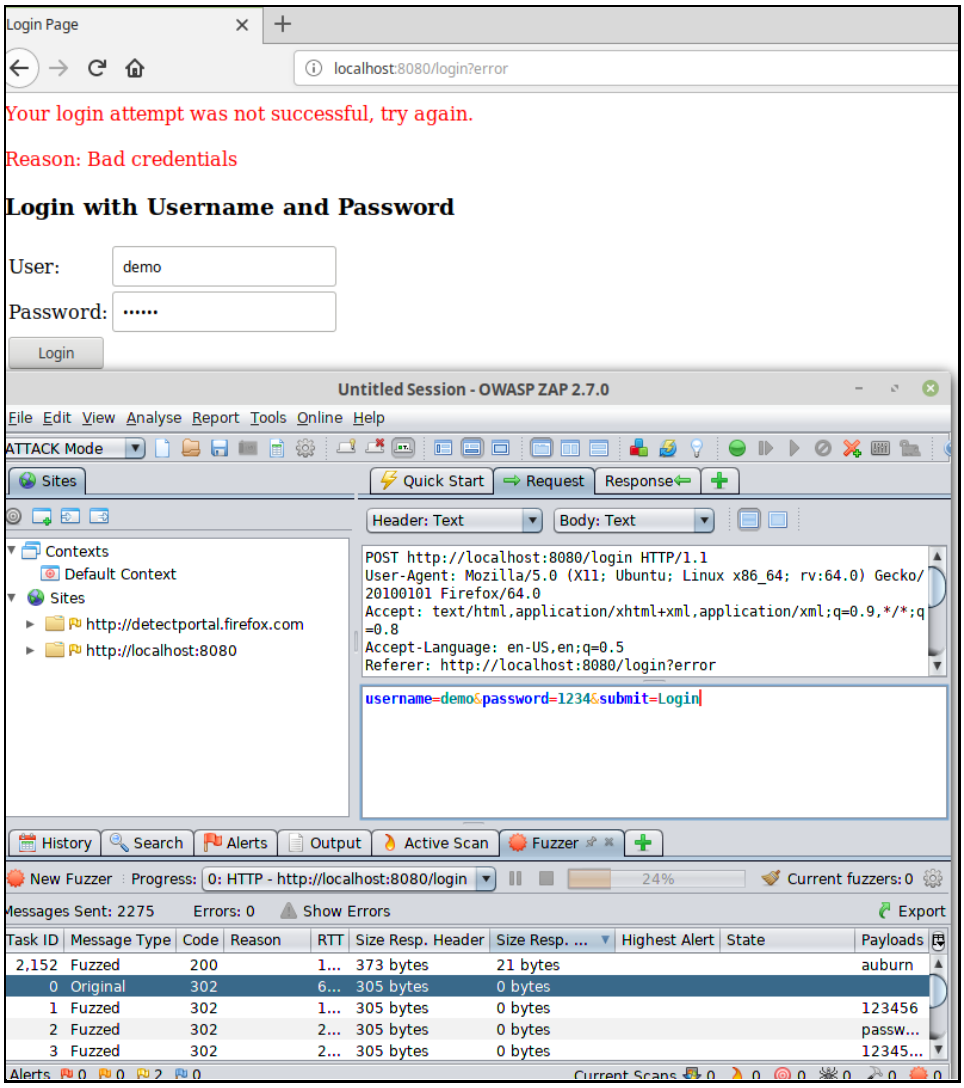
Järgmises ülesannes antakse üliõpilastele kaks tekstifaili, mis koosnevad kasutajakontode andmetest, näiteks nende täisnimedest, DES ja MD5 algoritmi abil krüpteeritud paroolidest ning muudest atribuutidest. Faili näide võib näha joonisel 12. Krüpteeritud paroolide murdmiseks üliõpilased peavad kasutama programmi John the Ripper ja eestikeelne sõnastik, et laiendada programmi võimalusi paroolide murdmisel. John the Ripper on programm, mis on mõeldud paroolide häkkimiseks ning võib kasutada sõnastikke, et võrrelda räsi miljonite või isegi miljardite lekkinud paroolidega [21]. Pärast edukat paroolide murdmist üliõpilased peavad kirjutama varem loodud TXT-failisse vähemalt kolm eestikeelset DES ja/või MD5 murtud parooli.

```
heli:ZPCJS6dj8AqSU:1010:1000:Helen Linnas:/home/heli:/bin/bash
olga_an:ryvdywQ.GBJy6:1011:1000:Olga Andrejev:/home/olga_an:/bin/bash
nurri:Q1fql3OURLJyw:1012:1000:Nurija Epel:/home/nurri:/bin/bash
lydra:QWJOrKh8zimmU:1013:1000:Lidia Raidmets:/home/lydra:/bin/bash
service:FWbN9RNnnHwyQ:1014:1000:Sergei Viik:/home/service:/bin/bash
test1:kD/hJ5fmU3sws:1015:1000:Peep Tulviste:/home/test1:/bin/bash
```

Joonis 12. DES abil krüpteeritud kasutajakontode andmed paroolifailis [9].

Kolmandas ülesandes peavad üliõpilased murdma parooli ja kasutama seda sisselogimiseks veebilehele, mis töötab lokaalses veebiserveris (ingl *local web server*) [9]. Paroolide murdmine toimub OWASP ZAP tööriista abil [9]. OWASP ZAP on veebirakenduste turvalisuse testimise tööriist, mis pakub automaatse ja käsitsi skaneerimise funktsioone [22]. Üliõpilased kasutavad seda tööriista ründe läbiviimiseks, kasutades nimekirja 10 000 kõige populaarsematest paroolidest [9]. Joonisel 13 on näidatud programmi kasutamine. Eesmärk on leida parool, mis vastab üliõpilase identifikaatorile, ja edukalt veebilehele sisse logida. Õpetajal on fail, mis sisaldab kõiki paroole, vastavalt iga kasutaja

ID-le, mis võimaldab kontrollida üliõpilase vastust. Oma parooli üliõpilane salvestab sama TXT-failisse kuhu lisas vastused esimese ja teise ülesannete jaoks.



Joonis 13 . Tööriista OWASP ZAP kasutamine kolmanda ülesande raames [9].

Pärast ülesannete täitmist üliõpilased esitavad Moodle'sse tekstifaili formaadis TXT, kus esimesel real on nende identifikaator ja järgmistel ridadel vastused. Nii õpetaja saab alla laadida kõigi õpilaste failid, et käivitada skript automaatseks ülesannete kontrollimiseks.

Allpool joonisel 14 on esitatud näide tekstifailist, mis sisaldab õigeid vastuseid kõikidele ülesannetele. Ülesannete 1a ja 1b maksimaalne punktide arv on 1, teise ülesande on 0,5 ja kolmanda ülesande on 2.

```
Minu id on 9

Ulesanne 1a ja 1b:
siin
vastused
õiged
ka
on
jälle

Ulesanne 2:
mustikas, tee, kass
Ulesanne 3:
carrot
```

Joonis 14. Esitatud üliõpilase fail, mis sisaldab õigeid vastused

Teise praktikumi kontrollimiseks kasutab skript alguses sama loogikat nagu esimese praktikumi skript. Mõlemad skriptid alustavad tööd tsükliga, mis läbib iga faili määratud kaustas. Võrreldes esimese praktikumi skriptiga, mis ühendas kõik failid ja lisas ühe puuduva parameetri faili üliõpilaste tulemustega, teise praktikumi skript loob faili sama parameetritega ning kontrollib üliõpilaste vastuseid, arvutades nende hindeid.

Esimesest reast failis eraldatakse kasutaja identifikaator *$userid* regulaaravaldisega *"/\d+/"*, kus *"\d"* vastab mis tahes numbrile ja *"+"* määrab, et tuleb leida üks või rohkem numbrit järjest. Iga faili rida jagatakse eraldi sõnadeks funktsiooni *preg_split("/[\s,]+/",$line)* abil, kus *[\s,]* tähendab tühikut *(\s)* või koma *(,)*, ja *"+"* näitab, et eraldajad võivad korduda üks või rohkem korda järjest. Eraldatud sõnad lisatakse massiivi *$unique_answers*, et õpilaste vastuseid saaks võrrelda etteantud õigete vastustega. Kui õige vastus sisaldub selles massiivis, siis suureneb õigete vastuste loendur, ülesannete 1a ja 1b puhul *$found_count1*, ülesande 2 puhul *$found_count2* (joonis 15).

```
    $correct_answers1 = ['siin', 'on', 'õiged', 'vastused', 'jälle', 'ka'];
    $correct_answers2 = ['puu', 'mustikas', 'tee', 'kohv', 'kass', 'koer',
'raamat', 'kruus', 'ilm'];
    $found_count1 = 0;
    $found_count2 = 0;
    foreach ($correct_answers1 as $correct_answer) {
        if (in_array($correct_answer, $unique_answers)) {
            $found_count1++;
        }
    }
    foreach ($correct_answers2 as $correct_answer) {
        if (in_array($correct_answer, $unique_answers)) {
            $found_count2++;
        }
    }
```

Joonis 15. Koodiblokk skriptist õigete vastuste loendamiseks esimese ja teise ülesande jaoks.

Kõikide kolmanda ülesande üliõpilaste paroolid loetakse failist “passwords.txt” ja salvestatakse assotsiatiivsesse massiivi $passwords, kus võtmeteks on üliõpilaste identifikaatorid ja väärtusteks nende vastavad paroolid. Näide faili “passwords.txt” sisust on esitatud joonisel 16. Kontroll algab sellega, kas massiivis $passwords on element võtmega, mis võrdub üliõpilase identifikaatoriga. Seejärel otsib funktsioon *in_array* üliõpilase parooli massiivist *$unique_answers*. Kui parool leitakse, lisatakse üliõpilase 2 punkti.

```
13 irish
9  carrot
8 shasta
7  shaved
6  foot
10  ferret
1  finger
2  whiskey
4 shui
3  rainyday
…
```

Joonisel 16. Näide “passwords.txt” faili sisust.

Skripti viimane samm on kogu tulemuste arvutamine ja saadud hinne salvestatakse muutujasse *$total_grade*. Pärast skripti töö lõpetamist luuakse automaatselt fail nimega “resultPraktikum2.csv” üliõpilaste tulemustega. Joonisel 17 on fail, mis on genereeritud üliõpilase näidistekstifaili kontrollimise põhjal.

	A	B	C
1	userid	assignName	grade
2	9	Praktikum 2	3.5
3	8	Praktikum 2	1
4	7	Praktikum 2	2.5
5	6	Praktikum 2	0.5
6	23	Praktikum 2	2

Joonis 17. Näide “resultPraktikum2.csv” faili sisust.

4.3 Hinnete automaatse ülekandmise põhiskripti arendamine

Enne hinnete automaatse ülekandmise skripti arendamist Moodle’i platvormile on oluline seada eesmärgid ja põhiskripti funktsionaalsed nõuded. Enne skripti kirjutamist on samuti oluline luua tõend ja seadistada veebiteenused.

4.3.1 Turvalisuse tõendi loomine ja veebiteenuste seadistamine Moodle’is.

Enne Moodle’i API kasutamist hinnete automaatseks ülekandmiseks tuleb seadistada veebiteenused ja luua tõend, mida kasutatakse API-le päringute autentimiseks. Veebiteenuste seadistamiseks tuleb kõigepealt aktiveerida vajalik protokoll, näiteks SOAP, REST, XMLRPC või muu. Käesolevas töös valiti REST protokoll, kuna see tagab kasutamise lihtsuse, seda toetavad paljud programmeerimiskeeled ning see võimaldab andmeid vahetada mugavas JSON vormingus [23].

Järgnevalt loodi unikaalne välisseade (ingl *external custom service*), kuhu lisati kolm vajalikku funktsiooni: *core_course_get_contents*, *core_grades_update_grades* ja *core_enrol_get_enrolled_users*. Need kolm põhifunktsiooni mängivad võtmerolli automatiseeritud andmete ülekandmisel. Järgnevalt kirjeldatakse nende definitsioone, põhinedes veebiteenuste API dokumentatsioonile[8]. Esimene neist on seotud kursuse info kogumisega kasutades *core_course_get_contents* funktsiooni. Selle funktsiooni abil saab välja tõmmata nimekirja kursuse töödest ja nende omadustest, sealhulgas nimetustest ja

[8] https://docs.moodle.org/dev/Web_service_API_functions

identifikaatoritest. Teise funktsiooni *core_grades_update_grades* abil hindeid saab lisada ja värskendada. Kolmas funktsioon *core_enrol_get_enrolled_users* on lisatud kontrollimiseks, kas kasutajad on registreeritud sellel kursusel, kuhu lisatakse hindeid või mitte. Kontroll on vajalik selleks, et vältida vigu, mis on seotud valesti määratud üliõpilaste identifikaatoritega nende failis. Kui üliõpilane ei ole kursusele registreeritud, siis skript ei vii läbi tema hinde ülekandmist ning õppejõul on võimalus üliõpilase tööd käsitsi kontrollida. Vastavalt Moodle'i dokumentatsioonile, mis on kättesaadav ainult administraatori vaates, nende funktsioonide skriptis väljakutsumiseks on vaja määrata parameetrid, mis on välja toodud tabelites 1,2 ja 3.

Tabel 1. Funktsiooni *core_course_get_contents* parameetrid.

Parameeter	**Kirjeldus**
wstoken	Moodle'i API juurdepääsu tõend
moodlewsrestformat	Andmete edastamise formaat
wsfunction	Kutsumiseks kasutatava funktsiooni nimi
courseid	Kursuse identifikaator, mille sisu tuleb hankida

Tabel 2. Funktsiooni *core_grades_update_grades* parameetrid.

Parameeter	**Kirjeldus**
wstoken	Moodle'i API juurdepääsu tõend
moodlewsrestformat	Andmete edastamise formaat
wsfunction	Kutsumiseks kasutatava funktsiooni nimi
source	Allikas, kust hindeid võetakse
courseid	Kursuse identifikaator, milles hindeid uuendatakse
component	Ülesande tüüp (nt ülesanne, test, foorum)
activityid	Kursuse tegevuse ehk ülesadne identifikaator
itemnumber	Tegevuse poolt edastatud mitmikhindade seerias oleva üksiku hinne
grades[0][studentid]	Õpilase identifikaator, kellele hinne lisatakse
grades[0][grade]	Hinne

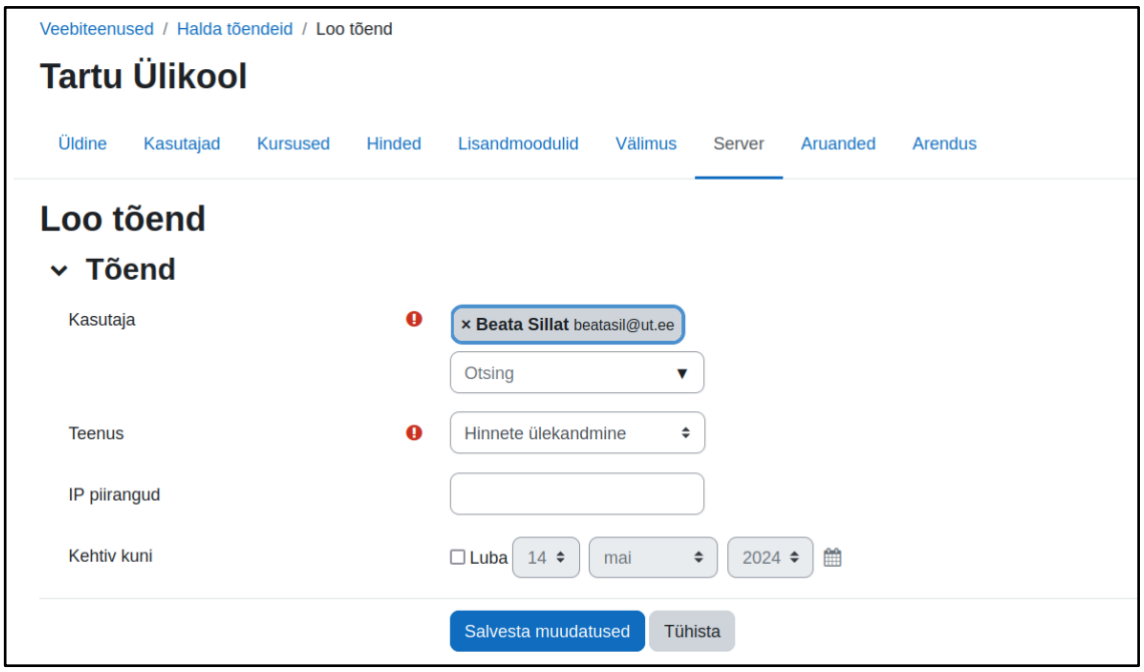
Tabel 3. Funktsiooni *core_enrol_get_enrolled_users* parameetrid.

Parameeter	**Kirjeldus**
wstoken	Moodle'i API juurdepääsu tõend
moodlewsrestformat	Andmete edastamise formaat
wsfunction	Kutsumiseks kasutatava funktsiooni nimi
courseid	Kursuse identifikaator, kuhu kasutajad on registreeritud

Kasutades skriptis funktisooni *core_course_get_contents* samuti saab määrata ülesande tüüpe. Allpool on toodud mõnede tüüpide definitsioonid:

- Ülesanne (ingl *Assignment*) – võimaldab õpetajatel hinnata ja anda kommentaare üles laetud failidele [24].
- Test (ingl *Quiz*) – võimaldab õpetajatel luua teste, mis võivad olla automaatselt hinnatud ning pakkuda tagasisidet ja/või kohe näidata õiget vastust [24]
- Foorum (ingl *Forum*) – võimaldab osalejatel läbi viia asünkroonseid diskussioone[24]

Kasutades neid kahte funktsiooni, saab luua mehhanismi, mis tagab koostöö välja töötatud lahenduse ja Moodle'i vahel, järgides samal ajal andmete turvalisuse ja konfidentsiaalsuse standardeid. Seejärel loodi turvalisuse tõend, tuntud ka kui wstoken, mis tagab turvalise juurdepääsu Moodle'ile välissüsteemidest [25]. Tõendi saab luua kasutaja administraatori õigustega Moodle'i halduspaneelil kaudu või isik kellel on sellele võimalusele juurdepääs (joonis 18).



Joonis 18. Tõendi loomine administraatori vaates.

Administraatoril on võimalus määrata, millised kasutajad võivad seda konkreetset tõendit kasutada, tagades, et juurdepääs on vaid piiratud arvul inimestel. Lisaks sellele saab administraator seostada tõendi kindla veebiteenusega, mis määrab, milliseid operatsioone on lubatud tõendi abil täita. Moodle võimaldab administraatoritel seadistada tõendeid arvestades kontekstipiiranguid [26]. Moodle'i dokumentatsiooni põhjal on võimalik tõendi seadistada nii, et see toimiks ainult kindlates tingimustes, näiteks ainult teatud kursustele või tegevustele juurdepääsuks [26]. See annab võimaluse luua tõend, mis omab juurdepääsu ainult vajalikele osadele, vältides volitamata juurdepääsu Moodle'i süsteemi teistele osadele. Tõendid võivad olla nii staatilised ehk lõpmatult kestev kui ka ajutised. Ajutiste tõendite puhul saab valida kehtivusaja, mille jooksul need on aktiivsed. Samuti on võimalik tõendi kasutust piirata IP-aadressi järgi, mis tähendab, et tõend toimib ainult ette määratud IP-aadressidelt.

4.3.2 Hinnete automaatse laadimise skripti eesmärgid

Pärast Moodle'i platvormi ettevalmistamist, praktikumide kontrollimise skriptide kirjutamist ja kõigi vajalike komponentide installimist alustati õpilaste hinnete automaatse ülekande skripti arendamist. Enne skripti loomist määrati järgmised eesmärgid, mida programm peaks täitma:

1. Igal õpetajal on valmis CSV-laiendiga fail, mis sisaldab vajalikke andmeid hinnete ülekandmiseks. Skript peaks avama faili ja lugema andmed edasiseks töötlemiseks.
2. Pärast faili avamist toimub andmete töötlemine, mille käigus määratakse iga veeru sisu. Seejärel salvestatakse andmed vastavatesse muutujatesse edasiseks kasutamiseks.
3. Turvalise andmeedastuse tagamiseks skripti ja Moodle'i süsteemi vahel kasutatakse eelnevalt genereeritud tõendit, mis annab skriptil juurdepääsu Moodle'i veebiteenustele ning hinnete ülekande toimingute tegemist minimaalse volitamata juurdepääsu riskiga.
4. Moodle'i veebiteenustega suhtlemiseks loob skript REST päringud. Skript koostab API Moodle'i kutsumiseks URL-aadressi, määrab HTTP-meetodi POST ja lisab vajalikud päringuparameetrid.

4.3.3 Hinnete automaatse laadimise põhiskripti analüüs

Skripti alguses määratakse muutujad, mida kasutatakse põhifunktsioonide kutsumisel. Sinna kuuluvad ligipääsu tõend, Moodle'i veebisaidi URL, andmeedastuse formaat ja tee CSV-failile, kus on kõik vajalikud andmed hinnete ülekandmiseks, näiteks failile, mis on loodud automaatselt pärast Praktikumi 1 või Praktikumi 2 kontrollimist. Andmed CSV-failist salvestatakse massiivi *$csvData* funktsiooni *file* abil, mis loeb faili rida-realt. Funktsiooni *array_map* koos *str_getcsv* kasutatakse iga failirea teisendamiseks komadega eraldatud väärtuste massiiviks. CSV-faili veergude päiseid eemaldatakse andmemassiivist funktsiooni *array_shift* abil.

Kuna peaaegu kõik parameetrid ühest peamisest funktsioonist *core_grades_update_grades* on ette teada, välja arvatud *component* ja *activityid*, loodi kaks lisa funktsiooni: *findAssignmentActivityId* ja *findComponentByActivityId*. Esimene funktsioon *findAssignmentActivityId* otsib kursuse sisust mooduleid ehk töid, mille nimi vastab CSV failis teise veerus olevale väärtusele. Selles funktsioonis on kaks tsüklit kursuse sisu kontrollimiseks. Esimene tsükkel läbib kursuse jagude massiivi (*$contents*), igaüks neist sisaldab mooduleid (*$module*). Teises tsüklis kontrollitakse mooduli nimi (*name*). Kui mooduli nimi kattub otsitava testi nimega, siis funktsioon tagastab selle mooduli identifikaatori (*activityid*). Joonisel 19 on esitatud funktsiooni *findAssignmentActivityId* kood.

```
function findAssignmentActivityId($assignmentName, $contents) {
    if (!empty($contents)) {
        foreach ($contents as $section) {
            foreach ($section['modules'] as $module) {
                if ($module['name'] === $assignmentName) {
                    return $module['id'];
                }
            }
        }
    }
    return null;
}
```

Joonis 19. Koodiblokk funktsiooniga *findAssignmentActivityId*.

Teine funktsioon *findComponentByActivityId* on vajalik ülesande tüübi määramiseks, kasutades varasemalt leitud tegevuse identifikaatorit. See funktsioon läbib ka kursuse sisu, analüüsides iga jaotist ja sellele alluvaid mooduleid. Funktsioon otsib moodulit, mille identifikaator (id) vastab activityId-le ja tagastab mooduli tüübi (modname), näiteks test (ingl *quiz*) või ülesanne (ingl *assignment*). Joonisel 20 on esitatud funktsiooni *findComponentByActivityId* kood.

```
function findComponentByActivityId($activityId, $contents) {
    if (!empty($contents)) {
        foreach ($contents as $section) {
            foreach ($section['modules'] as $module) {
                if ($module['id'] == $activityId) {
                    return $module['modname'];
                }
            }
        }
    }
    return '';
}
```

Joonis 20. Koodiblokk funktsiooniga *findComponentByActivityId*.

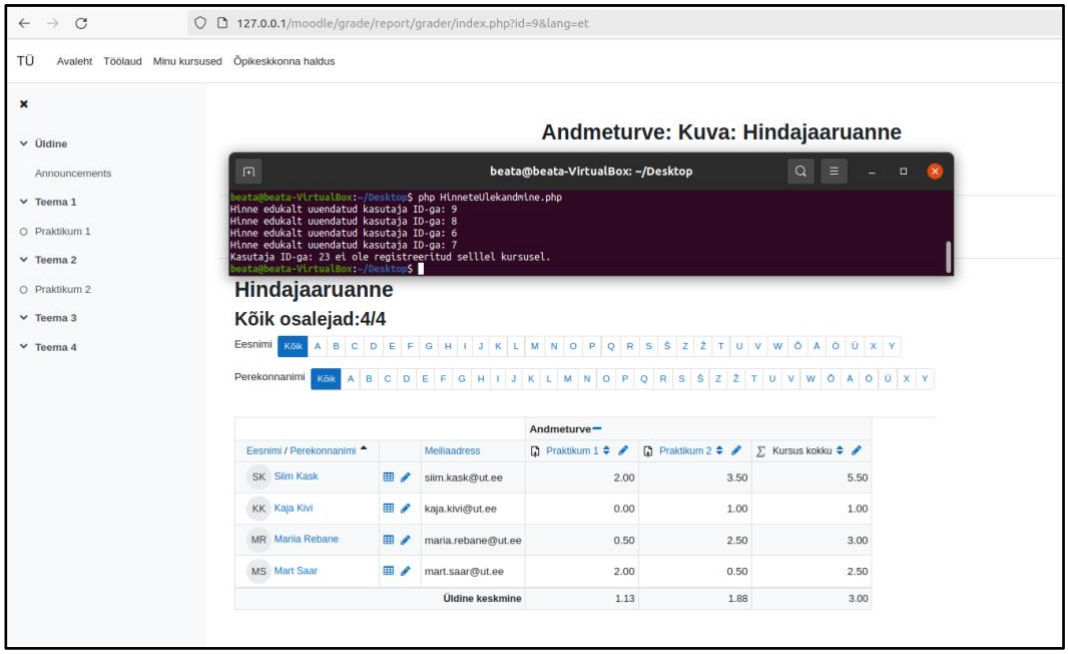
Skripti põhitsükkel käib läbi iga *$csvData* massiivi rea ja töötleb andmeid. Iga rea jaoks koostatakse Moodle'ile päring kursuse sisu saamiseks, kasutades *core_course_get_contents* funktsiooni, mis seejärel dekodeeritakse saadud vastus JSON-st assotsiatiivseks massiiviks *$contents*, mille osa on näidatud joonisel 21 *json_decode* abil. Terve JSON massiiv, mis sisaldab andmeid Moodle'i kursuse kohta, on lisatud Lisa 2 osas (Lisa II).

Joonis 21: Osa Moodle API poolt tagastatud assotsiatiivne massiiv.

Pärast ülesande tüübi leidmist, luuakse päring API funktsioonile *core_grades_update_grades* hinnete ülekandmiseks, kasutades POST-meetodit, mille päring näeb välja järgmine:

```
http://127.0.0.1/moodle/webservice/rest/server.php?wstoken=aa361a3ea3a50b6b222cd9796c220582&wsfunction=core_grades_update_grades&source=some_source&courseid=9&component=assign&activityid=23&itemnumber=0&grades[0][studentid]=9&grades[0][grade]=3.5
```

Päring saadetakse ja saadud vastus töödeldakse. Pärast päringu tulemuste saamist skript kontrollib, kas hinnete ülekandmine Moodle'i oli edukas. Lisaks jälgib skript võimalikke vigu ja teavitab kasutajat nende esinemisest. Allpool joonisel 22 on näide sellest, kuidas hinded on pärast skripti edukat käivitamist lisatud.

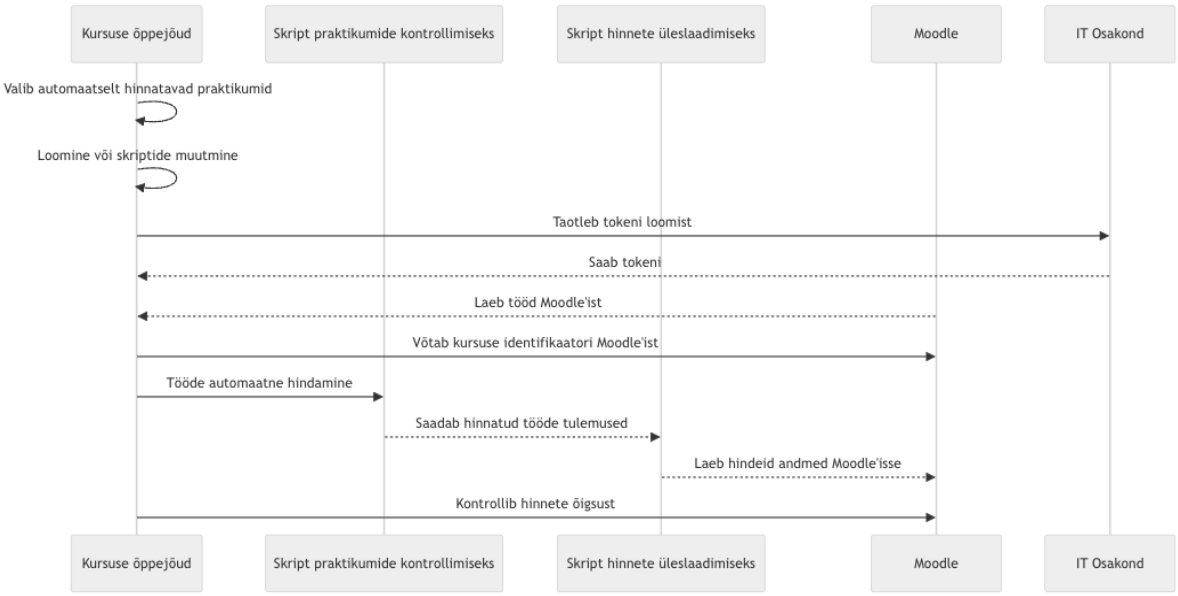


Joonis 22: Ülekantud hinded Moodle'i platvormil.

Skripti käivitamisel kasutati alguses "resultPraktikum1.csv" faili (joonis 10) ja seejärel "resultPraktikum2.csv" faili (joonis 17). Ühes neist failidest oli spetsiaalselt lisatud vale kasutaja identifikaator, et demonstreerida, kuidas terminalis kuvatakse vastav teade.

4.4 Skriptide koostöö visualiseerimine hariduskeskkonnas

Arendatud skriptide koostöö visualiseerimiseks ja potentsiaalse rakendamise demonstratsiooniks hariduskeskkonnas on allpool esitatud diagramm (joonis 23). Diagrammil on kujutatud suhtlemine erinevate protsessi osalejate vahel, samuti skriptide integreerimine ja kasutamine reaalses Tartu Ülikooli õpikeskkonna näitel. Diagramm kirjeldab põhjalikult sammude järjestust alates praktikumide valikust õpetaja poolt automaatseks hindamiseks kuni tulemuste lõpliku üleskandmeseni Moodle'i süsteemisse.



Joonis 23. Skriptide koostöö visualiseerimine hariduskeskkonnas.

1. Protsess algab sellega, et õppejõud valib, millised praktikumid saab automaatselt hinnata.
2. Sõltuvalt sellest valikust kasutatakse kas juba välja töötatud skripte või luuakse uusi.
3. Õppejõud pöördub ülikooli IT-osakonna poole, et saada API ühenduseks vajalikku ligipääsu tõendit, mis võimaldab hinnete ülekandmise skriptil Moodle'i süsteemiga suhelda.
4. Seejärel õppejõud võtab Moodle'i süsteemist kursuse identifikaatori, mis on vajalik skripti edasiseks tööks.
5. Kui kõik vajalikud juurdepääsud ja andmed on saadud, käivitatakse ülesannete automaatne hindamise skript ja saadud tulemused edastatakse hinnete ülekandmise skriptile, mis laadib need Moodle süsteemisse.
6. Protsessi lõpus teeb õppejõud pistelise hinnete kontrolli, et veenduda, et kõik hinded on edukalt üle kantud ja tehnilisi vigu ei esinenud.

4.5 Edasiarendusvõimalused

Välja arendatud süsteem Moodle'isse hinnete automaatseks ülekandmiseks veebiteenuste ja REST API kaudu ning praktiliste ülesannete automaatseks kontrollimiseks on võimalik veel täiendada ja mitmes suunas edasi arendada. Tulevikus saab loodud näidet laiendada, lisada uusi funktsioone ja adapteerida erinevate teiste kursuste jaoks. Olulise järgmise sammuna

tuleb käesoleva töö näitel veenda Tartu Ülikooli IT-osakonda loodud automaathindamise turvalisuses ja funkstionaalsuses. Juhul kui Tartu Ülikooli IT-osakond nõustub ligipääsu andmisega, siis tulevikus samuti süsteeme võiks testida Tartu Ülikooli Moodle'i keskkonnas.

Üks võimalus funktsionaalsuse laiendamiseks on kontrollisüsteemi loomine ülejäänud praktiliste ülesannete jaoks. Enamike ülesannete raames üliõpilased esitavad ekraanipilte tekstifailide asemel, seega võib kaaluda ka ekraanipiltide kontrollimise süsteemi loomis, kasutades pilditöötluse vahendeid. Ekraanipiltide automaathindamise arendamine on käesoleva töö raames liiga mahukas ja see nõuaks eraldiseisvat käsitlust.

Lisaks võiks arendada lihtne liides, mis võimaldaks skripti kasutada hinnete ülekandmiseks Moodle'isse ilma vajaduseta seda käivitada terminali kaudu. Programm võiks olla arusaadava ja lihtsa disainiga, mis teeb skripti kasutamise mugavamaks. See on eriti vajalik ennekõike ainetes, kus õppejõud ei oma kõrgeid tehnilisi oskusi ega ole harjunud töötama käsureaga, mis muudab skripti kasutamise neile kättesaadavamaks ja arusaadavamaks.

Kokkuvõtte

Bakalaureusetöö eesmärk oli luua lahendused, mis aitaksid optimeerida üliõpilaste tööde kontrollimise ja hinnete ülekandmise protsessi Moodle'i platvormile, kasutades näitena kursust "Andmeturve (LTAT.06.002)". Selleks loodi PHP-keeles kirjutatud skriptid, mis on mõeldud antud kursuse esimeste kahe praktilise töö kontrollimiseks ning samuti skript automaatseks hinnete ülekandmiseks Moodle'i platvormile.

Käesolevas töös paigaldati Moodle'i keskkond lokaalsele serverile, kasutades Apache veebiserverit ja MySQL andmebaasi, mis võimaldas luua turvalise keskkonna arendamiseks ja testimiseks. Moodle'i platvormil loodi demokursus, kuhu lisati testkasutajad, võimaldades süsteemi reaalsete kasutustingimuste imiteerimist.

Üks oluline tehnoloogia, mida kasutatakse hinnete ülekandmise skripti ja Moodle'i platvormi vahelise suhtluse jaoks, on Moodle'i veebiteenused. Need veebiteenused võimaldavad juurdepääsu Moodle'i funktsioonidele ja andmetele väljastpoolt Moodle'i keskkonda, kasutades REST API-d, mis võimaldab Moodle'iga suhelda HTTP päringute kaudu. Need päringud sisaldavad HTTP POST-meetodeid, mida skript kasutab hinnete Moodle'isee laadimiseks ja värskendamiseks. Arendatud lahendused ei kasuta üliõpilaste isikuandmeid, vaid nende nime asemel kasutatakse Moodle'i platvormile registreerumisel automaatselt genereeritud identifikaatoreid, mis aitab tugevdada andmekaitset GDPR nõuetele vastavalt. Hinnete automaatse ülekande skripti käivitamine testimiseks paigaldatud Moodle keskkonnas saavutas oma eesmärgi edukalt, kuna kontrollitud tööde tulemused kuvati demokursusel korrektselt.

Lisaks skriptide arendamisele viidi läbi analüüs olemasolevate hinnete ülekandmise meetodite kohta Tartu Ülikoolis. Analüüsi käigus uuriti erinevaid lähenemisviise, mida kasutatakse ülikooli kursustel välissüsteemide integreerimiseks haridusplatvormi Moodle'iga. Samuti vaadeldi töös välja arendatud lahenduste edasise arenduse võimalusi.

Kuigi käesolevas töös ei edastatud tulemusi Moodle.ut.ee keskkonda Andmeturve kursusele, on tehtud töö kasulik suure osalejate arvuga aine õppejõududele, et nad saaksid hinnata automaathindamise realiseerimise teostatavust, tehnilisi lahendusi ja keerukust võrreldes klassikalise inimese poolt sooritatud hindamisega.

Viidatud kirjandus

[1] Tartu Ülikooli Moodle'i juhend. https://teaduskool.ut.ee/et/tartu-ulikooli-moodlei-juhend (04.12.2023)

[2] Al-Ajlan A., Zedan H. Why Moodle. *12th IEEE International Workshop on Future Trends of Distributed Computing Systems*, 2008, pp. 58–64. https://doi.org/10.1109/FTDCS.2008.22

[3] e-Teatmik. IT ja sidetehnika seletav sõnaraamat. http://www.vallaste.ee/

[4] Cybernetica AS. Andmekaitse ja infoturbe leksikon. https://akit.cyber.ee/

[5] Moodle Dokumentatsioon. LTI and Moodle, 2023. https://docs.moodle.org/404/en/LTI_and_Moodle (16.04.2024).

[6] Nokhbeh Zaeem R., Barber K. Suzanne. The Effect of the GDPR on Privacy Policies: Recent Progress and Future Promise. *ACM Transactions on Management Information Systems*, 2020, 12(1), pp. 1–20. https://doi.org/10.1145/3389685

[7] Li H., Yu L., He W. The Impact of GDPR on Global Technology Development. *Journal of Global Information Technology Management*, 22(1), 2019, pp. 1–6. https://doi.org/10.1080/1097198X.2019.1569186

[8] Gruschka N., Mavroeidis V., Vishi K., Jensen M. Privacy Issues and Data Protection in Big Data: A Case Study Analysis under GDPR. *2018 IEEE International Conference on Big Data*, 2018, pp. 5027–5033. https://doi.org/10.1109/BigData.2018.8622621

[9] Tartu Ülikool. Arvutiteaduse instituut. Andmeturve (LTAT.06.002) õppeaastal 2023/24. https://courses.cs.ut.ee/2024/turve/spring (10.02.2023)

[10] VirtualBox. https://www.virtualbox.org/ (24.03.2024)

[11] Moodle Dokumentatsioon. About Moodle, 2022. https://docs.moodle.org/401/en/About_Moodle (05.03.2024).

[12] Buchner A. Moodle 4 Administration: An administrator's guide to configuring, securing, customizing, and extending Moodle. Birmingham: Packt Publishing Ltd. 2022. https://books.google.ee/books?id=p1OZEAAAQBAJ

[13] Dondorf T. Learning analytics for Moodle: facilitating the adoption of data privacy friendly learning analytics in higher education. *RWTH Aachen University*, 2022, pp. 1-196. https://doi.org/10.18154/RWTH-2022-04002

[14] Moodle Dokumentatsioon. External Services, 2024. https://moodledev.io/docs/4.4/apis/subsystems/external (15.04.2024)

[15] Moodle Dokumentatsioon. LTI External tools, 2024. https://docs.moodle.org/404/en/LTI_External_tools (15.04.2024)

[16] Metoodiline juhend õpetajale gümnaasiumi informaatika uute valikkursuste õpetamiseks, 25lk. https://media.voog.com/0000/0034/3577/files/Metoodiline%20juhend%20%C3%B5petajale%20g%C3%BCmnaasiumi%20informaatika%20uute%20valikkursuste%20%C3%B5petamiseks.docx.pdf (08.05.2024)

[17] MathWorks. MATLAB Grader. https://uk.mathworks.com/help/matlabgrader/ (08.05.2024)

[18] Panopto. How to Add a Quiz to a Video, 2023. https://support.panopto.com/s/article/How-to-Add-a-Quiz-to-a-Video (08.05.2024)

[19] Moodle Dokumentatsioon. PHP, 2024. https://docs.moodle.org/403/en/PHP (21.03.2024)

[20] Moodle Dokumentatsioon. Moodle 4.1, 2023. https://moodledev.io/general/releases/4.1 (05.03.2024)

[21] Marchetti K., Bodily P. John the Ripper: An Examination and Analysis of the Popular Hash Cracking Algorithm. *2022 Intermountain Engineering, Technology and Computing (IETC)*, 2022, pp. 1-6, doi: 10.1109/IETC54973.2022.9796671.

[22] Zap. https://www.zaproxy.org/getting-started/ (25.03.2024)

[23] Barbaglia G., Murzilli S., Cudini S. Definition of REST web services with JSON schema. *Software Practice and Experience*, 2017, 47: 907–920, doi: 10.1002/spe.2466.

[24] Moodle Dokumentatsioon. Activities, 2023. https://docs.moodle.org/403/en/Activities (29.04.2024)

[25] Moodle Dokumentatsioon. Security Keys, 2024.
https://docs.moodle.org/403/en/Security_keys (03.05.2024)

[26] Moodle Dukemntatsioon. External services security, 2021.
https://docs.moodle.org/dev/External_services_security (03.05.2024)

OpenAI (2024). ChatGPT (3.5): https://chat.openai.com.

Lisad

I. Lähtekoodid GitHub repositooriumis

Käesoleva töös arendatud skriptid, sealhulgas esimese kahe praktikumi automaatseks kontrollimiseks ja hinnete automaatseks ülekandmiseks Moodle'isse, on kättesaadavad aadressilt: https://github.com/BeataSl/automatic-grading-uploading

II. Moodle'I kursuse struktuur JSON-vormingus

core_course_get_contents funktsiooni päringu tulemus:


```
[
    {
        "id": 29,
        "name": "Üldine",
        "visible": 1,
        "summary": "",
        "summaryformat": 1,
        "section": 0,
        "hiddenbynumsections": 0,
        "uservisible": true,
        "modules": [
            {
                "id": 21,
                "url": "http://127.0.0.1/moodle/mod/forum/view.php?id=21",
                "name": "Announcements",
                "instance": 8,
                "contextid": 51,
                "visible": 1,
                "uservisible": true,
                "visibleoncoursepage": 1,
                "modicon":
"http://127.0.0.1/moodle/theme/image.php/boost/forum/1700943456/monologo?filteri
con=1",
                "modname": "forum",
                "modplural": "Foorumid",
                "availability": null,
                "indent": 0,
                "onclick": "",
                "afterlink": null,
                "customdata": "\"\"",
                "noviewlink": false,
                "completion": 0,
                "downloadcontent": 1,
                "dates": []
            }
        ]
    },
    {
        "id": 30,
        "name": "Teema 1",
        "visible": 1,
        "summary": "",
        "summaryformat": 1,
        "section": 1,
        "hiddenbynumsections": 0,
        "uservisible": true,
        "modules": [
            {
                "id": 22,
                "url": "http://127.0.0.1/moodle/mod/assign/view.php?id=22",
                "name": "Praktikum 1",
                "instance": 9,
                "contextid": 52,
                "visible": 1,
                "uservisible": true,
                "visibleoncoursepage": 1,
                "modicon":
"http://127.0.0.1/moodle/theme/image.php/boost/assign/1700943456/monologo?filter
icon=1",
                "modname": "assign",
                "modplural": "Ülesanded",
```

```
                "availability": null,
                "indent": 0,
                "onclick": "",
                "afterlink": null,
                "customdata": "{\"allowsubmissionsfromdate\":\"1715547600\"}",
                "noviewlink": false,
                "completion": 1,
                "completiondata": {
                    "state": 0,
                    "timecompleted": 0,
                    "overrideby": null,
                    "valueused": false,
                    "hascompletion": true,
                    "isautomatic": false,
                    "istrackeduser": true,
                    "uservisible": true,
                    "details": []
                },
                "downloadcontent": 1,
                "dates": [
                    {
                        "label": "Opened:",
                        "timestamp": 1715547600,
                        "dataid": "allowsubmissionsfromdate"
                    }
                ]
            }
        ]
    },
    {
        "id": 31,
        "name": "Teema 2",
        "visible": 1,
        "summary": "",
        "summaryformat": 1,
        "section": 2,
        "hiddenbynumsections": 0,
        "uservisible": true,
        "modules": [
            {
                "id": 23,
                "url": "http://127.0.0.1/moodle/mod/assign/view.php?id=23",
                "name": "Praktikum 2",
                "instance": 10,
                "contextid": 53,
                "visible": 1,
                "uservisible": true,
                "visibleoncoursepage": 1,
                "modicon":
"http://127.0.0.1/moodle/theme/image.php/boost/assign/1700943456/monologo?filter
icon=1",
                "modname": "assign",
                "modplural": "Ülesanded",
                "availability": null,
                "indent": 0,
                "onclick": "",
                "afterlink": null,
                "customdata": "{\"allowsubmissionsfromdate\":\"1715547600\"}",
                "noviewlink": false,
                "completion": 1,
                "completiondata": {
                    "state": 0,
                    "timecompleted": 0,
                    "overrideby": null,
                    "valueused": false,
                    "hascompletion": true,
```

```
                    "isautomatic": false,
                    "istrackeduser": true,
                    "uservisible": true,
                    "details": []
                },
                "downloadcontent": 1,
                "dates": [
                    {
                        "label": "Opened:",
                        "timestamp": 1715547600,
                        "dataid": "allowsubmissionsfromdate"
                    }
                ]
            }
        ]
    },
    {
        "id": 32,
        "name": "Teema 3",
        "visible": 1,
        "summary": "",
        "summaryformat": 1,
        "section": 3,
        "hiddenbynumsections": 0,
        "uservisible": true,
        "modules": []
    },
    {
        "id": 33,
        "name": "Teema 4",
        "visible": 1,
        "summary": "",
        "summaryformat": 1,
        "section": 4,
        "hiddenbynumsections": 0,
        "uservisible": true,
        "modules": []
    }
]
```

III. Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks

Mina, Beata Sillat

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose **“Andmeturve” kursuse näitel automaathindamise realiseerimine ja hinnete automaatne laadimine moodle keskkonda**, mille juhendajad on **Alo Peets**, 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.

Beata Sillat

15.05.2024