TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Maksim Potapov 155241IAPB

MOODLE'I PISTIKPROGRAMMI ARENDAMINE LÕPUTÖÖDE HALDAMISEKS

Bakalaureusetöö

Juhendaja: Ago Luberg
MSc

Tallinn 2019

Autorideklaratsioon

Kinnitan, et koostasin selle lõputöö iseseisvalt ning seda ei ole varem kaitsmisele esitatud. Töös kasutatud kirjandusallikad, olulised seisukohad, teiste autorite tööd ja mujalt võetud andmed on töös viidatud.

Autor: Maksim Potapov

04.04.2019

Annotatsioon

Töö eesmärgiks oli luua Moodle'i õppekeskkonna jaoks ülesannete moodul. Moodul võimaldab tudengitel ja õppejõududel mugavalt ja kiiresti hallata oma lõputööga seotud tegevusi. Pistikprogrammi ülesanne on anda tudengitele mugav lahendus nii sobiliku teema otsimiseks kui ka lõputöö haldamiseks. Süsteem peab võimaldama tudengil pakutud teemadele kandideerida. Õppejõu jaoks peab süsteem andma võimaluse näha ülevaadet kandidaatidest, mugavalt ja kiirelt jälgida tööprotsessi ning anda hinnangut. Pistikprogramm võimaldab lõputöö teemade haldamist, samal ajal on kõik tööga seotud tegevused ja üleslaadimised kursusel näha. Selleks loodi Moodle'isse tegevuse moodul. Moodul võimaldab teemade väljapanemist, otsimist, kandideerimist ning haldamist ja jälgimist. Lisaks kasutati Dockerit, et Moodle'i keskkond paigaldada ja sellel testida oma loodud lahendust lokaalselt. Pistikprogramm luuakse sellisel moel, et sellega võimalik ained.ttu.ee Moodle õppekeskkonnas teostada installimine ja uuendamine. Töö jaguneb analüüsiks, arendusprotsessiks, metoodikaks ja lahenduseks. Analüüs määrab loodud mooduli funktsionaalsust, kirjeldab andmebaasi skeemi ja salvestab kasutajale kõik olulisemad juhiseid. Peatükis arendusprotsessist ja metoodikast räägitakse, milliseid vahendeid on lõputöö realiseerimisel kasutatud.

Lõputöö on koostatud eesti keeles ning töös on 37 lehekülge teksti, 5 peatükki ja 15 joonist.

Abstract

Moodle plugin development for bachelor theses management

The work includes an activity module for the Moodle learning system. The module allows students and faculty to manage their thesis work quickly and easily. The task of the plugin is to provide a convenient way for students to find a suitable topic and to manage their work. The system must allow the student to apply to the proposed topics. For the lecturer, the system must provide an opportunity to see an overview of the candidates, to monitor the work process in a convenient and fast way and to give an assessment. The plugin allows the process of managing the thesis topic, at the same time, all the activity in the course is connected to the thesis. For this purpose, a PHP activity module for Moodle was created. The module includes displaying, searching, applying for, managing and monitoring topics. In addition, Docker was used to install the Moodle environment locally and to test the plugin. It is possible to install the plugin to the ained.ttu.ee learning environment based on Moodle. The work contains four parts: analysis, development process, methodology and solution. The analysis part has defined the required functionality, database schema and describes the user instructions. The development process and methodology part describes what was used for developing the plugin.

The thesis contains 37 pages of text, 5 chapters, 15 figures and is written in Estonian.

Lühendite ja mõistete sõnastik

Git	Versioonihaldussüsteem
Moodle	Õpihaldussüsteem
Repositoorium	Giti salv
PHP	Programmeerimiskeel
Plugin	Tarkvaramoodul, mis laiendab mõne teise tarkvarasüsteemi funktsionaalsust, edaspidi pistikprogramm.
Assignment	Ülesanne Moodle'i keskonnas
Submission	Tudengi ülesande esitus Moodle'i keskkonnas (Assignmenti kohta)

Sisukord

Jooniste loetelu

1 Sissejuhatus

Lõputöö teemade käsitsi lisamine ja muutmine on õppejõududele suure õpilaste arvu korral väga ajamahukas ülesanne. Samuti on ka tudengitele keeruline leida endale sobiv teema, see välja valida ja sellele registreeruda. Lõputöö teemade lisamiseks kasutavad õppejõud täna TTÜ veebi, kuhu teemade nimekirjad lisatakse. Tudeng saab teemade otsimiseks kasutada veebiteksti TTÜ veebilehel. Lisaks on suure tudengi arvu korral õppejõu jaoks väga keerukaks ja ajamahukaks tööks tudengite tegevuste jälgimine. Parem oleks siduda need tegevused kasutusel oleva Moodle'i õppekeskkonnaga nii, et võimaldada kõik lõputööga seotud tegevused teha mugavalt ühes kohas.

1.1 Eesmärk

Antud töö eesmärgiks on luua uus Moodle`i pistikprogramm, mis võimaldaks teostada kõiki lõputöö protsessis ette nähtud ja vajalikke tegevusi. Õppejõud, või siis nende asemele määratud isikud, saaksid sinna lisada vajalikke teemasid ning samas ka jälgida tudengite poolset kandideerimist nendele teemadele. Tudeng omaks lihtsat võimalust valida endale sobiv teema ja õppejõud ning kandideerida. Nii õppejõud kui ka tudeng omaksid võimalusi oma teemasid hallata ja ka näha, milliseid lõputööga seotud ülesandeid (*Assignment*) on vaja täita. Samuti oleks võimalik näha seda, milliseid vastuseid (*Submissions*) on antud. Vajadusel oleks komisjon võimeline hindeid haldama ja muutma.

Töö teiseks eesmärgiks on kirjutada kood sellisel viisil, et pistikprogrammi edasine arendamine oleks lihtne ning hiljem saaks seda publitseerida Moodle'i ametlike laienduste portaalis.

1.2 Metoodika

Üks tähtsamaid eesmärke on teha pistikprogramm nii standardne, et seda oleks võimalik paigaldada ükskõik millisesse Moodle'isse ja tagada lihtne versiooni muutmine.

Mooduli dokumentatsioon peab olema detailne ning lähtekood peab olema puhas kood. Moodul pakitakse installatsiooni arhiivina, mida saab kasutada erinevatesse Moodle'i keskkondadesse paigaldamiseks. Automaatne mooduli uuendamine, mis annab võimaluse uuendada ilma kasutuses oleva lehe tööd häirimata.

1.3 Oodatav tulemus

Oodatav tulemus on, et pistikprogramm oleks lihtne ja mugav kasutada nii tudengitele, õppejõududele ja halduritele. Luua infosüsteem kogu teemade valikuga, kust on kiiresti võimalik otsida ning saada tudengi ja õppejõu seoseid. Süsteem annab võimaluse eksportida kogu info CSV-sse. Seoste tekkimine tudengite ja õpejõudude vahel peab olema lihtsam kui praegune lahendus. Kui kandideerimisavaldus teemale on kinnitatud, tekib eelnimetatud seos automaatselt.

Moodle'i keskkonnas saab kasutada olemasolevaid funktsioone ja mooduleid (*Chat*, *Submission*, *Grade* jms) kogu lõputöö protsessi vältel. Kogu pistikprogrammi peab saama raskusteta edasi arendada ning uusi funktsioone lisada.

2 Analüüs

See peatükk kirjeldab loodud mooduli funktsionaalsust. Lisaks luuakse andmebaasi skeem ja kirjutatakse lahti peamised kasutusjuhud. Analüüsi teoreetilised tulemused on rakendatud töös lahenduse peatükis.

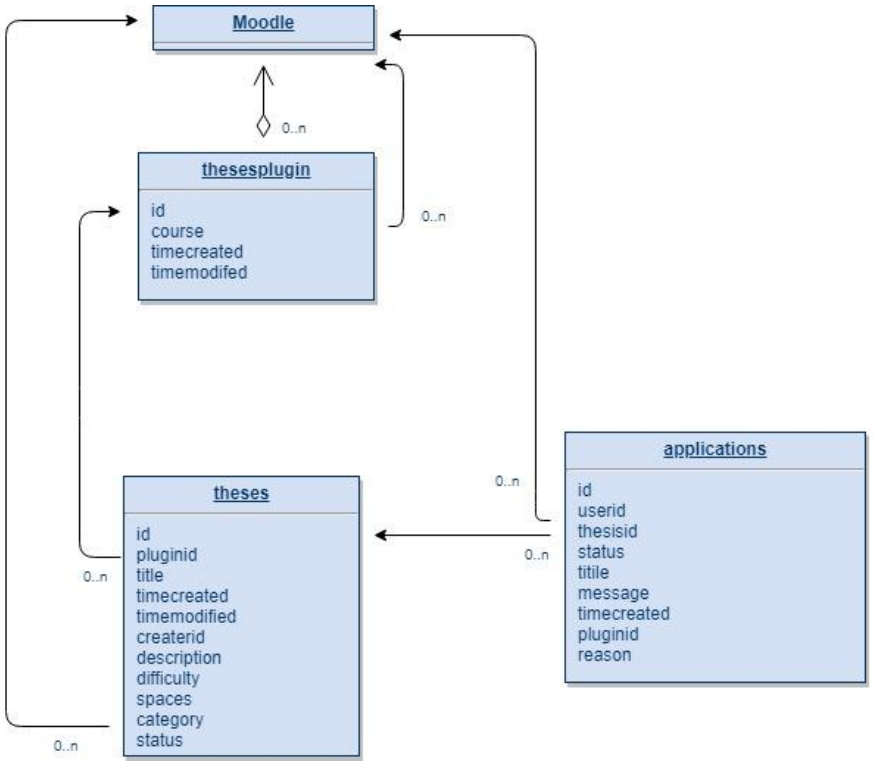
2.1 Tulemuse spetsifikatsioon

Loodud pistikprogramm (järgnevalt Thesesplugin) pakub võimaluse luua Moodle'isse rakenduse, mis annab võimaluse lõputöö teemade lisamiseks ja otsimiseks. Teema lisamisel saab määrata kogu teema informatsiooni, näiteks töö pealkirja, kategooria, raskuse, vabade kohtade arvu ja kirjelduse. Teema kategooriad on Moodle'is olemasolevad *tag*'id ehk sildid, mille järgi saab otsida ning aru saada töö valdkonnast . Üks teema võib sisaldada mitut erinevat tudengit, kes sellega tegelevad. Iga teema kohta võib olla mitu erinevat ülesannet (*Assignement*) ja tudeng peab laadima nendele lahenduse (*Submission*), et hinne saada.

Thesespluginat kasutav tudeng saab teemat otsida ja sellele kandideerida ning hiljem ka oma tulemusi jälgida. Õppejõule võimaldab Thesesplugin oma töö teemasid hallata ning tudengite tegevusi jälgida. Haldur saab süsteemi abil hallata kõiki töö teemasid ning üle vaadata tudengite tulemusi.

2.2 Andmebaas

Joonisel 1 on toodud välja lihtsustatud andmebaasi arhitektuur koos selgitustega. Pistikprogramm Thesesplugin koosneb kolmest tabelist, muu info võetakse Moodle'i andmebaasist.



Joonis 1. Andmebaasi skeem

Pistikprogrammis on kasutatud kolme tabelit, mis on järgnevad:

`thesesplugin` – on antud pistikprogrammi instants

`id` – unikaalne identifikaator

`course` – viide kursusele, kus plugin asub

`timecreated` – aeg, millal plugin on lisatud

`timemodified` – aeg, millal pluginat muudetud

`theses` – on pistikprogrammi tabel, kus hoitakse kõiki lisatud teemasid

`id` – teema ID, mida kasutatakse programmis teema tuvastamiseks

`pluginid` – plugin ID näitab, millises pluginas on teema loodud

`title` – teema pealkiri

`timecreated` – aeg, millal teema on lisatud

`createrid` – kasutaja id, kelle nime all on teema lisatud

`description` – teema kirjeldus

`difficulty` – määrab teema raskusastme

`spaces` – määrab teema tudengite mahus

`category` – määrab teema kategooria Moodle'i sildi abiga

`status` – määrab teema seisundi

`applications` – on pistikprogrammi tabel, kus hoitakse kõiki tudengite avaldused

`id` – identifikaator, mida pistikprogramm kasutab avalduse tuvastamiseks

`userid` – kasutaja id, kes avalduse saatis

`thesisid` – teema id, mille all on avaldus saadetud

`status` – määrab avalduse staatuse (seisund)

`title` – avalduse pealkiri

`message` – avalduse sõnum tudengilt õppejõule

`timecreated` – avalduse loomise aeg

`pluginid` – pistikprogrammi instantsi id, mille kohta avaldus on tehtud

`reason` – vastuse (tühistamise või kustutamise) põhjus

2.3 Kasutusjuhud

Loodava Moodle'i pistikprogrammi kasutusjuhtudeks on:

1 Õppejõud

1.1 Uue teema lisamine.

1.2 Teema muutmine.

1.3 Teemale kandideerimisavalduste ülevaade

1.4 Kandideerimisavalduse vastuvõtmine

1.5 Tudengite tulemuste jälgimine

2 Tudeng

2.1 Teema otsimine

2.2 Teemale kandideerimine.

2.3 Oma tulemuste jälgimine.

2.4 Kandideerimisavalduse lisamine.

3 Haldur

3.1 Uute teemade lisamine teise õppejõu nime all.

3.2 Teema muutmine teise õppejõu nime all.

3.3 Kandideerimisavalduse vastuvõtmine teise õppejõu nime all.

3.4 Kõigi tudengite tulemuste jälgimine kõikides aktiivsetes teemades.

3.5 Exceli faili alla laadimine, kus on näidatud kõik tudengi ja õppejõu seosed teema järgi.

2.4 Nõuded

Loodava Moodle'i pistikprogrammi funktsionaalsed ja mittefunktsionaalsed nõuded on kirjeldatud selles peatükis.

Funktsionaalsed nõuded määratlevad pistikprogrammi võimalused. Mittefunktsionaalsed nõuded määratlevad paigaldamise ja keskkonna integreerimise võimalused.

2.4.1 Funktsionaalsed nõuded

1. Pistikprogramm toetab Moodle'i sisseehitatud funktsionaalsust (näiteks kustutamine, dubleerimine, taastamine jne).
2. Õppejõul on võimalus määrata oma teemade informatsiooni..
3. Õppejõul on võimalus muuta oma teemade informatsioon.
4. Õppejõul on võimalus saata ülevaate kandideerimisavaldustest ning neile vastata.
5. Pistikprogrammis on oluline, et teema saadaval kohtade arvu oleks võimalik määrata.
6. Pistikprogrammis on oluline, et oleks võimalik kasutada Moodle'i *tag*'ide funktsionaalsust.
7. Tudeng peab saama otsida teemat ning peab saama sellele kandideerida.
8. Tudeng peab saama vaadata vastuseid (*Submission*) oma ülesannetele (*Assignements*), mis on samal kursusel.

9. Süsteem peab tudengile ja õppejõule andma võimaluse tudengi ülesannete vastuste tulemusiläbi vaadata.
10. Süsteem peab andma haldurile võimaluse kõiki teemasid redigeerida, lisada uusi teemasid teise õppejõudu nime all ja ka vastata kandideerimisavaldustele.
11. Süsteem võimaldab haldurile kogu protsessi ja tulemuste jälgimise.

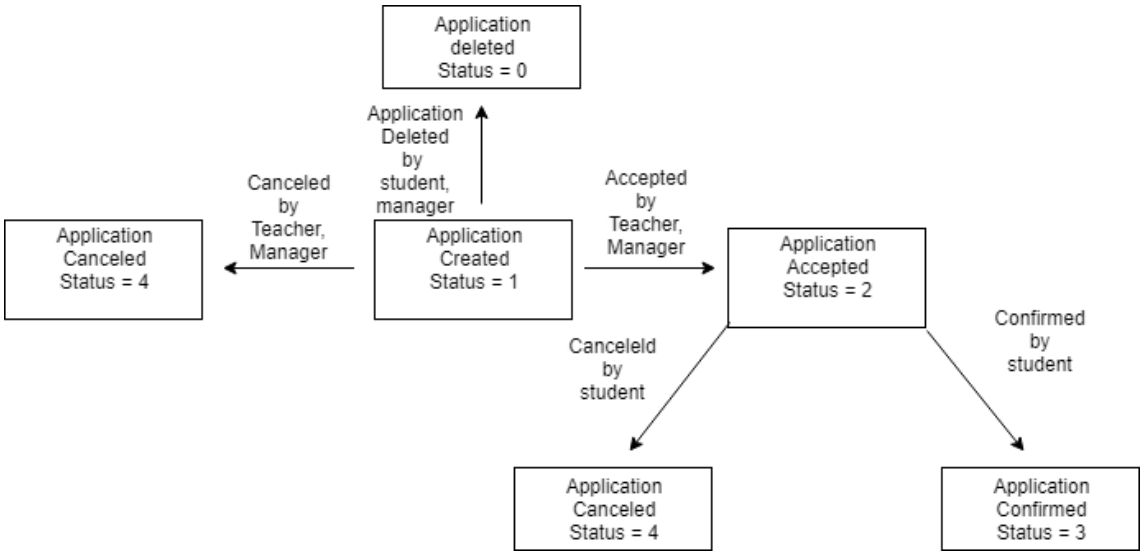
2.4.2 Mittefunktsionaalsed nõuded

1. Loodavat pistikprogrammi peab olema võimalik installida Moodle'i keskkonda.
2. Pistikprogramm luuakse tegevuse moodulina.
3. Loodavat pistikprogrammi peaks olema võimalik lisada kursusele.
4. Pistikprogrammis lähtekoodis on vajalik kasutada inglise keelt.
5. Pistikprogrammi kasutajaliides realiseeritakse inglise keeles ja eesti keeles.
6. Pistikprogramm peab olema dokumenteeritud.

2.5 Sündmuste diagramm

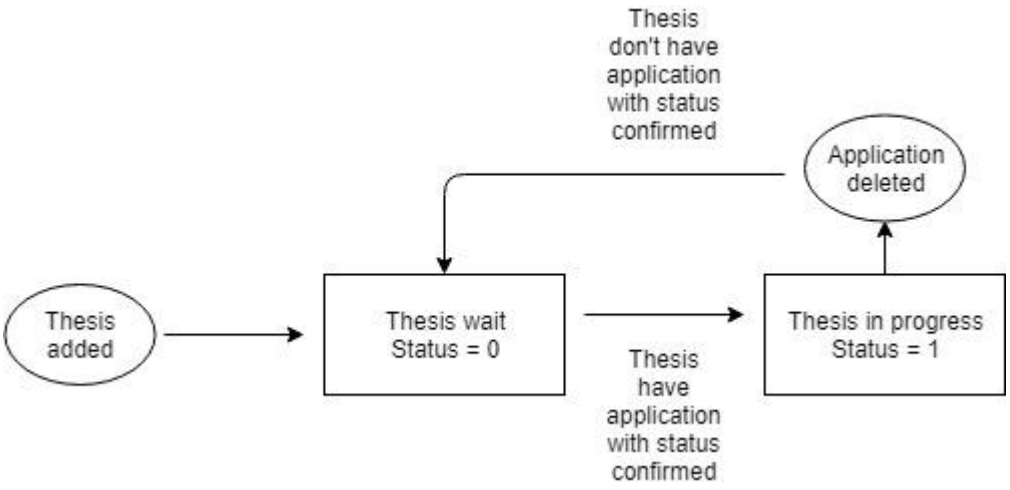
Järgmised diagrammid näitavad objektide olekuid erinevates etappides.

Joonisel 2 on välja toodud kandideerimisavalduse seisundi diagramm, millel on näidatud erinevad staatused pärast teatud tegevusi programmis.



Joonis 2. Avalduse seisundid

Joonisel 3 on välja toodud teema seisundi diagramm ning näidatud erinevad staatused pärast teatud tegevusi programmis.



Joonis 3. Teema seisundid

3 Arendusprotsess ja metoodika

Antud töös kasutati mitmeid levinuimaid arendusprotsesside võtteid ja metoodikaid.

3.1 Arendusmetoodika

Selles töös on kasutatud mitmeid projekteerimismeetodeid. Mooduli väljatöötamisel on oluline versioonikontrollisüsteemi arendamine ja seadmete vahel liikumise lihtsus. Giti hästi korraldatud ajalugu võimaldab tuvastada probleeme ja defekte ning tuvastada nende põhjuseid.

Arendamise protsess on aeganõudev ja veaohtlik Kasutades kahte erinevat keskkonda aitab see vältida vigu testimises, näiteks tööst eraldatud keskkond on mõeldud testimiseks. Test keskkonnas on võimalik kasutada erinevaid kasutuselevõtu protseduure. Oluline on võimalus kasutada erinevaid testkeskkonna andmebaasi konfiguratsioone. Kui ühest keskkonnast viia muudatus teise keskkonda, programmi testida ning teha kindlaks, et muudatus töötab vastavalt soovitule. Testimise keskkonnaks kasutatakse programme Docker ning Compose. Järgnevalt on täpsemalt kirjeldatud kasutatud programme.

Moodle'i pistikprogrammi arendamiseks on kasutatud pideva integratsiooni metoodikat.

3.1.1 Pidev integratsioon

Pidev integreerimine (CI - *Continuous Integration*) - tarkvaraarenduse tava, mis on tööalaste koopiate pidev ühendamine üldise peamise arengu haruga (kuni mitu korda päevas) ja sagedaste automatiseeritud projektikogumite rakendamine võimalike defektide varajaseks tuvastamiseks ja integratsiooniprobleemide lahendamiseks [8]. Tüüpilises projektis, kus arendaja töötab iseseisvalt süsteemi erinevates osades, on integratsioonietapp lõpmatu. Ta võib töö lõpetamist ettearvamatult edasi lükata. Üleminek pidevale integratsioonile vähendab integreerimise keerukust ja muudab selle ennustatavamaks, kuna vigade ja vasturääkivuste avastamine toimub varasemas staadiumis ja see annab võimaluse vigade kõrvaldamiseks samuti varasemas staadiumis.

Seega vähendab pidevkooste kasutamine vea parandamise kulusid selle varase avastamise tõttu.

Antud töös paigaldatakse testimata pistikprogramm esmalt testkeskkonnale virtuaal masinas ning läbitakse käsitsi kõik vajalikud funktsionaalsed testid. Moodle'i testimise keskkonnas on olemas erinevad kasutajad erinevate rollidega ning nende abiga testitakse kõik kasutusjuhud. Testimise protsess on kirjeldatud testimise protsessi peatükis. Kui kõik testid on läbitud, paigaldatakse pistikprogramm juba ained.ttu.ee keskkonda.

3.1.2 Git

Git on hajutatud versioonihaldussüsteem. Kogu arendusprotsess oli tehtud Giti hoidla abiga. Git-hoidla on failisüsteemi kataloog, mis sisaldab hoidla konfiguratsioonifaile, logifaile, mis talletavad hoidlas tehtud toiminguid, failide asukohta kirjeldavat indeksit ja faile ise sisaldavat hoidlat. On märkimisväärne, et ükski toiming ei muuda olemasolevate failide talletamise sisu [9].

Esmalt luuakse ülesanded Gitlabi keskkonda. Kui ülesanne või ülesanded on valmis, siis testitakse pistikprogramm käsitsi virtuaalmasinas Moodle'i keskkonnas ja alles pärast seda laaditakse töötav versioon Giti hoidlasse. Ja alustatakse teiste ülesannete tegemist.

3.1.3 Docker

Docker on avatud platvorm rakenduste arendamiseks, jagamiseks ja käivitamiseks. Docker võimaldab rakendusi infrastruktuurist eraldada, et saaks tarkvara kiiresti pakkuda. Dockeri abil saab infrastruktuuri hallata samal viisil kui rakendusi hallata. Kasutades Dockeri meetodeid, mis on ette nähtud koodide jagamiseks, testimiseks ja kasutuselevõtuks, saab märkimisväärselt vähendada koodi kirjutamise ja tootmise vahelist viivitust [10].

Docker paigaldatakse koos Compose tööriistaga ning docker-compose abiga installeeritakse ja seejärel käivitatakse Moodle'i keskkond pistikprogrammi testimiseks ning kontrollimiseks. Docker-compose kasutab rakenduse teenuste, võrkude ja köidete konfigureerimiseks YAML-faili

3.1.4 Moodle

Moodle (*Modular Object-Oriented Dynamic Learning Environment*) on avatud lähtekoodiga keskkond, mis pakub õpetajatele ja tudengitele isikupärastatud õpikeskkonda. Keskkond on jõuline, turvaline ja integreeritav. [1]. Martin Dougiamas avaldas Moodle'i esimese versiooni aastal 2001. [2]. Nüüdseks Moodle'i keskkond on installeeritud enam kui 80 000 serverile, omab üle 100 miljoni kasutajat ning üle 12 miljoni kursust, mida hallatakse 233 riigis[3].

Moodle'is on võimalik määrata erinevaid kursuste struktuure ja ülesandeid erinevatest ülesandetüüpidest ning kasutada palju erinevaid lisafunktsioone. Lisaks võib Moodle'i funktsionaalsust veelgi laiendada vastavalt kasutaja soovidele, on Moodle kasutusele võtnud pistikprogrammid. Moodle'il on suur kasutajate arv ja tänu sellele on ametlikus laienduste kataloogis saadaval üle 1300 erineva pistikprogrammi [4]. Ametlikule laienduste kataloogile on võimalik lisada oma pistikprogramme. Seetõttu on süsteemis juba olemas turvaline sisselogimine ning on võimalik kasutada olemasolevaid rolle. Moodle'i välimust on võimalik kergesti muuta ning süsteemi saab tõlkida erinevatesse keeltesse, nüüdseks on Moodle tõlgitud juba üle 100 erinevasse keelde [5].

Kasutatud on Moodle 3.6 versiooni, ning läbitud selles keskkonnas kõik tegevused, mis on seotud pistikprogrammi testimise ja kontrollimisega.

3.1.5 PHP

PHP on laialt levinud avatud lähtekoodiga üldotstarbeline skriptikeel. PHP on programmeerimiskeel, mis on spetsiaalselt loodud veebiserveril töötavate veebirakenduste kirjutamiseks. PHP tähistab *"Hypertext Preprocessor"*. Keele süntaks on pärit C-st, Javast ja Perlist, PHP on piisavalt lihtne programmeerimiskeel [11].

PHP oluline eelis Perli ja C keelte ees on võime luua HTML-dokumente varjatud PHP-käskudega. Moodle'i keskkond on arendatud PHP keeles ja pistikuprogrammide arendus tehakse samas keeles. Kogu keskkonnas on suur kogus klasse, mille funktsionaalsus lihtsustab arendamist. PHP annab väga laialdased võimalused objektorienteeritud programmeerimises.

PHP installeerimine on vajalik serveri poolt ning antud virtuaalmasinas oli kasutatud kõige stabiilsemat eelviimast PHP versiooni 7.0 .

3.1.6 Arenduskeskkonna seadistamine

Pistikprogrammi arendamiseks on vajalik tööle panna Moodle'i arenduskeskkond, kus saab programmi testida enne kasutamist. Operatsioonisüsteemiks sai valitud Ubuntu 18.04, pärast selle installimist paigaldati süsteemile Docker ja Compose, millega käivitati Moodle'i keskkond pistikprogrammi testimiseks.

Kasutades Dockerit ei ole vaja installida ja konfigureerida veebiserverit, PHP-d ega andmebaasiserverit. Kõik vajalikud moodulid paiknevad konteineri sees ja installitakse docker-compose abiga. Selline meetod lubab arenduskeskkonna käivitada lihtsalt ja kiirelt. Testimise protsessis saab kasutada iga kord värsket installatsiooni ja vältida kogunenud muutusi infrastruktuuril.

Moodle'i jaoks on loodud valmis Dockeri tõmmis (image). Ühtlasi pakutakse Moodle'i jaoks ka tervet Composeri konfiguratsioonifaili selleks, et Moodle'i keskkond ja andmebaas püsti panna.

Dockeri paigaldamise jaoks kasutakse tarkvara paigalduse rakendust apt-get. Selle abiga paigaldatakse docker.io moodul. Lisaks laaditakse alla docker-compose rakendus ning seejärel käivitatakse docker ja docker-compose.

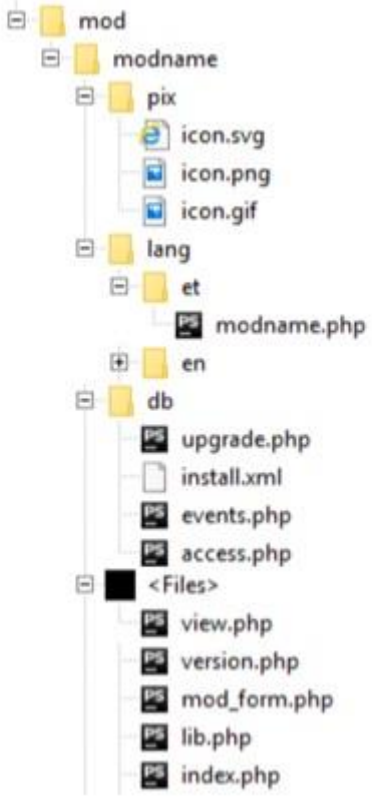
Docker-compose paigaldab vajalikud rakendused vastavalt YAML konfiguratsioonile ja käivitatakse Moodle'i keskkond.

3.1.7 Moodle'i pistikprogrammi arendamine

Plugin ehk pistikprogramm on kõige lihtsam viis Moodle'ile lisafunktsionaalsuse lisamiseks. Moodle'i keskkonnas on määratud üle 50 erineva standardiseeritud pistikuprogrammitüübi. Kogu nende loomise töökäik on dokumenteeritud. Kui standardiseeritud pistikprogrammi tüüpidest ei leidu sobivat, saab luua ka täiesti uut tüüpi kohaliku (*local*) pistikprogrammi [6]. *Activity* tüüp on kõige kõlblikum selle pistikprogrammi loomiseks kuna selline tüüp omab vajalikku funktsionaalsust ülesande täitmiseks.

Activity tüüpi pistikprogramm on kõige enam kasutatav, sellega on võimalik lisada kursuste alla tegevusi. Moodle'i keskkonnas on määratud kaustad erinevate tüüpide jaoks ja kaustas `/mod` asuvad *Activity* tüüpi pistikprogrammid. Uue pistikprogrammi installerimise protsessi jooksul luuakse uus kaust `/mod/<modname>`, kus `<modname>`

on pistikprogrammi nimi. Iga pistikprogrammi kaustas on olulised kohustuslikud failid ja kaustad ning kohandatud failid, mida arendaja saab kasutada oma funktsionaalsuse realiseerimises. Kohustuslikud failid määravad pistikuprogrammi installeerimist, deinstalleerimist ning suhtlemist ja andmevahetust Moodle'i süsteemiga. *Activity* tüüpi pistikprogrammi kohustuslike failide struktuur on näidatud joonisel 4 [7].



Joonis 4. Moodle'i failide struktuur

Kohustuslike failide sisu kirjeldus:

`db` – andmebaasi struktuur, süsteemi õigused ning sündmuse defineerimine

`db/install.xml` – määratakse andmebaasi struktuur XML (*Extensible Markup Language*) formaadis, mis pistikprogrammi tegevuses vajalik.

`db/events.php` – määratakse sündmuste kuulajad, näiteks kui luuakse või muudetakse ülesannet.

`db/upgrade.php` – sisaldab funktsioonid mis käivitatakse uue pistikprogrammi versioon paigaldamisel.

`db/access.php` – sisaldab õiguste ja konteksti määramine kasutatud pistikprogrammis.

`pix` – kaust pistikprogrammi ikoonide jaoks erinevates formaatides.

`lang` – kaust mis sisaldab pistikprogrammi kasutajate nähtavad teksti erinevates keeltes, kasutatakse kasutajaliidesena tõlkimises.

`lang/<keele abreviatuur>/<modname>.php` – tõlkimise fail.

`view.php` – sisaldab pistikprogrammi vaate funktsionaalsust.

`version.php` – sisaldab pistikprogrammi versiooni ja vajalikku Moodle'i keskkonna versiooni pistikprogrammi funktsioneerimiseks.

`mod_form.php` – sisaldab mooduli initsialiseerimise, muutmise ning valideerimise funktsionaalsust.

`lib.php` – sisaldab funktsioone vajaliku instantsi lisamiseks, uuendamiseks ning valideerimiseks.

`index.php` – põhifail, mis määrab pistikprogrammi vaadet kõikides moodulites.

Kasutatavad globaalsed Moodle muutujad:

`$DB` – viide Moodle'i andmebaasi objektile

`$USER` – sisaldab Moodle'i kasutaja andmeid

3.2 Testimisprotsess

Arendusprotsessi lõpus läbitakse vajalikud testid käsitsi, need on kirjeldatud allpool. Testimiskeskkonnal on olemas kasutajad rollidega: tudeng, õpetaja ja haldur. Järgmised sammud kirjeldavad ära kõige tähtsamad tegevused testimisel:

Kontrollimise testid:

1. Teema haldamine (õppejõud)

1.1. Lisatakse teema

(teema on salvestatud andmebaasis, õppejõud saab muuta ja kustutada)

1.2. Lisatakse teema teise õppejõudu nime all

(Õppejõud saab muuta ainult oma teemat, tudeng ei saa ühtegi)

2. Teemale registreerimiseavalduse saatmine (tudeng)

2.1. Lisatakse 5 teemat

2.2. Otsitakse lisatud teemasid

(Kõik lisatud teemad on nimekirjas)

2.3. Saadetakse 5 kandideerimisavaldust lisatud teemadele (igale teemale üks)

(Tudeng saab teha kandideerimisavaldusi mitmel teemal, kandideerimisavaldused on nähtavad nimekirjas ja neid näeb õppejõud)

2.4. Kustutatakse üks kandideerimisavaldus

(tudeng saab kustuta oma kandideerimisavaldust)

3. Teema kandideerimisavaldus vastuvõtmine (õppejõud)

3.1. Ühe tudengi kandideerimisavaldus tühistatakse, teine võetakse vastu

(Kandideerimisavaldus on seisundis „ootab kinnitamist“)

3.2. Tudeng kinnitab kandideerimisavalduse

3.3. Kontrollitakse õige kandideerimisavalduse näitamist

(Õppejõud, tudeng ja haldur näevad kandideerimisavaldusi õiges seisundis)

4. Kandideerimisavaldus haldamine (tudeng)

4.1. Üks vastuvõetud kandideerimisavaldus tühistatakse ning kustutatakse.

4.2. Üks vastuvõetud kandideerimisavaldus kinnitatakse

4.3. Vaadatakse aktiivset teemat ja selle ülevaadet

(Nüüd näeb tudeng, et teema ja seos õppejõuga on tekkinud. Tudeng ei saa enam kandideerimisavaldusi saata.)

4.4. Kontrollitakse õige kandideerimisavalduse näitamist

(Õppejõud, tudeng ja haldur näevad kandideerimisavaldusi õige seisundis)

5. Ülesande haldamine (tudeng/õppejõud)

5.1. Õppejõud lisab ülesande kursusele

5.2. Tudeng saadab oma vastuse ülesandele

(Ülesande vastus on nähtav teema läbivaatamisel tudengile ning õppejõule)

6. Halduri lisavõimalused

6.1. Vaadatakse läbi kõik kandideerimisavaldused ja kustutatakse viimane ootel kandideerimisavaldus tudengilt.

6.2. Süsteemi lisatakse uus teema teise õppejõu nime all ja läbitakse selle teema muutmine.

(Teema on lisatud ja seda võib muuta)

6.3. Võetakse eksportimise vaheleht ja kontrollitakse, et kõik seosed õpetaja ja tudengi vahel on tekkinud ning seda, et eksport töötab korrektselt. (CSV failis on kõik õppejõud, teemad ja tudengite teemad tegevusel)

6.4. Kontrollitakse kõiki olemasolevaid vahelehti.

(Kõik teemad ja kandideerimisavaldused on olemas õigel lehel)

6.5. Kustutatakse kinnitatud tudengi kandideerimisavaldus .

(Ei tudengil ega õppejõul ei ole enam tegevusel teemat. Tudeng saab uuesti kandideerimisavaldusi saata)

7. Kandideerimisavalduse vastuvõtmine halduri rollis

7.1. Tudeng saadab kandideerimisavalduse, haldur tühistab kandideerimisavalduse

(Kandideerimisavaldus on tühistatud seisundis)

7.2. Tudeng saadab kandideerimisavalduse, haldur võtab kandideerimisavalduse vastu

(Kandideerimisavaldus aktsepteeritud seisundis)

7.3. Tudeng kinnitab kandideerimisavalduse ja tühistab (ning kustutab) teise

(Tühistatud kandideerimisavaldused on õiges seisundis ja kinnitatud kandideerimisavaldusel on tehtud seos õppejõuga)

7.4. Haldur kontrollib, et kõik seosed tudengi ja õppejõudu vahel on tekkinud

4 Lahendus

Järgnev kirjeldus annab ülevaate sellest, kuidas saavutati eesmärk. Selgitus hõlmab nii mooduli funktsionaalsuse ülevaadet kui ka projekti struktuuri.

4.1 Funktsionaalsus

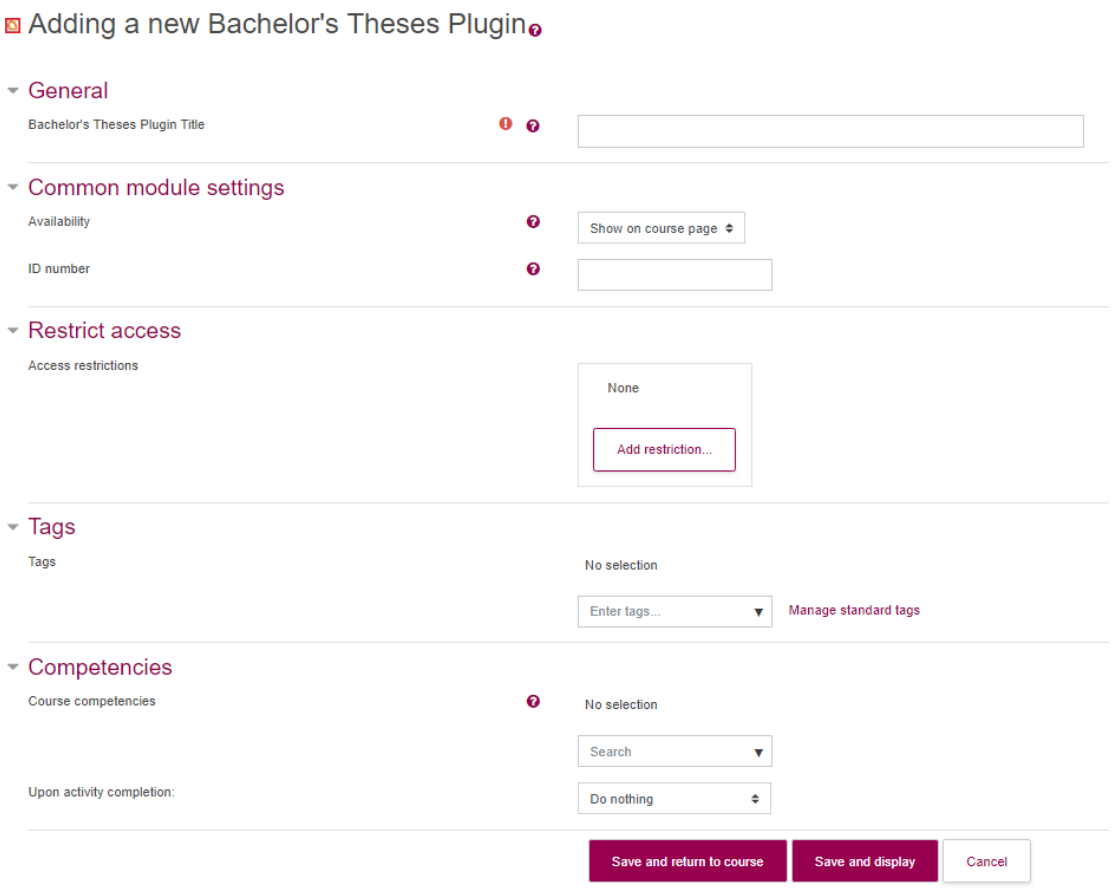
Pistikprogrammi funktsionaalsuse ülesanne on tavaliselt täita Moodle'i pistikprogrammi tegevusi nagu teemade lisamine, ülevaatamine ja redigeerimine. Nendele ülesannetele peab pistikprogramm vastu võtma infot, kui tudeng saadab kandideerimisavalduse õppejõu teemale, ning seejärel edastama infot õppejõule koos teavitusega. Õppejõud edastab süsteemile kandideerimisavalduse tulemuse ning moodul salvestab andmed tekitades tudengi ja õppejõu vahel seose. Iga kursusega on seotud moodul, mida kasutatakse teemade lisamiseks. Igal kursusel on omad teemad, õppejõud, haldurid ja tudengid. Kasutaja rollid on määratud Moodle'i süsteemis ja pistikprogramm kasutab neid.

4.1.1 Vormid

Töös on kasutusel kaks olulisemat vormi: pistikprogrammi seadete vorm ja uue instantsi lisamise vorm. Lisamise vormis on võimalik märkida pistikprogrammi nimi. Järgnevalt kirjeldatakse neid vorme.

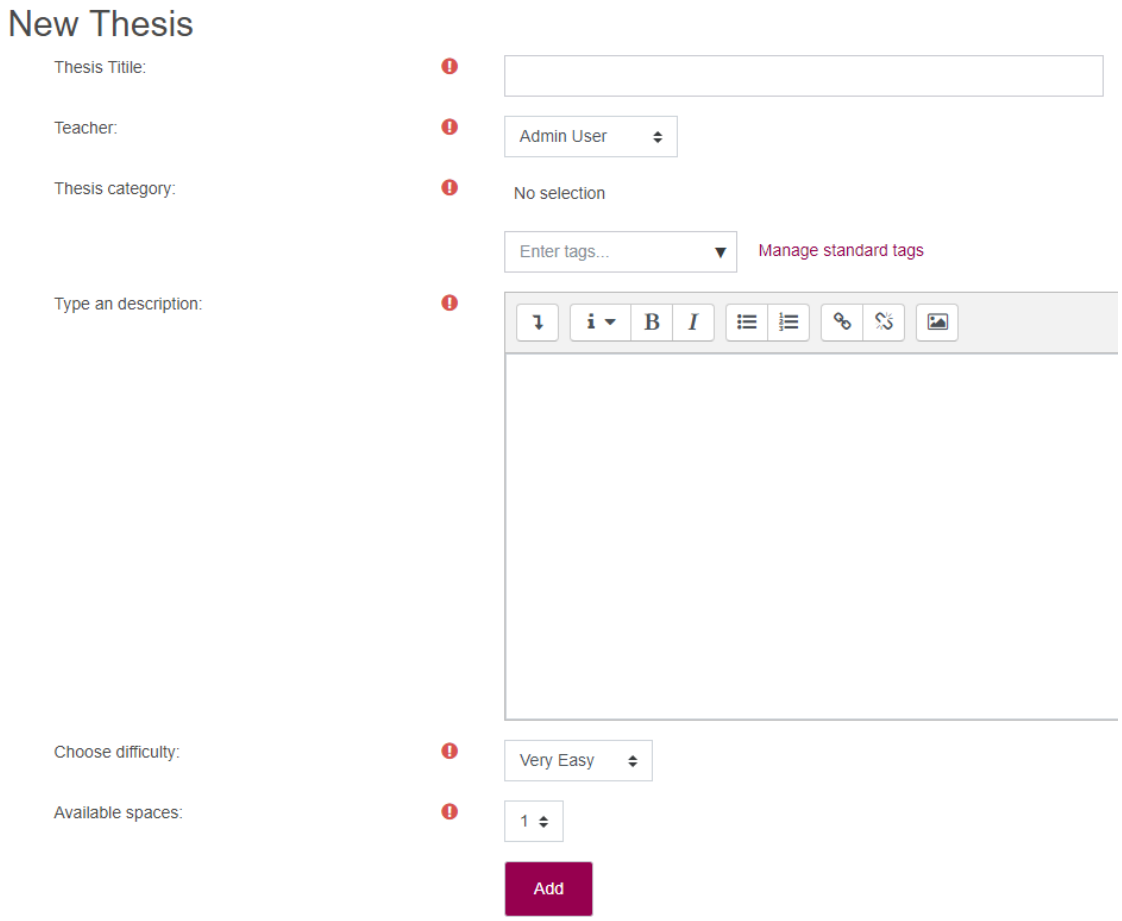
Iga mooduliga on seotud kursi seadeid. Seadetes saab kursusele lisada erinevaid inimesi rolli järgi, kui inimene ise paneb ennast kursusele kirja siis tema rool määratakse tudengina. Kõik neid inimesi pärast valmis pistikuprogrammi kasutamiseks.

Pistikuprogrammi lisamise vorm, ehk *instance form*, koosneb programmi nimest ja Moodle'i tegevuse tavapärastest lahtritest. Joonisel 5 on toodetud pistikuprogrammi lisamise vorm.



Joonis 5. Pistikprogrammi lisamise vaade

4.1.2 Teema lisamine



Joonis 6. Teema lisamise vaade

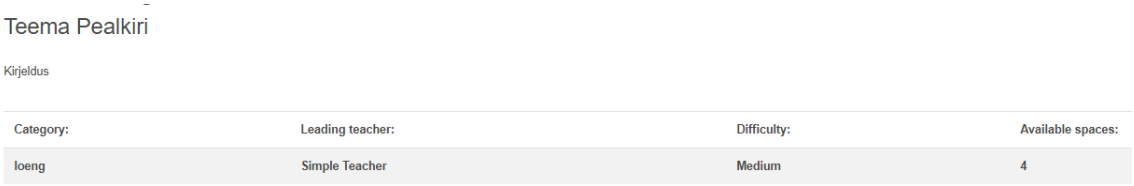
Joonisel 6 on välja toodetud teema lisamise vorm.

Teemade lisamise vormis on väljad: teema pealkiri, õppejõu valik halduri poolt või õppejõu puhul automaatselt valitud kasutaja, kategooria valik, kirjeldus, raskustase ja tudengite maht (mitu tudengit saab seda lõputööd teha, vaikimisi 1).

Edaspidiselt kasutakse neid lahtreid nii:

Teema pealkiri, õppejõu nimi, kategooria ja raskus on nähtavad teema ülevaatamisel ning otsingul. Raskuse väli on mõeldud tudengile, et mõista valitud teema keerukust. Kohad määrab hetkel vabade kohtade arvu antud teemal, mis muutub, kui tudengi kandideerimisavaldus on vastu võetud või tühistatud.

4.1.3 Teema ülevaatamine

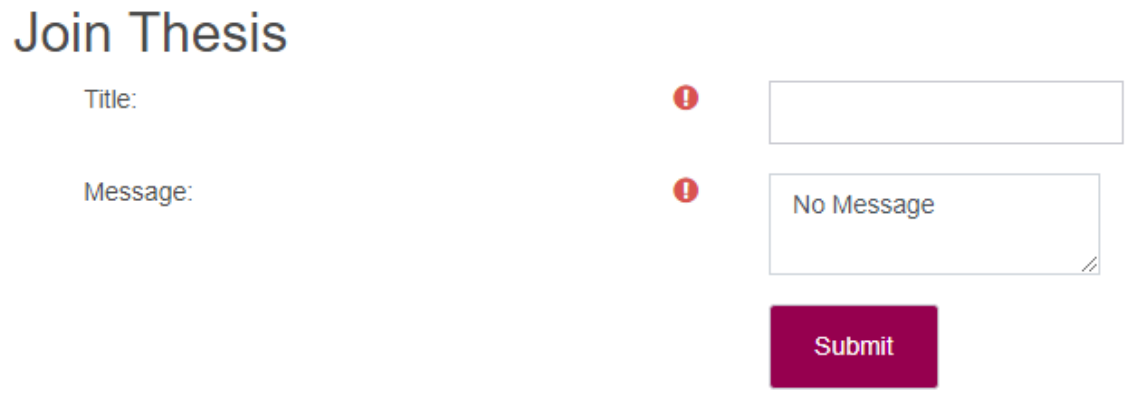


Joonis 7. Teema ülevaatamise vaade

Joonisel 7 on toodetud teema ülevaatamise vaade.

Teemade ülevaatamine näitab kogu teema informatsiooni ja võimaldab tudengil kandideerida (lisada kandideerimisavaldus).

4.1.4 Kandideerimise lisamine

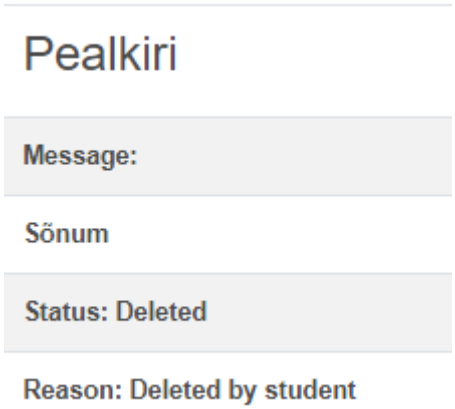


Joonis 8. Kandideerimisavalduse lisamise vaade

Nägu Joonisel 8 on toodetud kandideerimisavalduse lisamise vormis määrab tudeng pealkirja ja sõnumi.

Kandideerimisavalduse pealkiri on hiljem nähtav otsingus. Kandideerimisavalduse detailses vaates on näha nii pealkiri kui sõnumi sisu.

4.1.5 Kandideerimisavalduse ülevaatamine



Joonis 9. Kandideerimisavalduse ülevaatamise vaade

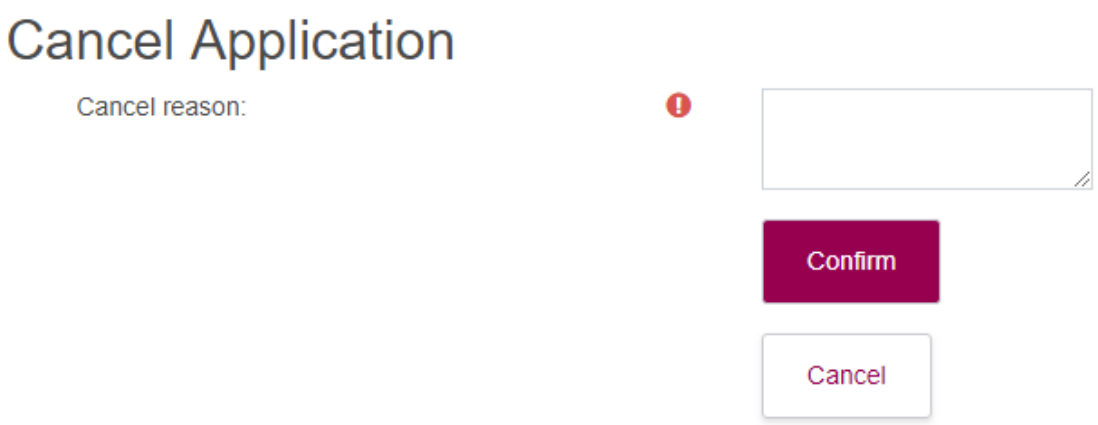
Joonisel 9 on näidatud, et kandideerimisavalduse ülevaatamisel näidatakse kogu kandideerimisavalduse informatsioon.

Ülevaate all on olemas kandideerimisavaldusele vastamise nupud haldurile ja õppejõule ning kandideerimisavalduse kustutamise nupp tudengile.

Õppejõul või halduril on võimalus vastu võtta või tühistada tudengi kandideerimisavalduse nuppude abil. Vajutades tühistamise nuppu avatakse kandideerimise tühistamise vaade.

Tudengil või halduril on võimalus kustutada kandideerimisavaldus kustutamis nupuga.

4.1.6 Kandideerimisavalduse tühistamine

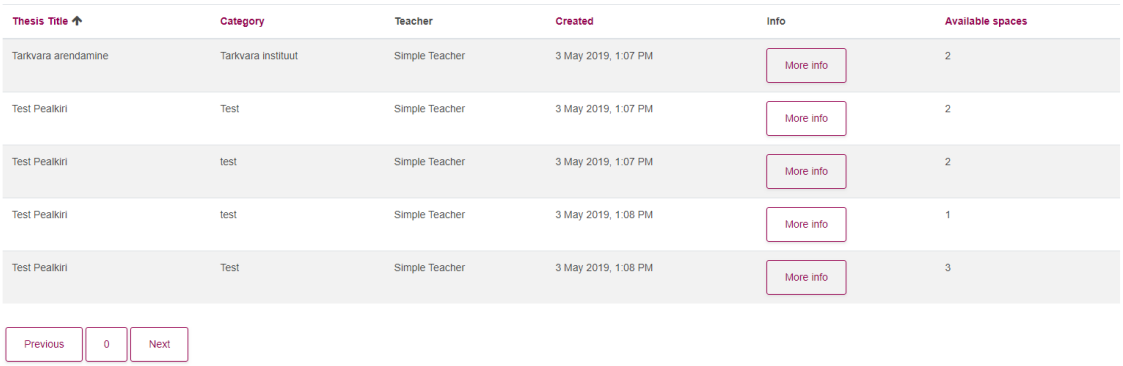


Joonis 10. Kandideerimisavalduse tühistamise vaade

Joonisel 10 on kandideerimise tühistamise vaade, kus õppejõud või haldur määrab tühistamise põhjuse, mis on hiljem nähtav kandideerimise ülevaatamisel.

4.1.7 Teemade nimekiri

Teemade ja Kandideerimisavalduste läbivaatamiseks ja otsimiseks kasutakse Moodle'i andmetabeli funktsionaalsust, mida on mõnedel juhtudel võimalik sorteerida ning filtreerida. Joonisel 11 on tabeli vaade, mida kasutatakse teema otsimiseks, sorteerimiseks ja filtreerimiseks.



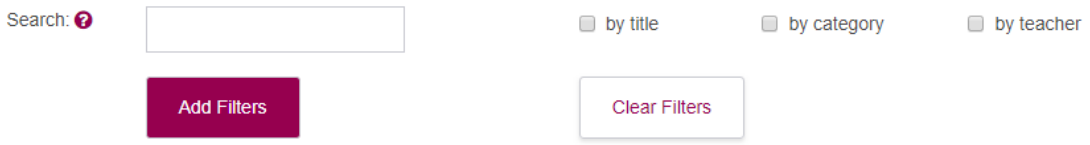
Joonis 11. Teemade tabel

Igal tabelil on oma otsingu ja sorteerimise võimalused, mis on kirjeldatud järgnevalt. Veerud vahetuvad vastavalt valitud vahelehele muutmata tabeli formaati.

4.1.8 Filtreerimine

4.1.8.1 Teema filtrid

Õppejõu, tudengi ja halduri jaoks –



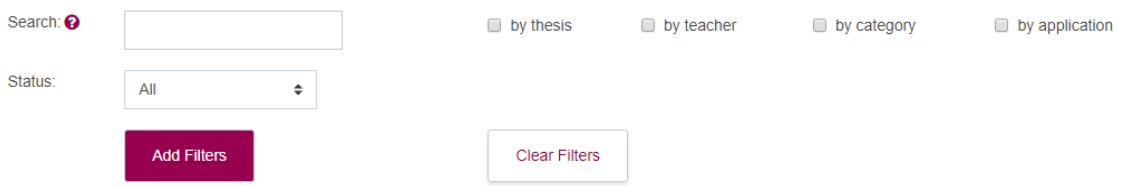
Joonis 12. Teema otsingu filtrid

Joonisel 12 on nähtav, et märkeruudu abiga on võimalik täpsustada, millistest väljades väärtust otsitakse.

Ilma valikuta otsitakse kõikidest lahtrites – teema pealkirjast, kategooriast ja õppejõu nimest.

4.1.8.2 Kandideerimise filtrid

Õppejõu ja tudengi jaoks –

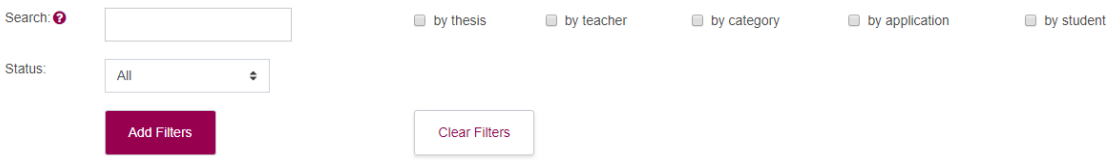


Joonis 13. Kandideerimisavalduse otsingu filtrid

Joonisel 13 on nähtav, et märkeruudu abiga on võimalik täpsustada otsingu parameetreid.

Ilma valikuta otsitakse kõikidest lahtrites – teema pealkirjast, õppejõu nimest, kategooriast ja kandideerimisavalduse nimest.

Halduri jaoks –



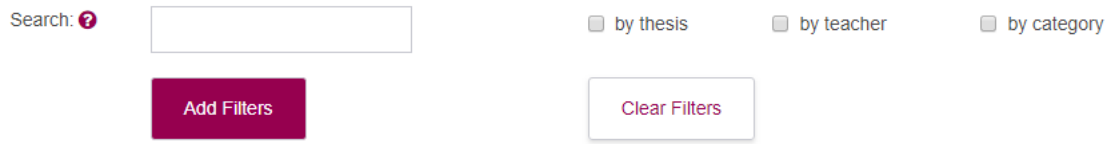
Joonis 14. Kandideerimisavalduse otsingu filtrid haldurile

Joonisel 14 on nähtav, et märkeruudu abiga on võimalik täpsustada otsingu parameetreid.

Ilma valikuta otsitakse kõikidest lahtrites – teema pealkirjast, kandideerimisavalduse pealkirjast, tudengi nimest, kategooriast ja õppejõu nimest.

4.1.8.3 Aktiivsete teemade filtrid

Halduri jaoks –



Joonis 15. Kõigi tegevuses olevate teemade filtrid

Joonisel 15 on näha, et märkeruudu abiga on võimalik täpsustada otsingu parameetreid.

Ilma valikuta otsitakse kõikidest lahtrites – teema pealkirjast, kategooriast ja õppejõu nimest.

4.2 Struktuur

Pistikprogrammi failid on tavaliselt jagatud kolmeks. Esiteks on vajalikud Moodle failid, mis on igal moodulil kohustuslikud ja peavad asuma juurkaustas. Teiseks on olemas PHP failid, mis vastutavad funktsionaalsuse eest ja asuvad pistikprogrammi kaustas. Kolmas jaotus on failid andmebaasi installimiseks. Allpool on kirjeldatud projekti osade struktuur.

4.2.1 Moodle'i failid

Moodle eeldab, et pistikprogrammis on failid, mida Moodle saab käivitada vajadusel või kasutada. Moodle'ile on vajalik järgmised failid:

- `version.php` – failis on mooduli ja Moodle'i versiooni numbrid. Moodle vaatab versiooni numbreid. Kui versioon on uuem siis Moodle käivitab mooduli uuendamise skripti. Uuendamise skriptis on tavaliselt andmebaasi uuendamine või andmete muutmine.
- `lib.php` – failis on funktsioonid, mida Moodle kasutab mooduli instantsi lisamiseks, muutmiseks või kustutamiseks õppekeskkonnas. Mooduli instantsi lisamiseks käivitab Moodle moodulis olemas oleva funktsiooni `thesesplugin_add_instance`, muutmiseks `thesesplugin_update_instance` ja kustutamiseks `thesesplugin_delete_instance`.
- `settings.php` – mooduli seadete asukoht, mida saab muuta ainult administraator. Need sätted kehtivad Thesesplugini kohta ning kogu õppekeskkonna ulatuses.

- Kaust `lang/` - failis on pistikprogrammi tõlked. Thesesplugini kaustas on kaks tõlkefaili, kus on eesti ja inglise keele tõlked.
- Kaust `db/` - kaustas on mooduli installeerimiseks ja uuendamiseks skriptid. Fail install.xml määrab XML kujul andmebaasi skeemi. Mooduli installeerimise ajal genereerib Moodle failist SQL laused ja käivitab need andmebaasi täitmiseks.
- `db/events.php` – faili kasutatakse Moodle'i sündmuste teostamiseks. Moodle käivitab failist erinevaid sündmusi, milliseid moodulis toimuvad.
- `db/access.php` – Thesesplugini õiguste kirjeldamine rollide järgi. Selles failis on kirjas näiteks, mis rolliga kasutajad saavad Thesesplugina teemat lisada ja mis rolliga kasutajad saavad teemat muuta jne.
- Kaust `classes/` - sisaldab klasse, mida on kasutatud pistikprogrammis.
- Kaust `pix/` - kaustas on kõik mooduli pildid ja ikoonid. Ikoon on saadaval erinevates formaatides (näiteks png, gif).

Kõiki ülaltoodud faile kasutab Moodle programmiliides uue mooduli instantsi lisamiseks. Failist lib.php genereerib Moodle päringud mooduli instantsi lisamiseks, muutmiseks ning kustutamiseks, annab need rakendusele ette ja käivitab [7].

5 Kokkuvõte

Töö eesmärgiks oli luua Moodle'i õppekeskkonna pistikprogramm, mis võimaldab lisada lõputöö teemasid, neid otsida ja tudengitel kandideerida. Pistikprogramm peab võimaldama mugava teemade lisamise ja neile kandideerimise, ülevaatamise ja vastamise. Samal ajal peab moodul andma ka tegevuste haldamise õigused õppejõule ja haldurile. Tudengid peavad saama teemade otsimise, kandideerimise ja jälgimise võimaluse. Viimaseks eesmärgiks oli vaja valmistada pistikprogramm nii, et seda oleks võimalik automaatselt uuendada ained.ttu.ee keskkonnas.

Eesmärgi täitmiseks tehti pistikprogramm, mis võimaldab kõiki lõputööga seotud tegevusi. Pistikprogramm on tehtud PHP programmeerimiskeeles. Tehti 20 PHP faili 3500 koodireaga. Lisaks kasutati teisi liiki faile nagu XML failid.

Järgmise eesmärgi valmistamiseks on tehtud ained.ttu.ee keskkonna automaatne teavitamise ning mooduli uuendamise võimalus. Tulemusena on ühe nupuvajutusega võimalik loodud moodul ained.ttu.ee keskkonda installeerida või uuendada seda.

Sügissemestril on plaanis kasutada pistikprogrammi õppekeskkonnas. Planeeritakse otsida inimesi (peamiselt tudengeid), kelle ülesandeks oleks pistikprogrammi hooldamine ja lisafunktsionaaluste arendamine ning lisamine. Pistikprogrammi lähtekood on saadaval TTÜ Giti salve aadressil https://gitlab.cs.ttu.ee/mapota/iapb

5.1 Edasised arendused

Selles peatükis on välja toodud potentsiaalsed pistikprogrammi Thesesplugin edasised arendused. Kogu pistikprogramm on kirjutatud ainult Moodle'i standardsete olemasolevate objektide abiga, seetõttu on programmi mugav ja lihtne edasi arendada.

5.1.1 Automaatne testimine

Teostada automaatsed testid keskkonnale. Realiseerida testimine teemade lisamiseks, muutmiseks, kandideerimiseks ning kandideerimisavalduse vastuvõtuks. Automaatsed

testid peavad võimaldama automatiseerida kogu testimise protsessi ning asendama käsitsi testid.

5.1.2 Andmete ekspordi edasiarendamine

Ette valmistada andmete eksportimine integreerimiseks teiste ülikooli süsteemidega ning lisada ekspordi tulemusse vajalik informatsioon tudengi ülesannete vastustest, näiteks lõplik lõputöö teema eesti ja inglise keeles. Informatsioon, mis on vajalik eksportimiseks, plaanitakse saada läbi intervjuu inimesega, kes kõiki lõputöid pärast haldab, näiteks lisab lõputöö andmed raamatukogu digiarhiivi.

5.1.3 Teavitused

Automatiseerida teavituste saatmine tudengitele ning õppejõududele kui mingi sündmus toimub pistikprogrammis, näiteks tudengi kandideerimisavalduse saatmisel või kandideerimisavalduse vastuse saabumisel õppejõult.

Kasutatud kirjandus

[1] About Moodle. [WWW]
https://docs.moodle.org/32/en/About_Moodle (27.04.2019)

[2] Moodle HQ. [WWW]
https://moodle.com/hq/ (27.04.2019)

[3] Moodle statistics. [WWW]
https://moodle.net/stats/ (27.04.2019)

[4] Moodle plugins directory. [WWW]
https://moodle.org/plugins/ (27.04.2019)

[5] Moodle language packs. [WWW]
https://docs.moodle.org/32/en/Language_packs (27.04.2019)

[6] Moodle plugin types. [WWW]
https://docs.moodle.org/dev/Plugin_types (27.04.2019)

[7] Moodle activity modules. [WWW]
https://docs.moodle.org/dev/Activity_modules (27.04.2019)

[8] Continuous integration. [WWW]
https://en.wikipedia.org/wiki/Continuous_integration (01.05.2019)

[9] Git documentation. [WWW]
https://git-scm.com/doc (01.05.2019)

[10] Docker documentation. [WWW]
https://docs.docker.com/ (01.05.2019)

[11] Php documentation. [WWW]
https://www.php.net/docs.php (01.05.2019)

[12] Official Ubuntu documentation. [WWW]
https://help.ubuntu.com/ (01.05.2019)

[13] How To Install Docker. [WWW]
https://phoenixnap.com/kb/how-to-install-docker-on-ubuntu-18-04 (01.05.2019)

[14] How To Install and Use Docker Compose. [WWW]
https://linuxize.com/post/how-to-install-and-use-docker-compose-on-ubuntu-18-04/
(01.05.2019)

[15] Composer Getting Started. [WWW]
https://getcomposer.org/doc/00-intro.md (01.05.2019)

[16] Linux vs Ubuntu. [WWW]
https://www.educba.com/linux-vs-ubuntu/ (01.05.2019)