TARTU ÜLIKOOL
MATEMAATIKA-INFORMAATIKATEADUSKOND
Arvutiteaduse instituut
Infotehnoloogia õppekava

Alar Kirikal

Aine "Süsteemihaldus" õppesüsteemi praktikumijuhendaja veebiliides

Bakalaureusetöö (6 EAP)

Juhendaja: Artjom Lind

Tartu 2014

Sisukord

Infoleht

Bakalaureusetöö pealkiri inglise keeles

Web Interface for the practice supervisors of the course "System Administration"

Lühikokkuvõte

Antud bakalaureusetöö käigus luuakse aine MTAT.08.021 "Süsteemihaldus" praktikumijuhendajate jaoks veebiliides, mis tsentraliseerib kursuse juhendajate ressursid ja mille abil oleks tulevikus lihtsam kursust administreerida. Hetkel on kursuse juhendajatel kasutusel mitmeid erinevaid skripte ning tööriistu, kuid neid kasutatakse vastavalt oma äranägemisele suvalises järjekorras ning tulevastel kursusejuhendajatel on raske neid kasutada. Seetõttu vajaminevad skriptid automatiseeritakse ja integreeritakse töö käigus valminud veebikeskkonda, mis muudab skriptide kasutamise lihtsamaks.

Short Summary

The purpose of this Bachelor's Thesis was to develop a website for the supervisors of the System Administration course in Tartu University. This web will act as a centralized server for the resources, that are currently in use by the supervisors. Currently these resources, which are mostly scripts, are used randomly and whenever supervisor needs them. In case new supervisors join the course, they will have little idea of how to use them and which scripts even exists. These scripts will be integrated with the webserver and are made easy to use. Some of them will also be automated.

Märksõnad

Süsteemihaldus, Python, Flask, Twitter Bootstrap, PostgreSQL, Jinja2, veebirakendused

Keywords

System administration, Python, Flask, Twitter Bootstrap, PostgreSQL, Jinja2, web development

Terminoloogia

ÕIS - Tartu Ülikooli Õppeinfosüsteem, mille kaudu tudengid ennast ainetele registreerivad ning kust ainete juhendajad üliõpilaste info võtavad.

Testserver - server, mis käitab ning haldab skripte, mis kontrollivad tudengite edasijõudmist aine "Süsteemihaldus" praktikumides.

Testserveri veebiliides - veebiliides, kuhu testserver tulemusi üles paneb ning mille kaudu tudengid oma tulemusi vaadata saavad.

Praktikumijuhendite veebiliides - veebiliides, kust tudengid saavad lugeda praktikumijuhendeid, mille järgi nad hakkavad praktikumi virtuaalmasinat seadistama.

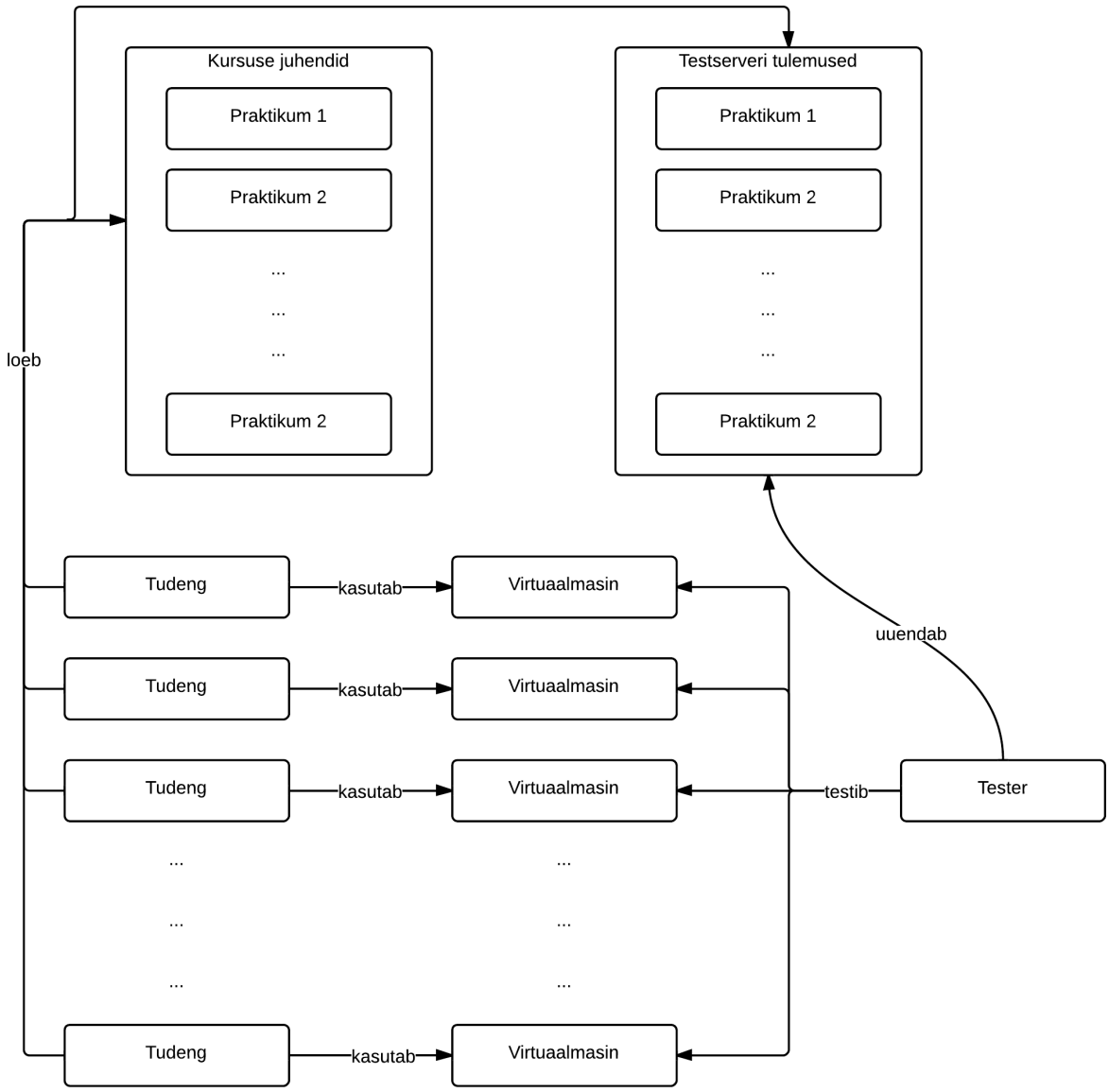
Admin server - server, mis sisestab enda andmehoidlasse Õppeinfosüsteemist saadud tudengite informatsiooni ning loob tudengitele neile vastava virtuaalmasina, mida tudengid praktikumides seadistama hakkavad.

Admin veebiliides - veebiliides, mille kaudu kursuse juhendajad admin serverit hallata saavad.

Sissejuhatus

Antud lõputöö ülesandeks on valmistada Tartu Ülikooli aine "Süsteemihalduse" administreerimiseks veebiliides. Kuna hetkel tehakse suurem osa administreerimistöödest käsitsi ning paljuski ühe õppejõu teadmiste põhjal, siis oleks vaja aine jaoks sellist veebiliidest, mille kasutamisega aine uued õppejõud kergema vaevaga hakkama saaksid.

"Süsteemihaldus" on Tartu Ülikooli õppeaine, millest võtavad osa enamasti 2. ja 3. aasta Arvutiteaduse Instituudi erialade tudengid. Aine eesmärk on õpetada tudengitele, kuidas seadistada UNIX süsteemi, seda hallata ning teada saada, kuidas see üldse töötab. Aine raames saab iga üliõpilane endale virtuaalmasina koos Debian operatsioonisüsteemiga, mida ta seadistama hakkab. Aine juhendajate kasutuses on ka testserver, mis kontrollib pidevalt kõiki aktiivseid virtuaalmasinaid, vaatab läbi selle konfiguratsioonifaile ning salvestab üliõpilase progressi. Tudengid saavad kursuse veebilehe kaudu vaadata, millised ülesanded on neil õigesti sooritatud ja millised mitte. Kuna veel paar aastat tagasi oli igal tudengil kindlas praktikumiklassis oma arvuti, mida ta seadistas, siis oli selle administreerimine ka lihtsam, kuid nüüd on võimalik kõigil osalejatel virtuaalmasinad enda arvutisse kopeerida ning kodus praktikume teha. See on aga kaasa toonud probleeme, millega varem kokkupuuted puudusid.



Joonis 1. Konseptuaalne praktikumide süsteem.

Selleks, et aru saada, kui tähtis antud töö aine "Süsteemihaldus" praktikumijuhendajatele on, tuleb esmalt teada, mida praktikumijuhendajad läbi teevad, enne kui üliõpilased üldse praktikumidest osa võtta saavad.

Iga semestri alguses tuleb praktikumijuhendajal laadida Tartu Ülikooli Õppeinfosüsteemist alla nimekiri üliõpilastest, kes on antud ainele registreerinud ning seejärel seadma üles iga tudengi jaoks virtuaalmasin unikaalsete määrajatega, kus tudeng oma praktikumitöid sooritama hakkab.

Unikaalsete määrajate järgi saab praktikumijuhendaja ning testserver aru, millise üliõpilasega tegemist on, kui kontrollitakse üliõpilase edasijõudmist. Kuna esimese kahe nädala jooksul muutub registreerinute arv tihti, siis iga õpilasega uuesti tegelemine ja temale virtuaalmasina genereerimine on aeganõudev, kuid tegelikult automatiseeritav. See funktsioon antud bakalaureusetöö raames tähendaks üliõpilaste lihtsat lisamist ja eemaldamist ainest veebiliidese kaudu.

Juhul kui info registreeritud tudengite kohta on kätte saadud, on vaja kindlaks teha, et iga virtuaalmasin saaks kindla unikaalse IP-aadressi, kuna aines kasutatav testserver, mis kontrollib üliõpilaste edasijõudmist aines, kasutab seda samuti saamaks aru, millise üliõpilase arvutit ta antud hetkel kontrollib. Seega tuleks sellel veebiliidesel ka kindlaks teha, et iga üliõpilase virtuaalsed arvutid omaks unikaalseid määrajaid nii selle suhtes, et kahe üliõpilase IP-aadressid kattuma ei hakkaks kui ka selle, et andmete salvestamisel pärast kursuse lõppu ei tekiks juhtumeid, kus kahe erineva aasta üliõpilaste virtuaalarvutid on eristamatud samasuguste määrajate tõttu.

Järgmise sammuna on vaja need virtuaalmasinad luua ning konfigureerida üliõpilastele kasutamiseks. Sellest lähtuvalt on vaja luua kursuse jaoks LDAP andmete hoiustamiseks ning kasutamiseks, virtuaalmasinad ning need signeerida, arhiveerida ja anda märku ka testserverile, mis kontrollib tudengite edasijõudmist praktikumis, et sellist virtuaalmasinat on nüüdsest vaja ka kontrollida.

Pärast initsialiseerimist on oluline kõige esimene ülesanne, mida iga üliõpilane praktikumi tulles tegema peab – sertifikaadi loomine. Hetkel peab iga üliõpilane endale sertifikaadi looma, seejärel selle praktikumijuhendaja e-mailile saatma ning ootama vastust. Juhendaja osa on oma meilist üles otsida tudengite saadetud sertifikaadid, kontrollima nende õigsust ning siis selle signeerima. Juhul kui sertifikaadis esineb vigu, saadab juhendaja tudengile tagasiside ning jääb ootama üliõpilaselt järgmist signeerimise päringut, kus oleks leitud vead parandatud. See võtab palju aega ning on tegelikult samuti suuresti automatiseeritav veebiliidese kaudu, mis haldaks üliõpilaste saadetud päringuid ja kontrolliks automaatselt sertifikaatide õigusust, võimaldades juhendajatel teha kõiki eelnevaid töid vaid mõne sekundi ning paari hiireklikiga.

Hetkel saadakse küll kõigega hakkama, kuid tegemist on üleliigse vaevaga. Samuti on ebakindel, kuidas saaksid hetkel hakkama aine teised juhendajad, kui testserveri looja Artjom Lind juhendajate rivist lahkuma peaks. Kuna juhendaja on antud programmid kirjutanud ning süsteemi suures osas välja mõelnud, siis tema juhenduseta on uue kursuse alustamine hetkel raskendatud. Vastavad skriptid kõikide ülesannete jaoks on küll olemas, kuid nende kasutust pole kusagil dokumenteeritud ning hetkel kasutab juhendaja neid vastavalt vajadusele sellises järjestuses nagu tarvis. Uue veebiliidese kaudu, mis selle kõigega "ise" hakkama saab, saaks uued juhendajad tunduvalt lihtsamini hakkama ning veebiliidese enda administreerimine ja konfigureerimine oleks vaid ühe inimese töö. Samuti oleks loodud täpsem kord, kuidas need skriptid töötaks ning seetõttu oleks neid ka parem hakata optimeerima, kui nende kasutuse struktuur on täpselt paigas.

Bakalaureusetöö koosneb veebiserveri programmeerimisest keeles Python ning veebilehe kirjutamisest kasutades *HTML*i, *CSS*i ning *Javascripti*.

1. Probleemi püstitus

Kuna hetkel puudub Tartu Ülikooli Arvutiteaduse Instituudi aine "Süsteemihaldus" jaoks korralik keskkond administreerimaks kursust, siis on autori eesmärgiks luua selline keskkond. Selleks, et veebitööriist tehtud saaks, on vaja välja veel mõelda, milliseid tehnoloogiaid ning teeke kasutada, et süsteem oleks võimalikult optimiseeritud nii kiiruse kui ka kasutajamugavuse poolest. Dokumendi eesmärk on selgitada, milline rakendus täpselt välja näeb ning seletab lahti tehnoloogiaid, mida süsteem kasutab.

1.1 Teekide ning keelte valik

Tegemist on veebikeskkonnaga ning seetõttu tuleb kasutusse HTML (*HyperText Markup Language*), CSS (*Cascading Style Sheets*) ning Javascript. Samuti tuleb kasutusele võtta lisateeke, mis teeksid hõlpsaks *back-end* sidumise eelkõige HTMLi ning Javascriptiga.

Süsteem ise kirjutatakse programmeerimiskeeles Python tänu selle lihtsusele, paljudele võimalustele ning sellegi tõttu, et hetkel testserveris jooksvad skriptid on ise kirjutatud just selles keeles. Niimoodi on võimalik kergemini integreerida veebikeskkonda olemasoleva koodiga ilma mingeid vahekihte kasutamata.

Veebiserverite jaoks on Pythonit kasutades mitu head valikut, kuid suurimateks konkurentideks sellel alal on Flask [1] ning Django [2]. Teekide kasutuse näiteid on mitmeid mõlemate teekide jaoks. Djangot kasutavad sellised suuremad veebilehed nagu näiteks Instagram [12], Mozilla [13] ning OpenStack [14]. Kuulsamad Flaski näited on Hyves (kinnipandud Hollandi sotsiaalvõrgustik) ning Disqus [15]. Flaski näited pole küll nii kuulsad kui Django omad, kuid sellel on ka põhjuseid, mis pole seotud nende funktsionaalsuste vahega - nimelt on kergem leida arendajaid, kes on varem tegelenud Djangoga kui et Flaskiga. Kuigi Django tuleb kaasa rohkemate funktsionaalsustega, siis võib seda pidada ka selle miinuseks, kuna kõiki neid funktsionaalsusi ei hakka veebiliides kunagi kasutama. Django on rohkem reeglites kinni ning kõike, mida on võimalik saavutada Djangos, on võimalik saavutada ka Flaskis ning seejures teha seda omamoodi.

Flaski tugev külg on ka see, et selle tööle saamine on võimalik paari koodireaga ning kogu rakenduse struktuuri saab arendaja ise üles ehitada. Seevastu on Djangol tugevalt rakenduse selgroog juba paigas ning selle modifitseerimine on raskendatud ning liigne ajaraiskamine. Alustades uut Django projekti, on vaja käivitada eraldi skript, mis tekitab valmis juba andmestruktuuri, millesse edasi oma koodi hakata kirjutama. Flaski kasutades pole midagi sellist vaja ning projekti saab käima kuue koodireaga kasutamata selleks mingeid skripte.

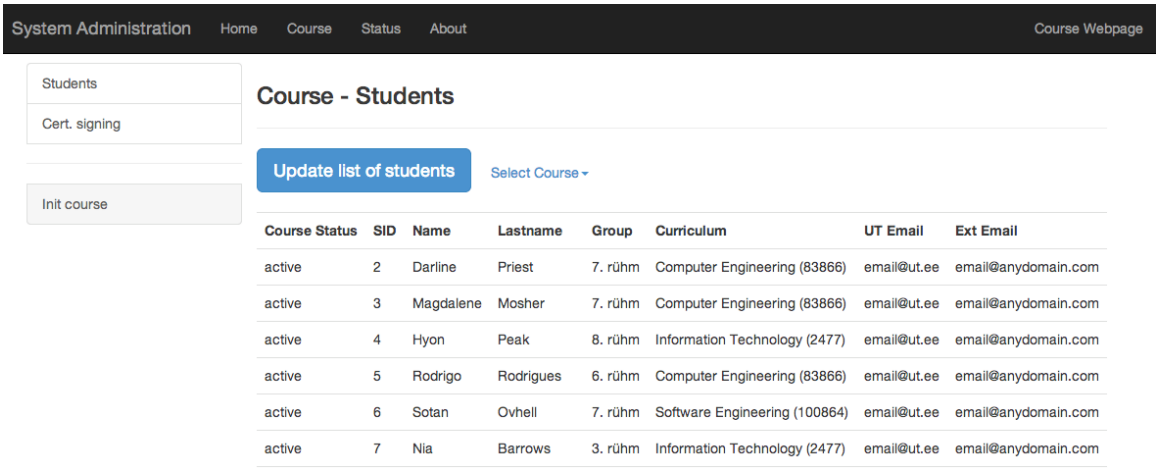
Flaskiga kaasneb ka selline tööriist nagu Jinja2 [3], mille eesmärk on muutujaid otse Pythonist HTMLi edastada. Samuti lubab ta kasutada HTMLis *if-else* tingismuslauseid ning itereerida *for* tsükleid kasutades üle listide. Seega on võimalik näiteks üles ehitada HTML kood, mis genereerib tabelisse täpselt vastava arvu ridu kasutamata selleks javascripti.

Selleks, et ka disaini kiiremini valmis saada, on otsutatud kasutusele võtta selline teek nagu *Bootstrap* [5], mis on kasutusel sellistel lehtedel nagu Twitter [16], WebDesign [17] ning CodeAcademy [18]. Antud tööriist teeb hõlpsaks elementide lisamise oma kodulehele, millele on valmis tehtud juba näiteks mobiilivaated ning Javascript ja CSS animatsioonide jaoks. Kasutusele on võetud sealt ülemine riba menüü jaoks, kus on veebilehe erinevate funktsioonidele ligipääsemiseks menüü, alumine riba, mis sisaldab veebilehe kohta informatsiooni ning erinevaid tabelite ja menüüde kujundusi. Abivahendi võlu peitub selles, et valmis on tehtud mobiilivaade, mille jaoks ei pea eraldi kujundust siis ise tegema hakkama. Bootstrapile on olemas ka alternatiive nagu näiteks Zimit Framework [19] ning Foundation [20], kuid eelneva kogemuse ning selle lihtsa kasutuse tõttu otsustati projekti raames kasutusse võtta just Bootstrap [5].

2. Veebi ülesehitus

2.1 Põhifunktsioonid

Veebileht hakkab täitma peamisi rolle seoses tudengite haldamisega aine raames ning tulevikus ka testserveri haldust. Veebilehel on olemas peamenüü, mille kaks alamharu on kursuste haldamine ning testserveri haldamine. Bakalaureusetöö raames saab valmis suurem osa kursuse administreerimise funktsionaalsustest ning hilisemad arendustööd testserveri jälgimiseks on võimalik serverile pärast lisada. Kuna kasutatav veebiraamistik on väga paindlik, siis pole uute funktsionaalsuste lisamine veebi sugugi aeganõudev ega raske töö. Samuti tasuks ära mainida fakt, et Pythoni puhul on programmeerimiskeele lugemine tunduvalt lihtsam kui mõne teise populaarsema veebikeele nagu näiteks Java või PHP puhul [23] ning seejures kirjutatakse Pythonile hulgaliselt kolmanda osapoole mooduleid [22], mis teeb Pythoni veebiserverid palju paindlikumaks ning skaleeritumaks. Samuti on näiteks PHP-ga seotud palju turvaprobleeme [21].



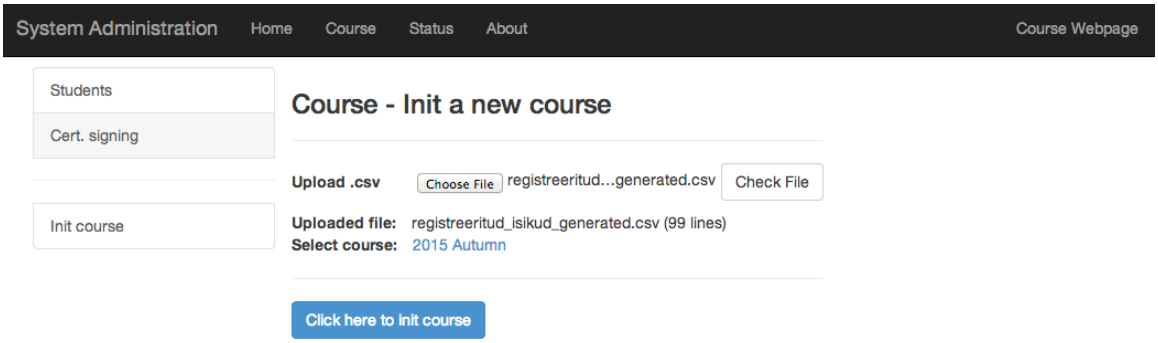
Joonis 2. Kursuse tudengite vaade.

2.2 Funktsioonide täpsemad selgitused

Järgnevates lõikudes selgitatakse, millised on suurimad funktsionaalsused, milleks antud veebiteenust vaja läheks ning kuidas need töötavad nii kasutaja poolt vaadates kui ka tehniliselt.

2.2.1 Uue kursuse loomine

Kui kasutaja soovib uut kursust luua, on võimalik seda teha kursuste haldamise lehelt. Joonisel 2 on näha vasakpoolsest menüüst valikut "Init course", millele vajutades jõuab kasutaja kursuse loomise lehele. Kursust luues palutakse kasutajal üles laadida ÕIS-ist saadud .csv fail, mis kontrollib selle õigsust ning laseb kasutajal edasi minna ainult juhul, kui üles laaditud fail on korrektne. Teine valik, mille kasutaja tegema peab, on kursuse aja valimine, mille valikud server automaatselt pakub olenevalt loomise aastast ja sellest, mis kursused on juba andmebaasis eelnevalt olemas. Joonisel 3 on näha kursuse loomise kuva hetkel, mil vajalik .csv fail on juba üles laaditud ning kursuse aeg valitud. Kui eelnevad kaks asja on tehtud, saab kasutaja vajutada nupule "Click here to init course", mis salvestab uued õpilased andmebaasi.



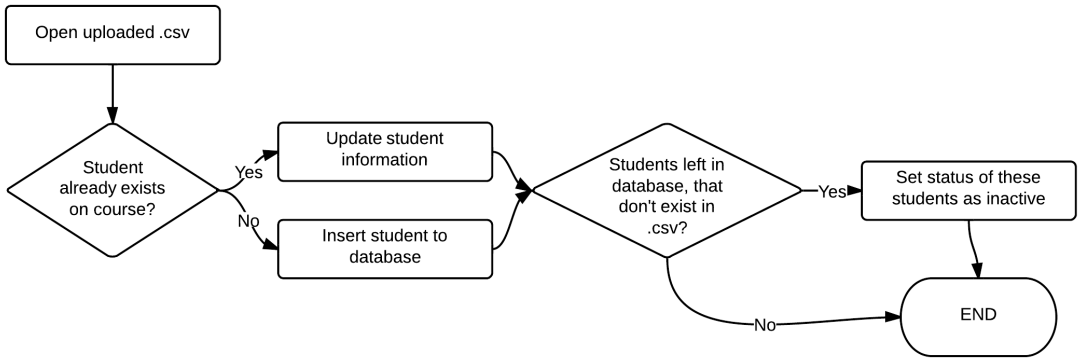
Joonis 3. Kursuse alustamise vaade.

Esmalt, kui jõutakse lehele, vaatab server, milliseid kursuste valikuid näidata kasutajale. Aastas saab toimuda kursus kahel ajal - sügisel ning kevadel. Lehe laadimisel võtab server 4 võimalust - praeguse ning järgmise aasta mõlemad aastajad ning seejärel jätab välja valikud, mis on juba kursusena registreeritud. Seega ei saa luua kaks korda 2014 aasta kevade kursust ning selle kursuse tudengite muutmisvõimalus on olemas kursusevaates, kuhu saab laadida samuti üles .csv faili, mis muudab vastava aja kursuse õpilasi uue faili järgi.

Kui leht on laetud, saab kasutaja üles laadida .csv faili, mis sisaldab kursusele registreeritud tudengite infot Tartu Ülikooli õppeinfosüsteemist ÕIS. Esialgu kontrollib server ainult faililaiendit ning selle õigsuse korral vaatab, mitu rida failis on. Seejärel tagastab *ajax* päringule vastuseks ridade arvu ning annab teada, et faililaiend on õige. Pärast kursuse valimist saab kasutaja vajutada nupule, mis andmed üles laadib. Juhul kui kõik andmed saavad probleemideta üles laaditud, suunab *Javascripti* koodijupp kasutaja kursuste lehele, kus kuvatakse talle automaatselt just loodud kursuse õpilasi.

2.2.2 Kursuse tudengite andmete vaatamine ning uuendamine

Kursuste lehe esimene menüüvalik on "Students", millele vajutades saab kasutaja vaadata kõikide kursuste õpilasi, kes andmebaasis on olemas. Lehele minnes võetakse kõige viimane kursus, mis on andmebaasis olemas ning näidatakse selle kursuse tudengeid. Joonisel 2 on näha ka nuppu "Update list of students", millele vajutades saab kasutaja üles laadida uue .csv faili valitud kursuse jaoks. Juhul kui fail on korrektne, uuendatakse andmebaasi ning kasutaja näeb juba uuendatud tudengite nimekirja.



Joonis 4. Kursuse tudengite uuendamise mudel.

Kui kasutaja laeb üles uue faili, siis kontrollib server esialgu tudengeid, kes failis on ning võrdleb neid andmebaasiga. Juhul kui tudeng on juba antud kursusel olemas, siis uuendatakse selle tudengi andmeid. See võib osutuda kasulikuks, kui tudeng on muutnud oma e-maili ning kursuse juhendaja tahab seda andmebaasis uuendada. Samuti saab server aru, kui .csv failis on puudu mõni tudeng, kes oli varem sellele kursusele registreerinud, kuid on hiljem otsustanud end

nimekirjast eemaldada. Sellisel juhul jätab server tudengi kohta andmed alles ning määrab tema staatuse ebaaktiivseks. Kui failis leidub õpilasi, keda andmebaasis varem sellel kursusel polnud, sisestatakse need andmebaasi.

2.2.3 Sertifikaatide signeerimine

Kui õpilased on kursusele registreerunud, siis esimesed ülesanded, mis neile kursusel antakse, on endale sertifikaatide loomine, mis signeeritakse kursuse juhendaja poolt. Eelnevalt on protsess toimunud nii, et pärast meiliga sertifikaadi saatmist kontrollib juhendaja selle üle, signeerib selle ning saadab tudengile signeeritud versiooni tagasi. Juhul kui midagi on sertifikaadi loomisel valesti tehtud, palutakse tudengil saata uus sertifikaat. Kogu protsessi annab palju lihtsamaks muuta seda automatiseerides.

Veebiserveris töötav skript hakkab kontrollima meili ning otsib sealt tudengite poolt saadetud sertifikaate. Kui sertifikaat on käes, kontrollib server sertifikaadi õigsust**.** Juhul, kui mõne osaga on midagi valesti, märgib server ära, mis osa on sertifikaadil vigane, kuidas see tekkis ning pakub ühe nupuvajutusega tudengile võimaluse meili kaudu sertifikaat uuesti saata, sisaldades sealjuures infot, mis sertifikaadi juures valesti tehtud oli.

Vastavad skriptid sertifikaatide ning meilide kontrollimiseks on praegu juba olemas, kuid need pole automatiseeritud. Antud bakalaureusetöö raames on nende integreerimine skoobist väljas ning sellega tegeleb edasi juhendaja, kellel on vaja skripte integratsiooni jaoks modifitseerida. Veebiliideses on valmis programmeeritud nende kasutuse loogika ning server võimalikult kaugele valmis ehitatud, et nende integratsioon oleks kiire.

Seega on veebirakendust ehitades tarvis need serveriga vaid integreerida. Kuna vastavad skripte modifitseerib kursuse juhendaja pidevalt, siis integreerib ta need skriptid koos vajaminevate muudatustega veebiserverisse ise juurde.

2.3 Veebikeskkonna tulevik

Kuna Flask veebikeskkond on äärmiselt paindlik, siis sellele funktsionaalsuse juurdekirjutamine ei kujune väga aeganõudvaks.

Veebikeskkonnas on valmis loodud juba ka koht testserveri integratsiooniks, et kõik juhendaja ainega seotud ressursid oleksid ühes kohas. Bakalaureusetöö skoobis selle tegemist ei ole, kuid koht selle jaoks on serveris loodud ning selle põhiülesandeks oleks testserveri töö jälgimine ning veateadete haldamine. Kuna skriptid, mida testserver käitab, on kirjutatud Pythonis, on nende haldamine veebiserveri kaudu lihtne. Veebiliidese kaudu annaks lihtsaks teha ka testserveri taaskäivitamise ning tulevikus selle tüüpvigade automaatse paranduse. Näiteks juhtub tihti olukordi, kus mõne testi kontrollimiseks on tarvis kindla käsu väljundit, kuid käsu täitmine annab tagasi veateate. Olemasolevad skriptid ei tunne ära kõikvõimalikke veateateid, mida käsud tagastada võivad ning seetõttu tuleb neid tihti uuendada. Hetkeseisuga sellised tüüpvead võivad testserveri töö halvata ning seetõttu ei näe tudengid testserveri veebiliidesest oma praktikumitulemusi koheselt.

Üheks võimalikuks lisafunktsionaalsuseks oleks andmebaasi vaatamine ning haldamine läbi veebi. Selline võimalus oleks äärmiselt kasulik selleks, et vaadata täpsemalt olemasolevaid andmeid, kuna veebiliidese teised kuvad ei näita kõike infot, mis õpilase kohta olemas on. Võimalus oleks anda sertifikaatide kaudu erinevatele veebiliideste kasutajatele erinevad õigused, mis lubaks neil siis kas andmebaasi ainult vaadata või sinna ka andmeid sisestada. Sellega kaasnevalt võiks kasutajatel olla võimalus oma SQL skripte hoiustada, mida nad sama kuva kaudu lihtsast taaskäivitada saaksid juhul kui tekib selleks tarvidus.

Samuti oleks võimalik koguda nii tudengite kui ka testserveri kohta statistikat, mida tulevaste kursuste ning arenduste jaoks ära kasutada. Selle kogumine oleks ühe keskse süsteemi tõttu lihtne ning tarvis oleks vaid luua selle jaoks andmebaasimudel ning välja mõelda, millist statistikat koguda.

Võimalikke elemente keskkonnale juurde lisada on aga palju - andmebaasi vaatamine/haldamine läbi veebi, testserveri haldamine, statistika kogumine jpm.

2.4 Disain

Disaini pool on suuremalt jaolt tulnud Bootstrapi kasutamisest, mis pakub väga lihtsat menüüde loomist ning ühtset nuppude ja muude elementide disaini. Võimalikult vähe on kasutatud ka erinevaid värvitoone, et kirjusus kasutajat häirima ei hakkaks. Peamised toonid on must, valge ning osad nupud on sinised. Värvivalik on peamiselt tulnud Bootstrapi enda CSSi põhjal, kus värvide sobivus on juba läbi testitud ning seetõttu ei pidanud ise hakkama omavahel sobivaid värve otsima.

Tänu Bootstrapile on automaatselt loodud ka mobiilivaade veebilehele, mis paigutab ümber menüüd ning sisuelemendid selliselt, et kasutaja võimalikult vähe sisse-välja suurendama peaks. Sarnane liides on olemas ka näiteks Tartu Ülikooli meiliserveri veebil [24], mis oleks üpris raske kasutada mobiili peal juhul kui eraldi mobiilivaade puuduks.

3. Arhitektuur

3.1 Tehnoloogiad

3.1.1 Python

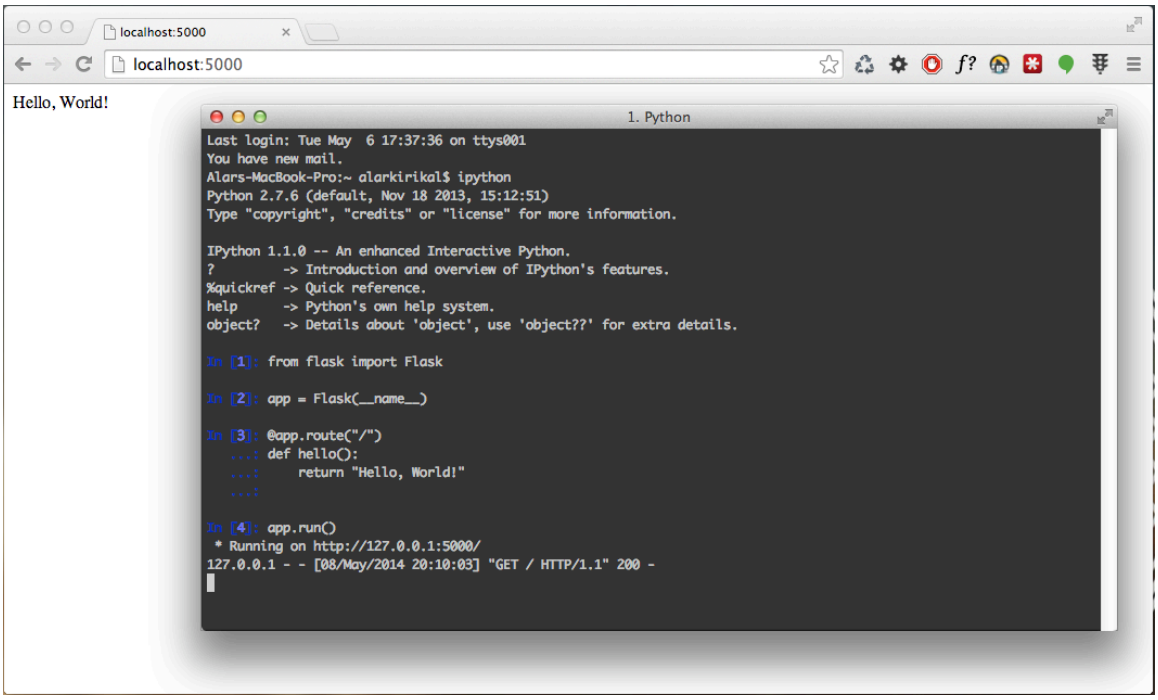
Kogu serveri pool on kirjutatud Pythonis. Otsus sai tehtud enda varasema kogemuse põhjal. Kuigi Java ning PHP on tunduvalt populaarsemad keeled veebiserverite kirjutamiseks, on Pythonil siiski palju eeliseid eelnevate ees. Näiteks on Python tänu selle ebapopulaarsusele ka tunduvalt turvalisem, kuna pahavara pole Python veebiserverite vastu kirjutada nii otstarbekas. Seda jooksutavad ainult 0,2 protsenti kõikidest veebiserveritest maailmas [28]. Populaarsemad valikud veebiserveri keelte seas on Java ja PHP, mille kasutus on vastavalt 2,7 protsenti ning 82 protsenti [29][30]. Samuti oli üheks määravaks teguriks otsustamisel fakt, et kõik eelnevad olemasolevad skriptid on kirjutatud Pythoniga või Bashiga, millede jooksutamine on Pythoniga tunduvalt lihtsam kui Javaga või PHPs.

3.1.2 Flask

Kogu serveri pool on kirjutatud Pythoniga ning selle jaoks on olemas palju erinevaid veebiraamistikke, mis teevad veebiserveri jooksutamise ning kirjutamise Pythoniga palju efektiivsemaks. Kõige populaarsem valik on kindlasti Django, kuid antud projekti raames otsustati Flask kasuks, kuna viimane on väiksem ning kergemini skaleeritav kui Django. Kui Djangol on ka failide struktuur tugevamalt paigas, siis Flask puhul tuleb kõik asjad ise alguses konfigureerida ning seetõttu annab süsteemi üles ehitada oma nägemusele vastavalt. Kui Flask saab jooksutada ka paari koodireaga, siis Django ülesehitamiseks läheb alguses tunduvalt kauem aega. Kuigi Django pakubki vaikimisi palju enamat, siis selle haldamine ja nende lisamine serveri funktsionaalsusesse on palju töömahukam ülesanne, kui selle ise implementeerimine Flaskiga. Seega on paljud samad asjad võimalik väga lihtsalt Flaskiga ise juurde kirjutada, mis teeb Flaskist kergekaalulisema veebiraamistiku, omades ainult sellist funktsionaalsust, mis päriselt ka kasutusse lähevad. Flaskil on olemas ka tugev dokumentatsioon paljude näidetega, tehes arendamise selle raamistiku kaudu hõlpsamaks.

Lisas 3 on näha võrdlust Django ja Flask "Hello World!" rakenduste loomisest.

Flask on ääretult lihtne jooksutada ning seda saab näha jooniselt 6, kui väheste koodiridadega saab "Hello, World!" veebirakenduse juba käima panna.



Joonis 5. Flask [1] kasutamise näide.

Flask ei määra ära ka täpselt kausta- ega failistruktuuri, vaid lubab kõike arendajal ise konfigureerida. Serveris on iga peamenüü alamharude kuvade failid viidud eraldi kaustadesse - *Home*, *Course*, *Status* ja *About*. Täiesti eraldi kaustadest võib leida andmebaasi sätted, serveri tuumafailid (andmebaasi raamistik jne), staatiliselt failid ning *HTML* failid.

Samuti tasub ära mainida, et Django paneb programmeerijat mängima "django reeglite" järgi: asju tuleb üles seadistada ning arendada vastavalt nende raamistikule. Selline lahendus ei ole hea muude lahenduste integreerimiseks, kuna väliste asjade sisestamine projekti võtab palju rohkem aega. Flask puhul sellised piirangud puuduvad.

3.1.3 Jinja2

Jinja2 on kergesti kasutatav mallikeel Python jaoks, mis lubab kergesti luua dünaamilisi veebilehti, tuua serverist üle objekte, üle nende itereerida ning need vastavalt vajadusele kuvada.

```
57 <table class="table" style="max-width:875px;">
58     <!-- Table head -->
59     <tr>
60         <th>Course Status</th>
61         <th>SID</th>
62         <th>Name</th>
63         <th>Lastname</td>
64         <th>Group</td>
65         <th>Curriculum</td>
66         <th>UT Email</td>
67         <th>Ext Email</td>
68     </tr>
69     <!-- Table rows -->
70     {% for student in students %}
71     <tr>
72         <td> {{ student.status }} </td>
73         <td> {{ student.sid }} </td>
74         <td> {{ student.name }} </td>
75         <td> {{ student.lastname }} </td>
76         <td> {{ student.group }} </td>
77         <td> {{ student.curriculum }} </td>
78         <td> {{ student.utemail }} </td>
79         <td> {{ student.extemail }} </td>
80     </tr>
81     {% endfor %}
82 </table>
83 </div>
84
85 {% endblock %}
```

Joonis 6. Jinja2 [3] kasutamise näide.

Joonisel 7 on näha, kuidas on loodud veebilehel kursust võtvate tudengite tabel. Jinja2 lubab kasutada ka raskemaid objekte, nagu näiteks sõnastikke, mille elemente siis välja kutsuda. Joonisel on näha, kuidas käiakse läbi list *students*, mis koosneb *student* sõnastikest, kus iga sõnastik koosneb ühe tudengi informatsioonist. Itereerides üle tudengite listi, saame me luua iga ringiga uue tabelirea ning ükshaaval välja kutsuda õige väärtuse sõnastikust tema võtme järgi. Sellised võimalused teevad koodi lugemise väga lihtsaks ning arusaadavaks juba esmapilgul seda lugedes.

Suur positiivne külg, kasutades Jinja2 mallikeelt, ongi tsüklite kasutamise võimalus, mis lubab dünaamiliselt luua veebilehele sisu. Samuti tuleb kasuks *if-else* kasutamise võimalus, tänu

millele saab kergelt luua näiteks veateateid, mis kirjutatakse HTMLi ainult selle olemasolul, mitte seda peites ning pärast Javascriptiga nähtavale tuues.

Sarnaselt Bootstrapile on ka Jinja2 antud rakenduses kujunduse rollis, aidates kergesti luua *front-end*.

3.1.4 Bootstrap

Bootstrap on *front-end* raamistik, mõeldud lihtsustama disaini loomist veebis ning paneb mõtlema samal ajal ka mobiilivaatele, kuna raamistik pakub automaatselt ka mobiilivaadet. Tava- ning mobiilivaateid veebilehest on näha lisadest 1 ja 2. See koosneb Javascript failist ning kujundusfailist, mis tuleb veebilehega koos laadida. Bootstrap vajab kasutamiseks ka jQuery teeki. Seejärel on võimalik raamistiku elemente juba kasutada.

Bootstrapil on lisaks olemas väga mahukas dokumentatsioon [6], kust leiab suurel hulgal koodinäiteid kõikide olemasolevate elementide kohta, mida kasutada saab.

Arhitektuuriliselt on Bootstrap kujundaja rollis, mis aitab arendajal lihtsasti ning kiiresti valmistada veebileht mõtlemata ülemääre selle kujundusele.

3.1.5 PostgreSQL

PostgreSQL [7] on vabavaraline relatsiooniline andmebaasihaldaja, mis sai projekti valituks selle vabavaralisuse ning hea Python toe tõttu. Kasutades *psycopg2* [8] Pythoni moodulit, on võimalik suhelda otse andmebaasiga tõhusalt ning kiirelt. Moodul on turvaline ning kaitstud ka *SQL Injection* [9] nimelise tehnika eest, mis võimaldab pahatahtlikul kasutajal jooksutada oma kirjutatud koodi veebi andmebaasis. Kuigi *SQL* skript luuakse koodis, täites vastavad väljad *.csv* failist võetud infoga, siis muudetakse info PostgreSQLi jaoks sobivaks andmetüübiks kasutades *psycopg2* moodulit.

```
# Pass data to fill a query placeholders and let Psycopg perform
# the correct conversion (no more SQL injections!)
>>> cur.execute("INSERT INTO test (num, data) VALUES (%s, %s)",
...       (100, "abc'def"))
```

Joonis 7. psycopg2 [8] kasutamise n2ide SQL Injectioni [9] vastu.

Alternatiiviks Postgresile oli MySQL, kuid tänapäeval on PostgreSQL kasutamine levinum, kui MySQL ning seda paljude põhjuste tõttu. Näiteks saab Postgres tabeli veergude nimesid ümber muuta ilma tabelit ümberkirjutamata. Samuti on toodud välja PostgreSQLi parem kiirus. [25][26]

3.2 Andmemudel

```
sysadmin=# \d+ students;
                                Table "public.students"
      Column      |       Type        | Modifiers | Storage  | Stats target | Description
------------------+-------------------+-----------+----------+--------------+-------------
 id               | character varying |           | extended |              |
 study_nr         | character varying |           | extended |              |
 name             | character varying |           | extended |              |
 lastname         | character varying |           | extended |              |
 faculty          | character varying |           | extended |              |
 year             | character varying |           | extended |              |
 study_degree     | character varying |           | extended |              |
 study_curriculum | character varying |           | extended |              |
 group_nr         | character varying |           | extended |              |
 email_ut         | character varying |           | extended |              |
 email_ext        | character varying |           | extended |              |
 course_status    | character varying |           | extended |              |
 course_time      | character varying |           | extended |              |
 student_uid      | character varying |           | extended |              |
 status_mail      | character varying |           | extended |              |
 status_smime     | character varying |           | extended |              |
 status_csr       | character varying |           | extended |              |
 status_certsent  | character varying |           | extended |              |
 username         | character varying |           | extended |              |
Has OIDs: no
```

Joonis 8. "Students" tabeli mudel.

Kuna andmetüübid võivad vastavalt skriptidele ning edasistele otsustele veel muutuda, on kõik andmetüübid seadistatud arenduse ajaks piiramatu pikkusega sõnedeks. Veerud *id, study_nr, name, lastname, faculty, year, study_degree, study_curriculum, group_nr, email_ut* ning *email_ext* tulenevad ÕISi *.csv* failist. Veergude *course_status* ning *course_time* väärtused genereeritakse tudengi objekti loomisel või muudetakse selle muutmisel. Viimased veerud on seotud tudengi sertifikaadi kontrollimisega.

Käesolev andmemudel on veel lahtine ning sinna võib juurde veel lisanduda õpilaste edasijõudmine aines ning muid olulisi väärtusi.

3.3 Esinenud probleemid

Serverit kirjutades tekkisid mitmeid probleeme, mis tasuvad mainimist. Peamine probleemide põhjustaja ning suurim raskus asja sujuvaks töölesaamiseks on tudengite nimed. Nendes esinevad tähed, mis ei kuulu ASCII [10] keelemärkide hulka. Kuigi suurem osa sõnedest, mis ÕIS-i *.csv* failist võetakse, konverteeritakse automaatselt UTF-8 [11] kodeeringule ümber, siis oli probleeme nende sisestamisega andmebaasi ning pärast nende lugemisega.

```
  File "/Users/alarkirikal/Projects/FlaskServer/server/templates/course/base.html", line 1, in top-level template code
    {% extends 'base.html' %}
  File "/Users/alarkirikal/Projects/FlaskServer/server/templates/base.html", line 51, in top-level template code
    {% block content %}
  File "/Users/alarkirikal/Projects/FlaskServer/server/templates/course/cert_sign.html", line 31, in block "content"
    <td> {{ student.name }} </td>
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in position 0: ordinal not in range(128)
```

Joonis 9. ASCII kasutamise tõttu tekkinud veateade.

Suurem osa UTF-8 probleemidest sai lahendatud Pythoniga iga kord sõne lugedes andmebaasist ja/või üleslaaditud *.csv* failist üle defineerides, et antud sõne UTF-8 kodeeringusse üle kirjutataks.

4. Kokkuvõte

Antud bakalaureusetöö käigus arendati välja veebikeskkond Tartu Ülikooli Arvutiteaduste Instituudi aine "Süsteemihaldus" jaoks, mille kaudu saavad kursuse juhendajad ainet õppivaid tudengeid ja selle aine jaoks loodud testserverit hallata.

Töö käigus arendati valmis nii serveri pool kui ka programmeeriti *front-end* selliseks, et edaspidi oleks neid tulevikuvajadusi arvesse võttes kergem edasi arendada. Kuna projekt on suur, siis funktsionaalsus, mis valmis tänu antud bakalaureusetööle, on ainult osa veebikeskkonna tõelisest potentsiaalist. Lõpusirgel on kursuste tudengite haldamise osa, mis sisaldab endas uue kursuse loomist andmebaasis koos Õppeinfosüsteemist saadava .csv faili importimisega. Antud funktsionaalsus võimaldab juhendajal paari liigutusega valmis luua uue kursuse, kus tudengite andmed automaatselt salvestatakse. Samuti saab loodud kursuse tudengite kohta infot uuendada ning vaadata nende protsessi sertifikaatide signeerimise kohta, mis on aine läbimise üks alustalasid.

Veebikeskkond on äärmiselt tähtis selle tõttu, et paljud aine jaoks valminud skriptid ning muud vahendid on kirjutatud aine ühe juhendaja poolt ning tema lahkumise puhul võib tekkida probleeme kursuse läbiviimisega. Kuna uued juhendajad ei oska ning ei tea kuidas kasutada vastavaid vahendeid, mille juhendaja kirjutanud on, siis on olukord kujunenud aja jooksul kursuse jaoks kriitiliseks. Uus veebikeskkond kasutab neidsamu vahendeid ning teeb nende kasutuse kas automaatseks või kõigile arusaadavalt kasutatavaks. Tänu sellele ei pea uued kursuse juhendajad omama nii palju teadmisi teema kohta, mis serveris tegelikult toimub ning saavad juhendamisega hakkama lihtsamini ning kiiremini.

Veebikeskkonna potentsiaal on veel suurem, kuna sinna saab hakata lisama funktsionaalsusi, mille jaoks juhendajatel eelnevalt lihtsalt aega polnud.

5. Viited

[1] Armin Ronacher (Mai 2014) Flask - A Python Microframework [Online] http://flask.pocoo.org/

[2] Django Software Foundation (Mai 2014) The Web framework for perfectionists with deadlines [Online] https://www.djangoproject.com/

[3] Armin Ronacher (Mai 2014) Jinja 2 - The Python Template Engine [Online] http://jinja.pocoo.org/

[4] Armin Ronacher (Mai 2014) Jinja 2 Documentation [Online] http://jinja.pocoo.org/docs/

[5] Mark Otto, Jacob Thornton (Mai 2014) Bootstrap [Online] http://getbootstrap.com/

[6] Mark Otto, Jacob Thornton (Mai 2014) CSS Bootstrap [Online] http://getbootstrap.com/css/

[7] The PostgreSQL Global Development Group (Mai 2014) PostgreSQL [Online] http://www.postgresql.org/

[8] Daniele Varrazzo (Mai 2014) PostgreSQL + Python | Psycopg [Online] http://initd.org/psycopg/

[9] W3Schools (Mai 2014) SQL Injection [Online] http://www.w3schools.com/sql/sql_injection.asp

[10] ASCII Table (Mai 2014) [Online] http://www.asciitable.com/

[11] UTF-8 (Mai 2014) [Online] http://www.utf-8.com/

[12] Instagram (Mai 2014) [Online] http://instagram.com/

[13] Mozilla Community (Mai 2014) Mozilla [Online] http://www.mozilla.org/en-US/

[14] OpenStack (Mai 2014) OpenStack Open Source Cloud Computing Software [Online] https://www.openstack.org/

[15] Disqus, Inc. (Mai 2014) The Web's Community of Communities [Online] http://disqus.com/

[16] Twitter Inc. (Mai 2014) [Online] https://twitter.com/

[17] Envato Pty Ltd. (Mai 2014) Tuts+ Web Design Tutorials [Online] http://webdesign.tutsplus.com/

[18] CodeAcademy (Mai 2014) CodeAcademy: Learn to Code [Online] http://www.codecademy.com/

[19] Jorge Garrido Oval (Mai 2014) Zimit Framework: The consistent HTML5 framework [Online] http://zimit.firezenk.com/

[20] ZURB Inc. (Mai 2014) Foundation: The most advanced responsive front-end framework from ZURB [Online] http://foundation.zurb.com/

[21] SecureReality (Mai 2014) A Study in Scarlet - Exploiting Common Vulnerabilities in PHP Applications [Online] http://old.lwn.net/2001/0704/a/study-in-scarlet.php3

[22] UserfulModules - Python Wiki (Mai 2014) [Online] https://wiki.python.org/moin/UsefulModules

[23] PythonVsPhp - Python Wiki (Mai 2014) [Online] https://wiki.python.org/moin/PythonVsPhp

[24] Tartu Ülikool (Mai 2014) Tartu Ülikooli kirjakast [Online] https://mailhost.ut.ee/mail/

[25] Reddit Inc. (Mai 2014) PostgreSQL or MySQL for Django? [Online] http://www.reddit.com/r/Python/comments/wg1qa/postgresql_or_mysql_for_django/c5dnobn

[26] Reddit Inc. (Mai 2014) PostgreSQL or MySQL for Django? [Online] http://www.reddit.com/r/Python/comments/wg1qa/postgresql_or_mysql_for_django/

[27] Github Inc. (Mai 2014) Github [Online] https://github.com/

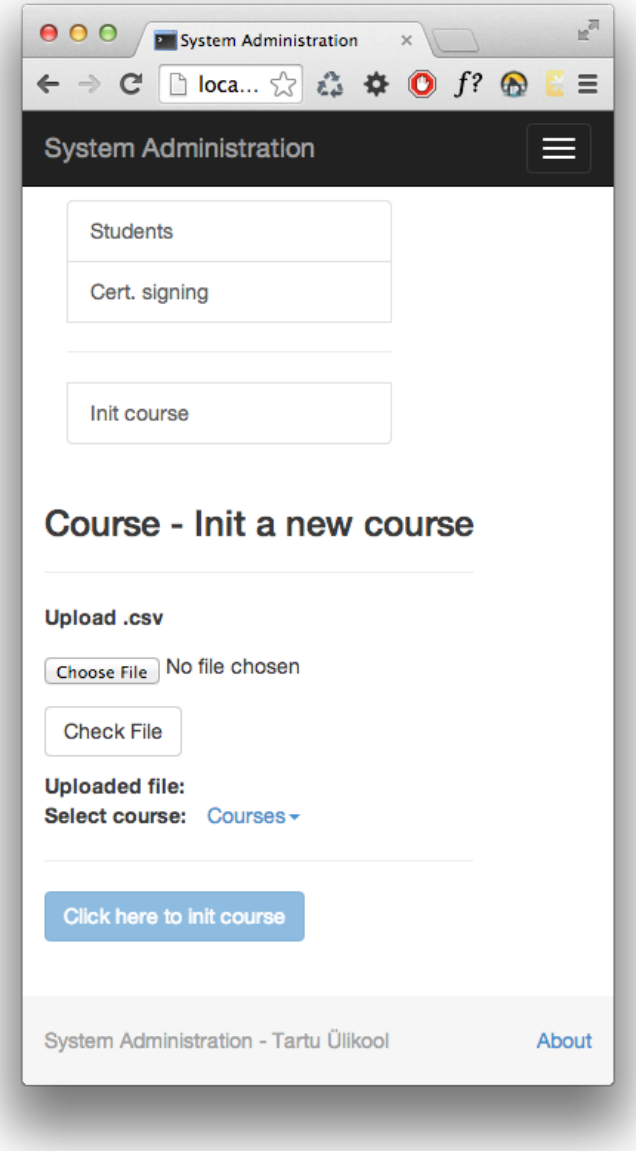
[28] Q-Success (Mai 2014) Usage statistics and market share of Python for websites [Online] http://w3techs.com/technologies/details/pl-python/all/all

[29] Q-Success (Mai 2014) Usage statistics and market share of Java for websites [Online] http://w3techs.com/technologies/details/pl-java/all/all

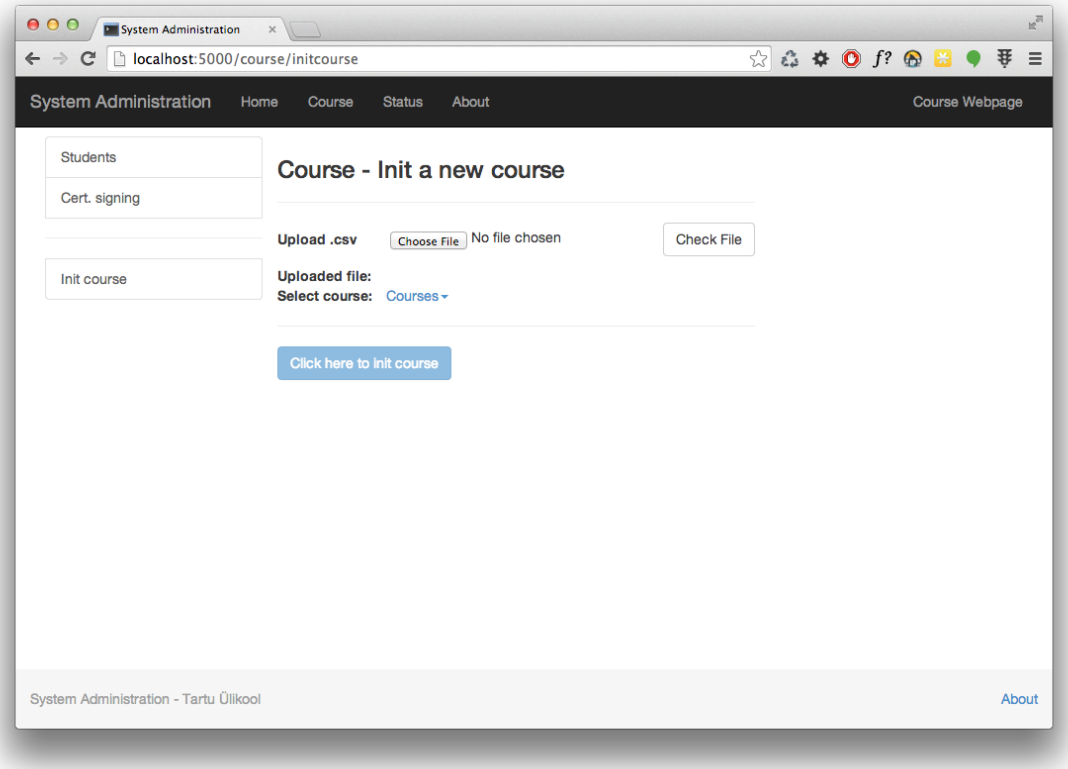
[30] Q-Success (Mai 2014) Usage statistics and market share of PHP for websites [Online] http://w3techs.com/technologies/details/pl-php/all/all

6. Lisad

Lisa 1 - Mobiilivaade



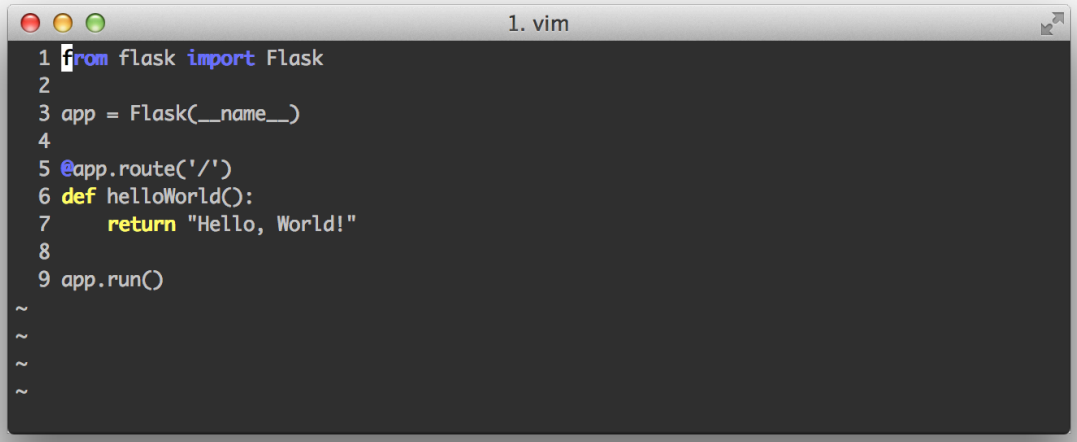
Lisa 2 - Tavavaade



Lisa 3 - Django vs Flask "Hello World!"

3.1 Flask

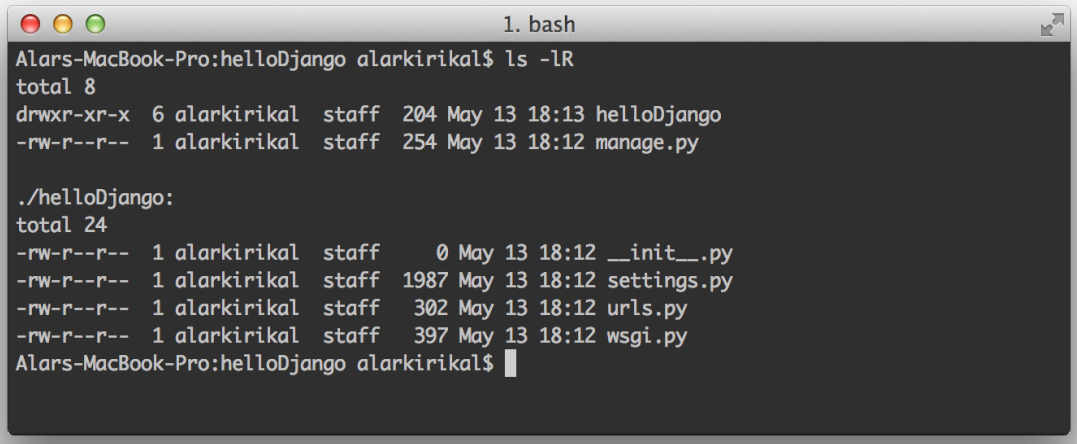
Flask jaoks on tarvis luua üks fail, mille sisu on näha allolevalt jooniselt. Selle käivitamine Pythonis on server juba käima pandud.



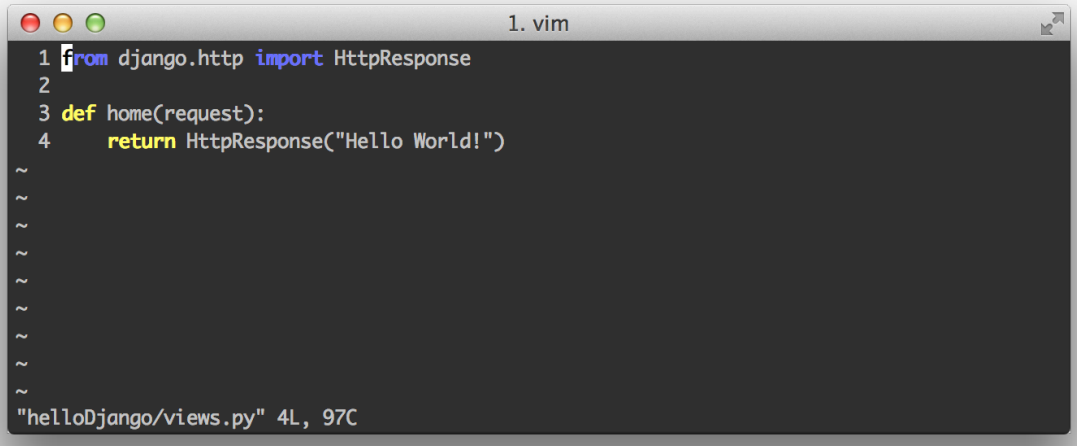
3.2 Django

Django jaoks on esimeseks ülesandeks Django skripti kasutades projekti loomine, mis automaatselt kaustastruktuuri ning vajaminevad Python failid loob.

Alars-MacBook-Pro:Django alarkirikal$ django-admin.py startproject helloDjango



Seejärel on vaja hakata konfigureerima urls.py faili, mis sisaldab endas infot selle kohta, milliseid meetodeid käivitada kindlat URL-i pärides. Lisades faili rea "*url(r'^$', 'helloDjango.views.home', name='home')*" suunab server kodulehe päringud vaatele "home".



Seejärel alles jookseb Django baasil "Hello World" rakendus.

3.3 Kokkuvõte

Nagu eelnevalt näha, läks palju rohkem tööd Django käimasaamiseks võrreldes Flaskiga, mille jaoks oli tarvis mõnda rida koodi ning sama fail lihtsalt käima panna.

Lisa 4 - Veebi kood

Veebiserveri ning *front-end* koodi on võimalik näha GitHubist [27] järgnevalt lingilt: https://github.com/alarkirikal/sysadmin-web

Litsents

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina **Alar Kirikal** (sünnikuupäev: 24.02.1992)

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose **Aine "Süsteemihaldus" õppesüsteemi praktikumijuhendaja veebiliides**, mille juhendaja on Artjom Lind,

   1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

   1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, **5/14/14**