TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Teele Liblik 174894IDDR

Kutsete saatmise automatiseerimine “Tagasi Kooli“ infosüsteemis

Diplomitöö

Juhendaja: Meelis Antoi
Magister

Tallinn 2021

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Teele Liblik

02.01.2021

Annotatsioon

“Tagasi Kooli“ algatuse eesmärgiks on koolide ja ülejäänud ühiskonna üksteisele lähemale toomine. Selle saavutamise keskseks tööriistaks on infosüsteem, kus õpetajad saavad kutsuda väljastpoolt kooli inimesi külla tunde andma ning organisatsioonid saavad kutsuda õpilasi õppekäikudele ja töö- ning tudengivarjuks olema. Küll aga, ei satu inimesed tihtipeale infosüsteemi kutseid otsima, mistõttu kutsete vastuvõtjate leidmine toimub suuresti manuaalselt “Tagasi Kooli“ töötaja poolt. Selline lahendus ei ole jätkusuutlik ega eskaleeritav. Käesoleva töö eesmärgiks on pakkuda kirjeldatud probleemile lahendus profiilide loomise ja kutsete automatiseerimise näol. Tulemuseks on tervikliku lahenduse kirjeldus, varasemate andmete põhjal profiilide tabeli loomine, profiilide selekteerimine vastavalt kutse meetodile ning relevantsuse skoori arvutamine, mille abil leitakse profiilid, keda sisestatud kutse võiks kõige rohkem kõnetada. Töös loodud lahenduse kasutamisel saab tõdeda, et “Tagasi Kooli“ algatusel on aastatega kogunenud laialdane kasutajaskond, kelle hulgast leiab antud funktsionaalsuse läbi potentsiaalseid huvilisi kutsetele.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 30 leheküljel, 5 peatükki, 17 joonist.

Abstract

Automation of Sending Invitations in “Back to School“ Infosystem

“Back to School“ is an initiative with the purpose of bringing society and schools closer to each other. It’s central tool for achieving this is the infosystem. In this infosystem teachers can invite others to give a class and organisations can invite schools to study trips or to shadow their studies or work. The problem here is that people don’t tend to find the invitations from the infosystem organically. Currently this is solved by “Back to School“ employees manually finding people who might be interested in an invitation. This approach is time-consuming and not scalable. The goal of the thesis is to offer a solution to this problem by creating profiles and finding the ones with matching interests. Based on the selection, an invitation is sent to those profiles via email. The solution is described in more detail and parts of it is developed in code.

As “Back to School“ has been an active organisation for many years, there is significant amount of data from the previous infosystem that had the logic of registered users. This thesis takes this forementioned data and creates profiles based on the guest lessons they have participated. The selection of relevant profiles is done by calculating a relevance score for each profile. This score comprises of matching together profiles and invitations, based on their tags, fields and areas. Also taken into account is the net promoter score, which will be based on the feedback from the people they have worked with.

The result is that using the functionality that is described and developed in the current thesis it is possible to effectively use the wide userbase “Back to School“ has accumulated to find the people who are potentially interested in a certain invitation.

The thesis is in Estonian and contains 30 pages of text, 5 chapters, 17 figures.

Lühendite ja mõistete sõnastik

Boolean	Andmetüüp, mille võimalikud väärtused on tõene või väär
CSV	*Comma Separated Values*, komadega eraldatud väärtustega failitüüp
Dataframe	Kahemõõtmeliste indekseeritud andmete manipuleerimiseks kasutatav objekt
Jupyter Notebook	Vabavaraline veebirakendus, kus saab koodi kirjutada
Kasutaja	Inimene, kes kasutab “Tagasi Kooli“ infosüsteemi
Registreeritud kasutaja	Inimene, kes on end registreerinud kasutajaks
Kohtumine	Külalistund, õppekäik, töö- või tudengivarjutamine
Pandas	Andmetöötlus- ja analüüsi teek andmetabelitega manipuleerimiseks
Python	Programmeerimise keel
Regular expression operations	Sõnede manipuleerimiseks mõeldud moodul

Sisukord

Jooniste loetelu

1 Sissejuhatus

Käesolev töö käsitleb probleemi, milleks on asjaolu, et inimesed ei sisusta oma aega "Tagasi Kooli" veebilehel olles ja seetõttu ei jõua sealne koostöö kutsete info nendeni. Täna toimub ülespandud kutsetele vastaspoole leidmine suuresti manuaalselt "Tagasi Kooli" kollektiivi poolt. Selline lahendus on väga kulukas, aeganõudev ning pole eskaleeritav.

"Tagasi Kooli" algatuse eesmärk on tuua koolist väljapoole jäävat teavet õpilastele lähemale. Selle saavutamise keskseks vahendiks on infosüsteem, mis lihtsustab koolidel ja organisatsioonidel koostööettepanekute nähtavakstegemist ja infovahetust. Praegune lahendus eeldab, et koostööst huvitatud inimesed peavad külastama veebilehte, et ülespandud kutseid näha. Käesolev töö pakub sellele probleemile lahenduse, mis võimaldab ülespandud kutsete infol jõuda automaatselt varasemalt algatusest osa võtnud inimesteni, keda kutse võiks huvitada.

2 “Tagasi Kooli” kutsete saatmise lahenduse analüüs

2.1 Visiooni kirjeldus ja skoobi määramine

“Tagasi Kooli“ on kodanikualgatus, mille eesmärk aidata 1.-12. klassi õpilastel õppida 21. sajandi oskusi. Selleks pakub Tagasi Kooli veebipõhist keskkonda näost näkku ja *online*-külalistundide, õppekäikude ning töö- ja tudengivarjutamiste kokku leppimiseks. Tagasi Kooli on avatud kõigile õpetajatele ja noortekeskuste juhendajatele ning koolidega koostööst huvitatud inimestele [1].

“Tagasi Kooli” eelmine infosüsteem oli ülesehitatud loogikale, et huvilised peavad end kõigepealt registreerima kasutajateks, andes sellega märku, mis rollis nad soovivad olla (õpetaja või külalisõpetaja) ning millest, kellele ja kuhu nad on valmis minema rääkima, et toimuks külalistund. Veebilehe kaudu sai vahetada kontaktandmeid, et kohtumine kokku leppida ning seejärel sai toimuvat tundi ka infosüsteemis registreerida. Viimasena mainitu on “Tagasi Kooli” algatuse jaoks väga oluline info, sest just toimunud kohtumiste info kajastab algatuse mõju kõige paremini. Toimunud kohtumiste ja osalenud õpilaste arv ning ulatus maakonniti on peamised mõõdikud, mida algatus enda tulemuslikkuse ilmestamise jaoks saab kasutada.

Eelmise infosüsteemi praktikas ilmnes, et huvilised ei tahtnud võtta aega, et endale veebilehel kasutajaid teha, ununema kippusid sisselogimise detailid ning tundide registreerimine. Seetõttu liiguti uue tööpõhimõtte juurde, mida võiks iseloomustada teadete tahvli põhimõttega. Kasutajaks registreerimine ei ole vajalik, koolid ja organisatsioonid saavad “Tagasi Kooli” veebilehel hõlpsasti üles panna kutse vastavalt oma soovidele, keda, kuhu ja millest rääkima nad ootavad. Huvilised saavad minna veebilehele, näha ülespandud kutseid ja võtta vastu need, mis kõnetavad. Edasine suhtlus toimub juba osaliste vahel, väljapool infosüsteemi, telefoni või meilivahetuse teel. Selline lahendus on tõestanud ennast, kui kasutajatele mugavam ja arusaadavam. Lisaks on väga tõenäoline, et kohtumise toimumisest statistikasse jälg jääb. Tuleb mainida, et uues

infosüsteemis saab olema võimalik end kasutajaks registreerida, et ligi pääseda tasulistele teenustele, mida “Tagasi Kooli“ läbi veebilehe pakkuma hakkab. See lahendus on peamiselt mõeldud organisatsioonidele ja koolidele, kes soovivad hallata suuremaid üritusi ja lihtsustada nende kommunikatsiooni. Edasipidiselt on töös kasutatud mõistet „kasutaja“, kui isik, kes kasutab “Tagasi Kooli“ infosüsteemi. Kui on soovitud viidata registreeritud kasutajatele, siis on see nii täpsustatud.

Kuivõrd e-külalistunnid ja e-õppekäigud korraldatakse ja teavitatakse “Tagasi Kooli” meeskonna poolt eraldi, ei vaatle käesolev töö neid kohtumise vorme. Vaatluse alla jäävad külalistund, õppekäik ning töö- ning tudengivarjutamine. Külalistunde on kahte tüüpi. Külalistund võib olla kohtumine, kus õpetaja (või keegi muu kooli administratiivsest personalist) on veebilehele pannud üles kutse, kus oodatakse kedagi väljastpoolt kooli rääkima märgitud teemast antud tunni raames. Külalistund võib alguse saada ka organisatsiooni initsiatiivil, kus organisatsioon paneb veebilehele üles info, millest nad tahaksid külalistundi andma tulla ning õpetaja saab sellele reageerida ja külla kutsuda. Õppekäigu puhul on kutsujaks organisatsioon, kes tahab õpilastele tutvustada oma ametit või töökohta. Töö- ning tudengivarjutamise puhul sisestab kutse vastavalt töötaja või tudeng, kes kutsub õpilasi end varjutama. Üldistades saab öelda, et “Tagasi Kooli” veebilehte kasutavad kutsujad, kes sisestavad süsteemi kutse ning sellele vastavad kutse vastuvõtjad. Kusjuures, sama inimene saab olla erinevate kutsete puhul erinevas rollis. Toimuva külalistunni, õppekäigu, töö- või tudengivarjutamise puhul kasutatakse töös lihtsustamiseks nimetust “kohtumine“.

“Tagasi Kooli” on hetkel disainimas endale uut infosüsteemi, mis jätkaks praegust loogikat, kus huvilised saavad panna üles kutseid koostööle, need on kõigile nähtavad ja neid saab veebilehel vastu võtta. Vajadus uue infosüsteemi järele on peamiselt seetõttu, et olemasoleval esineb erineva raskusastmega kasutajamugavuse probleeme (lehe ülesehituse loogikast kuni erinevate väljade funktsionaalsusteni) ning soovitakse lahendada probleemi, kus ülespandud kutsed leiaksid kiiremini ja suurema tõenäosusega kutse vastuvõtja. On näha, et kutseid sisestatakse veebilehele, kuid aktiivsus veebilehel kutseid vaatamas ja vastu võtmas käia, on madal.

“Tagasi Kooli” algatus on aastast 2012 pakkunud infosüsteemi koolidele ja õpilaste arendamise huvilistele koostööks [1]. Selle aja jooksul kogunenud kontakte kasutab täna “Tagasi Kooli” meeskond, et veebilehel üles pandud kutsetele leida koostööpartnereid.

“Tagasi Kooli” töötaja saadab meili teel edasi üleskutse, et inimene, kes varasemalt antud teemas ja piirkonnas on koostööd teinud läheks vaatama ülespandud kutseid. Kirjeldatud protsess on kulukas ning ei ole eskaleeritav. Seetõttu soovib “Tagasi Kooli” meeskond, et uue veebilehega kaasneks lahendus, mis leiaks varasemalt osalenud inimeste info põhjal üles huvilised, keda konkreetsed veebilehele lisatud kutsed võiksid kõnetada ning saadaks kutse automaatselt neile e-postkasti.

Funktsionaalsed nõuded:

- potentsiaalsete huviliste selekteerimise peamine lähtepunkt on kutses sisestatud märksõnad, kuna need kirjeldavad kohtumise sisu kõige täpsemini. Märksõnade all peetakse silmas kõiki neid kohtumise teemaga seonduvaid välju, mida kasutaja sisestab käsitsi (näiteks amet, õppekäigu pealkiri, külalistunni märksõnad). Seejärel tuleks lähtuda töövaldkonnast või pädevusest, mis on etteantud valikutega väljad.
- Potentsiaalsete huviliste selekteerimisel tuleb lähtuda eeldusest, et inimesed on nõus kokkusaamiseks liikuma teatud raadiuse ulatuses. Lähtume esialgu, et ollakse valmis liikuma maakonna piires.
- Potentsiaalsete huviliste selekteerimisel tuleks arvesse võtta soovitusindeksit (*Net Promoter Score*), mis tekib varasemate kohtumiste tagasiside põhjal koostööpartnerite poolt [2].
- Külalistunni kutset, mille on sisestanud õpetaja, tuleb soovitada külalisõpetajatele.
- Külalistunni kutset, mille on sisestanud organisatsioon, tuleb soovitada õpetajatele.
- Õppekäigu kutse tuleb soovitada õpetajatele.
- Töö- ja tudengivarju kutsed tuleb soovitada õpetajatele, kes on ka klassijuhatajad.
- Kui mõni inimene on nii õpetaja, kui külalisõpetaja rollis, siis peab arvesse võtma, et huvid erinevas rollis, on erinevad.
- Tulevikus võib tekkida soov huviliste selekteerimisel võtta arvesse asjaolu, mis aine õpetajaga on tegemist.
- Potentsiaalsete huviliste selekteerimisel tuleks arvesse võtta, et on optimaalne kogus inimesi, kellele tuleks kutse saata. Kui saata liiga väikesele grupile, võib juhtuda, et kutse ei kõneta kedagi. Kui saata liiga paljudele inimestele, võib

juhtuda, et soovijaid on mitu ning kuna kutse saab vastu võtta ainult üks isik, võivad ülejäänud tunda end halvasti.

- Andmestik peab täienema vastavalt infole, mida kasutaja annab kutsete sisestamisel, broneerimisel või tellimisel.
- Kasutaja saab märku anda, kui ta ei soovi enam, et "Tagasi Kooli" talle kutseid soovitab.
- „Tagasi Kooli" meeskonnal peab olema võimalik määrata kasutajaid, kellele kutseid ei soovitata.
- Kasutaja saab märku anda, et soovib jälle saada kutsete soovitusi, kui ta on eelnevalt palunud endale kutseid mitte soovitada.
- Kasutaja võib pöörduda „Tagasi Kooli" poole sooviga, et kõik temaga seonduvad andmed kustutataks ja see peab olema võimalik.
- Peab olema võimalik mõõta lahenduse efektiivsust.

2.2 Erinevad võimalused probleemi lahendamiseks

Üks võimalus, oleks vaadata toimunud kohtumisi, püüda leida sealt võimalikult sarnased kohtumised ning saata kutse kohtumisel osalenud inimesele, arvesse võttes tema soovitusindeksit. Sellise lahenduse tulemusel selekteerime välja need inimesed, kes on osalenud varasemalt võimalikult sarnasel kohtumisel. Antud lahendust oleks üpris lihtne luua ja see koosneks peamiselt kahe tabeli väljade võrdlemises. Ei oleks tarvidust andmebaasis muudatusi teha. Arvestades seda, et kasutajad viivad läbi kohtumisi väga erinevatel teemadel erinevates paikades, siis kitsendaksime oluliselt huviliste hulka. Näiteks, ei jääks sõelale inimene, kes on küll kutse piirkonnas kohtumisel osalenud, kuid antud teemaga seonduvalt osalenud mujal piirkonnas toimunud kohtumisel. Lisaks jääksid tahaplaanile need inimesed, kelle varasematele kohtumistele pole antud tagasisidet ning nende soovitusindeks puudub.

Teiseks lahendusviisiks, oleks luua kasutajate profiilid varasemalt toimunud kohtumiste põhjal, mida saab täiendada infoga, mida kasutaja sisestab kutset sisestades, broneerides või tellides.

Seda lahendust toetab asjaolu, et uues infosüsteemis saavad olema mitmed muutujad, mis on ühe konkreetse kasutajaga seotud, olenemata sellest, et kasutajad valdavalt ei registreeri end. Profiilide tabel annaks hea koha selliste andmete hoidmiseks nagu soovitusindeks ja asjaolu, kas soovitused on konkreetsele inimesele lubatud. Lisaks peegeldab antud lahendus paremini inimeste huvide muutust ja valmisolekut erinevates piirkondades kohtumistele. Selline lahendus võimaldab „Tagasi Kooli“ organisatsioonil saada paremat ülevaadet infosüsteemi kasutajatest ning on hea sisend turundus- ja teavitussõnumite täpsemaks edastamiseks.

Profiilide valikul võib kasutada mitmeid erinevaid lähenemisi. Üks võimalus oleks leida profiilid, millel leiduvad sisestatud kutse märksõnad, valdkond ja piirkond ning nad järjestada soovitusindeksi alusel. Kuna selline lahendus nõuaks väga täpset vastet, siis joonistuks välja väga väike hulk profiile. Seepärast, võiks kasutada lähenemist, et näiteks kaks kolmest muutujast peavad klappima või isegi üks kolmest, kui sõelale jääb vähe profiile.

Märksõnade väljad täidavad kasutajad käsitsi ja tavapäraselt sisestatakse mitu märksõna, mis tähendab, et kohtumiste teemade ring on väga lai. Toimuv kohtumine võib puudutada ka mitmeid erinevaid valdkondi, näiteks õpetaja valib kutse valdkonnaks „Infotehnoloogia“ ja „Avalik sektor“ ja külalistundi tuleb andma inimene, kes töötab avalikus sektoris infotehnoloogia alal, kuid varasemalt on kutsega kaasas olnud ainult „Avaliku sektori“ valdkond. Või, näiteks inimene, kes on valmis minema külalistundi andma üle Eesti, pole varasemalt seda teinud Võrumaal, mistõttu ta ei pruugi väga jäikades sobivuse otsimise tingimustes sõelale jääda, kuigi oleks külalistundi antud teemast huvitatud andma.

Profiilide valikul võib lähtuda ka lahendusest, kus arvutame igale profiilile relevantsuse skoori, mis näitaks, kui suure tõenäosusega konkreetne kutse inimest huvitaks. Skoori arvutamine võimaldab kaasata väga mitmeid erinevaid muutujaid selliselt, et saame anda neile kaalu. Selline lahendus annab vabaduse võtta hõlpsalt arvesse paljusid erinevaid kohtumisega seonduvaid asjaolusid ning võimaldab laiemal spektril profiilidel jääda sõelale, kes ühte või teistpidi võiksid olla kutsest huvitatud.

2.3 Kriteeriumid, mille alusel lahendus valida

- Uue infosüsteemi arendamine on alles disainifaasis, seega lahendus peab suutma võimalikult autonoomselt hakkama saada ja seda peab olema lihtne hiljem integreerida.
- Lahendus peab olema hea alus, mida täiendada ja edasi arendada vastavalt sellele, kuidas inimesed „Tagasi Kooli“ infosüsteemi kasutavad.

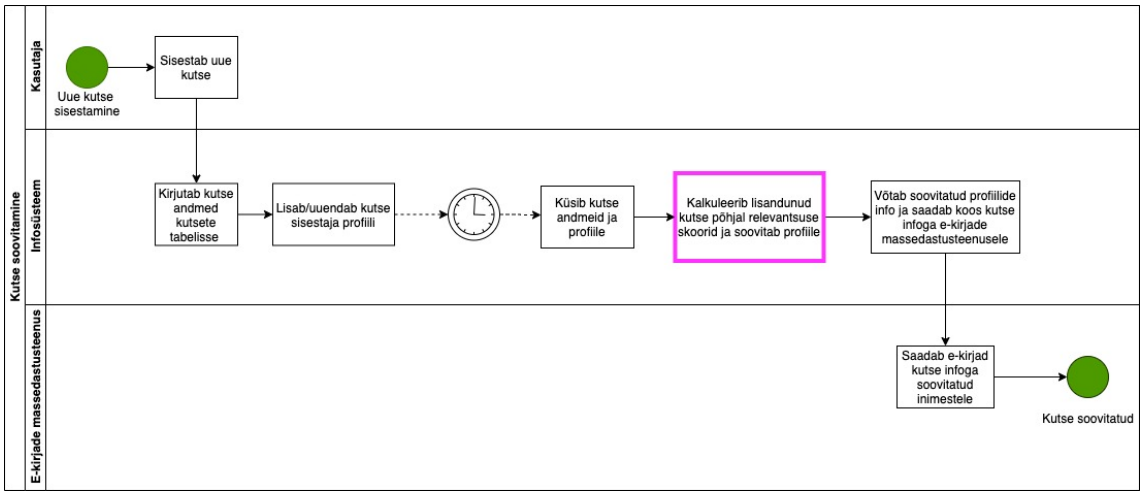
2.4 Valitud lahendus

Valitud lahenduseks osutus profiilide tabeli ja relevantsuse skoori kasutamine huviliste valikul. See lahendus annab hea aluse, mille peale kirjutatud soovituse loogikat saab hõlpsasti edasi arendada vastavalt „Tagasi Kooli“ soovidele ja kasutajate käitumisele.

Profiilide kasutamise puhul, saame mugavalt kõiki kasutajaga seonduvaid atribuute hoida ühes kohas. Kui otsiksime kattuvusi otse toimunud kohtumiste tabelist, siis peaksime küsima infot mitmest tabelist kokku, kuna sellised muutujad, nagu soovitusindeks ja asjaolu, kas kasutajale võib soovitusi saata, ei oleks tõenäoliselt samas tabelis. Suuremate andmemahtude puhul pole see kõige mõistlikum teguviis ja ei võimalda suurt paindlikust lisada muutujaid huviliste leidmisel. Lisaks, annab see hea põhja kasutajate kontode loomiseks, kui kunagi peaks „Tagasi Kooli“ otsustama pöörduda tagasi registreeritud kasutajate loogika juurde.

Kuna uue infosüsteemi arhitektuur pole veel käesoleva töö valmimisel otsustatud, siis oleks riskantne luua lahendust, mis toetub tabelile, mille ülesehitus ja käitumine pole kokku lepitud. Kui kasutame profiilide tabelit, siis lähtume vähematest eeldustest tulevase arhitektuuri osas.

Alloleval joonisel 1 on kujutatud kutse soovitamise töökäik valitud lahendusega. Antud töö raames arendatakse roosades ristkülikutes olevad lahenduse osad, luuakse profiilide tabel ning tuuakse välja olulised punktid profiilide tabeli käitumises, mida infosüsteemi loomisel tuleb arvesse võtta, et lahendus töötaks oodatava efektiivsusega.



Joonis 1. Kutse soovitamise töökäik.

Käesolev töö teostab lahenduse järgnevad osad:

- olemasolevate andmete põhjal profiilide loomine ja nende käitumise analüüs.
- Lahenduse loomine, mille sisendiks on kutse info ja väljundiks kutsest enim huvitatud profiilid.

Käesolevas töös on kasutatud *Python*'i programmeerimise keelt, kuna seda peetakse üheks parimaks keeleks, mida andmeanalüüsi puhul kasutada [3, 4]. Keskseks tööriistaks koodi kirjutamisel osutus *Pandas* andmetöötlus- ja analüüsi teek, mis võimaldab kasutada andmetabelite manipuleerimiseks mõeldud operatsioone ja andmestruktuure [5]. Lisaks kasutatakse *Regular expression operations* moodulit, mida kasutatakse sõnede võrdlemiseks [6]. Valitud arenduskeskkonnaks osutus *Jupyter Notebook* oma lihtsuse poolest. „Tagasi Kooli" soovil on tabelid, nende nimetused ja muutujate nimetused koodis inglise keeles.

3 Kutse soovitamise lahenduse teostamine

3.1 Profiilide tabel ja selle käitumine

Tabeli atribuudid:

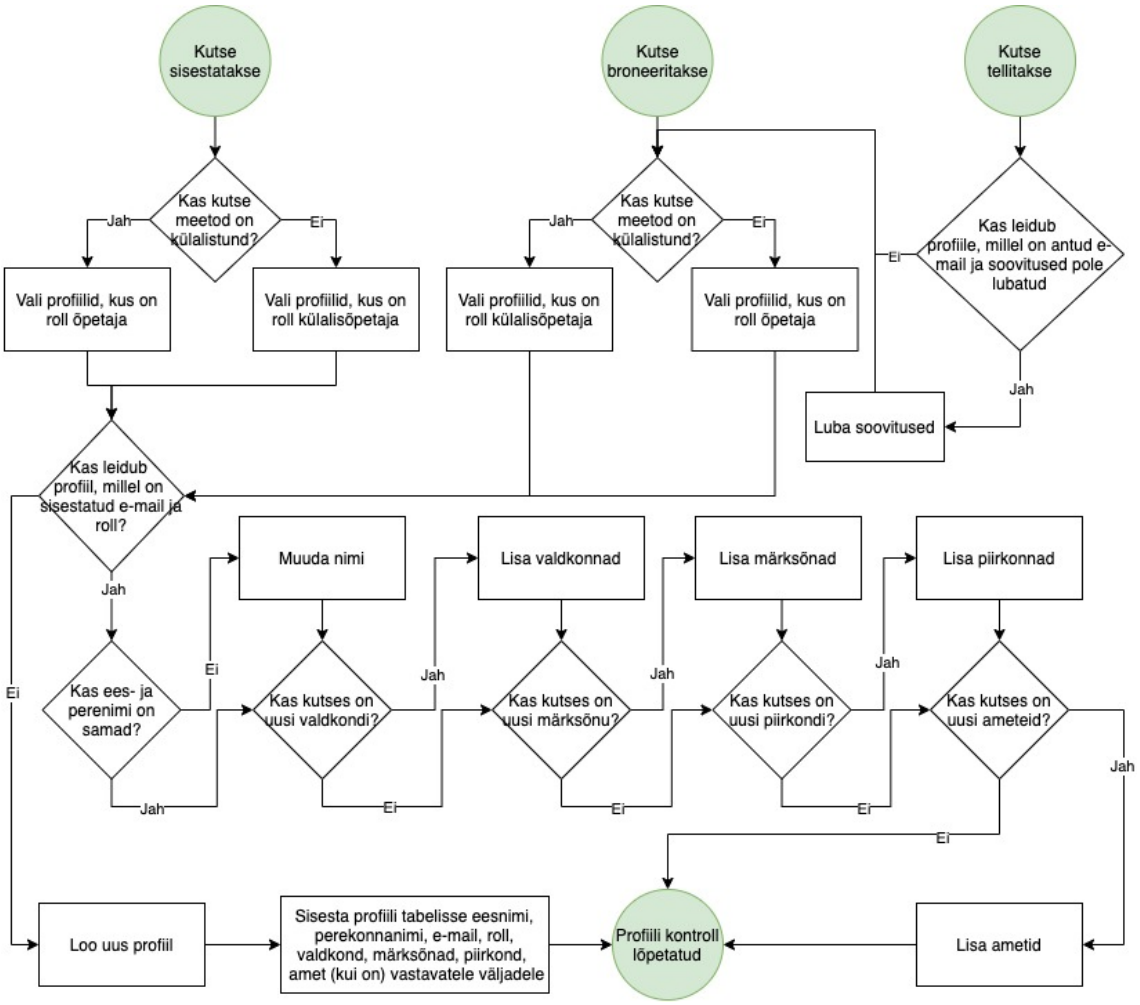
- *profile_id* ehk profiili ID sisaldab inkrementaalset välja, mis suureneb ühe võrra iga tabelisse kantud profiiliga ning on unikaalne ja täisarvu tüüpi.
- *first_name, family_name* sisaldavad kasutaja eesnime(sid) ja perekonnanime(sid). Need väljad ei saa tühjaks jääda ja sisaldavad sõne tüüpi andmeid.
- *email* sisaldab kasutaja e-posti aadressi. See väli ei saa olla tühi ja sisaldab sõne tüüpi andmeid.
- *role* ehk roll näitab, kas antud profiil on rollis külalisõpetaja (*guest_teacher*) või õpetaja (*teacher*). See väli ei saa olla tühi. Ühe e-maili kohta võib olla kuni kaks rolli (seega kaks profiili), mis annab meile teada, et inimene võib olla nii õpetaja, kui külalisõpetaja. Selline eristus on oluline, kuna teemad, mis inimest ühes rollis huvitavad, ei pruugi huvitada teises rollis. Andmetüüp on käesoleva töö raames valitud sõne, kuid lõplik tüüp jääb uue infosüsteemi arhitektide otsustada.
- *field* ehk valdkond sisaldab kohtumise teemaga seonduvaid välju, mida kasutaja valib etteantud valikust. Uue infosüsteemi senise analüüsi tulemusel on selgunud, et siia koguneksid kohtumise valdkonnad, pädevused, aga ka tudengite erialad. See väli ei saa olla tühi, aga võib sisaldada mitmeid väärtusi, mis kõik on sõne tüüpi ning hoitakse listina.
- *tag* ehk märksõna sisaldab kohtumise teemaga seotud väljasid, mille kasutaja on käsitsi täitnud. Siia kogunevad sõne tüüpi andmed, mida hoitakse listina. Kuna infosüsteemile tervikuna pole analüüs veel valminud, ei ole teada kõik kohad veebilehelt, kust siia info tuleb, kuid kindlasti tulevad väärtused märksõnade väljadest, mida täidetakse kutse sisestamisel, aga ka broneerimisel ja tellimisel. Lisaks, amet, tudengivarju puhul spetsialiseerumine ja õppekäigu pealkiri. See väli võib olla tühi.
- *occupation* ehk ameti atribuut hakkab hoidma esialgu infot, kas tegemist on klassijuhataja ja/või aineõpetajaga. Andmetüüp on käesoleva töö raames valitud

sõne, kuid lõplik tüüp jääb uue infosüsteemi arhitektide otsustada.. See väli võib olla tühi, sest esialgu külalisõpetajate ja organisatsioonide käest seda ei küsita.

- *area* ehk piirkond sisaldab sõne tüüpi andmeid, milleks on Eesti maakonnad. Andmeid hoitakse listina. Andmetüüp jääb uue infosüsteemi arhitektide otsustada. See väli ei saa olla tühi, kuid võib sisaldada mitmeid väärtusi.
- *recommendations_allowed* ehk soovitused lubatud sisaldab boolean väärtust, mis on TRUE, kui soovitused on lubatud, vastasel juhul FALSE. See väli ei saa olla tühi ja profiili loomisel on see automaatselt TRUE.
- *net_promoter_score* ehk soovitusindeks on täisarvu tüüpi väli võimalike väärtustega -100 kuni 100 ning tekib kohtumisejärgselt kogutud tagasiside põhjal, kus vastaspooled hindavad, millise tõenäosusega nad soovitaksid antud inimesega koostööd. Indeks leitakse 9 ja 10 palli andnute osakaalust lahutades nende osakaal, kes andsid tagasisideks 1–6 palli. See väli võib olla tühi.

Tuleb märkida, et infosüsteemi analüüsi käigus võib ilmneda, et antud tabel vajab lisavälju või muid muudatusi.

Oluline on, et profiilide tabel täieneks ja uueneks alati, kui kasutaja infosüsteemile kasutatavat infot edastab. Senise infosüsteemi analüüsi käigus on ilmnenud kolm tegevust, mille käigus saame profiilide tabelit täiendada: kutse sisestamine, broneerimine ja tellimine. Nende tegevuste toimumisel peab infosüsteem kasutaja profiili lisama või uuendama. Selleks, et kontrollida profiili olemasolu, tuleks kasutada e-posti aadressi ja rolli kombinatsiooni, kuna see on ainuke unikaalne kasutajat iseloomustav kombinatsioon tabelis. Sellise lahenduse puhul muutuvad kahjuks need profiilid kasutamatuks, kus inimene on vahetanud e-posti aadressi, kuid olukorras, kus kasutajad end ei registreeri, ei saa me profiile identifitseerida. Joonisel 2 on kujutatud profiili kindlakstegemise, lisamise ja uuendamise loogika.



Joonis 2. Profiili lisamise ja uuendamise loogika.

Soovitatud kutsete e-kirjas on võimalus kasutajal infosüsteemile märku anda, et ta ei soovi enam kutseid saada. Kui kasutaja seda kasutab, tuleb leida tema profiil ning märkida *recommendations_allowed* väli väärtusega FALSE. Vastavalt andmekaitse seadustele, võib kasutaja „Tagasi Kooli“ poole pöörduda ning paluda enda andmed infosüsteemist eemaldada. Sellisel juhul tuleb paluda kasutajal saata oma soov enda isikliku e-posti aadressilt, misjärel tuleb tagada, et antud e-posti aadressiga profiilide kirjed saavad samuti kustutatud.

Kuivõrd inimeste e-posti aadressid vahetuvad aja jooksul, tuleks hoia profiilide tabel ajakohane. See tähendab, et infosüsteem peab võtma vastu e-kirjade massedastusteenuse teavituse, kui mõnele e-posti aadressile ei jõua kirjad kohale ning uuendama vastavalt profiilide tabelit. Sellisel juhul, võib näiteks kasutada *recommendations_allowed* välja ning panna selle väärtuseks FALSE.

3.2 Profiilide loomine

Profiilide algandmeteks võetakse tabelid eelnevas "Tagasi Kooli" infosüsteemis registreeritud kasutajate kontaktandmete ja nende poolt aastatel 2012–2018 registreeritud külalistundide infoga. Registreeritud külalistundide tabelis on toodud iga tunni kohta muuhulgas ka osalenud õpetaja ja külalisõpetaja nimi, tunni valdkond, teema ning kooli aadress. Tunni valdkonnast saab sisendi profiili valdkondade väärtustele, tunni teemast kujuneb sisend profiili märksõnade väärtustesse. Kooli aadressist saab profiili piirkonna väärtusteks maakonna. Mainitud tabelis ei ole kontaktandmeid, mistõttu tuleb kasutada veel teist tabelit, kus on registreeritud kasutajate nimi, roll (külalisõpetaja või õpetaja), e-posti aadress jne. Järgnevalt on kirjeldatud võtmekohti profiilide loomise lahenduskäigul.

Eelmainitud tabelid laaditakse CSV(*Comma Separated Values*) formaadis *DataFrame* objektina. CSV on formaat, kus väärtused on eraldatud komadega [7]. *DataFrame* on objekt, mida saab kasutada kahemõõtmeliste indekseeritud andmete manipuleerimisel [8].

Järgnevalt grupeeritakse kõigi külalistundide info vastavalt tunnis osalenud külalisõpetaja ja seejärel õpetaja järgi [9]. Selliselt saame iga profiili täiendada kõigi valdkondade, märksõnade ja piirkondadega, millega seonduvates külalistundides kasutaja varem osalenud on. Mõlemale grupeeringule lisatakse rolli veerg ning antakse vastavalt kas külalisõpetaja või õpetaja väärtus [10]. Seejärel tõstetakse kaks *DataFrame*'i kokku. Alloleval joonisel 3 on näha nende toimingute kood [11].

```
guest_teachersdf = guest_lessonsdf.groupby(["Osalev_KÕ"]).agg(lambda
col: ",".join(col))
guest_teachersdf["role"] = "guest_teacher"
teachersdf = guest_lessonsdf.groupby(["Osalev_Õ"]).agg(lambda col:
",".join(col))
teachersdf["role"] = "teacher"
concatenateddf = pd.concat([guest_teachersdf, teachersdf])
```

Joonis 3. Andmete grupeerimine rolli järgi ja rolli lisamine.

Loodud *DataFrame*'is muudetakse aadressi, tunni valdkonna ning teema andmed väikeste tähtedega sõne-tüüpi listideks ning eemaldatakse kordused. Kuna aadress on esitatud kujul "Koidu 97,Tallinn linn (Harju maakond)", siis esmalt otsitakse kõik sulgudes olevad sõned ning peale korduste eemaldamist, kaotatakse sulud. Joonisel 4 on toodud näitena aadressi andmeid manipuleeriv kood [12, 13, 14, 15, 16, 17, 18].

```
profiles_raw_renamedf["address_raw"] =
profiles_raw_renamedf["Aadress"].astype(str)
profiles_raw_renamedf["address_lowercase"] =
profiles_raw_renamedf["address_raw"].str.lower()
profiles_raw_renamedf["address_area"] =
profiles_raw_renamedf.address_lowercase.apply(lambda x:
re.findall('\(.*?\)',x))
profiles_raw_renamedf["address_area_unique"] =
profiles_raw_renamedf.address_area.apply(lambda x:
list(dict.fromkeys(x)))
profiles_raw_renamedf["area_string"] =
profiles_raw_renamedf.address_area_unique.apply(lambda x:'
'.join(map(str, x)))
profiles_raw_renamedf["area_no_parenthesis"] =
profiles_raw_renamedf.area_string.apply(lambda x: x.replace('(',
'').replace(')', ''))
profiles_raw_renamedf["area"] =
profiles_raw_renamedf.area_no_parenthesis.apply(lambda x:
x.split(","))
```

Joonis 4. Aadressi andmete muutmine väikeste tähtedega unikaalsete sõne väärtuste listiks.

Järgmiseks valmistatakse ette kasutajate kontaktandmete tabel, et saaks andmed eelnevalt loodud profiilide infoga kokku tõsta. Muuhulgas muudetakse kasutajate märgitud roll profiilide *Dataframe*'i rolliga samasuguseks. See on oluline, kuna kahe tabeli ühendamisel kasutatakse võtmeveergudena just nime ja rolli. Joonisel 5 on toodud kasutajate *Dataframe*'i rolli muutmise kood.

```
def role_transformer(role):
    if role == ("Õ"):
        return "teacher"
    elif role == "KÕ":
        return "guest_teacher"
usersdf["role"] = usersdf.Roll.apply(lambda role:
role_transformer(role))
```

Joonis 5. Kasutajate rolli väärtuse kohandamise kood.

Joonisel 6 on toodud kood, millega profiilide ja kasutajate kontaktandmete *Dataframe*'id ühendatakse üheks objektiks [19, 20]. Oluline on ühendada nii nime, kui rolli järgi, kuna üks inimene võib olla mitmes erinevas rollis ja tema huvid neis rollides võivad erineda.

```
profilesdf = pd.merge(profiles_raw_renamedf, usersdf, on=["name",
"role"])
```

Joonis 6. Profiilide ja kasutajate tabeli ühendamise kood.

Viimaks lisatakse *DataFrame*'ile puuduolevad veerud vastavalt peatükis 3.1 kirjeldatud struktuurile [10]. Ameti ja soovitusindeksi väljad märgitakse väärtusega "NaN", andmaks märku, et need ei ole kohaldatavad. Kutsete soovitused on kõigile loodud profiilidele võimalikud, kuna need inimesed on nõustunud teavituse lubavate kasutajatingimustega

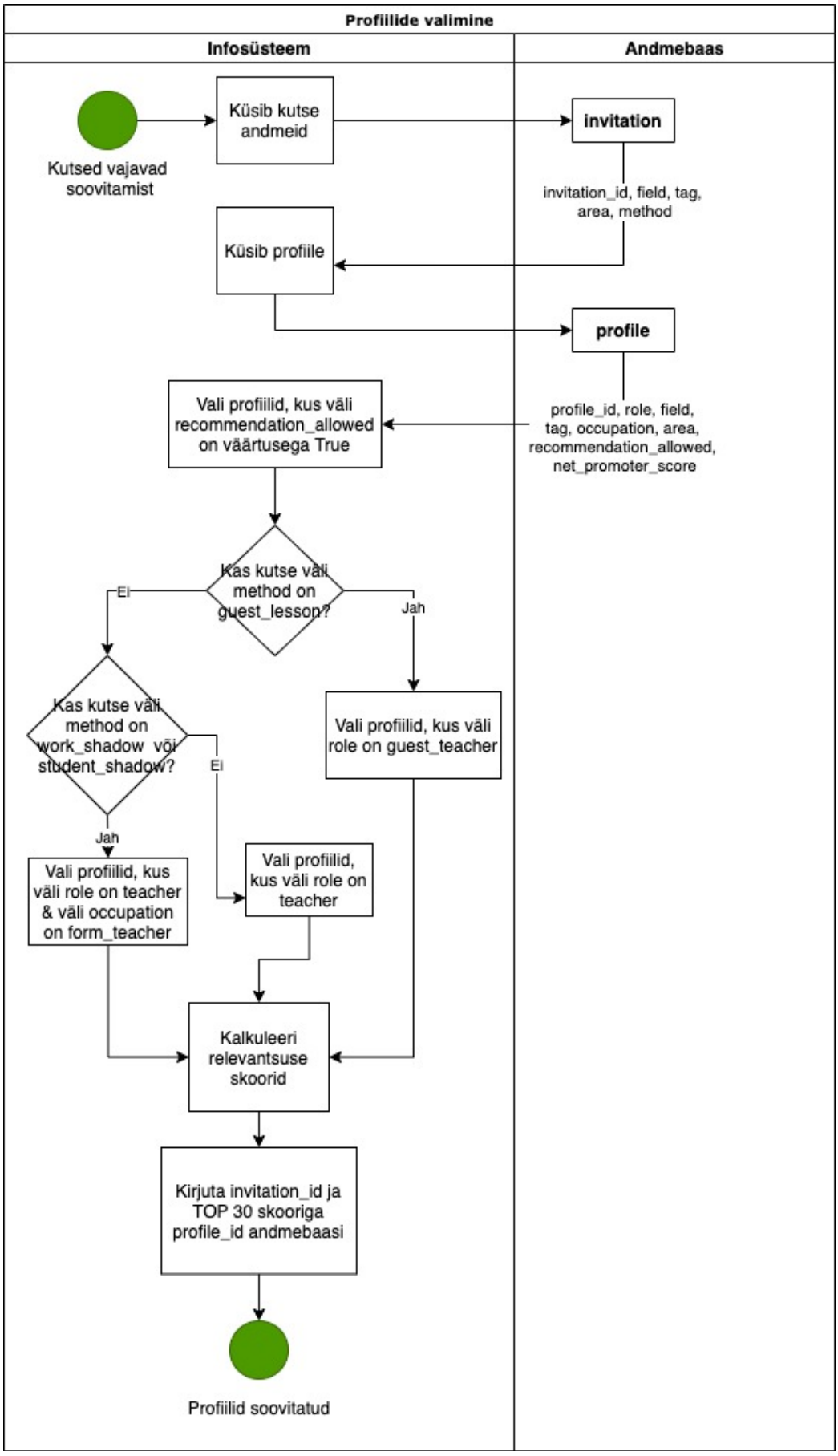
infosüsteemi registreerudes. Joonisel 7 on toodud *DataFrame*'ile puuduolevate veergude ja nende väärtuste lisamise kood.

```
profilesdf.insert(7, "occupation", "NaN", True)
profilesdf.insert(8, "recommendations_allowed", "TRUE", True)
profilesdf.insert(9, "net_promoter_score", "NaN", True)
```

Joonis 7. Profiilide DataFrame'ile veergude ja nende väärtuste lisamise kood.

3.3 Profiilide selekteerimine

Joonisel 8 on kujutatud on kujutatud profiilide selekteerimise loogika. Kutsete tabeli (joonisel märgistatud sõnaga „*invitation*") struktuur pole täpselt teada, aga uue infosüsteemi arendusmeeskond on seda meelt, et tõenäoliselt hoitakse kõiki kutseid ühes tabelis ning seal on ka väljad, mis hoiavad kutse valdkonda, märksõnu, piirkonda ja meetodit. Profiilide selekteerimise kood saab olema põhimõttel, et sisendinda võetakse ühe kutse andmed. Asjaolu, mis hetkel koodi välja kutsutakse, jääb uue infosüsteemi arhitektide otsustada.



Joonis 8. Profiilide selekteerimine.

Profiilide soovitamine algab kutse ja profiilide andmete küsimisega. Seejärel selekteeritakse need profiilid, kellele on luba saata kutseid. Järgmiseks tehakse kindlaks, mis rollis inimesele on kutse mõeldud. Kui kutse meetod on õpetaja sisestatud külalistund (*guest_lesson*), siis see on mõeldud külalisõpetajatele (*guest_teacher*). Õppekäigu (*study_trip*) ja organisatsiooni sisestatud külalistunni (*guest_lesson_org*) puhul on kutse mõeldud õpetajatele (*teacher*). Töö- ja tudengivarju (*work_shadow*, *student_shadow*) puhul on oluline, et õpetaja oleks ka klassijuhataja (*form_teacher*).

Eelmainitud tabelid laaditakse CSV(*Comma Separated Values*) formaadis *DataFrame* objektina. Joonisel 9 on toodud kirjeldatud selekteerimise protsessi kood [21].

```
 profilesdf = profilesdf[profilesdf["recommendations_allowed"] ==
True]
if (invitationdf.loc[0]["method"]== "guest_lesson") :
    profilesdf = profilesdf[profilesdf["role"] == "guest_teacher"]
elif invitationdf.loc[0]["method"] == "work_shadow" :
    profilesdf = profilesdf[(profilesdf["role"] == "teacher") &
(profilesdf["occupation"] == "form_teacher")]
elif invitationdf.loc[0]["method"] == "student_shadow" :
    profilesdf = profilesdf[(profilesdf["role"] == "teacher") &
(profilesdf["occupation"] == "form_teacher")]
elif invitationdf.loc[0]["method"] == "study_trip" :
    profilesdf = profilesdf[profilesdf["role"] == "teacher"]
elif invitationdf.loc[0]["method"] == "guest_lesson_org" :
        profilesdf = profilesdf[profilesdf["role"] == "teacher"]
```

Joonis 9. Profiilide selekteerimine vastavalt kutse meetodile ja kutsete saatmise võimalikkusele.

Järgnevalt ühendatakse kutse ja profiilide *DataFrame*'i selliselt, et igal profiili real oleks ka kutse andmed. Selliselt tabeli ühendamiseks on vajalik luua võti, mis hiljem kustutatakse [22].

```
invitationdf["key"] = 1
profilesdf["key"] = 1
joineddf = pd.merge(invitationdf, profilesdf, on ='key').drop("key",
1)
```

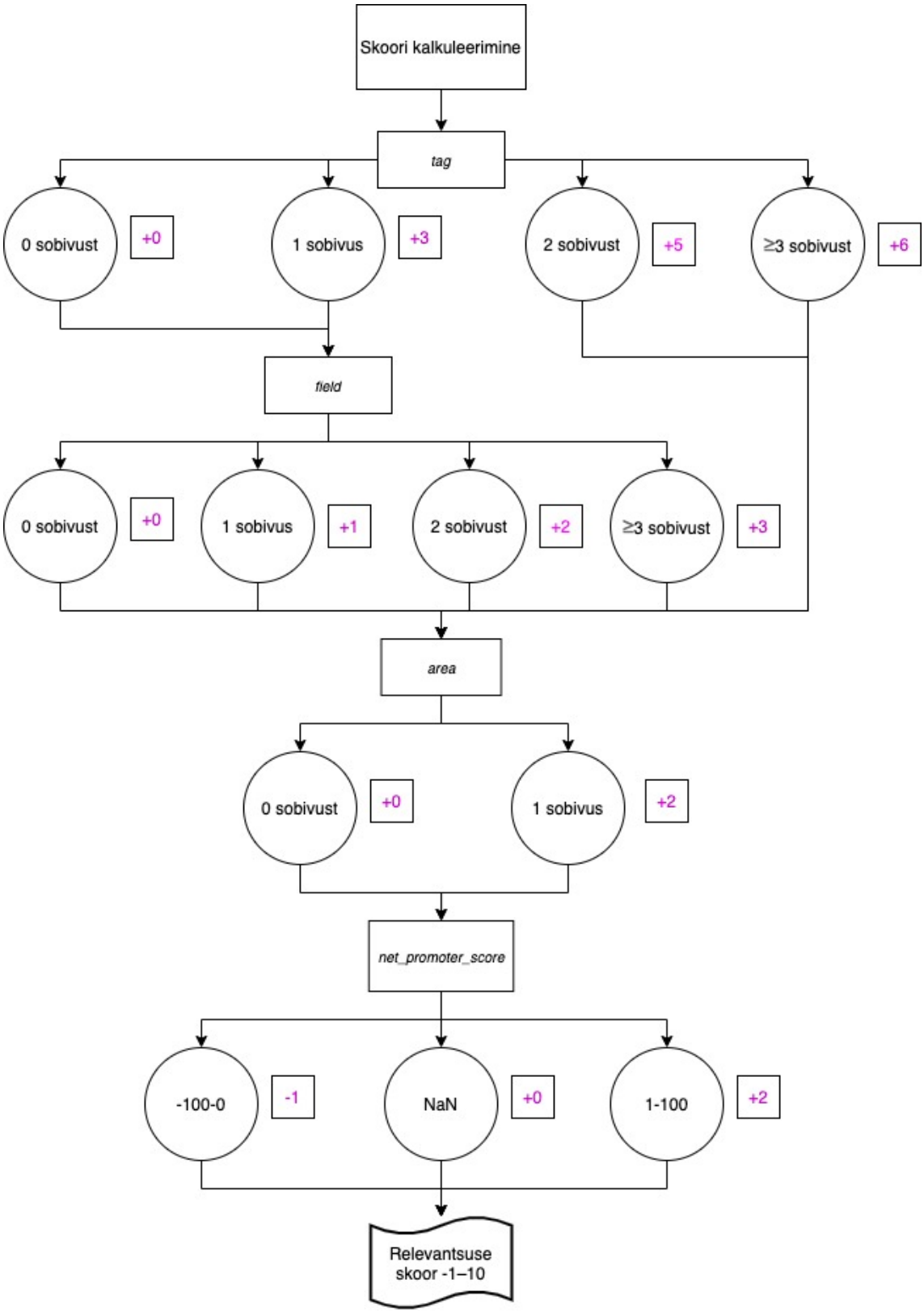
Joonis 10. Kutse ja profiilide *DataFrame*'ide ühendamine.

3.4 Relevantsuse skoori arvutamine

Peatükis 3.3. ettevalmistatud *DataFrame*'i põhjal saab hakata arvutama igale profiilile relevantsuse skoori vastava kutse kohta. Joonisel 11 on toodud skoori ja kaalude loogika. Skoor antakse vahemikus -1–10 punkti.

Skoori kõige suurema panuse saab anda kolme või enama märksõna (*tag*) kokkulangevus kutsel ja profiilil. Sellisel juhul, võib järeldada, et kutse temaatika ootus vastab üsna täpselt vastaspoole huvidega. Eeldus on, et selline huvide kokkulangevus võib pakkuda piisavalt tugeva motivatsiooni liikuda kohtumiseks uutesse maakondadesse. Kahe märksõna kattuvuse puhul, võib järeldada samuti küllaltki täpset huvide kattuvust, seetõttu ka sel juhul valdkonda (*field*) ei vaadata. Ühe või mitte ühegi märksõna kattuvuse puhul, vaadatakse valdkonna kattuvusi. Kui märksõna kattuvusi on üks ja valdkonna kattuvusi on kolm või enam, siis võib järeldada huvide kattuvust üsna täpseks ja skoor annab samaväärse tulemuse, kui kolme või enama märksõna kokkulangevus. Ühe märksõna ja kahe valdkonna kattuvust võib pidada samaväärseks kahe märksõna kattuvusega.

Piirkonna (*area*) kattuvuse korral, lisandub skoorile kaks punkti, kuna tegemist on „Tagasi Kooli“ meeskonna hinnangul olulise aspektiga huviliste valmisolekul kohtumisteks. Järgnevalt vaadatakse soovitusindeksit (*net_promoter_score*) ning siit on võimalik saada maksimaalselt lisa kaks punkti. Soovitusindeksi panus pole suurem, kuna uue infosüsteemi tööle hakkamise hetkel, pole seda veel profiilidel kujunenud. Aja möödudes, kus valdaval osal profiilidel on piisavas koguses kohtumiste tagasiside põhjal kujunenud soovitusindeks, on mõttekas selle kaal üle vaadata ning anda kesksem koht skoori kujunemisel. Profiilide, mille soovitusindeks on 0 või vähem, relevantsuse skoor vähendatakse 1 palli võrra. Selliselt soovitame halvema tagasiside saanud profiilidele kutseid ainult juhul, kui kutse teema ja piirkond klapivad profiili omadega suures osas, mis võiks anda suure panuse õnnestunud kohtumisse.



Joonis 11. Relevantsuse skoori kalkuleerimine.

Joonisel 12 on toodud kood, mis loetleb kutse ja profiilide märksõnade kokkulangevusi [23]. Vastavalt sellele antakse panus relevantsuse skoori, nagu joonisel 11 on toodud.

```
joineddf["tag_match"] = joineddf.apply(lambda row: len([x for x in
row.invitation_tag_list if x in row.profile_tag_list]), axis=1)
def tag_logic(tag_match):
    if tag_match == 0 :
        return 0
    elif tag_match == 1 :
        return 3
    elif tag_match == 2 :
        return 5
    elif tag_match >= 3 :
        return 6
joineddf["relevance_score"] = joineddf.tag_match.apply(lambda x:
tag_logic(x))
```

Joonis 12. Märksõnade kattuvuse leidmine ja vastavalt relevantsuse skoori loomine.

Juhul, kui märksõnade kattuvusi oli vähem, kui kaks, vaadatakse ka valdkonna kattuvusi ning uuendatakse relevantsuse skoori vastavalt [21, 23, 24, 25]. Joonisel 13 on toodud valdkondade kattuvuse leidmise ja relevantsuse skoori uuendamise kood.

```
joineddf["field_match"] = joineddf.loc[joineddf["tag_match"] <
2].apply(lambda row: len([x for x in row.invitation_field_list if x in
row.profile_field_list]), axis=1)
def field_logic(field_score):
    if field_score == 0 :
        return 0
    elif field_score == 1 :
        return 1
    elif field_score == 2 :
        return 2
    elif field_score >= 3 :
        return 3
    elif np.isnan(field_score) :
        return 0
joineddf["relevance_score"] =
joineddf["relevance_score"].add(joineddf.field_match.apply(lambda x:
field_logic(x)))
```

Joonis 13. Valdkondade kattuvuste leidmine ja vastavalt relevantsuse skoori uuendamine.

Järgmisena leitakse kattuvused kutse ja profiilide piirkondade väärtuste vahel ning uuendatakse relevantsuse skoori vastavalt. Viimasena annab relevantsuse skoori oma panuse soovitusindeks. Piirkonna ja soovitusindeksi kaalude kood on toodud joonisel 14.

```
def area_logic(area_score):
    if area_score == 0 :
        return 0
    elif area_score == 1 :
        return 2
def net_promoter_score_logic(net_promoter_score):
    if -100 <= net_promoter_score <= 0 :
        return -1
    elif 0 <= net_promoter_score <= 100:
        return 2
    elif np.isnan(net_promoter_score) :
        return 0
```

Joonis 14. Piirkonna ja soovitusindeksi kaalude kood.

Lõpetuseks valitakse vaid need profiilid, mille relevantsuse skoor on suurem kui neli, reastatakse kahanevas järjekorras ning pannakse kokku CSV-vormingus fail, mis sisaldab kuni 30 suurima skooriga profiili ID'd ja kutse ID'd [21, 26, 27]. Joonisel 15 on toodud vastavaid toiminguid tegev kood.

```
highest_scoredf = joineddf.loc[joineddf["relevance_score"] > 4]
highest_scoredf = highest_scoredf.nlargest(30,"relevance_score")
recommended_profilesdf = highest_scoredf[["invitation_id",
"profile_id"]]
recommended_profilesdf.to_csv(r"/Users/teeleliblik/Jupyter/Recommended
_profiles.csv")
```

Joonis 15. 30 suurima relevantsuse skooriga profiili ID'd ja kutse ID'd CSV-failiks tegev kood.

Lahendusele on plaanis juurde ehitada mõõdik, mille alusel hinnata selle efekti. Selleks tuleb infosüsteemil võrrelda, kui paljud inimesed, kellele kutse soovitatakse, selle ka vastu võtavad. Mõõdiku edasiarendamisena on plaanis lisada info, kas kasutaja leidis endale mõne muu sobiva kutse, kui just soovitatud kutset vastu ei võtnud.

3.5 Lahendusel ilmnenud probleemid ja nende lahendamine

Ajamahukamaks probleemiks käesoleva lahenduse loomisel oli seotud andmetüübi olemusega CSV-faili sisselugemisel. Probleem tuli esile märksõnade, valdkondade ja piirkondade kattuvuste leidmisel, kus erinevate testandmete kasutamisel, ei andnud kood õiget tulemust. Kõigepealt tegi töö autor kindlaks, et kattuvuste otsimise algoritm oleks õige. Seejärel tekkis kahtlus, et viga võib olla sisendiks antavates andmetüüpides. Paistis, et midagi muutis sõnedest koosnevad listid millekski muuks. Peale erinevate andmete testimist tabelite ühendamise koodiga, sai selgeks, et viga ei olnud seal. Lõpuks jõudis

autor mõttele, et CSV-faili sisselugemisel võivad andmetüübid muutuda. Nimelt CSV-faili puhul loeti sõnedest koosnev list sisse ühe sõnena. Seda olukorda sai vältida joonisel 16 toodud lahendusega, kus täpsustatakse faili sisselugemisel veerud, mille puhul tuleb säilitada algne andmetüüp [28].

```
profilesdf = pd.read_csv(r"/Users/teeleliblik/Jupyter/Profiles.csv",
index_col = 0, converters={"profile_id": eval, "field": eval, "tag":
eval, "area": eval})
```

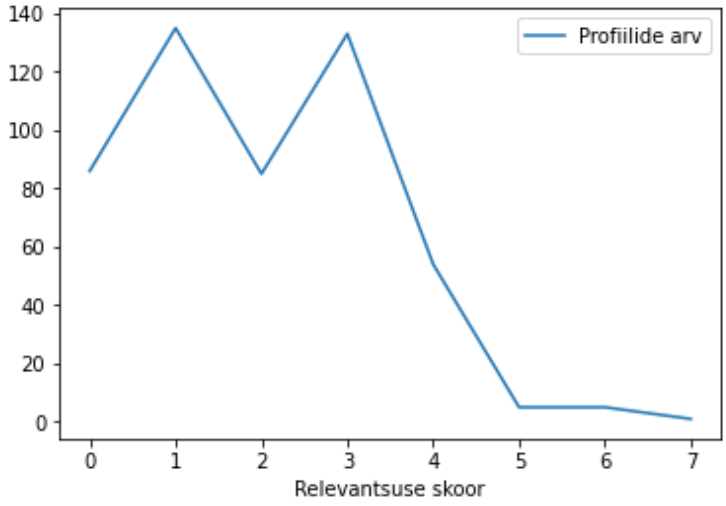
Joonis 16. CSV-faili sisselugemise kood andmetüüpide muutumatuse täpsustusega.

Ootamatult keerukaks kujunes aadressi väärtustest piirkonna kätte saamine. Autor proovis siin kasutada *re.search()* ja *pandas.Series.str.extract()* funktsionaalsusi, kuid erinevatel põhjustel need ei andnud soovitud tulemust. Lahendus leiti *re.findall()* funktsionaalsuse kaudu, kuid peab tõdema, et see ei ole kõige elegantsem, kuna väärtus muudetakse sõnest listiks ja tagasi mitmeid kordi.

4 Lahenduse testimine

Kuna loodud kood manipuleerib andmetabelitega, siis oli lahenduse testimine küllaltki lihtne, sest peale käskluse rakendamist on andmetest kohe näha, kas saadi soovitud tulemus. Lahenduse testimisel lõi autor mitmeid erinevaid profiile ja kutseid, et veenduda soovitud tulemuse saamises.

Loodud funktsionaalsuse testimiseks võeti sisendiks “Tagasi Kooli“ lehele ülespandud kutsed. Joonisel 17 on toodud loodud profiilidele relevantsuse skoori arvutamise tulemused külalistunni kutsega, kus oodatakse inimest Harju maakonda rääkima oma tööst majanduse ja matemaatika valdkonnas, täpsemalt rahatarkusest ja ärimaailmast.



Joonis 17. Relevantsuse skoori arvutustulemused.

Jooniselt 17 on näha, et hetkel paika pandud skoori kaalude ja loogika abil õnnestub selekteerida üsna täpselt kutsele vastavad profiilid.

5 Kokkuvõte

Käesolev töö pakub “Tagasi Kooli“ veebilehel kutsete vastuvõtmise madala aktiivsuse probleemile lahenduse kutsete saatmise automatiseerimise näol. Selleks lõi töö autor varasemalt osalenud kasutajate andmete põhjal profiilid, mille hulgast saab leida inimesed, keda lisandunud kutse võiks kõige rohkem kõnetada ning kellele see e-kirja teel saata. Profiilide valikul rakendatakse relevantsuse skoori põhimõtet, kus iga kutse puhul kalkuleeritakse, kui suurel hulgal sarnaneb see inimese varasemalt osaletud kohtumistega ehk kui tõenäoliselt võiks see huvi pakkuda.

“Tagasi Kooli“ algatuse keskseks tööriistaks on infosüsteem, kuhu saab sisestada kutseid koostööle. Küll aga, ei toimi orgaaniline kutsete leidmine infosüsteemist väga efektiivselt, mistõttu tihtipeale toimub kutse vastuvõtja leidmine manuaalselt “Tagasi Kooli“ töötaja poolt. Loodud funktsionaalsuse kasutamise järgselt hetkel infosüsteemi sisestatud kutsete andmetega saab järeldada, et Tagasi Kooli“ algatusel on aastatega tekkinud laialdane kasutajaskond, mille hulgast leiab ehitatud skoori arvutamise tulemusel potentsiaalseid huvilisi kohtumisele. Käesolevas töös toodud lahendus võimaldab “Tagasi Kooli“ kollektiivil vabanenud ajaressurssi suunata olulisematesse toimingutesse ning aitab hoida infosüsteemi koostöö keskpaigana, kus toimub aktiivne osalemine nii kutsujate, kui kutse vastuvõtjate poolt.

Kasutatud kirjandus

[1] Tagasi Kooli. [WWW] https://registreeru.tagasikooli.ee/ (10.09.2020)

[2] Net Promoter. [WWW] https://en.wikipedia.org/wiki/Net_Promoter (10.10.2020)

[3] Top Programming Languages for Data Science in 2020. [WWW] https://www.geeksforgeeks.org/top-programming-languages-for-data-science-in-2020/ (28.10.2020)

[4] And the Best Programming Language for Data Science Goes to…[WWW] https://medium.com/@harshit_tyagi/and-the-best-programming-language-for-data-science-goes-to-f80b9a5b439c (28.10.2020)

[5] Pandas. [WWW] https://pandas.pydata.org (28.11.2020)

[6] Regular Expression HOWTO. [WWW] https://docs.python.org/3/howto/regex.html (28.11.2020)

[7] Comma-separated values. [WWW] https://en.wikipedia.org/wiki/Comma-separated_values (28.11.2020)

[8] Pandas. [WWW] https://et.wikipedia.org/wiki/Pandas (28.11.2020)

[9] Python Pandas: concatenate rows with unique values. [WWW] https://stackoverflow.com/questions/27174009/python-pandas-concatenate-rows-with-unique-values (12.11.2020)

[10] Adding new column to existing Data-Frame in Pandas. [WWW] https://www.geeksforgeeks.org/adding-new-column-to-existing-dataframe-in-pandas/ (12.11.2020)

[11] Patak, M. (2018) Joining DataFrames in Pandas. [WWW] https://www.datacamp.com/community/tutorials/joining-dataframes-pandas (12.11.2020)

[12] Kartikaybhutani. (2018) Python | Pandas Series.astype() to convert Data type of series. [WWW] https://www.geeksforgeeks.org/python-pandas-series-astype-to-convert-data-type-of-series/ (18.11.2020)

[13] How do I lowercase a string in Python? [WWW] https://stackoverflow.com/questions/6797984/how-do-i-lowercase-a-string-in-python (08.11.2020)

[14] Regular expression to return text between parenthesis. [WWW] https://stackoverflow.com/questions/4894069/regular-expression-to-return-text-between-parenthesis (20.11.2020)

[15] How to Remove Duplicates From a Python List. [WWW] https://www.w3schools.com/python/python_howto_remove_duplicates.asp (18.11.2020)

[16] Shivam_k. (2019) Python program to convert a list to string. [WWW] https://www.geeksforgeeks.org/python-program-to-convert-a-list-to-string/ (10.11.2020)

[17] How to remove parentheses from string [duplicate]. [WWW] https://stackoverflow.com/questions/56375850/how-to-remove-parentheses-from-string (22.11.2020)

[18] Python String split() Method. [WWW] https://www.w3schools.com/python/ref_string_split.asp (10.11.2020)

[19] Combining DataFrames with Pandas. [WWW] https://datacarpentry.org/python-ecology-lesson/05-merging-data/index.html (21.11.2020)

[20] How to merge two data frames based on particular column in pandas python? [WWW] https://stackoverflow.com/questions/37697195/how-to-merge-two-data-frames-based-on-particular-column-in-pandas-python (21.11.2020)

[21] (2020) How to Select Rows from Pandas DataFrame. [WWW] https://datatofish.com/select-rows-pandas-dataframe/ (10.10.2020)

[22] Maryamnadeem20. (2020) Python Program to perform cross join in Pandas. [WWW] https://www.geeksforgeeks.org/python-program-to-perform-cross-join-in-pandas/ (15.10.2020)

[23] Python 3 - counting matches in two lists (including duplicates). [WWW] https://stackoverflow.com/questions/13323851/python-3-counting-matches-in-two-lists-including-duplicates (16.10.2020)

[24] How to compare one value with 'NaN' in if statement: Python 3.5.2 | Anaconda 4.1.0 (64 bit) For example: a!=Nan where a is either a float value. [WWW] https://stackoverflow.com/questions/45061088/how-to-compare-one-value-with-nan-in-if-statement-python-3-5-2-anaconda-4-1 (23.10.2020)

[25] Shubham__Ranjan. (2018) Python | Pandas dataframe.add(). [WWW] https://www.geeksforgeeks.org/python-pandas-dataframe-add/ (23.10.2020)

[26] (2019) How to Select Top N Rows with the Largest Values in a Column(s) in Pandas? [WWW] https://cmdlinetips.com/2019/03/how-to-select-top-n-rows-with-the-largest-values-in-a-columns-in-pandas/ (23.10.2020)

[27] (2020) Pandas to CSV. [WWW] https://www.datacamp.com/community/tutorials/pandas-to-csv (23.10.2020)

[28] Pandas DataFrame stored list as string: How to convert back to list? [WWW] https://stackoverflow.com/questions/23111990/pandas-dataframe-stored-list-as-string-how-to-convert-back-to-list (26.11.2020)

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Teele Liblik

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Kutsete saatmise automatiseerimine "Tagasi Kooli" infosüsteemis", mille juhendaja on Meelis Antoi.
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

02.01.2021

1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – “Tagasi Kooli“ juhi Teibi Tormi tagasiside

01.12.2020

Lugupeetud lõputööde kaitsmise komisjon

Tagasi Kooli haldab ja arendab alates 2012. aastast infosüsteemi külalistundide, õppekäikude ja töövarjude kokku leppimiseks. Teele Liblik on varasemalt töötanud MTÜ Tagasi Koolis nii projektijuhi kui vabatahtlikuna. Rääkisime Teelele planeeritava uue süsteemiga seotud väljakutsetest ja ta oli valmis ühe väljakutse oma lõputööga lahendama.

Kõigepealt põgusalt väljakutsest. Tagasi Kooli praeguse infosüsteemi probleemiks on see, et õpetajate poolt sisestatud külalistunni kutsed ei jõua õigete külalisõpetajateni. Põhjuseks on see, et kutseid kuvatakse veebis ja Tagasi Kooli saadab pidevalt inimestele infot kutseid vaatama tulla aga just sellel hetkel kui inimene tuleb, ei pruugi sobival teemal, ajal ja piirkonnas kutset üleval olla. Seetõttu jäävad ära sajad kohtumised. Soovime uue süsteemiga luua olukorra, kus eelnevalt kokku lepitud tundide andmete põhjal saame saata inimestele võimalikult täpselt neile sobivaid uusi kutseid.

Teele töötas kirjeldatud väljakutse lahendamisega süsteemselt ja paindlikult. Meil toimus mitmeid kohtumisi, kus Teele uuris tausta ja näitas meile enda poolt tehtud tööd ja küsis tagasisidet. Meie jaoks oli suurepärane, et Teele eesmärgiks oli meid aidata ning ta arvestas meie poolt antud tagasisidega ja tegi vajadusel muudatusi. Oleme Teele tehtud tööga väga rahul, sest alustame peagi uue süsteemi loomist ja saame tema tehtud koodi uude süsteemi integreerida.

Head soovides
Teibi Torm
MTÜ Tagasi Kooli juht
5664 2380
teibi@tagasikooli.ee
tagasikooli.ee