Tallinna Ülikool
Digitehnoloogiate Instituut

VIRTUAALNE TUNNUSTAMISE SÜSTEEMI OPEN BADGES ÜLESSEADMINE JA KASUTAMINE

Bakalaureusetöö

Autor: Joonas Vaino
Juhendaja: Marina Kurvits

Autor: …… „ ..... „2016
Juhendaja: …….. „ ..... „2016
Instituudi direktor: …….. „ ..... „2016

Tallinn 2016

Autorideklaratsioon

Deklareerin, et käesolev bakalaureusetöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

(kuupäev)(autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina Joonas Vaino (sünnikuupäev: 22.04.1992)

1. annan Tallinna Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Virtuaalne tunnustamise süsteemi Open Badges ülesseadmine ja kasutamine", mille juhendaja on Marina Kurvits, säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.
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.

Tallinnas,

....

(kuupäev) (allkiri)

Sisukord

Sissejuhatus

Tunnustuse andmine ja saamine on inimeste elu igapäevane osa. Olgu see kas või lihtne hommikune tere oma kaaskodanikule või siis midagi suuremat, nagu elutöö preemia. Tunnustuse jagamise alguse otsimiseks peab kindlasti appi võtma ajalooraamatu. Väiksemaid ja suuremaid preemiaid, auraha või diplomeid jagatakse väljapaistvatele inimestele kindlasti rohkem kui kord aastas. Arvuti tulekuga meie igapäevaellu on välja arenenud ka virtuaalsed tunnustamise viisid, mida saab rakendada positiivsete tegevuste tegemise motiveerimiseks. Autor kohtus oma juhendaja Marina Kurvitsaga, kellel oli soov rakendada virtuaalset tunnustamist oma igapäeva töös.

Autor valis selle bakalaureusetöö teema, kuna on ise suur tunnustus kogumis huviline ja on olnud huvitatud endi projektides rakendama virtuaalseid tunnustamise viise kasutajate ja rakenduse vahelise koostoimimise parandamiseks. Eeluurides erinevaid tunnustamise süsteeme jäi silma just Open Badges, kuna seda on arendatud eesmärgil kehtestada virtuaalsetes tunnustamise süsteemides üldstandard.

Käesoleva bakalaureusetöö eesmärgiks on anda ülevaade Open Badges virtuaalse tunnustamise süsteemi ülessseadmiseks ja kasutamiseks ning anda hinnang selle kasutusele võtmise keerukusest. Lisaks tutvustab bakalaureusetöö virtuaalsete tunnustamise süsteemide ajalugu ja rakendamist töö -ja õppevaldkondades. Töö võiks pakkuda huvi iga valdkonna inimestele, kes soovivad virtuaalset tunnustust hakata jagama, jälgides hetkel populaarsemat üldstandardit.

Autor tutvub tehnoloogia spetsifikatsiooniga ning katsetab praktikas olulisemaid võimalusi. üritades anda võimalikult hea ülevaate, et töö lugeja oleks võimeline protsessi hiljem järgi tegema.

Mõisted

DELETE - HTTP protokolli meetod. Kasutakse veebiserverist andmete kustutamiseks.

GET - HTTP protokolli meetod. Kasutakse veebiserverist andmete küsimiseks.

HMAC-SHA256 - Krüpteerimisalgoritm, mis kasutab Sha256 räsifunktsiooni.

HTTP protokoll - Andmevahetusprotokoll, mida kasutakse internetis dokumentide vahetamiseks.

JSON (JavaScript Object Notation) - Avatud standard inimloetava teksti saatmiseks andmeobjektidena.

POST - HTTP protokolli meetod. Kasutakse veebiserverile andmete saatmiseks.

PUT - HTTP protokolli meetod. Kasutakse andmete salvestamiseks.

Sha256 - Räsifunktsioon mis genereerib sisendsõnast 32-bitise väljundi.

Token - Autoriseerimisvõti võrguligipääsu saamiseks, reeglina krüpteeritud sõne.

1. Virtuaalne tunnustus

Virtuaalne tunnustus on tunnustus, mis on arvutiandmete kujul. Tunnustuse saamise kriteeriumi võib täita digitaalsest maailmast väljaspool, aga kui saadud tunnustus jaguneb kuskil bitide jadaks, on tegemist virtuaalse tunnustusega. See võib olla midagi lihtsat, nagu sulle suunatud Facebook'i pöidla märk, või midagi suuremat, nagu digitaalne diplom. Antud peatükis uurib autor virtuaalse tunnustuse ajalugu ja uurib virtuaalse tunnustamise väärtust professionaalses elus.

1.1 Virtuaalse tunnustuse ajalugu

Virtuaalsed tunnustamise süsteemid on välja kasvanud arvutimängudest. Tänapäeval on raske leida uut mängu, mis ei kaasa endaga mingisugust virtuaalset tunnustamise süsteemi. Nimetusi võib olla saadud tunnustusel erinevaid, näiteks: *achievement*, trofee(*trophy*), märk(*badge*), medal, väljakutse(*challenge*) või lihtsalt auhind(*award*). Kõiki eelpool nimetatud tunnustuse näitajaid ühendab nende loomus. Reeglina on need tunnustused teisejärgulised eesmärgid, mis ei mõjuta edasist mängu ja mille haldus toimub mängu piiridest väljaspool. Mis on väga sarnane reaalelus saadava tunnustusega, näiteks koolis õppides ei ole eesmärk diplomit saada, vaid isiklik areng. (Hamari & Eranti, 2011)

Arvutimängudes tunnustuse jagamise idee lähtekohaks võib pidada ameerika mängulooja Activision'i kampaaniat: Patches for high scores (märgid rekordpunktide vastu). Mis toimus aastatel 1982 kuni 1983. Kampaania seisnes selles, et mängu käsiraamatutes oli kirjas punktide arv, mille täitmisel pidi mängija tegema oma ekraanist pildi, selle ilmutada laskma ja siis postiga Activision'i peakontorisse saatma. Sealt saadeti neile vastu tunnustusmärk koos kirjaga, mis õnnitles mängijat tema saavutuse puhul. (Hillard, 2013)

Peale Activision'i kampaaniat hakkasid paljud mängutegijad oma mängudesse lisama salalisasi või tunnustusauhindu. Alles interneti laialt levimisajal arendati välja esimene ühtlustatud tunnustamise süsteem mängudele. Microsoft tutvustas *Gamescore* süsteemi aastal 2005. See sundis mängu valmistajatel lisama oma mängudele kõrvaleesmärke, mille täitmisel autasustati mängijat punktidega. Hiljem hakkasid sarnast mudelit kasutama Valve oma

mänguhaldus- ja poe tarkvara Steam peal kui ka Sony oma Playstation võrgus (Giantbomb, 2016). (Giantbomb, 2015)

1.2 Virtuaalse tunnustuse kasutamine professionaalses elus

Kõik inimesed, kes on internetis end mingisugusele teenusele registreerinud, on endale loonud internetipersoona. See persoona koosneb kõikidest kontodest, mis on otseselt sinu reaalmaailma persoonaga kokkuviidavad, sinna alla kuuluvad näiteks: e-posti konto, sotsiaalmeedia kontod, foorumite ja teiste interneti kogukondade kontod. Personaalset interneti persoonat saab kasutada iseenda reklaamimiseks reaalmaailmas. (Holmes, 2014)

Sisu, mis on sinu endi poolt loodud, vajab tõestust. Väga tähtsal kohal virtuaalse tunnustuse likviidsusel on kogukonna olemasolu. Kogukond annab sinu poolt toodetud sisule sotsiaalse tõestusmaterjali, et sinu oletus või teos on tõene või aktsepteeritav. Sotsiaalne tõestusmaterjal on müügialaselt väga laialt levinud võte ja reeglina personaalse interneti persoona üks eesmärke ongi otseselt või kaudselt sinu enda müümine. Olgu see siis sotsiaalmeedia kuulsus või mõni kõrgem eesmärk, mille sa soovid oma persoonaga saavutada. (Fletcher, kuupäev puudub)

Oma interneti persoona väärtuse tõstmine on küllaltki lihtne. Veebipõhiseid tööriistu, mille abiga saab enda kompetentsi tõestada, on mitmeid ja samuti on nende olemused ning valdkonnad seinast seina. Eksisteerivad eraldi sotsiaalmeedia platvormid oma töö- või haridusalase kompetentsi näitamiseks, interneti kursused või interneti ülikoolid, kus saab läbida õppetunde tunnustuse saamiseks või lihtsalt tööalaste suunitlustega foorumid, mis jälgivad sinu vastuste abistavust teistele. Kõikidel nendel veebipõhistel tööriistadel on kogukonnad, mis kehtestavad tunnustuse tähtsuse standardi, mis omakorda aitab näidata oma interneti persoonat heast küljest.

Teiselt küljelt, nende platvormide poolt väljastatud tunnustuste väärtust võib teinekord olla raske hinnata isegi kogukonna olemasolul. Näiteks läbides veebikursuse, mis koosneb ainult videotest ja saades selle eest tunnustust, ei pruugi tähendada veel, et sa kogu materjali oled omandanud. Siin kohal on hea näide arendusalase õppekeskonna Treehouse'i (https://teamtreehouse.com/) esindaja Hernandez Guil'i vastus kogukonna poolt esitatud

sooritatud kursuste sertifikaatide väljastamise funktsionaalsuse soovile: “Parim sertifikaat, minu arvamusel, on disainida ja arendada midagi väga vägevat, kasutades oma oskusi ja siis avalikustada selle maailmale vaatamiseks ja kasutamiseks.”. (Zomick, 2013)

Sarnase murega tegelesid ka ühe tuntuma arendus foorumi Stack Overflow kasutajad. Stack Overflow töötab küsimus ja vastust põhimõttel. Murega kasutaja küsib küsimuse ja teised kasutajad pakuvad sellele diskuteerides vastuseid, mida kasutajad saavad kas üles või alla hääletada vastuste nimekirjas. Kogu selle protsessi käigus teenivad kasutajad punkte asjakohaste küsimuste küsimise, heade vastuste kirjutamise, vastuse õigesti redigeerimise ja palju muu eest. Kuna leht on väga lugupeetud ja suure kogukonnaga, tekkis küsimus, et millisest hetkest alates mainida Stack Overflow punkte oma cv-s. Arutelus tulid välja arvamused, et punktide väärtus on liiga suhteline otsese väärtuse lisamiseks, aga huvide näitamise jaoks väga hea väljund. (StackExchange, kuupäev puudub)

Kokkuvõtlikult on virtuaalse tunnustuse väärtus professionaalses elus kahepoolne ja lõpliku hinnangu sellele väärtusele saab anda ainult vaatleja. Siin kohal tuleks appi üldistatud virtuaalse tunnustuse süsteem, mis suudaks lihtsalt kokku koguda kasutaja virtuaalse tunnustuse märgid erinevatelt veebilehtedelt, tehes selle näitamise kasutajale ajalises mõttes odavamaks.

2. Virtuaal tunnustuse tarkvara Open Badges

Open Badges on Mozilla poolt arendatud tasuta tarkvara virtuaalse tunnustuse jagamiseks. Open Badge'i eesmärk oli luua tarkvara, mille abil oleks mugav kasutajatel anda, saada ja näidata enda virtuaalset tunnustust. Mozilla visiooni järgi peaks Open Badges saama virtuaalse tunnustuse väljaandmise standardiks, mida kõik tunnustust andvad veebileheküljed saaksid lihtsalt oma süsteemi integreerida. (Badges/FAQs, kuupäev puudub)

2.1 Open Badges'i ülesehitus

Open Badge'iga tehtud tunnustuse märk koosneb digitaalsest pildist ja andmetest. Andmed virtuaalse märgi küljes kirjeldavad:

- Mida märk esindab (mille eest välja antud).
- Kellele on märk antud.
- Kes andis märgi välja (kool, tööandja, kogukond).

Open Badge'i süsteemis on kasutajatel neli rolli:

- Emitent (*Issuer*)
- Märgi teenija (*Earner*)
- Märgi näitaja (*Displayer*)
- Märgi tarbija (*Consumer*)

Märgi emitent loob virtuaalsed märgid ja annab kasutajatele võimaluse neid teenida. Lisaks otsustab märgi emitent, kes märgi teenivad. Olenevalt märgist võib siin kohal olla määratud märgisaamiskriteeriumid, mis defineeritakse märgi loomise ajal. Peale loomist on kasutajatel ehk märgi teenijatel võimalus märke teenida. Märkide teenimise protsess on tavaliselt ühendatud mingi rakendusega või hindamisprotsessiga. (GitHub, 2014d)

Kui emitent on otsustanud autasustada märgi teenijat märgiga, võib märgi teenija lisada märgi oma märgi seljakotti. Seljakott on rakendus, mis võimaldab märgi saajatel oma märke hallata. Seljakoti funktsionaalsuse hulka kuulub märkide sorteerimine, grupeerimine, kollektsioonide

loomine ja nähtavaks tegemine või peitmine. Uued märgid, mis kasutaja teenib, on vaikimisi privaatsed, mis tähendab, et kasutaja saab ise valida, milliseid märke ta kellegagi jagada soovib. Mozilla on ka loonud oma seljakoti rakenduse Mozilla Backpack: https://backpack.openbadges.org/. (Mozilla support, kuupäev puudub)

Kui kasutaja avalikustab grupi märke, muutub see grupp nähtavaks märgi näitajatele. Märgi näitajad saavad teha päringuid kasutaja meiliaadressiga, mis tagastab kasutaja poolt avaldatud märgid. Lisaks saavad märgi näitajad teha päringuid märgi küljes olevate andmetega, tehes kindlaks kas märk on õiguspärane, peale mida võimaldab märgi näitaja kuvada kasutaja märke veebilehtedel, rakendustes või sotsiaalmeedias. Viimaseks tegijaks protsessis on veel tarbijad, kes saavad kasutaja märke vaadata. (GitHub, 2014d)

2.2 BadgeKit'i ülesseadmine

Open Badges'iga märkide jagamiseks on vaja kasutada BadgeKit rakendusliidest. BadgeKit rakendusliides on ehitatud node.js'i põhjal, kuigi rakendusliidese käima saamiseks ei ole vaja node.js'is erilisi teadmisi. Selles alapeatükis näitab autor, kuidas üles seada lokaalne BadgeKit'i rakendusliides. Ülesseadmisel kasutab autor järgmiseid avatud lähtekoodiga tarkvarasid:

- Npm - node.js paketihaldur.
- Foreman - Serveri haldustarkvara.
- Git - Versiooni haldustarkvara.
- MySQL - Andmebaasi haldustarkvara.

Esimese sammuna tuleb luua kaust, kus hoiustada BadgeKit'i rakenduliitme eksemplari. Järgmisena tuleb avada käsurida ja liikuda loodud kausta, peale mida tuleb alla laadida Mozilla poolt loodud BadgeKit'i rakendusliide nende ametlikust Git'i hoidlast. Käsurea käsk on järgnev:

```
git clone https://github.com/mozilla/badgekit-api.git
```

Kui käsu jooksutamine õnnestub, on käsureal õnnestumine teada ja kausta on loodud uus kaust nimega badgekit-api (vt joonis 1).

```
➜ OpenBadges  git clone https://github.com/mozilla/badgekit-api.git
Cloning into 'badgekit-api'...
remote: Counting objects: 3164, done.
remote: Total 3164 (delta 0), reused 0 (delta 0), pack-reused 3164
Receiving objects: 100% (3164/3164), 547.58 KiB | 365.00 KiB/s, done.
Resolving deltas: 100% (2117/2117), done.
Checking connectivity... done.
➜ OpenBadges  ls
badgekit-api
```

Joonis 1 - Käsurida peale edukat git clone käsku, näha on äsjaloodud kausta.

Järgmisena liigume loodud badgekit-api kausta ja laeme alla tarkvara, millest BadgeKit'i rakendusliides sõltub. Selleks kasutame npm'i mis on node.js'i paketi haldur. Käsureale sisestatav käsk on järgnev:

```
npm install
```

Kõik sõltuvad paketid on kirjas package.json failis. Peale käsu jooksutamist tekib terminali nimekiri allalaetud pakettidest ja teade allalaadimise eduka sooritamise korral.

Järgmise sammuna tuleb seadistada BadgeKit'i rakendusliidese keskkonna muutujad. Seadistamiseks tuleb luua tekstifail, mille sisuks on:

- DB_HOST - Andmebaasi aadress.
- DB_NAME - Andmebaasi nimi.
- DB_PASSWORD - Andmebaasi parool.
- DB_USER - Andmebaasi kasutajanimi.
- MASTER_SECRET - Kasutaja poolt määratud BadgeKit'i rakendusliitme salasõna.
- PORT - Loogilise ühenduse lõpp-punkti number, kuhu BadgeKit'i rakendusliide seadistakse.

BadgeKit'i rakendusliidest lokaalset jooksutades näeb faili sisu välja selline:

```
export DB_HOST=localhost
export DB_NAME=badgekitapi
```

```
export DB_PASSWORD=parool
export DB_USER=root
export MASTER_SECRET=salasona
export PORT=8080
```

Järgmise sammuna tuleb luua BadgeKit'i rakendusliidesele MySQL andmebaas. MySQL'i sisenemiseks tuleb sisestada käsureale järgnev käsk:

```
mysql --user=root --password=parool
```

Kus `--user` on kasutatava andmebaasi kasutaja nimi ja `--password` vastava kasutaja parool. Kui MySQL avaneb, tuleb järgmisena luua uus andmebaas, mille nimi peab olema sama, mis BadgeKit'i rakendusliidese keskkonna muutujate tekstifailis on defineeritud `DB_NAME` abil. Uue andmebaasi loomiseks tuleb MySQL'i käsureal sisestada järgnev käsk:

```
CREATE DATABASE badgekitapi;
```

Näite puhul on autori andmebaasi nimi badgekitapi. Kui andmebaas on loodud, tuleb jooksutada BadgeKit'i rakendusliidese andmebaasi migratsiooni pakkfaili. Mis asub bin kaustas ja on nimega db-migrate. Et aga pakkfail saaks edukalt andmebaasi uuendada, tuleb BadgeKit'i rakendusliitmele sisestada eelnevenalt loodud keskkonna muutujate tekstifailis olevad muutujad. Muutujate uuendamiseks tuleb käsureale sisestada järgnev käsk:

```
source env_local
```

Näite puhul oli autori keskkonna muutujate tekstifaili nime env_local, env_local asemel tuleb sisestada loodud keskkonna muutujate tekstifaili nimi. Järgmisena saab nüüd jooksutada migratsiooni pakkfaili, olles badgekit-api kaustas, tuleb käsureale sisestada:

```
bin/db-migrate up
```

Kui migratsioon on end jooksutanud edukalt, peab käsureale tulema selle kohta teada (vt joonis 2).

```
➜  badgekit-api git:(master) ✗ source env_local
➜  badgekit-api git:(master) ✗ bin/db-migrate up
[INFO] Processed migration 20140603150606-initial
[INFO] Processed migration 20140606190514-45-badge-uniqueness
[INFO] Processed migration 20140609142108-evidence-type
[INFO] Processed migration 20140609165236-standards-alignment
[INFO] Done
```

Joonis 2 - Käsurida peale edukat migratsioonifaili jooksutamist.

Kui nüüd liikuda tagasi MySQL'i ja sisestada käsk:

```
show tables
```

Peab olema andmebaasi loodud kõik vajalikud tabelid BadgeKit'i rakendusliidese toimimiseks (vt joonis 3).

```
mysql> show tables;
+------------------------+
| Tables_in_badgekitapi  |
+------------------------+
| alignments             |
| applications           |
| badgeInstances         |
| badges                 |
| categories             |
| claimCodes             |
| consumers              |
| criteria               |
| evidence               |
| images                 |
| issuers                |
| migrations             |
| milestoneBadges        |
| milestones             |
| programs               |
| reviewItems            |
| reviews                |
| systems                |
| tags                   |
| webhooks               |
+------------------------+
20 rows in set (0.00 sec)
```

Joonis 3 - MySQL'i käsurida peale migratsiooni faili jooksutamist ja show tables käsu sisestamist.

Viimaseks sammuks andmebaasi loomisel on ühe süsteemi lisamine systems tabelisse. BadgeKit'i rakendusliides vajab vähemalt ühte süsteemi oma tabelisse töötamiseks. Süsteemi tabelisse süsteemi lisamise süntaks on järgnev:

```
INSERT INTO systems (slug, name, url) VALUES ('badgekit', 'Your System', 'http://localhost:3000');
```

Kus tabeli tulpade väärtused peavad olema:

- slug - loodud BadgeKit'i rakendusliidese nimi inimloetaval muutuja kuhul. Seda kasutakse hiljem BadgeKit'i rakendusliidesega suhtlemisel.
- name - Loodud BadgeKit'i rakendusliidese nimi või organisatsiooni nimi, kes BadgeKit'i rakendusliidestkasutab.
- url - BadgeKit'i rakendusliidese aadress.

Järgmise sammuna tuleb BadgeKit'i rakendusliides käivitada. Selleks tuleb jooksutada käsureal järgnev käsk:

```
nf start
```

Kui BadgeKit'i rakendusliides alustab tööd edukalt, peab tekkima käsureale vastav kirje (vt joonis 4).

```
➜ badgekit-api git:(master) ✗ nf start
[WARN] No ENV file found
[OKAY] Trimming display Output to 86 Columns
22:23:54 web.1 |  badgekit-api listening at http://0.0.0.0:8080
```

Joonis 4 - Käsurida peale BadgeKit'i rakendusliidese edukat käivitamist.

Viimaseks kontrolliks tuleb avada veebilehitseja etteantud aadressil, kui BadgeKit'i rakendusliides on töökorras, väljastab veebilehitseja BadgeKit'i rakendusliidese info (vt joonis 5).

Joonis 5 - Veebilehitsejas olev tekst peale BadgeKit'i rakendusliidese edukat käivitamist

3. BadgeKit’i rakendusliidese kasutamine

BadgeKit’i rakendusliidese kasutamiseks on sisuliselt kaks võimalust. Esimeseks võimaluseks on kasutada juba olemasolevaid graafilisi kasutajaliideseid, teiseks võimaluseks on kirjutada BadgeKit’i rakendusuliidesega suhtelmisfunktsionaalsus ise. Kuna kogu suhtlus käib kasutades HTTP meetodeid, on võimalik BadgeKit’i rakendusliidest kasutada enamiku raamistikkudega, millel on HTTP kasutamisfunktsionaalsus olemas.

Nimekirja graafilistest kasutajaliidestest võib leida Badge Alliance’i kodulehelt (http://www.badgealliance.org/badge-issuing-platforms/). Badge Alliance on organisatsioon, mis edustab Open Badge’i tarkvara kasutamist. Mozilla on ka enda BadgeKit’i rakendusliidesegraafilise tarkvara välja arendanud, kuid see on hetkel ainult saadaval avatud beetaversioonina (https://badgekit.org). Tarkvara kood on aga avatud ja seda saab alla laadida Mozilla ametilikust GitHub’i kontolt: https://github.com/mozilla/openbadges-badgekit.

BadgeKit’i rakendusliides on ehitatud eesmärgiga, et seda saab integreerida kõikidesse veebilehtedesse. Selles peatükis uurib autor rakendusliidese kasutamiseks vajalikke osasi ja toob välja näiteid põhilistest BadgeKit’i rakendusliidese kasutusvõimalustest.

3.1 BadgeKit’i rakendusliidese andme hierarhia

BadgeKit’i rakendusliitmel on kolmetasandiline hierarhia, mis on loodud eesmärgiga, et üks virtuaalmärkide väljastamise mootor oleks kasutatav mitme organisatsiooni ja organisatsioonisisese osakonna vahel. Andme hierarhia määrab kasutaja õigused märkide üle, mida kasutaja haldab, samuti määrab hierarhia sidumispunktide aadressid. Hierarhia kolm konteinerit, välimisest sisemiseni on:

- Süsteemid (*systems*) - Süsteem on kõrgeima taseme admistraator ühes BadgeKit’i eksemplaris. Ühes BadgeKit’i eksemplaris võib olla üks või rohkem süsteemi ja üks süsteem võib sisaldada null kuni mitu väljastajat. Ilma ühegi süsteemita ei saa BadgeKit’i rakendusliide töödata. (GitHub, 2014a)

- Väljastajad (*issuers*) - Väljastaja on keskmise taseme administraator. Iga väljastaja kuulub ühte süsteemi. Välajastaja võib sisaldada null kuni mitu programmi. Väljastaja on reeglina üks organisatsioon näiteks kool või ettevõte. (GitHub, 2014b)
- Programmid(*programs*) - Programm on madalaima taseme administraator. Iga programm kuulub ühe väljastaja alla. Programm on reeglina organisatsiooni alamüksus, mis väljastab märke ühes konkreetses valdkonnas, näiteks koolis mingi aine raames väljastatavad märgid. (GitHub, 2014c)

BadgeKit'i rakendusliidesega tehtud märke saab kõigi hierarhia taseme konteineritega ühendada, ehk kui soovid jooksutada rakendusliidest ainult isiklikuks tarbeks, ei ole vaja erinevaid tasemeid luua ja võib salvestada kõik märgid ühe süsteemi külge. Kõik eelpool mainitud konteinerid salvestakse rakendusliidese andmebaasi, kus neile antakse inimloetav unikaalne ID-kood, mida kasutakse rakendusliidese väljakutsumisel konkreetse konteineri poole pöördumiseks.

3.2 BadgeKit'i rakendusliidese sidumispunktid

Sidumispunktid(*endpoints*) on staatilised aadressid mis on defineeritud igas serveripoolses veebi rakendusliideses, aadressid tähistavad kohti kus kolmanda osapoole tarkvara saab ligipääsu rakendusliidesesse. (Rohlin, kuupäev puudub)

BadgeKit'i rakendusliidesega suhtelmiseks saab kasutada HTTP meetoteid: GET, POST, PUT, DELETE. BadgeKit'i rakendusliidese sidumispunktid on ülessehitatud kasutades eelpool mainitud hierarhia loogikat. Kõik päringu aadressid algavad BadgeKit'i rakendusliidese instalatsiooni aadressiga, millile lisatakse sidumispunkti aadress. Kogu süsteemi juur on aadressil /systems, see tähendab, et kõik sidumispunkti aadressid algavad:

```
/systems
```

Kui eelpool mainitud aadressile GET päring teha, tagastab ta nimekirja süsteemidest, mis on loodud selles BadgeKit'i rakendusliidese instalatsioonis.

Lisaks saab aadressile kaasa anda eel pool mainitud unikaalse konteineri inimloetava ID-koodi mis teeb iga päringu spetsiifilisemaks. Näiteks, eelpool mainitud juhedis tekitas

autor süsteemi mille inimloetav ID-kood oli badgekit, päring ainult sellele süsteemile näeks välja selline:

`/systems/badgekit`

Kui eel pool mainitud aadressile GET päring teha, tagastab ta konkreetse süsteemi nimega badgekit mis on loodud selles BadgeKit'i rakendusliidese instalatsioonis. Aadressidele saab lisada ka väljastajad ja programmid, nendele omakorda inimloetavad ID-koodid, et päringut veel täpsemkas teha. Üldmudelina näevad päringute aadressid välja sellised:

`/systems`

`/systems/<slug>/<endpoint-name>`

`/systems/<slug>/issuers/<slug>/<endpoint-name>`

`/systems/<slug>/issuers/<slug>/programs/<slug>/<endpoint-name>`

Kus `<slug>` on temast eespool oleva konteineri või objekti inimloetav ID-kood ja `<endpoint-name>` on konkreetse sidumispunkti nimi, mille poole pöördutakse. Igale BadgeKit'i rakendusliidese sidumispunktile vastab andmebaasis tabel, millega tehakse muudatusi. Sidumispunktiteks on:

- Systems - Eelpool mainitud süsteemide haldamiseks.
- Issuers - Eelpool mainitud väljastajate haldamiseks.
- Programs - Eelpool mainitud programmide haldamiseks.
- Badges - Märkide haldamiseks, mis süsteem saab väljastada.
- Claim codes - Märkide lunastamise koodide haldamiseks. Märgi lunastmise koodiga saavad kasutajad märke seljakotti lisada. Samuti saab vaadata kas kasutaja on koodiga enda märgi lunastanud.
- Issuing - Märkide väljaandmine konkreetsele kasutajale. BadgeKit'i rakendusliidesega märgi väljastamine tähendab sisuliselt badgeInstance tabelisse kirje tegemist, ehk iga välja antud märgi kohta luuakse uus märgi eksemplar.
- Applications - Märke väljastakse reeglina peale mingite kriteeriumite täitmist. Application haldab märgi väljastamise taotluste esitamist. Taotlusega saadetakse

tõendusmaterjali, mille põhjal see hiljem ülevaadatakse ja siis kas kinnitatakse või lükatakse tagasi.

- Reviews - Taotluste ülevaatamise haldus. Siin saab taotlusele tagasisidet anda ja teha kirjeid, millised kriteeriumid on täidetud ja millised ei ole.
- Milestones - Väljastab tähtsündmusmärke. Tähtsündmuse märgid on autasuks teiste märkide teenimiste eest, näiteks: kui kasutaja on teeninud mingi eeldefineeritud grupi märkidest väljastatakse talle automaatselt märk selle saavutuse eest.

Nimekiri kõikidest BadgeKit'i rakendusliidese sidumispunktidest koos võimalike HTTP meetodite ja võimalike päringu andmetega, leiab BadgeKit'i ametliku GitHub lehelt: https://github.com/mozilla/badgekit-api/blob/master/docs/api-endpoints.md.

3.3 BadgeKit'i rakendusliideses päringute tegemine

Nagu eel pool mainitud toimub suhtlus BadgeKit'i rakendusliidesega kasutades HTTP protokolli. Päringute tegemist võib jagada kahe astmeliseks protsessiks, esimesena tuleb genereerida autoriseerimis *token*, teisena tuleb tuleb teha HTTP päring. Selles peatükis uurib autor kuidas autoriseerimis *token* luua ja mida see teeb ja mis peab päringu sisu olema.

3.3.1 Päringu autoriseermine JSON Web Token'iga

Kõik päringud peavad olema autoriseeritud kasutades JSON Web Token'it (edasipidi JWT). JWT on JSON'i baasil ehitatud avatud standard veebirakendustele oma vaheliseks suhtluseks. JWT *token* on krüpteeritud sõne (*string*) mis koosneb kolmest punktiga eraldatud osast:

- Päis (*header*) - Tavaliselt sisaldab päis kahte muutujat. Esimene tähistab mis tüüpi *token*'iga on tegemist ja teine tähistab mis krüpteerimisalgoritmi on kasutatud (vt joonis 6).

Joonis 6 - Päise sisu kui JWT *token* krüpteeritakse kasutades HMAC-SHA256 algoritmi.

- Kasulik koormus (*payload*) - Sisaldab päringuga saadetavat infot. Kolm välja on alati kohustuslikud, nendeks on:
    - Key - Tuleviku perspektiivis lisatud väli. Hetkel seeb BadgeKit'i rakendusliitmele ligi ainult ühe võtmega ja selle välja väärtus peab olema: `master`.
    - Method - HTTP meetod mida päringus kasutakse. Võimalikud väärtused on: `GET`, `POST`, `PUT`, `DELETE`.
    - Path - Sidumispunktini viiv veebiaadressi tee. Peab vastama aadressile kuhu konkreetne päring tehakse, näiteks kui päring tehakse aadressile `http://0.0.0.0:8080/systems/badgekit/badges?archived=true`, siis path väärtus peab olema `/systems/badgekit/badges?archived=true`. Päringusõne kaasa arvatud.

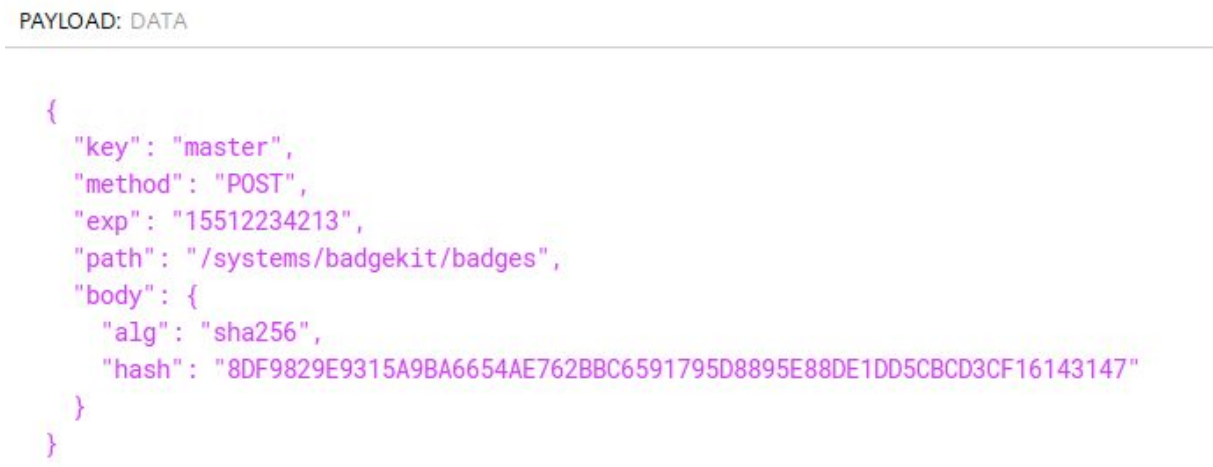
    Kui HTTP päringu meetod BadgeKit'i rakendusliidesele on kas `POST` või `PUT`, peab lisama:

    - Body - Body väärtus peab olema objekt tüüpi muutuja kahe väärtusega, milleks on:
        - Alg - Räsifunktsiooni nimi millega body teine muutuja hash on räsitud. Soovituslikult sha256 räsifunktsioon.

- hash - Objekt muutuja päringuga saadetavatest andmetest, mis on räsitud algmuutujas kirjas olevas räsifunktsiooniga (vt joonis 7).

Soovituslik, aga mitte nõutav väli, mida kasutakse serverivaheliste sünkroniseerimisprotsesside puhul:

- Exp - JWS *token*'i aegumis aeg Unix'i ajatemplina. Soovituslikult väikest väärtust peaks kasutama, näiteks 30-60 sekundit.



Joonis 7 - Näide kasulikust koormusest POST päringu puhul.

- Allkiri (*signature*) - Turva element, mis tagab rakendusele teadmise, et saatja on usaldusväärne ja et selle sisu ei ole saatmise ajal muudetud. Allkiri koosneb eelpool mainitud krüpteeritud kujul päisest ja kasulikust koormusest, mis krüpteeritakse veel omakorda päises äramainitud algoritmiga, kasutades rakenduse parooli võtmena (vt joonis 8).

Joonis 8 - Allkirja loomisfunktsioon, kasutades HMAC-SHA256 krüpteerimisalgoritmi. Funktsioonis on näha päise, kasuliku koormuse ja parooli kasutamist HMAC-SHA256 krüpteeringu sooritamiseks.

Token näeb lõppkujul välja selline:

```
xxxxx.yyyyy.zzzzz
```

Kui päring on rakendusliidese serverisse jõudnud, kontrollitakse päringu sisu, meetoteid ja päist JWT *token*'is olevate andmete vastu. See tegevus kaitseb võimaluse eest, et keegi päringu sisu saatmise ajal modifitseerib. Kui *token* ja päringu andmed ei ühtu, ei viida päringut lõpuni. JWT pakub õppe ja silumise eesmärgil väga head veebipõhist tööriista, mis päris ajas krüpteerib ja dekrüpteerib JWT *token'*eid: https://jwt.io/.

Kõik HTTP päringud BadgeKit'i rakendusliitme süsteemi tehakse JWS *token*'iga. Päringu päisesse tuleb lisada autoriseerimis päis (*Authorization*), mille väärtus on:

```
JWT token = "xxxxx.yyyyy.zzzzz"
```

Kus xxxxx.yyyyy.zzzzz on päringust genereeritud JWS *token.* (JWT, kuupäev puudub)

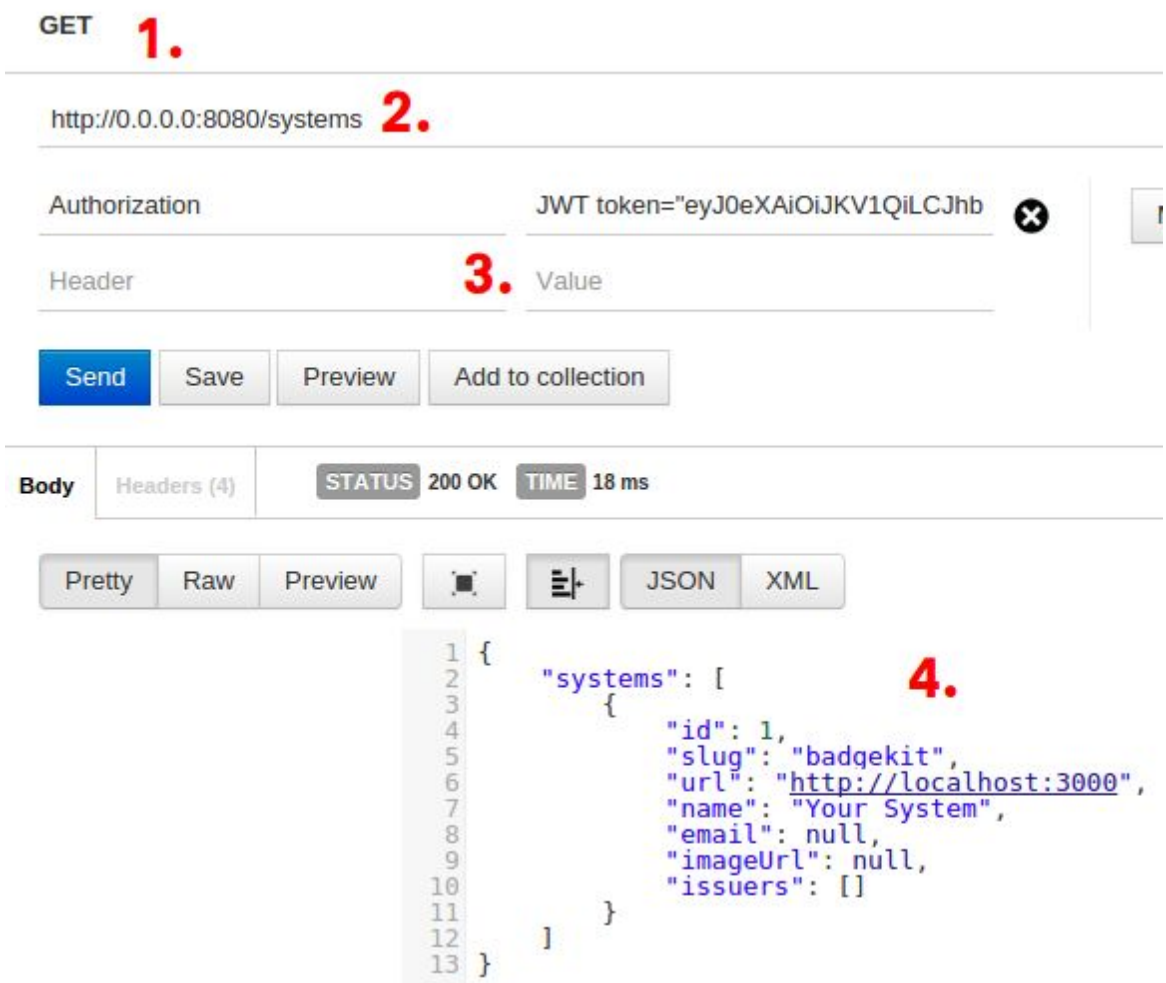
3.3.2 Päringute sisu GET ja DELETE päringute puhul koos näitega

GET ja DELETE päringute tegemiseks on vaja ainult seada eelpool mainitud autorisatsiooni päis. Päringuks vajaminev informatsioon on JWT *token*'i sees kirjas. BadgeKit'i

rakendusliides kontrollib *tokeni*'i sees olevat informatsiooni päringu aadressi ja HTTP meetodi vastu. Näidis GET päringu puhul nägi välja JWT *token*'i kasulik koormus selline:

```
{
      "key": "master",
      "method": "GET",
      "path": "/system"
}
```

Kus *token*'i kasuliku koormuse method muutuja sobib päringu HTTP meetodiga (vt joonis 9.1), path muutuja on päringu aadressi ilma domeeni nimeta (vt joonis 9.2). JWT *token* on saadetud päringu päises autoriseerimispäisena (vt joonis 9.3).



Joonis 9 - Edukas GET päring BadgeKit'i rakendusliidese süsteemi kasutades Postman REST Client'i. Joonisel on näha: 1. kasutatavat HTTP meetodit, 2. päringu aadressi, 3. päringu autoriseerimispäise olemasolu JWT *token*'iga, 4. positiivset vastust, et päring on edukalt sooritatud.

3.3.3 Päringute sisu POST ja PUT päringute puhul koos näitega

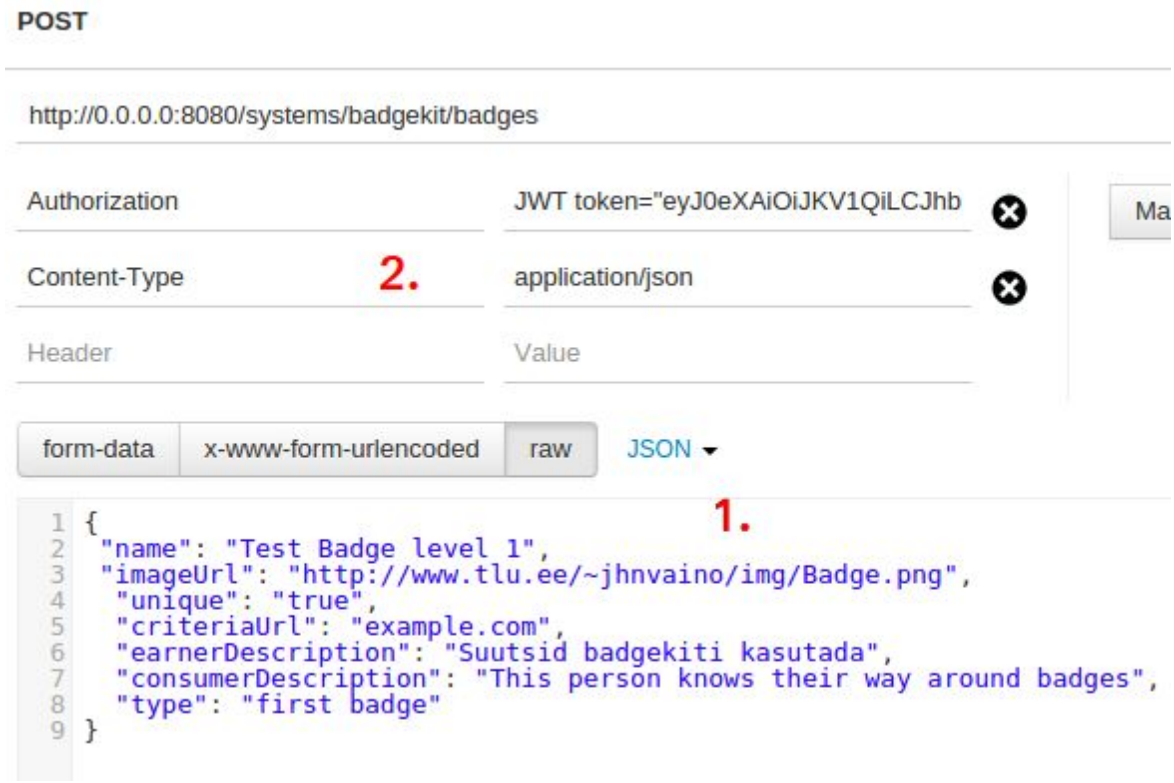
POST ja PUT päringute puhul on päringu ülesehitamiseloogika enamjaolt sama, mis GET ja DELETE päringute puhul, aga esineb 3 erinevust:

- Eelpool mainitud JWT *token*'i kasulik koormus tuleb lisada body väli, mis sisaldab kasutatud räsifunktsiooni ja sama räsifunktsiooniga räsitud päringu andmeid (vt joonis 10).


```
{
  "key": "master",
  "method": "POST",
  "exp": "15512234213",
  "path": "/systems/badgekit/badges",
  "body": {
    "alg": "sha256",
    "hash":
"3bea0ee9a16e2641e98ecd71b3778bf3837c
024f5c9efeee299bbef8f3afc183"
  }
}
```


Joonis 10 - Näidis POST meetodiga tehtava päringu JWT *token*'i kasulikust koormust. On näha, et kasutakse body väljakontroll andmete saatmiseks.

- Päringule tuleb lisada andmed. Andme tüübiks võib olla (vt joonis 11.1):
  - application/json - JSON'i objekt.
  - application/x-www-form-urlencoded - Päringusõne.
  - multipart/form-data - HTML vormi andmed.
- Päringu päisesse tuleb lisada, mis tüüpi andmeid POST või PUT päringuga saadetakse (vt joonis 11.2).



Joonis 11 - Edukas POST päring BadgeKit'i rakendusliidese süsteemi kasutades Postman REST Client'i. Joonisel on näha: 1. on lisatud andmed JSON objektina, 2. päringu päisesse on lisatud andme tüübi päis, mis täpsustab, et andmed on JSON objekti kujul.

Kokkuvõte

Käesoleva bakalaureusetöö eesmärgiks oli anda ülevaade Open Badges virtuaalse tunnustamise süteemi ülesseadmisest ja kasutamisest ning kasutusele võtmise keerukuse hindamine. Lisaks uuris autor virtuaalsete tunnustamise süteemide ajalugu ja selle väärtust töö- ja õppe valdkondades.

Töö käigus kujunes välja hea ülevaade Open Badges süsteemi võimekusest, ülesseadmis lihtsusest ja funktsionaalsusest. Internetist leitav materjal jäi kohakuti poolikuks, aga õnneks peale katsetamist ja lisainfo otsimist sai katsetus rakendusliidest edukalt kasutada. Autori hinnangul on rakendusliidesee kasutuselevõtmine juba eksisteerivas virtuaalse tunnustamise süsteemis väga lihtne. Tuleb ainult lisada vajalikud BadgeKit'i rakendusliidese päringud eksisteerivasse rakendusse. Täiesti personaalse graafilise liidese loomine on aga suurem projekt, õnneks saab selleks kasutada teiste poolt loodud tarkvara.

Üldhinnanguna usub autor, et Open Badges on väga hea raamistik virtuaalse tunnustuse jagamiseks. Kuna teda on lihtne integreerida, kasutada ja kasutajasõbralik tunnustuse saajatele. Lisaks, kuna ta on loodud üldstandardina, annab ta võimaluse koguda virtuaalset tunnustust ühte kohta paljudest virtuaalse tunnustuse süsteemidest. Nimekiri Open Badge'i partnerlehtedest, kes Open Badge'i kasutab, on pikk (http://openbadges.org/participating-issuers/), siia lisanduvad veel ka rakendused, mis kasutavad Open Badge'i seda ise hostides.

Summary

The purpose of this bachelor thesis was to give an overview of the virtual recognition system Open Badges. The author describes how to set it up, how to use it and evaluates how hard it is to integrate Open Badges to a system. In addition the author researched the history of virtual recognition systems and their worth in a work or school environment.

During the process of writing this thesis, a good overview of the capabilities, setup difficulty and functionality of an Open Badges system took shape. The material found online was not great, but thankfully after some testing and searching for extra information the author got the test API to working condition. In the author's opinion, integrating the BadgeKit API into an already existing virtual recognition system is quite easy. You would just need to add the needed functionality into the existing application. Setting up your own graphical interface is a slightly bigger project, but it is possible to use already pre-existing ones made by other people.

All in all, the author believes that Open Badges is a great framework for building virtual recognition systems. It is easy to integrate, use and it is user friendly towards people receiving recognition. In addition, it is built with the idea of it being the overall standard for virtual recognition systems, making it have the functionality of gathering the user's virtual recognition in one place. The list of partner sites using Open Badges is quite long (http://openbadges.org/participating-issuers/) and that does not count for applications that host it on their own.

Kasutatud kirjandus

Hamari, J., & Eranti, V. (2011). Framework for Designing and Evaluating Game Achievements. Loetud aadressil: http://www.digra.org/wp-content/uploads/digital-library/11307.59151.pdf

Hillard, K. (2013, 26. oktoober). Activision Badges - The Original Gaming Achievement [ajaveebipostitus]. Loetud aadressil: http://www.gameinformer.com/b/features/archive/2013/10/26/activision-badges-the-original-gaming-achievement.aspx

Giantbomb. (2015). Let the Achievement Hunt Continue. Loetud 27. aprillil 2016 aadressilt: http://www.giantbomb.com/steam-achievements/3015-3280/

Giantbomb. (2016). PlayStation Trophies. Loetud 20. aprillil 2016 aadressilt: http://www.giantbomb.com/playstation-trophies/3015-198/

Holmes, B. (2014, 30. juuni). Boost College Applications With a Positive Online Presence [ajaveebipostitus]. Loetud aadressilt: http://www.usnews.com/education/blogs/college-admissions-playbook/2014/06/30/boost-college-applications-with-a-positive-online-presence)

Fletcher, L. (kuupäev puudub). 7 Ways to Prove Your Worth on Your Resume [ajaveebipostitus]. Loetud 14. aprillil 2016 aadressilt: http://www.blueskyresumes.com/blog/7-ways-to-prove-your-worth-on-your-resume-with-examples/

Zomick, B. (2013, 14. september). What's It Worth? Certificates, Badges and Online Portfolios [ajaveebipostitus]. Loetud 19. aprillil 2016 aadressilt: http://www.skilledup.com/articles/whats-it-worth-certificates-badges-online-portfolios

StackExchange. (kuupäev puudub). At what point do you put your SO reputation in your resume? [closed]. Loetud 28. aprillil 2016 aadressilt: http://meta.stackexchange.com/questions/58947/at-what-point-do-you-put-your-so-reputation-in-your-resume

Badges/FAQs. (kuupäev puudub). Mozilla wiki. Loetud 18. aprillil 2016 aadressilt: https://wiki.mozilla.org/Badges/FAQs#Who_can_issue_badges.3F

Mozilla support. (kuupäev puudub). What is a Mozilla Backpack?. Loetud 20. aprillil 2016 aadressilt: https://support.mozilla.org/ee/kb/what-is-a-mozilla-backpack

GitHub. (2014a, 16. juuni). Systems. Loetud 20. aprillil 2016 aadressilt: https://github.com/mozilla/badgekit-api/blob/master/docs/systems.md

GitHub. (2014b, 16. juuni). Issuers. Loetud 20. aprillil 2016 aadressilt: https://github.com/mozilla/badgekit-api/blob/master/docs/issuers.md).

GitHub. (2014c, 16. juuni). Programs. Loetud 20. aprillil 2016 aadressilt: https://github.com/mozilla/badgekit-api/blob/master/docs/programs.md).

GitHub. (2014d, 29. september). Get Started with Open Badges [ajaveebipostitus]. Loetud 20. aprillil 2016 aadressilt: https://github.com/mozilla/openbadges-backpack/wiki/Get-Started-with-Open-Badges

Rohlin, A. (kuupäev puudub). The Definition of Web Service EndPoint. Loetud 21. aprillil 2016 aadressilt: http://www.ehow.com/info_12212371_definition-service-endpoint.html

JWT. (kuupäev puudub). Introduction to JSON Web Tokens. Loetud 21. aprillil 2016 aadressilt: https://jwt.io/introduction/