Samuel Ristolainen

STEAM WEB-RAJAPINNAN KÄYTTÖ

Datan hakeminen Web-rajapinnasta

STEAM WEB-RAJAPINNAN KÄYTTÖ

Samuel Ristolainen
Steam Web-rajapinnan käyttö
Syksy 2018
Tietojenkäsittelyn tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietojenkäsittely, Web-sovelluskehitys

Tekijä(t): Samuel Ristolainen
Opinnäytetyön nimi suomeksi: Steam Web-rajapinnan käyttö
Työn ohjaaja(t): Jouni Juntunen
Työn valmistumislukukausi ja -vuosi: Syksy 2018
Sivumäärä: 27

Opinnäytetyön toimeksiantajana toimi OAMK:n yrityshautomoprojekti Gamership, jonka pyrkimyksenä on auttaa yksinäisiä ja peliseuraa kaipaavia pelaajia löytämään seuraa heidän pelaamiinsa nettimoninpeleihin.

Tämän opinnäytetyön tavoitteena oli tutustua peliyhtiö Valven tarjoamaan Steam-pelipalvelun rajapintaan, joka tarjoaa palvelun käyttäjistä ja peleistä keräämää tietoa. Tämän lisäksi tarkoituksena oli toteuttaa tiedonhaku kyseisestä rajapinnasta verkkosivulle käytettäväksi.

Tässä opinnäytetyössä käsitellään askeleita, jotka vaaditaan Steam ohjelmointirajapinnan käyttöön ja sieltä tiedonhakuun, tässä tapauksessa web-sovelluksessa käytettäväksi. Työn toteuttamiseen käytettiin PHP-ohjelmointikieltä, ja sille kehitettyä Codeigniter-sovelluskehystä käyttäen.

Lopputuloksena opinnäytetyöstä saatiin toimiva rajapintaintegraatio sivustolle, jota kautta käyttäjästä haettu data pystytään tallentamaan sovelluksen tietokantaan. Työtä olisi myös mahdollista jatkokehittää muuttamalla käyttäjästä saatavaa tilastokokonaisuutta. Tällä mahdollistettaisi saman tasoisten pelaajien saattaminen yhteen.

Asiasanat: ohjelmointirajapinta, web-sovelluskehitys, sovelluskehykset

ABSTRACT

Oulu University of Applied Sciences
Degree programme, option

Author(s): Samuel Ristolainen
Title of thesis: Steam Web-API integration
Supervisor(s): Jouni Juntunen
Term and year when the thesis was submitted: Fall 2018
Pages: 27

This thesis was made for Gamership, a project in OUAS-incubator program, whose goal was to help lonely video gamers find company to play multiplayer games with. The purpose of the thesis was to become familiar with video game developer Valve's software distribution platform Steam and its web-API, which offers interfaces from which software can fetch data about Steam's users. The goal was to then develop a web-based application which could fetch its user's data from Steam to help them find similarly skilled people to play games with.

This thesis depicts the different steps required to integrate Steam Web API to a website and how to get the data needed from it. The web application was developed using Codeigniter, a programming framework made for PHP.

As the result we have an application with working API-integration that can authenticate people using an OpenID authentication to enable requesting data for a user. The application is then able to process the data that it got from the interface and save relevant data to its database. The project could easily be improved by tweaking what data gets saved for users, perhaps even implementing a ranking system to further help matchmaking.

Keywords: API, web-development, frameworks

SISÄLLYS

1 JOHDANTO

Peliyhteisöt ovat pelien ympärille muodostuneita yhteisöjä, joita on ollut olemassa jo jossain muodossa vuodesta 1972, jolloin peliyhtiö Atari aloitti ensimmäisen kaupallisen pelin, Pongin, myynnin. (The International Game Museum, viitattu 21.02.2018)

Pongin julkaisun jälkeen alkoi arcade-pelikoneita ilmestymään julkisille paikoille, esimerkiksi kauppakeskuksiin, baareihin ja keilahalleihin ympäri maailmaa. Ensimmäisten videopelikonsoleiden saavuttua markkinoille pelaaminen oli jonkin aikaa suurelle yleisölle lähinnä mahdollista arcade-pelihalleissa. Kotipelaamisen vähyys johtui verrattain vielä vähäisestä määrästä väritelevisioita ihmisten kodeissa, sekä konsoleiden korkeista hinnoista. (BMIGaming.com, viitattu 21.02.2018)

Vuonna 1980 Atarin VCS-pelikonsolille julkaistu Space Invaders antoi kehittyvälle alalle suuren sysäyksen pelikonsoleiden myynnin noustessa kahteen miljoonaan kappaleeseen tuona vuonna. Koti- ja arcadesalipelaamisen kukoistaessa alkoivat myös ensimmäisten pelaamiseen liittyvien harrastelehtien myötä varsinaisia peliyhteisöjä muodostumaan niiden ympärille. Seuraavan suuren sysäyksen peliyhteisöille antoi lähiverkkopelaaminen Pathway to Darkness -pelin tullessa markkinoille vuonna 1993. Tämän jälkeen Windows 95 -julkaisun sekä edullisimpien verkkokorttien markkinoille tulo lisäsi vielä entisestään lähiverkkopelaamisen suosiota. (Techcrunch. 31.08.2015)

Lähiverkkopelaaminen, sekä myöhemmin internetin välityksellä tapahtuva pelaaminen mahdollistivat pelaajien kesken käymän kilpailun ja kanssakäymisen, joka entisestään paransi videopelaamisen sosiaalisia puolia. Internet-yhteyksien yleistyminen, ja niiden nopeuksien kasvaminen 2000-luvulla mahdollisti nykyään tunnetussa muodossa olevan verkkopelaamisen syntymisen. Lisäksi kehitys mahdollisti nykyään suosittujen, internetissä sijaitsevien verkkopeliyhteisöjen synnyn ja toiminnan.

Nykyään eri peleille ja pelikonsoleille omistettuja peliyhteisöjä voi löytää lukemattomia määriä internetistä. Peliyhteisöjä löytyy myös keskitettyinä hyvin erilaisille, ja joissain tapauksissa todella yksinkertaisille aihealueille, kuten esimerkiksi pelien speedrunnaukselle (pelin läpäisy mahdollisimman lyhyessä ajassa) omistetut yhteisöt.

2 TAUSTA

Opinnäytetyön toimeksiantajana toimi OAMK:n yrityshautomoprojekti Gamership. Projektin tavoitteena oli auttaa yksinäisiä pelaajia löytämään peliseuraa heidän pelaamiinsa nettimoninpeleihin.

Ohjelmointikielenä työssä käytetään PHP-ohjelmointikieltä, jolla alustava Gamership-verkkosivusto luotiin käyttäen PHP-sovelluskehys CodeIgniteria. Tämän lisäksi PHP:tä käytetään OpenID-tunnistautumiseen sekä Steam Web rajapintahakuihin, joita varten työssä hyödynnetään valmiiksi saatavissa olevia software development kittejä ja ohjelmointikirjastoja. Steamin API:a hyödynnetään työssä johtuen Steamin suuresta käyttäjämäärästä, sekä sen tarjoamasta mahdollisuudesta ylipäätään hakea sen käyttäjien tietoja, jonka avulla taitotasoltaan samanlaiset pelaajat voitaisi yhdistää toisiinsa. Steamin rajapinta on myös tarkoitukseen hyvin sopiva, sillä iso osa suosituimmista tietokoneella pelattavista moninpeleistä toimii Steamin kautta. Täten yhtä alustaa hyödyntämällä saavutetaan suurin mahdollinen määrä potentiaalisia käyttäjiä.

Opinnäytetyön tavoitteena on soveltaa tekijän omaa web-ohjelmointiosaamista toimivan web-sovelluksen kehittämiseen, sekä tutustua toisen osapuolen tarjoamien rajapintojen hyödyntämiseen. Idea opinnäytetyöhön syntyi Gamership-verkkosivuston ollessa alustavassa kehityksessä, johtuen sivuston tarpeesta saada edellä mainittuja tilastoja käyttöön toimintaansa varten.

Työn kehitystehtävänä on ohjelmoida kehityksessä olevalle Gamership-verkkosivustolle mahdollisuus tiedon hakuun Steam ohjelmointirajapinnasta, jonka helpottamiseksi toteutetaan tunnistautuminen Steamiin internetin välityksellä. Tunnistautumista varten opinnäytteessä käytetään OpenID-autentikointiprotokollaa. OpenID-tunnistautumisen kautta Steamin rajapinta palauttaa käyttäjän SteamID: n. Tämän avulla sivustolle voidaan hakea Steamin rajapinnasta pelipalvelun keräämää tietoa käyttäjän pelaamista peleistä sekä muuta käyttäjään liittyviä tilastoja. Opinnäytetyössä käsitellään aluksi opinnäytteeseen liittyviä asioita teoriaosuudessa, jossa selvitetään muun muassa mitä ovat Steam, Steam Web API ja OpenID. Työssä tutustutaan myös hieman verkkopelaamiseen ja peliyhteisöihin, sekä perehdytään myös tarkemmin SDK ja API -käsitteisiin.

Opinnäytteen käytännön osuudessa tarkastellaan sovelluksen toteutusta ja toimintaa. Lisäksi paneudutaan tarkemmin OpenID-tunnistautumisen toteutukseen sekä Steamin rajapinnan käyttöön ja sieltä tuleviin tietoihin.

Codeigniter on alun perin Ellislab-yhtiön toimitusjohtajan Rick Elliksen vuonna 2006 kehittämä MVC (Model view controller) periaatetta noudattava sovelluskehys PHP-ohjelmointikielelle. Codeigniterin perustana toimivat suurimmalta osalta Ellislabin ExpressionEngine-julkaisujärjestelmästä lainatut luokkakirjastot, avustajat ja alajärjestelmät. Codeigniteristä on poistettu kaikki sovelluskohtaiset toiminnallisuudet, jotta se olisi mahdollisimman yksinkertainen ja kevyt, jolla on pyritty mahdollistamaan nopea verkkosivu- ja verkkosovelluskehitys. (Ellislab, 2014)

Kun Codeigniteriä kehittävät tahot vuonna 2014 julkaisivat halunsa löytää uuden ylläpitäjän projektilleen, ilmoittautui British Columbia Institute of Technology halukkaaksi ottamaan projektin omakseen, jonka jälkeen Ellislab luovutti projektin omistuksen BCIT:lle. Nykyään Codeigniterin kehityksen kärjessä toimii edellämainitusta Reactor Teamista huolella valitut kehittäjät. (Ellislab, 2014).

3 STEAM

Steam on yhdysvaltalaisen peliyhtiö Valven vuonna 2003 julkaisema, alun perin vain Counter-Strike -räiskintämoninpelin päivitysten jakeluun tarkoitettu ohjelma, joka on aikojen saatossa päätynyt yhdeksi maailman suurimmista online-pelialustoista. Julkaisunsa jälkeen Steam saavutti todellisen suosionsa Valven avatessa Steamin kaupan muidenkin peliyhtiöiden pelien myynnille, ja alkoi muillakin tavoilla kehittää alustaa monipuolisemmaksi. Nykyään Steamissa on tarjolla tuhansia eri pelejä, ja aktiivisia käyttäjiä alustalta löytyy Valven mukaan 35 miljoonaa. Tarjolla olevista peleistä osa on maksullisia, sekä vuoden 2009 jälkeen Steamista voi myös löytää suuren määrän ilmaispelejä Valven avattua mahdollisuuden ilmaispelien (engl. free-to-play) jakelulle. (Valvesoftware. 2003, 2017)

Pelien lisäksi Steamissa on myynnissä myös erilaisia hyötyohjelmia, kuten esimerkiksi videonkäsittely- ja äänenkäsittelyohjelmia. Näiden lisäksi Steamissa on mahdollista lähettää pikaviestejä ystävälistalla oleville henkilöille, jakaa videoita ja lähettää livekuvaa pelaamisestaan Steam Communityssä, käydä kauppaa pelien kosmeettisilla esineillä Steam Marketissa, sekä jakaa itse tehtyjä modifikaatioita Steam-peleihin Steam Workshopissa.

3.1 Steam-yhteisöt

Steam-yhteisö (engl. Steam Community) on Steamin peleille ja hyötyohjelmille suunnattu osuus Steamia, josta löytyy yhteisön sekä virallisen tahon niihin tekemää sisältöä. Sisältöön kuuluvat peleihin liittyvien kuvankaappausten, taideteosten, suoratoiston, videoiden, uutisten, oppaiden ja arvosteluiden jakamismahdollisuus.

Steam-yhteisöjen toisena osana toimii Steam Workshop, jossa käyttäjät voivat etsiä ja ladata toisten käyttäjien tekemää sisältöä peleihin ja ohjelmistoihin, sekä jakaa itse niihin tekemäänsä sisältöä. Muut käyttäjät voivat myös arvostella ja käydä keskustelua sisällöstä, jonka lisäksi sisältöön tehdyt muutokset ovat automaattisesti käyttäjien nähtävillä sisältösyötteessä.

3.2 Steamworks

Steamworks on Valven pelien ja muiden ohjelmistojen kehittäjille tarjoama kokoelma ilmaisia työkaluja. Näihin työkaluihin kuuluu muun muassa Steamworks SDK, joka tarjoaa kehittäjille

valmiita ominaisuuksia, joiden tarkoituksena on avustaa ohjelmiston tai pelin julkaisussa Steam-alustalle. Steamworks SDK:n käyttö antaa kehittäjille mahdollisuuden ladata pelin tai ohjelman sisältöä Steamiin, esimerkkinä tästä voitaisi käyttää pelien sisältämät saavutukset. (Steamworks, viitattu 18.01.2018)

Steamworks SDK:n lisäksi Steamworks tarjoaa Steamin verkkorajapinnan ja dokumentaatiot sen käytölle. Steamin verkkorajapinnan käyttö ei välttämättä vaadi erillistä API-avainta, mutta jos haluaa toteuttaa vaikkapa Steamiin tunnistautumisen OpenID:tä käyttäen verkkosivulla, on sen hankinta pakollista. Perus API-avain on kaikkien Steam-käyttäjien saatavilla täyttämällä hakulomakkeen ja hyväksymällä Steam API:n käyttäjäehdot. (Steamworks, viitattu 18.01.2018)

4 TEKNOLOGISET TYÖKALUT

Tämän opinnäytetyön, ja vastaavanlaisten sovellusten kehityksessä rakennetaan hyvin harvoin kokonainen sovellus kokonaan itse, jättäen hyödyntämättä esimerkiksi erilaisia sovelluksen kehitystä nopeuttavia työkaluja. Nämä käytettävissä olevat työkalut tuovat mukanaan monia hyötyjä, ne esimerkiksi helpottavat työn toteutusta niille valmiiksi olemassa olevien ohjeiden ja dokumentaatioiden ansiosta. Monesti sovelluksen kehityksessä voidaankin hyödyntää joko jotain software development kittiä, jollekin ohjelmointikielelle tehtyjä ohjelmointikirjastoja, tai vaikkapa opinnäytetyössä käytetyn OpenID:n kaltaisia, jonkin tietyn asian toteutukseen luotuja apuvälineitä. Myös JSON:in tyyliset standardit ovat omiaan helpottamaan tämän opinnäytteen ja sen tapaisten sovellusten toteutusta tarjoamalla selkeän syntaksin datan siirrolle eri järjestelmien välillä.

4.1 Software development kit

Software development kit (SDK, tästä eteenpäin devkit), on sovellusten kehitystä varten oleva kokoelma teknologisia työkaluja. Devkitin avulla voidaan luoda sovelluksia tietyille ohjelmistopakkauksille, ohjelmistoarkkitehtuureille, laitteistoille, tietokonesysteemeille tai vastaaville alustoille. Esimerkkeinä eri devkiteistä voitaisiin käyttää Windows 7 devkittiä tai Max OS X devkittiä, jotka sisältävät työkalut kyseisille alustoille kehitettäviä ohjelmistoja varten. Yleisimmin devkittiä kullekin alustalle tarjoava osapuoli on myös kyseisen alustan kehittäjä, esimerkiksi Windows devkit on saatavilla Microsoftilta.

Devkitit sisältävät yleensä jonkin Integrated Development Environmentin (suom. ohjelmointiympäristö), jonka mukana tuleviin työkaluihin kuuluvat työkalut, joilla voidaan etsiä ja korjata vikoja (debug) sekä muut apuvälineet, joita löytyy yleisimmin eri ohjelmointiympäristöistä. Devkitit sisältävät usein myös näytekappaleita koodista ja avustavia teknisiä yksityiskohtia tai muuta dokumentaatiota, jolla pyritään selventämään alkuperäismateriaalissa olevaa tietoa. (Techterms. 15.04.2010)

Koska useimmat devkittejä valmistavat yhtiöt haluavat rohkaista kehittäjiä kehittämään ohjelmistoja heidän alustoilleen, ovat devkitit lähestulkoon poikkeuksetta ilmaisia. Devkitin kehittäjät voivat ladata sitä tarjoavan yhtiön verkkosivuilta ja aloittaa kehitystyön heti latauksen jälkeen. (Techterms.

15.04.2010)

4.2 JSON

JSON (JavaScript Object Notation) on JavaScript ohjelmointikielessä alun perin käytetty datan esitysmuotoilu, jonka tarkoituksena on tarjota standardoitu tiedonvaihtomuoto yksinkertaista tekstiä käyttäen. JSON:n etuna on sen muotoisena datan olevan helppolukuista ihmisille, ja samalla helppoa tietokoneille ja järjestelmille jäsentää sekä generoida. (JSON, 2018)

JSON on tekstiformaatti, joka on täysin itsenäinen varsinaisista ohjelmointikielistä, mutta käyttää eri C-ohjelmointikielistä, Javasta, JavaScriptistä ja Pythonista tuttuja käytäntöjä. Tämän mahdollistaakin sen käytön sovellusten väliseen keskusteluun huolimatta siitä millä ohjelmointikielellä kukin sovellus on toteutettu. (JSON, 2018)

JSON on käytettävästä ohjelmointikielestä riippuen yhdessä kahdesta mahdollisesta muodosta. Se voi olla kokoelma nimi-arvo pareja, joka useissa ohjelmointikielissä tunnistetaan muun muassa joko objektina, hajautustauluna tai assosiatiivisena listana. Toinen mahdollinen käytettävä muoto on järjestetty lista arvoja. Nämä muodot ovat universaaleja tietorakenteita, joita käytännössä kaikki modernit ohjelmointikielet tukevat jommassakummassa muodossa. (JSON, 2018)

Esimerkiksi REST-rajapinnoista haettava data on yleensä JSON muodossa johtuen sen helposta käännettävyydestä eri ohjelmointikielten välillä. Tämän lisäksi sen helppolukuisuus on suuri etu sen käytössä rajapinnoissa, sillä rajapintaintegraatioita tehdessä kehittävä osapuoli voi yksinkertaisesti lukea mitä dataa rajapinnasta saadaan.

4.3 Application programming interface

Application programming interface, API (suom. ohjelmointirajapinta), on kokoelma komentoja, funktioita, protokollia ja olioita, joiden avulla ohjelmoijat voivat luoda ohjelmistoja, jotka joko jollain tavalla hyödyntävät ulkoista järjestelmää, tai ovat vuorovaikutuksessa sellaisen kanssa. Ohjelmointirajapinnat tarjoavat ohjelmoijille standardinmukaiset komennot, joilla suorittaa yleisiä toimenpiteitä ilman, että jokaista toimintoa varten tarvitsisi kirjoittaa koodia alusta-alkaen itse. (Techterms.20.06.2016)

Erilaisia ohjelmointirajapintoja on tarjolla sekä mobiili- että työpöytäjärjestelmille. Windows ohjelmointirajapinta esimerkiksi tarjoaa ohjelmoijille mahdollisuuden hallita eri käyttöliittymäelementtejä, muun muassa ohjelmaikkunoita, vierityspalkkeja sekä dialogilaatikoita. Tämän lisäksi Windows-rajapinta tarjoaa komennot, joilla voidaan päästä käsiksi tietojärjestelmään ja sen toimenpiteisiin, kuten esimerkiksi tiedostojen luomis- ja poisto toimenpiteisiin. Lisäksi Windows ohjelmointirajapinta sisältää verkkokomentoja, jotka mahdollistavat tiedon lähetyksen lähiverkon ja Internetin välityksellä. (Techterms.20.06.2016)

Mobiililaitteiden ohjelmointirajapinnat, kuten esimerkiksi Applen iOS rajapinta, tarjoavat mahdollisuuden havaita kaikkia kosketusnäytön syötteitä, kuten näpäytyksiä, pyyhkäisyjä sekä näytön kääntöä mobiililaitteen asentoa muuttaessa. Edellä mainittujen ominaisuuksien lisäksi niihin sisältyvät yleisimmät käyttöliittymäelementit, kuten pop-up näppäimistö, hakupalkki sekä näytön alareunassa oleva navigointipalkki. IOS rajapinta, kuten myös muutkin mobiilirajapinnat, tarjoaa valmiiksi määritellyt funktiot laitteen kameran, mikrofonin ja kaiuttimien käyttöä varten. (Techterms.20.06.2016)

Käyttöjärjestelmien ohjelmointirajapinnat tulevat yleensä integroituina kunkin käyttöjärjestelmän omaan devkittiin. Eri käyttöjärjestelmien rajapintojen ollessa yleensä kokoelma varsin laajoja ominaisuuksia, voivat toisenlaiset rajapinnat olla hyvinkin yksinkertaisempia. Esimerkiksi jokin verkkosivu voi tarjota rajapintaa verkkosivujen kehittäjille, joka päästää kehittäjät käsiksi vain johonkin tiettyyn tietoon. Verkkorajapinta voi yksinkertaisimmillaan olla vain pieni kokoelma XML-elementtejä muutamalla peruskomennolla tiedon hakua varten. (Techterms.20.06.2016)

Ohjelmointirajapintojen hyödyllisyyden pohjalla on niiden tarjoama mahdollisuus tehdä usein hyvinkin monimutkaisista prosesseista helposti uudelleenkäytettäviä vain yhdellä tai muutamalla rivillä koodia, josta eritoten kehittäjät hyötyvät kohottamalla heidän tuottavuuttaan. Esimerkiksi sen sijaan, että älypuhelimelle karttasovellusta kehitettäessä tarvitsisi ohjelmoida alusta alkaen paikannus ominaisuutta itse, voidaan lähettää kutsu mobiililaitteen käyttöjärjestelmän GPS-API:lle, joka palauttaa ohjelmalle tarvittavat tiedot paikannusta varten. Täten ohjelmaa kehittävä taho säästää aikaa joutuessaan tutustumaan ainoastaan rajapinnan dokumentaatioon ja joissain tapauksissa kirjoittamaan vain minimaalisen määrän koodia jonkin ominaisuuden toteuttamiseksi.

4.4 REST

REST, eli lyhenne sanoista Representational State Transfer, on arkkitehtuurityyli ja lähestymistapa verkkopalvelujen kehityksessä. RESTful-rajapinnat pilkkovat tapahtumat sarjoiksi pieniä moduuleja, jossa jokainen moduuli vastaa jostain tietystä osasta tapahtumaa. RESTful-tyyliä käyttävien rajapintojen eri komponentit ovat jaoteltu resursseiksi. Nämä resurssit toimivat musta laatikko -periaatteella, jonka tarkoituksena on, että rajapintaan pyyntöjä lähettävän tahon ei tarvitse tietää sen toteutuksesta mitään. Resurssilla REST-tyylissä tarkoitetaan mitä tahansa nimettävissä olevaa tietoa, esimerkkeinä resursseista voitaisi käyttää vaikkapa kirjaa tai jotain dokumenttia. (REST API Tutorial, hakupäivä 18.01.2018)

REST-tyyli, sisältää kuusi omaa ohjaavaa rajoitettaan, joiden tulisi toteutua, jotta rajapintaa voidaan kutsua RESTful-rajapinnaksi. Näihin rajoitteisiin kuuluvat asiakkaan ja palvelimen erottaminen, tilattomuus, välimuistin käytettävyys, rajapinnan yhdenmukaisuus, järjestelmän kerroksittaisuus sekä koodin ladattavuus. (REST API Tutorial, hakupäivä 18.01.2018)

Asiakas- ja palvelinpuolen erottamisella pyritään parantamaan skaalautuvuutta yksinkertaistamalla palvelimen komponentteja. Tämän lisäksi mahdollistetaan käyttöliittymän siirrettävyyttä eri alustojen välillä. Erottamisen ansiosta saman rajapinnan käyttö on mahdollista niin www-selaimella kuin myös jollain muulla alustalla. Myöskin asiakas- ja palvelinpuolen jatkokehittäminen on mutkattomampaa, kunhan niiden välinen rajapinta pidetään samanlaisena. (REST API Tutorial, hakupäivä 18.01.2018)

Oletuksena rajapintaan lähetettävät kutsut ovat niin sanotusti tilattomia. Tilattomuudella tarkoitetaan sitä, että pyyntöjen kuuluisi sisältää kaikki tarvittava tieto niihin liittyen, eikä niiden tulisi hyödyntää mitään palvelimella säilynyttä tietoa. Tilattomuuden ansiosta REST-tyyli on käytännöllinen tapa esimerkiksi pilvipalvelujen kehittämisessä. Tarvittaessa tilattomat komponentit voidaan vapaasti käyttää uudelleen esimerkiksi jonkin osion mennessä epäkuntoon. Sen lisäksi ne skaalautuvat helposti myös palvelun kokeman taakan mukaan. Nämä edut juontuvat siitä, että rajapintaan lähetettävät pyynnöt voidaan kohdistaa mihin tahansa komponentin instanssiin, sillä mitään tietoa lähetetystä pyynnöstä ei jää rajapintaan. (REST API Tutorial, hakupäivä 18.01.2018)

Välimuistin käytettävyys tulisi ilmetä jokaisessa rajapinnan haulle palauttamassa vastauksessa, jotta asiakasohjelmisto saa tietää voidaanko välimuistiin tallentaa dataa hausta ja käyttää sitä seuraavissa samanlaisissa hauissa. Tällä voidaan saavuttaa pienempi määrä tarvittavia rajapintakutsuja, sillä tarvittava data tai osa siitä on jo saatavilla välimuistista edellisen kutsun jäljiltä. Välimuistin käytettävyydellä onkin täten palvelimen kokemaa kuormaa pienentävä vaikutus. (REST API Tutorial, hakupäivä 18.01.2018)

Rajapinnan yhdenmukaisuus pyritään RESTful-rajapintojen komponenteissa saavuttamaan käyttämällä ohjelmistokehityksestä tuttua yhdenmukaisuusperiaatetta, jonka ansiosta systeemin arkkitehtuuri saadaan mahdollisimman yksinkertaiseksi ja komponenttien väliset kanssakäynnit läpinäkyviksi. Yhdenmukaisen rajapinnan toteuttamista varten joudutaan käyttämään useaa eri arkkitehtuurillista rajoitetta, joita REST-tyylissä on neljä kappaletta. Niihin kuuluvat resurssien tunnistaminen, resurssien manipulointi esityksillä, itsekuvaavat viestit sekä hypermedia sovelluksen tilakoneena. Hypermedialla viitataan tässä mihin tahansa sisältöön, joka sisältää linkkejä toisessa muodossa oleviin medioihin kuten esimerkiksi kuviin, elokuviin ja tekstiin. Arkkitehtuurityyli mahdollistaa hypermedialinkkien käytön vastauksen sisällössä, jotta asiakasohjelmisto voi dynaamisesti navigoida linkkejä käyttämällä kyseessä olevaan resurssiin. (REST API Tutorial, hakupäivä 18.01.2018)

Järjestelmän kerroksittaisuus mahdollistaa arkkitehtuurityylin hierarkkisuuden, joka rajoittaa komponenttien käytöstä estämällä niitä näkemästä niiden välittömässä kanssakäynnissään olevia tasoja kauemmas.
Koodin ladattavuus REST-tyylissä mahdollistaa jonkin koodinpätkän lataamisen ja ajamisen asiakasohjelmistolla pyynnön mukana ja täten laajentaa sen toiminnallisuutta. Tämä yksinkertaistaa asiakasohjelmaa vähentämällä tarvittavaa määrää valmiiksi toteutettuja toiminnallisuuksia.
(REST API Tutorial, hakupäivä 18.01.2018)

4.5 Steam API

Steam API on HTTP-pohjainen verkkorajapinta, joka mahdollistaa käsiksi pääsyn useimpiin Steamworks-toimintoihin. Rajapinta sisältää julkisia metodeja, joihin voi päästä käsiksi millä tahansa HTTP-pyyntöihin kykenevällä sovelluksella, kuten esimerkiksi pelin asiakasohjelmalla tai

pelipalvelimella. Rajapinnan metodit voivat palauttaa tuloksensa kolmessa eri formaatissa, jotka ovat JSON, XML ja VDF (Valve Data Format).
Julkisten metodien lisäksi rajapinta sisältää suojattuja metodeja, joihin käsiksi pääseminen vaatii tunnistautumista. (Steamworks, viitattu 18.01.2018)

Steamin verkkorajapinnan julkisen osion kutsut tulee lähettää HTTP- tai HTTPS protokollalla osoitteeseen api.steampowered.com. Verkkorajapinta on jaoteltu useisiin eri rajapintoihin, jotka sisältävät niihin liittyvät metodit. Rajapintakutsua tehtäessä URI-formaatti lähetettäville kutsuille tulee olla seuraavassa muodossa:https://api.steampowered.com/<rajapinta>/<metodi>/v<versio>/. Useimmat rajapinnan metodeista tukevat luetelman vaadittuja tai valinnaisia parametreja. Metodista riippuen nämä parametrit tulee syöttää GET tai POST parametreina kutsussa. (Steamworks, viitattu 18.01.2018)

Tavallisten verkkorajapintakutsujen lisäksi Steam rajapinta sisältää myös erilaisia palvelurajapintoja. Nämä palvelurajapinnat toimivat hyvin samalla tavalla kuin muutkin rajapinnat, suurimpana erona kuitenkin niiden toiminnassa on palvelurajapintojen kyky hyväksyä argumentit yhtenä JSON-palana GET- ja POST-parametrien sijaan. Palvelurajapinnan tunnistaa rajapinnan nimen lopussa olevasta Service-tunnisteesta. (Steamworks, 2018)

Opinnäytteessä käytetään lähestulkoon yksinomaan ISteamUserStats- ja ISteamUser-rajapintoja, josta löytyvät kaikki verkkosivuprojektin tarvitsemat, Steam-käyttäjää koskevat tiedot. Näiden rajapintojen lisäksi Steam Web API tarjoaa muun muassa ISteamNews-rajapinnan pelejä koskeville uutisille, sekä ITFItems_440-rajapinnan steam käyttäjien Team Fortress 2 -verkkotoimintapelissä omistamien kosmeettisten esineiden hakuun.

Työssä käytettäviä rajapintoja varten tarvitaan käyttäjän Steam-tunnuksen uniikki, 64-tavuinen SteamID. Vaikka SteamID onkin mahdollista käyttäjän itse hakea manuaalisesti, on nopein tapa sen saamiseksi kirjautua Steamiin verkon välityksellä. Tätä varten Steam voi toimia OpenID tarjoajana, jolla käyttäjän SteamID voidaan todentaa oikeaksi ilman, että käyttäjän tarvitsee syöttää Steam-käyttäjätunnusta ja salasanaa verkkosivulla. Tunnistaminen voidaan toteuttaa hakemalla OpenID kirjaston käytettävälle ohjelmointikielelle ja käyttäen http://steamcommunity.com/openid -osoitetta OpenID tarjoajana. Tunnistautuminen tapahtuu kolmannen osapuolen verkkosivulta siten,

että verkkosivu ohjaa käyttäjän Steamin yhteisösivuston kirjautumislomakkeelle, jossa käyttäjä syöttää kirjautumistietonsa, jonka onnistuessa tämä palautetaan sivustolle. Kirjautumisen jälkeen Steamin muodostamassa paluu-URL:ssa mukana palautuu tietoa käyttäjästä, muun muassa käyttäjän SteamID. Valve edellyttää sivustolta Steamiin kirjautuessa käytettävän selkeästi Valven brändin mukaisia kuvia kirjautumisnapeissa.

4.6 OpenID

OpenID on vuonna 2005 avoimen lähdekoodin yhteisön kehittämä palvelu, jonka tavoitteena oli ratkaista ongelma, jota ei ollut helposti mahdollista ratkaista silloin olemassa olevilla identiteettiteknologioilla. OpenID on keskittämätön palvelu, jota kukaan taho ei omista, ja jota kuka tahansa voi käyttää ilmaiseksi ilman rekisteröintiä tai miltään organisaatiolta haettavaa lupaa. (OpenID, viitattu 08.01.2018)

OpenID:n taustalla toimiva OpenID säätiö perustettiin vuonna 2007 tarjoamaan sille laillisen entiteetin sekä tarvittavaa infrastruktuuria avustamaan avoimen lähdekoodin projektia. Tämän lisäksi säätiön tehtäviin kuuluu projektin luoman palvelun käytön edistäminen muun muassa sitä mainostamalla. (OpenID, viitattu 08.01.2018)

OpenID mahdollistaa valmiiksi olemassa olevan käyttäjätunnuksen käytön useille verkkosivuille kirjautumiseen ilman, että käyttäjän tarvitsee luoda uutta käyttäjätunnusta ja salasanaa jokaiselle verkkosivulle. Käytännössä tämä suojaa käyttäjää vähentämällä muistettavien salasanojen määrää ja täten mahdollistaa voimakkaamman salasanan käyttöä. OpenID antaa käyttäjälle mahdollisuuden päättää, mitä tietoa heidän käyttämänsä verkkosivut heistä niillä käydessään saavat. OpenID:llä kirjautuessa käyttäjän salasana annetaan identiteetin tarjoajapalvelulle, joka todentaa henkilöllisyytesi verkkosivulle kirjautuessa. Tarjoajapalvelua lukuun ottamatta mikään verkkosivu ei täten näe koskaan käyttäjän salasanaa. (OpenID, viitattu 08.01.2018)

Palvelun käytön eräs etu on myös, kuinka se pienentää tietomurtovaaraa vähentämällä tietomurtoja yrittävien tahojen hyökkäysvektoreita. Myöskään kyseenalaiset tietosuojakäytännöt verkkosivuilla eivät ole niin suuri riski käyttäjän salasanan suojassa pysymiselle.

Opinnäytetyössä toteutettu Steam-autentikaatio tapahtuu käyttäen LightOpenID kirjastoa Steamin toimiessa OpenID tarjoajana. LightOpenID on PHP:lle tehty OpenID-kirjasto, jonka tavoitteena on keveys sekä mahdollisimman helppo käytettävyys. Steamin OpenID autentikaatio on toteutettu käyttäen jo nyt vanhaksi jäänyttä, mutta silti edelleen käytettävissä olevaa OpenID 2.0 spesifikaatiota, jonka OpenID säätiö korvasi uudella OpenID Connect spesifikaatiolla. OpenID:tä sekä eritoten vanhentunutta OpenID 2.0 -spesifikaatiota käyttäviä kirjastoja käytetään opinnäytetyössä kuitenkin siksi, että se on tällä hetkellä ainoa Steamin tarjoama tapa joko yhdistää Steam-käyttäjä kolmannen osapuolen verkkosivulla olevaan käyttäjään, tai toteuttaa Steamin kautta tapahtuva autentikaatio verkkosivulle. (Steamworks, viitattu 07.02 2018)

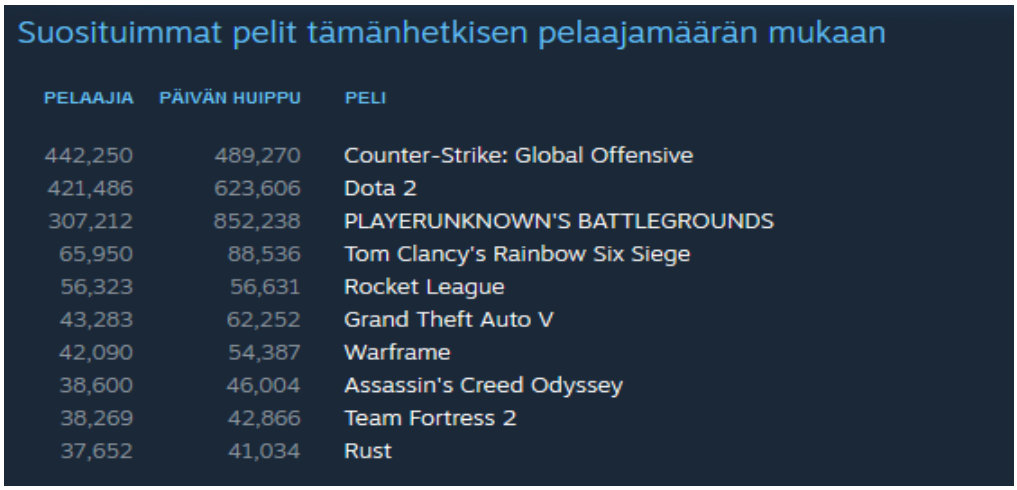
5 SOVELLUKSEN RAKENNE JA TOIMINTA

5.1 Kehityksen alkupiste

Sovelluksen kehitykseen käytettiin XAMPP-ohjelmaa omalla tietokoneellani pyörivän palvelimen simulointiin, joka mahdollistaa kehitettävän sovelluksen toiminnan testauksen kehityksen ohessa. Sovelluksen kehityksen alkupäässä perehdyin Steamin rajapintaan ja kuinka sieltä saadaan haettua tietoa. Selvitettyäni tämän olevan helpointa toteuttaa OpenID kirjautumisella sekä tuleville käyttäjille, että itse sovelluksen kehitystyössä, aloin tutkimaan kuinka tämän toteutus tulisi tehdä. Mahdollisuuksia tutkittuani päädyin hyödyntämään GitHubista löytyvää SteamAPI-kirjastoa, joka sisältää metodit kirjautumiselle ja yleisimmille rajapinnan kutsuille.

XAMPP-ohjelmaan sisältyvässä MySQL-ylläpitonäkymässä loin aluksi tietokannan, johon lisäsin yksinkertaisen tietokantataulun käyttäjälle. Tähän tietokantatauluun suunnittelin tallentavani onnistuneen kirjautumisen jälkeen kirjautuneen käyttäjän SteamID: n. Tämän lisäksi suunnittelin lisääväni tietokantaan taulut eri pelien tilastoille, joihin tallentaisin relevantiksi lukemani rajapinnasta saatavat pelitilastot.

Koska sovelluksen päätavoitteena on auttaa pelaajia löytämään peliseuraa pelaamiinsa moninpeleihin, koin luonnolliseksi tutkia Steamin internetissä tarjoamia pelitilastoja. Tilastoista oli mahdollista nähdä eri pelaajien käyttäjämäärä, joiden pohjalta päädyin lisäämään tietokantaan taulut kolmelle Steamissa eniten pelatulle moninpelille. Nämä pelit ovat jo jonkin aikaa olleet Dota2, Counter-Strike: Global Offensive sekä PlayerUnknown's Battlegrounds. Kuvassa 1 on nähtävissä Steamin verkkosivuilta otettu kuvakaappaus, jossa näkyy Steam-alustalla eniten pelatut pelit.



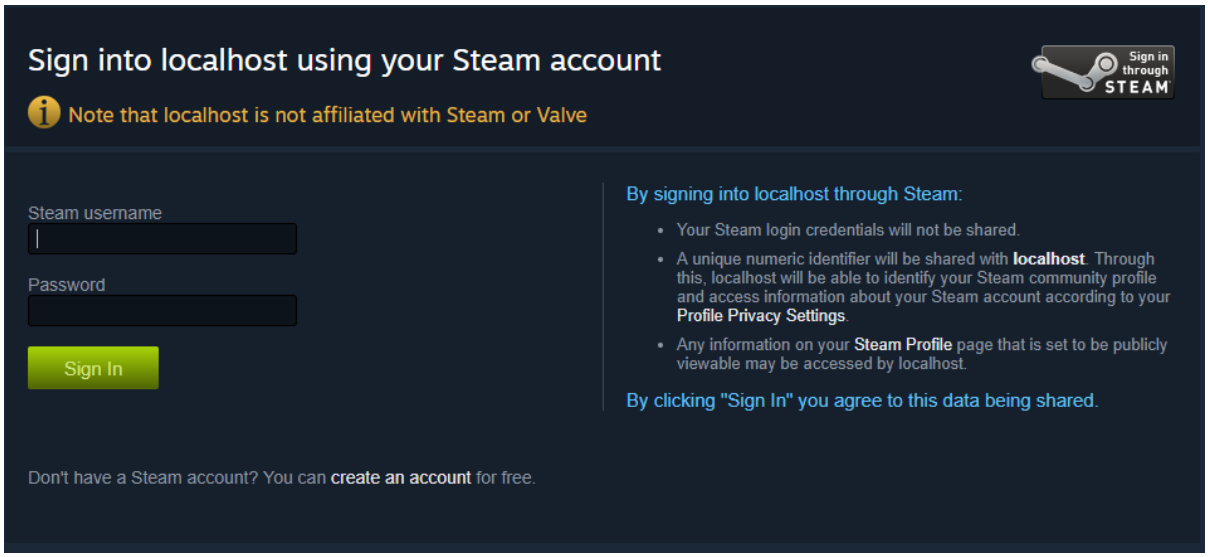
Kuva 1: Kuvakaappaus Steamin verkkosivuilta, jossa näkyvät päivän pelatuimmat pelit.

5.2 OpenID-tunnistautuminen

Steam edellyttää, että sen kautta tapahtuvaa tunnistautumista toteuttavan osapuolen täytyy käyttää selkeästi Steam-brändättyä kirjautumispainiketta. Kuvassa 2 nähtävissä esimerkki sivuilla käytetystä kirjautumispainikkeesta.

Kuva 2: Steamin edellyttämä, brändätty kirjautumispainike

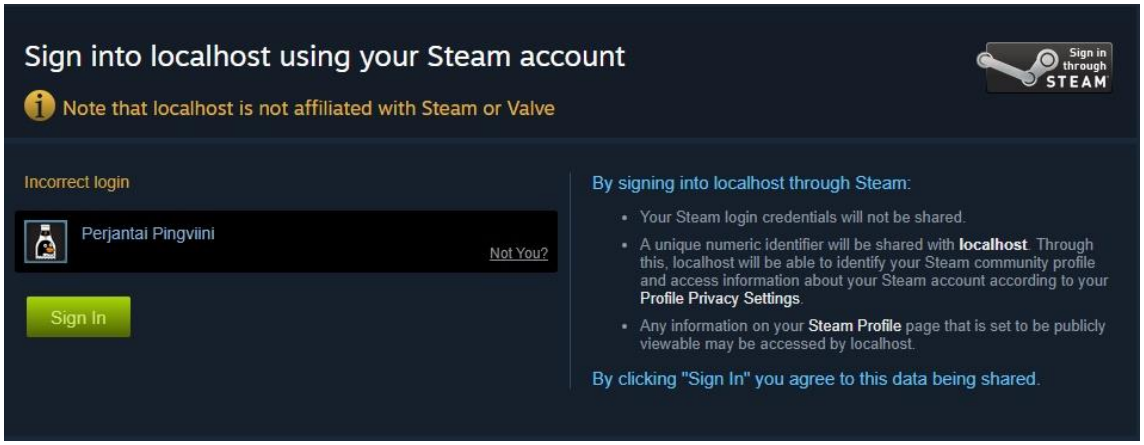
Kirjautumispainiketta painaessa controlleri lähettää kutsun SteamAPI-kirjaston metodille, joka aloittaa OpenID-kutsun Steamin palvelimelle ja uudelleenohjaa yhteyden muodostuessa Steamin sivustolle. Kuvassa 3 kuvakaappaus uudelleenohjauksen jälkeen käyttäjälle avautuvasta kirjautumisformista.



Kuva 3: Kuvakaappaus Steamin kirjautumisformista

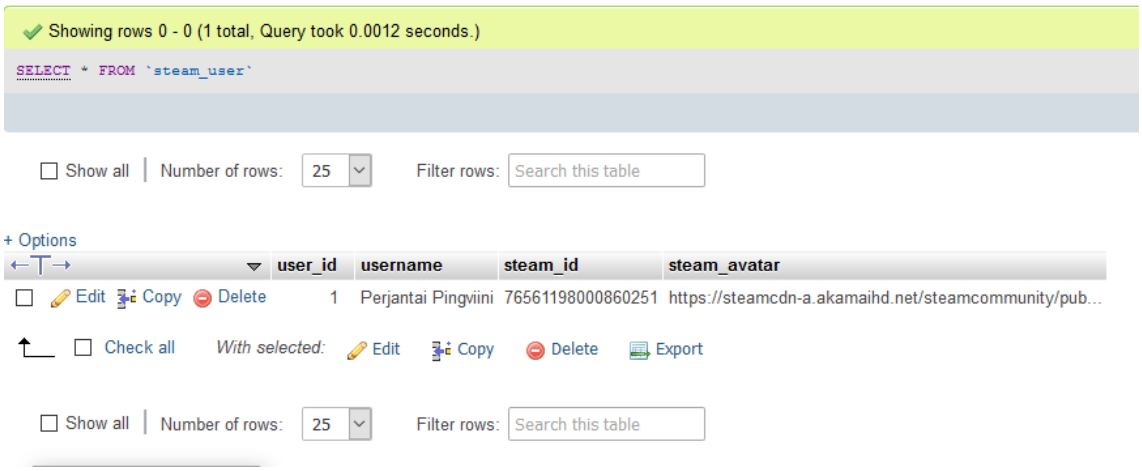
Kirjautumisformilla ilmaistaan selkeästi, että sivu, jolle on kirjautumassa Steamin kautta ei ole Steamin eikä Valven yhteistyökumppani. Formilla tiedotetaan myös siitä, mitä tietoja kirjautumalla antaa sivustolle.

Onnistuneen kirjautumisen jälkeen avautuvassa näkymässä käyttäjä näkee oman Steam-tunnuksensa tiedot sekä vielä kertaalleen näkyvän kirjautumisnapin, jota painamalla käyttäjä ohjautuu takaisin kirjautumassa olevalleen sivustolle. Kuvassa 4 on nähtävissä ennen takaisinpalautusta kirjautumista haluavalle sivustolle sille kirjautumassa oleva käyttäjä.



Kuva 4: Ennen sivustolle palautusta näytettävä, kirjautumisen varmistava näkymä

Käyttäjän palatessa takaisin sivustolle, palautuu käyttäjän mukana myös tämän SteamID, jonka SteamAPI-kirjasto tallentaa istuntomuuttujaan. Tämän jälkeen SteamID:tä käyttäen tehdään rajapintahaku, jolla haetaan käyttäjän tiedot kuten nimi sekä avatar-kuva, jotka tallennetaan tietokantaan. Kuvassa 5 kuvakaappaus steam_user -tietokantataulusta, johon käyttäjän tiedot on tallennettu.



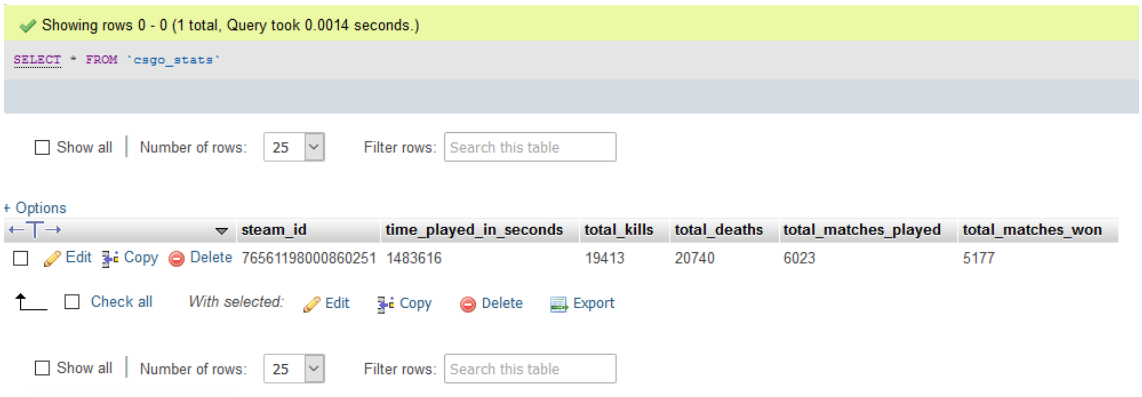
Kuva 5: Tietokantataulu steam-käyttäjän tiedoille

5.3 Tilastojen haku rajapinnasta

Kirjautumisen jälkeen käyttäjän Steam ID:n saatua voidaan alkaa suorittamaan rajapintahakuja, joista saatavat käyttäjän tilastot voidaan tallentaa ja täten hyödyntää sovelluksen käyttötarkoituksiin. Haettavan testidatan aiheeksi valikoin Counter-Strike: Global Offensive -pelin tilastot johtuen pelin suosiosta, sekä siitä saatavan datan kattavuudesta. Tämän ansioista on mahdollista saada luotua hyvinkin tarkka kokonaisuus pelaajan pelitaidoista.

Kyseiselle pelille rajapinta palautti 166 eri tilastoa, joista suurin osa oli käyttäjän eri aseilla ampumien laukausten määrä sekä tarkkuus, sekä eri kenttiin liittyviä tilastoja kuten niissä voitettujen ja hävittyjen pelien määrä. Näiden lisäksi mukana tuli myös suuri kokoelma käyttäjän pelin sisäisistä saavutuksista. Pelissä olevan ranking-systeemin pelaajalle antamaa rankingia rajapinta ei ikävä kyllä tarjoa, sillä tämä olisi ollut yksinkertaisin ja tarkin tapa nähdä pelaajan taso.

Saadusta datasta pyrin valikoimaan mielestäni parhaimmin käyttötarkoitukseen relevantit tiedot. Näiksi valikoin käyttäjän pelin pelaamiseen käytetyn ajan, pelissä tehtyjen tappojen ja kuolemien kokonaislukumäärän sekä pelattujen pelien ja voitettujen pelien määrän. Valikoituani haluamani tiedot ohjelmoin ne tallennettavaksi niille tarkoitettuun tietokantatauluun. Tilastojen lisäksi tauluun tallennetaan myös käyttäjän oma Steam ID, jotta ne voitaisi tunnistaa juuri tämän käyttäjän omiksi tilastoiksi. Kuvassa 6 nähtävissä kuvakaappaus pelille haetuista käyttäjän tilastoista.



Kuva 6: Tallennetut tilastot tietokannassa

6 POHDINTA

Opinnäytteen tavoitteena oli toteuttaa hakumahdollisuus Steamin verkkorajapintaan opiskelijaprojektin verkkosivulle. Tämä toteutettiin käyttäen PHP-framework Codeigniteria, sillä web-sovellus oli alun perinkin ollut suunniteltu toteutettavan sitä käyttäen. Tämän lisäksi työssä päädyttiin hyödyntämään OpenID-autentikointiprotokollaa, jotta käyttäjät pystyisivät kirjautumaan sivulle käyttäen Steam-käyttäjätunnuksiaan, jonka tarkoituksena oli helpottaa käyttäjien SteamID:n saanti. SteamID:n tietäminen on vaatimuksena käyttäjän omien pelitilastojen hakuun. Työn lopputulos oli odotetun lainen, kirjautumisen toteuttaminen sekä tietojen haku toimivat yhtä poikkeusta lukuun ottamatta juuri halutulla tavalla.

Kehitystyössä ilmeni jonkin verran eri syistä johtuneita haasteita. Suurimpana haasteena koin palauttaa takaisin mieleen PHP:n ja Codeigniterin käytön, opinnäytteen toteutuksessa tapahtuneiden pitkien taukojen johdosta.

Haasteeksi ilmeni myös alun perin suunnittelemani Dota2-pelin tilastojen haku testausta varten. Koska OpenID-tunnistautuminen oli helpointa testata omilla Steam-käyttäjätunnuksillani, olisi ollut yksinkertaista hakea tätä kautta saatavalla käyttäjäni SteamID:llä tietoja Steamin rajapinnasta. Tämä osoittautui kuitenkin todella monimutkaiseksi, sillä rajapinnan GetUserStatsForGame -metodi, joka palauttaa haussa spesifioidun käyttäjän ja sovellustunnuksen mukaan dataa ei sisältänytkään mitään tilastoja Dota2:lle. Jonkin aikaa asiaa tutkittuani selvisi, että pelin tilastot olivat saatavilla vain pelille itselleen omistetusta rajapinnasta. Tutkiessani asiaa minulle selvisi myös, että toteutus tilastojen haulle tätä kautta olisi hyvin aikaa vievää työtä. Siinä missä GetUserStatsForGame -metodi palauttaa yhdellä haulla esimerkiksi Counter-Strike -tilastoja hakiessa suoraan pelattujen pelien määrän, Dota2:lle tarkoitetusta rajapinnasta täytyisi hakea kaikki pelatut pelit, joissa käyttäjän SteamID:stä erillinen accountID on ollut osallisena. Haun tuloksesta olisi täytynyt poimia jokaisen käyttäjän pelaaman pelin oma matchID, jota käyttäen jokaisen pelin tilastot olisi erikseen haettu rajapinnasta. Vasta tätä kautta olisi voitu saada käyttäjän pelaamien pelien tilastot tietoon ja kasattua jonkinlainen kokonaisuus pelaajan taidoista.

Projektin lopputulokseksi saadut tulokset olivat mielestäni hyvät, sillä sille asetetut tavoitteet tulivat täyteen ja ominaisuuden hyödyntäminen jatkokehityksessä olisi täysin mahdollista. Selkeästi negatiivisin osa työssä oli sen aivan liian pitkä kesto, joka johtui puhtaasti saamattomuudesta. Pelkkien työhön upotettujen tuntien perusteella työ olisi voinut olla valmis huomattavasti aiemmin, suurimpana ongelmana olivat pitkät välit, jolloin työ ei edennyt käytännössä ollenkaan. Pitkien taukojen jälkeen työn jatkaminen oli myöskin hankalampaa, kuin jos työ olisi tehty kerralla lyhyen ajan sisällä loppuun. Haastavinta projektissa olikin työn teon aloittaminen, projektin toteutus tapahtui useiden lyhyiden mutta tehokkaiden spurttien aikana, jolloin aina yksi suunniteltu kokonaisuus kerrallaan valmistui. Työn viitekehyksen koen olevan täysin kelvollinen, työhön liittyvät käsitteet ovat asiantuntevan lukijan helposti ymmärrettävissä teoriaosuuden pohjalta.

Ominaisuuden jatkokehityksessä suurin työ olisi perehtyä Dota2:n rajapintaan ja toteuttaa jokin ratkaisu käyttäjien tilastojen kasaamisen kyseiselle pelille. Koska hakuja pitäisi tehdä useampi jokaiselle käyttäjälle heidän tilastojensa saamiseksi, täytyisi siitä saada tehtyä mahdollisimman hyvin optimoitu, ettei sovelluksen suorituskyky kärsisi. Tämän lisäksi myös kyseisestä pelistä saatavan datan määrä on sen verran laaja, että tallennettava tieto täytyisi valikoida huolellisesti tietokannan toimivuuden vuoksi. Myös muiden pelien tilastokokonaisuuksien laajentamista olisi hyödyllistä miettiä siten, että tulevaisuudessa saman tasoiset pelaajat pystyttäisiin mahdollisimman luotettavasti saattamaan yhteen. Tätä varten voitaisi pyrkiä selvittämään kuinka pelien sisäiset ranking-systeemit toimivat ja pyrkiä niitä mukaillen web-sovelluksen tietokantaan tallennetun datan perusteella laskelmoimaan heidän mahdollinen rankinginsa. Sovelluksen sisäisen ranking-systeemin toteuttaminen helpottaisi myös sovelluksen toimintaa, kun käyttäjiä voitaisi vertailla nopeasti keskenään heidän rankinginsa perusteella. Tämän ansioista monimutkaisia käyttäjien tilastojen vertailuja ei tarvitsisi tehdä joka kerta käyttäjän etsiessä peliseuraa.

Työn toteutuksessa opin kenties eniten omista työskentelytavoistani ja kuinka ne tuntemalla kykenen maksimoimaan aikaansaamani työt. Sen lisäksi myös tiedonhakutaidot ja ajatuksien jäsentäminen kirjoitettuun muotoon saivat harjoitusta. Teoriapuolen asioista myös tarkempi tieto RESTful-arkkitehtuurityylistä on todennäköisesti hyödyllinen jatkon kannalta sovelluskehitystöitä tehdessä.

LÄHTEET

BMIGaming.com. The Golden Age Of Video Arcade Games. Hakupäivä 21.02.2018,
https://www.bmigaming.com/videogamehistory.htm

Codeigniter. 25.11.2017,
https://codeigniter.com/user_guide/general/credits.html

Ellislab. Hakupäivä 27.3.2017,
https://ellislab.com/codeigniter

Derek Jones. Expressionengine. 10.06.2014,
https://expressionengine.com/blog/your-favorite-php-framework-codeigniter-has-a-new-home

JSON. Introducing JSON. Hakupäivä 20.01.2018
https://www.json.org/

OpenID. OpenID Foundation. Hakupäivä 08.01.2018,
http://openid.net/foundation/

OpenID. What is OpenID? Hakupäivä 08.01.2018,
http://openid.net/what-is-openid/

Programmableweb. Hakupäivä 2.12.2015,
http://www.programmableweb.com/api-university/what-are-apis-and-how-do-they-work

REST API Tutorial. What is REST. Hakupäivä 18.01.2018,
https://restfulapi.net/

Steamworks. Web API Overview. Hakupäivä 18.01.2018,
https://partner.steamgames.com/doc/webapi_overview

Steamworks. Web Browser based authentication with OpenID. Hakupäivä 07.02.2018,
https://partner.steamgames.com/doc/features/auth#website

Techcrunch. The History Of Gaming: An Evolving Community. 31.08.2015.
https://techcrunch.com/2015/10/31/the-history-of-gaming-an-evolving-community/

Techterms. API. 20.06.2016,
https://techterms.com/definition/api

Techterms. SDK. 15.04.2010,
http://techterms.com/definition/sdk

The International Arcade Museum. Pong. Hakupäivä 21.02.2018,
https://www.arcade-museum.com/game_detail.php?game_id=9074

Valvesoftware. 2015. Hakupäivä 2.12.2015,
http://www.valvesoftware.com/company

Valvesoftware. 2003. Hakupäivä 2.12.2015,
http://store.steampowered.com/news/183/

Webopas. Hakupäivä 2.12.2015,
http://www.webopas.net/orajapinta.html