Jarno Horn

Tiedon reaaliaikainen visualisointi verkkosivulla

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Mediatekniikan koulutusohjelma

Insinöörityö

13.5.2015

Metropolia

Tekijä Otsikko Sivumäärä Aika	Jarno Horn Tiedon reaaliaikainen visualisointi verkkosivulla 25 sivua 13.5.2015
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Mediatekniikka
Suuntautumisvaihtoehto	Digitaalinen media
Ohjaaja	Lehtori Aarne Klemetti

Insinöörityössä selvitettiin, mitä asioita tarvitaan kerätyn tiedon reaaliaikaiseen visualisoimiseen internetsivuilla. Työ tehtiin osana projektin prototyyppiä, jossa yritysasiakkaat voivat kerätä henkilöasiakkaista tietoa siten, että se hyödyttää myös henkilöasiakkaita.

Insinöörityössä käytettiin Google Charts -kaaviointiohjelmaa ja erittäin suosittua MySQL-tietokantapalvelua. Ohjelmoinnin osalta työssä käytettiin tunnetuimpia ohjelmointikieliä. Palvelintason ohjelmointiin käytettiin PHP-ohjelmointikieltä ja selaintason ohjelmointiin käytettiin JavaScriptiä ja HTML:ää.

Google Charts -palvelun lisäksi kokeiltiin muita vastaavia ohjelmia, mutta niiden kohdalla ongelmaksi muodostui JavaScript-kirjastojen heikko toimiminen ja vähäinen ohjeistus kaavioiden käyttöönotossa. Google Charts -palvelussa käyttäjälle informoidaan selkeästi, millaisessa muodossa tiedon on oltava, jotta käytetyt JavaScript-kirjastot saavat siitä selvää.

Tiedon visualisoimiseen on paljon eri syitä, ja jokaiseen tilanteeseen on oma kaavio, jolla asia saadaan esitettyä parhaiten. Tärkeää on tiedostaa, että kaavioihin ei päädy niin sanottua valehtelukerrointa. Insinöörityössä käytettiin palkkikaaviota, piirakkakaaviota, pistekaaviota ja taulukkokaaviota. Google Charts -taulukkojen ulkonäköä voidaan muokata, ja muokkaamista kokeiltiin piirakkakaavion osalta. Insinöörityö onnistui hyvin ja kaavioiden näyttämä tieto oli helposti integroitavissa tietokantaan.

Avainsanat	Reaaliaikainen, tietokanta, visualisoiminen, Google Charts

Author Title Number of Pages Date	Jarno Horn Realtime visualization of data on a website 25 pages 13 May 2015
Degree	Bachelor of Engineering
Degree Programme	Media Technology
Specialisation option	Digital Media
Instructor	Aarne Klemetti Senior lecturer

This thesis investigates the things that are needed for visualizing wanted data on a website. It is part of a project that allows companies to collect data from visitors so that it is beneficial to both parties.

To visualize the data, the Charts program from Google and the MySQL database program, which is highly popular, were used. In addition, a few commonly known programming languages were used, such as PHP for back end programming and JavaScript and HTML for front end programming.

Besides Google Charts a few other similar programs were tested, but they were problematic because of their lack of instructions and the functioning of their JavaScript libraries.

There are many reasons to visualize data and there are plenty of charts to choose from. It is important to realize that when a chart is made it should have a truthful representation. Column, pie, scatter and table charts were used in this project. There are plenty of options on how to alter the charts and this thesis presents some of them.

Keywords	Real time, database, visualization, Google Charts

Sisällys

Lyhenteet

HTML	Hypertext Markup Language on verkkosivujen tukiranka. Koko sivusto kootaan HTML:n ympärille.
PHP	Hypertext Preprocessorilla saadaan verkkosivuille dynaamisia toimintoja ja voidaan tehdä esimerkiksi kutsuja tietokantaan.
JSON	JavaScript Object Notation on ketterä tiedonsiirtoon käytetty formaatti. Se perustuu osittain JavaScriptiin, ja sitä käytetään muun muassa PHP:llä kerätyn datan muokkaamiseen sellaiseen muotoon, että JavaScript-ohjelmointikieli pystyy sitä käsittelemään.
JQuery	JavaScript-kirjasto.
MySQL	Avoimen lähdekoodin tietokantasovellus. MySQL:n My tulee sovelluksen luojan tyttären nimestä My ja SQL tulee sanoista Structured Query Language.
NFC	Near Field Communication on lyhyen matkan radiotaajuuksilla toimiva keskusteluyhteys eri laitteiden välillä.
CSS	Cascading Style Sheets on ohjelmointikieli, jonka avulla verkkosivut saavat oman tyylinsä.

1 Johdanto

Insinöörityö on osa yritysidean prototyyppiä, jossa ajatuksena on toimia yritysasiakkaan ja yrityksen asiakkaiden rajapinnassa ja tarjota yrityksille muun muassa mahdollisuutta seurata asiakkaiden toimia reaaliajassa omalla verkkosivustolla. Tarkoituksena on luoda erilaisissa tapahtumissa käyville asiakkaille heti tapahtuman aluksi henkilökohtainen käyntikortin kokoinen NFC-sirulla varustettu kortti, joka sisältää asiakkaan henkilötiedot. NFC on lähilukua varten tarkoitettu pieni tallennustila. Tätä korttia käytettäisiin tapahtumaan osallistuvien yritysten kilpailuihin osallistumiseen ja erilaiseen tiedon keräämiseen ja jakamiseen. Rajapintaa on ajateltu käytettäväksi muun muassa messutilanteissa, jolloin yhdessä paikassa voi olla useita yrityksiä ja esimerkiksi kilpailuihin ja mainostukseen kerättävien sähköpostiosoitteiden kerääminen helpottuisi. Vielä nykyäänkin yhteystiedot kerätään täytä lappu -tekniikalla, mikä on aikaa vievää ja vähentää keskittymistä asiakaskontaktien luomiseen ja asiakaspalveluun.

Insinöörityö on osa projektia, johon kuuluu kolme osa-aluetta: tapahtumissa käytettävän mobiilisovelluksen suunnittelu ja luonti sekä yritysasiakkaille ja henkilöasiakkaille tarvittavien sivustojen suunnittelu ja luominen. Mobiilisovelluksen on määrä käyttää NFC:tä asiakastietojen tallentamiseen ja käyttämiseen mobiilisovelluksen eri käyttökohteissa. Yritysasiakkaan sivuilla on tarkoitus mahdollistaa erilaisten tietojen tarkastelu ja muokkaaminen tapahtumaan liittyen, ja henkilöasiakkaiden sivuilta voisi käydä tutkimassa itseään kiinnostavien yritysten tietoja. Kaikki kolme toimintoa käyttäisivät yhteistä tietokantaa, jolloin esimerkiksi yritysten tekemät muokkaukset omiin tietoihin näkyisivät saman tien myös henkilöasiakkaan verkkosivuilla.

Insinöörityön osa-alueena projektissa on yritysasiakkaille näkyvän sivuston suunnittelu ja luominen. Projekti alkaa selvittämällä, minkälainen sivusto olisi sopiva yritysasiakkaalle ja minkälaisia ominaisuuksia siinä tulee olla. Tunnen entuudestaan erilaisia verkkojulkaisualustoja, kuten Wordpress ja Foundation, joten aloitan sivuston luomisen Foundation-julkaisualustaa käyttäen. Foundation on vahvasti responsiivisuuteen keskittynyt julkaisualusta, joten sivusto toimii hyvin niin tietokoneiden selaimilla kuin mobiililaitteillakin. Tämä on tärkeää, sillä henkilöasiakkaat todennäköisesti käyvät heille tarkoitetulla sivustolla mobiililaitteita käyttäen jo tapahtuman aikana, eivätkä he mahdollisesti odottaisi kotiin ja tietokoneelle pääsyä. Myös yritysasiakasta saattaa kiinnostaa tapahtuman kulku, jolloin se voi käydä mobiiliselaimella katsomassa, miten tapahtuma

sujuu. Haluan mahdollistaa yritysasiakkaille selkeän tavan seurata tapahtuman kulkua omilta sivuiltaan. Tämä toteutetaan tietojen reaaliaikaisella visualisoinnilla. Käytännössä sivuille kerätään tietoa tietokannasta reaaliajassa ja näytetään se sivuilla erilaisin kaavioin ja taulukoin. Insinöörityössä perehdytään siihen, mitä reaaliaikainen tiedonvälitys visuaalisesti tarvitsee tietokannalta ja ohjelmoinnin kannalta. Lisäksi selvitetään erilaisten kaavioiden käyttö- ja muokkaamismahdollisuuksia ja tutkitaan erilaisten visuaaliseen selainpohjaiseen tiedonvälitykseen tarkoitettujen ohjelmien eroja.

2 Tietokannat

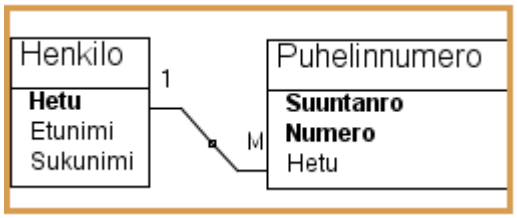
Tietokannat ovat kokoelmia toisiinsa liittyvistä tiedoista. Tietokantoihin liitetään kuitenkin teknisiin ominaisuuksiin liittyviä lisävaatimuksia, jotta ne erottuisivat perinteisistä ohjelmointiin käytettävistä tiedonvarastoimismuodoista. Tiedon täytyy olla riippumatonta sitä käyttävästä ohjelmasta, jolloin jokaisella ohjelmalla tai eri ohjelmoimiskielillä ei tarvitse olla erillisiä paikkoja tallentaa tietoja. Usean käyttäjän on voitava käyttää tietoja samaan aikaan. Tietokannoissa pyritään tarjoamaan mahdollisimman laajaa tiedostojen rinnakkaiskäyttöä samalla minimoiden sen mukanaan tuomat ongelmat. Tietokannat antavat myös avaimet tietojen monipuolisempaan käsittelyyn. Tavallisesta tiedostosta poiketen tietokannasta pystyy hakemaan ja käsittelemään tietoa sen sisällön perusteella. Tietokannat tarjoavat mahdollisuuden tiedon sisällön perusteella tapahtuvaan kohteiden valintaan ja käsittelyyn. Tällöin ohjelman ei tarvitse käydä läpi kaikkia tietoja peräkkäin, kuten tiedostosta haettaessa. [1.]

Tietokannat mahdollistavat käyttöjärjestelmiä tai tiedostoja paremmat käyttörajoitukset, ja suojaukset ovat huomattavasti paremmalla tasolla. Tietokannat pyrkivät pitämään mahdollisimman päivittynyttä tietoa yllä siten, että se ei voi vahingossa kadota. Tiedostotasolla tämä hoidetaan varmuuskopioinnilla, mutta varmuuskopioinnin huono puoli on, että se vaatii ajanjakson, jossa tieto ei ole kenenkään käytössä, esimerkkinä tekstin muokkaaminen. Yleensä tallennettaessa tiedoston käyttö hetkellisesti estyy, kunnes kopio on valmis, ja jos tiedostolla on enemmän kuin yksi käyttäjä, saattaa sopivan varmuuskopioinnin ajankohdan löytyminen olla hankalaa. [1.]

2.1 Relaatiotietokannat

Tietokannat tehdään yleisimmin relaatiomallilla. Relaatiomallissa jokaisessa tietokannan taulussa on aina oma muuttumaton tekijä eli perusavain. Taulut saavat relaationsa viiteavaimien avulla. Viiteavain on isäntätaulun perusavain, jolla uusi taulu yhdistetään isäntätaulun kanssa. Merkittävää on, että perusavaimen on oltava uniikki, eikä taulusta saa löytyä identtisiä perusavaimen arvoja. [1.]

Kuvasta 1 näkyy, että Hetu, eli yksittäisen henkilön henkilötunnus, eli perusavain, on otettu puhelinnumerotaulun yhteiseksi tekijäksi eli viiteavaimeksi. Tällöin henkilön hetulla haettaessa saadaan tiedot molemmista tauluista. Hetua voidaan tarpeen mukaan käyttää viiteavaimena useassa taulussa, jos sille nähdään tarvetta. Henkilötunnus on valittu perusavaimeksi siksi, että se ei voi missään tilanteessa olla sama kahdella eri henkilöllä. Toissijaiset avaimet toimivat rajoitteina ja nopeuttavat taulujen järjestämistä.



Kuva 1. Esimerkkirelaatio, jossa on kaksi taulua ja molemmissa kolme kenttää [2]

Tauluille tulee antaa selkeä ja kuvaava nimi, mutta erikoismerkkien tai skandinaavisten merkkien käyttöä kannattaa välttää. Kuvassa 1 vasemmanpuoleisen taulun nimeksi on annettu ”Henkilö” ja oikeanpuoleisen taulun nimeksi on annettu ”Puhelinnumero”. Tauluissa on aina yksi tai useampia kenttiä. Kenttiä luotaessa niille valitaan erilaisia ominaisuuksia:

- kentän nimi
- tietotyyppi
  - numeerinen
  - merkkijono
  - aika
- kiinteän tai vaihtuvamittaisen merkkijonon valinta

- merkkijonon enimmäispituus
- merkkijonon tarkkuus
- tiedon pakollisuus
- oletusarvo (esimerkkinä lukuarvo voisi olla oletuksena 0 tai 1).

Kentän nimeämisessä on syytä käyttää samoja rajoitteita kuin taulukonkin nimeämiseen käytetään. Jos tietotyyppi on numeerinen, kenttään ei voi tallentaa muuta kuin numeroarvoja, esimerkiksi paino tai pituus. Merkkijonoon taas on mahdollista tallentaa niin numeroita kuin tekstiäkin, esimerkiksi henkilöllisyystunnus tai etu- ja sukunimi. Kun tietotyyppi asetetaan aikaa määrittäväksi, siihen voidaan tallentaa vain aikaa määrittäviä tietoja, kuten syntymäaika tai työvuoro. Jos haluttu tieto on ennalta määritellyn pituinen, voidaan tallennus tehdä kiinteämittaisena. Tällöin merkkijonossa on oltava aina tietty määrä merkkejä. Vaihtuvamittaisessa merkkijonossa määritetään tallennettavalle tiedolle ennalta valittu enimmäispituus. Merkkijonon tarkkuuteen voidaan vaikuttaa esimerkiksi, jos halutaan tallentaa tieto yhden desimaalin tarkkuudella. Lukumääräkentän tarkkuudeksi taas riittää hyvin kokonaisluku. [2.]

Kenttien tiedoille on mahdollista asettaa myös rajoituksia siitä, minkälaisessa muodossa tieto on mahdollista syöttää kenttään. Tällä voidaan tarkistaa tiedon oikeellisuus tai sallittujen arvojen joukko. Esimerkiksi henkilötunnus on jokaisella suomalaisella samanlainen: alussa on oltava kuusi numeroa, minkä jälkeen tulee välimerkiksi joko + tai -, sitten kolme numeroa ja viimeisenä tarkistusmerkki, joka voi olla numeraali tai kirjain. Kuvassa 1 Henkilö-taulussa olevat Hetu, Etunimi ja Sukunimi ovat taulun kenttiä.

Kenttiin tallennetaan itse tieto. Tärkeimmät standardisoidut tiedostotyypit ovat

- char
- varchar
- numeric
- integer
- date
- time
- timestamp

- interval.

Char ja varchar ovat merkkijonoja. Char on kiinteämittainen, kun taas varchar on vaihtuvamittainen ja sille määritetään enimmäispituus. Numeric on tarkka numeerinen arvo, jolle voidaan määrittää desimaalien määrä. Integer on kokonaisluku, jolla on käyttökohteesta riippuva enimmäiskoko. Date on päiväys, kuten vuosi, kuukausi ja päivämäärä. Timellä taas tallennetaan tarkka kellonaika, eli tunti, minuutti ja sekunti. Timestamp on date ja time yhdistettynä. Intervallilla määritetään kulunut aika eli ajanväli. [2.]

MySQL

Projektin tietokantasovellukseksi valittiin MySQL-relaatiotietokantaohjelmisto. MySQL oli ennestään tuttu palvelu ja sopi käyttötarkoitukseen erityisen hyvin. Se on maailman suosituin avoimen lähdekoodin tietokantasovellus, jota on ladattu yli satamiljoonaa kopiota. Useat maailman suurimmista ja nopeimmin kasvavista yrityksistä, kuten Google, Nokia ja YouTube, käyttävät MySQL-sovellusta, sillä se poistaa yritysten suurimpia ongelmia, joita ovat verkkosivujen alhaallaoloajat, huoltotyöt ja sivustojen hallinnointiin liittyvät ongelmat.

MySQL soveltuu siis erittäin hyvin juuri web-palveluiden tietokannaksi. Sovellukset keskustelevat tietokannan kanssa yleensä PHP-, Python- tai Perl-ohjelmointikielellä. Tässä projektissa käytettiin tietokannan ja verkkosovelluksen väliseen yhteydenpitoon PHP-ohjelmointikieltä. MySQL:n kanssa on mahdollista olla yhteydessä muun muassa C:llä, C++:lla, C#:lla, Smalltalkilla, Javalla, Rubyllä ja TCL:llä. Tietokannan kanssa keskustellaan SQL-pyyntöjen avulla. MySQL:ssa tiedot esitetään tauluina, joiden yksittäistä riviä kutsutaan tietueeksi. Taulun jokaisella rivillä on oltava sama määrä tietueita. [2; 3.]

2.2 NoSQL-tietokanta

Tavallisen SQL-tietokannan haastajaksi on pikkuhiljaa noussut uudenlainen tietokantamalli. Tätä tietokantamallia kutsutaan NoSQL-tietokannaksi. NoSQL tulee sanoista Not Only SQL, mikä tarkoittaa sitä, että NoSQL-tietokanta kattaa laajalti erilaisia tietokantateknologioita, joita on kehitelty sitä mukaa, kuin tietokannat ovat kasvaneet tallennettavan datan määrässä. Datan määrän lisäksi on huomattu, että datan saatavuu-

teen, tietokannan nopeuteen ja prosessien lisääntyneeseen tarpeeseen on ollut tarvetta kehitellä paremmin skaalautuvaa järjestelmää. Nykyaikaiset sovellukset vaativat enemmän ketteryyttä ja skaalautuvuutta, kuin mitä SQL-pohjaiset tietokannat mahdollistavat.

NoSQL:n datan varastoimismenetelmä on SQL:ään verrattuna ylösalaisin käännetty. SQL-tietokannassa täytyy olla valmiiksi rakennettuja taulukoita, jotka sisältävät valmiiksi määriteltyjä kenttiä datan tallentamista varten. Mikäli muutoksia on tehtävä, on koko tietokantaa muunneltava, ja tällöin tietokannan tulee olla pois käytöstä eli offline-tilassa.

NoSQL-tietokantaan voidaan syöttää uusia taulukoita ja muutoksia taulukoihin ilman minkäänlaista käyttökatkoa tai tietokannan muokkaamista, suoraan sovelluksen koodista. [4.]

Koska insinöörityön sovellus oli vasta prototyyppivaiheessa ja koska perinteinen SQL-tietokanta oli ennestään tuttu, päätettiin jättää NoSQL vielä tässä vaiheessa taka-alalle, mutta lopulliseen sovellukseen se otetaan käyttöön.

3 Ohjelmointikielet

Jotta tietokantaan tallennettu tieto saataisiin verkkosivuille visualisoitua, täytyy tuntea muutama yleisesti käytössä oleva ohjelmointikieli. Vaikka nykyaikaisempi malli käsitellä tietokantoja olisi tehdä se suoraan JavaScriptin avulla, esimerkiksi NodeJs:n avulla tehdä muutoksia ja ottaa yhteys NoSQL-tietokantaan, päätin käyttää itselleni jo entuudestaan tuttuja ohjelmointikieliä ja hieman vanhempaa MySQL-tietokantasovellusta. Tämän valinnan takana on se, että alkaessani rakentaa esimerkkisivua oli Googlen kaaviopalvelun tietokantasovellusten integroimisohjeistus erittäin niukkaa, jolloin näin tarpeelliseksi yrittää ensin ymmärtää sen toimintaa, ennen kuin mahdollisesti vaihtaisin palvelintason ohjelmat nykyaikaisempiin. Ohjelmointikielet, joita tässä insinöörityössä käytettiin, ovat hypertext preprocessor -kieli, eli PHP, joka on palvelintason ohjelmointikieli, eli sen avulla tietokantaan tehdään kyselyjä ja tiedot käsitellään edelleen lähetettäväksi. Tämän jälkeen käytettiin JavaScript Object Notation -kieltä, eli JSON:a, jolla PHP:n muodostama tapahtuma voidaan lähettää JavaScriptille. JavaScript on objektiorientoitunut ohjelmointikieli, jota yleensä käytetään interaktiivisten efektien tekoon verkkosivuilla. JavaScript voi olla pelkästään selaintason ohjelmoitikieli, mutta kuten

aiemmin mainittiin, nykyään sillä voidaan tehdä myös palvelintason ohjelmointia. Lopullinen sivu eli varsinainen selaintason ohjelmointi tehtiin Hypertext Markup Languagella, eli HTML:llä, jonka avulla JavaScript-efektikin saadaan sivuilla näkymään.

3.1 PHP-ohjelmointikieli

PHP-kieli on laajasti käytetty avoimen lähdekoodin yleiskäytännöllinen ohjelmointikieli. Sitä käytetään verkkosivujen kehityksessä palvelintason ohjelmointikielenä, ja sen dynaamiset toiminnot on helppo syöttää HTML:llä ohjelmoiduille sivuille.

PHP:llä on kolme pääasiallista käyttökohdetta:

- Palvelinpuolen ohjelmointi, joka on kaikista yleisimmin käytössä oleva käyttökohde. Palvelinpuolen ohjelmoimiseen tarvitaan kolme asiaa: palvelimen moduuli (PHP parser), internetpalvelin ja verkkoselain. Verkkoselaimella käyttäjä voidaan yhdistää PHP:lla tehtyyn ohjelmaan palvelimen kautta.
- Komentoriviohjelmointi, jossa ei tarvita palvelimia tai selaimia, vaan PHP:ta käytetään käyttöjärjestelmien, kuten LINUX tai Windows, toimintojen käyttämiseen.
- Sovellukset työpöytäkäyttöön, mutta tähän toimintoon se ei ole kaikista optimaalisin ratkaisu. [6.]

Yksi tärkeimmistä ja vahvimmista toiminnoista PHP:ssä on, että se tukee monenlaisia tietokantoja. PHP:n avulla voi hakea, muokata tai lisätä tietoa tietokantaan. Tätä PHP:n toimintoa käytettiin tässä projektissa. [5; 6.]

3.2 JavaScript-ohjelmointikieli

JavaScript on oliopohjainen ohjelmointikieli, jonka on kehittänyt Netscape. JavaScriptin ensisijainen tarkoitus on tuottaa dynaamisia toimintoja verkkosivuille. Muun muassa esitarkastettavat lomakkeet, ulkoasun muutokset hiirtä linkin päälle vieden ja erilaiset ponnahdusikkunat voidaan ohjelmoida JavaScriptin avulla. Selain tulkitsee JavaScriptiä suoraan, mikä mahdollistaa sen, että JavaScript ei tarvitse erillistä tiedostoa, josta koodi suoritetaan, vaan se voidaan ohjelmoida suoraan HTML-tiedoston sisään.

JavaScript soveltuu parhaiten interaktiivisten toimintojen ja pienten animaatioiden tekoon. Liiallinen JavaScriptin käyttö ja varsinkin suuri määrä JavaScript-animaatioita tekee sivustosta raskaan ja hidastaa sivuston latautumista.

Nykyään JavaScriptiä voidaan käyttää myös palvelintason ohjelmointiin. Tällaisia JavaScriptiin pohjautuvia kieliä ovat muun muassa AngularJs, ExpressJs ja NodeJs. [7.]

JavaScript-kirjastot — JQuery

JavaScript-kirjastot ovat valmiita JavaScript-ohjelmointikielellä ohjelmoituja tiedostoja, jotka on tehty helpottamaan JavaScriptillä ohjelmointia. JavaScript-kirjastot voivat sisältää satojen koodirivien valmiita toiminnallisuuksia, joita voidaan ottaa käyttöön verkkosivustoille JavaScriptiä käyttämällä tekemällä niin sanottuja kutsuja kirjastosta. Kirjastojen käyttö aina mahdollisuuksien mukaan on suotavaa, sillä niiden tarkoitus on vähentää niin sanottua pyörän uudelleenkeksimistä. [8.]

3.3 HTML-ohjelmointikieli

HTML eli Hypertext Markup Language on koko selainpohjaisen internetin ja verkkosivujen tukirakenne. HTML kertoo verkkoselaimelle, miten verkkosivun kuvat ja teksti näytetään. HTML:llä rakennetaan niin sanotusti verkkosivujen runko käyttämällä tageja ja attribuutteja. Verkkosivut koostuvat HTML:n osalta pääosasta (head) ja vartalosta (body). Pääosaan tallennetaan tietoja, jotka eivät välttämättä näy suoraan selaimella sivustoa katsottaessa, vaan sinne tallennetaan tieto siitä, mistä sivustossa on kyse. Pääosassa kerrotaan myös, tuleeko selaimen käyttää ulkoisia tiedostoja, kuten PHP-tiedostoja tai tyylitiedostoja (CSS). Vartalo-osaan tallennetaan kaikki se tieto, mitä verkkosivulla halutaan näyttää. Vartalo-osassa kuvataan myös sivuston rakenne esimerkiksi erilaisin divein ja lomakkein. [9; 10.]

HTML5

HTML5 on viides ja uusin versio HTML-kielestä. Sen suurin muutos vanhempiin versioihin verrattuna on se, miten selain lukee ja havainnoi sivuille ohjelmoidut sovellukset. Muita uusia funktioita HTML5:ssä on se, miten sivuille upotetaan grafiikkaa, ääntä, videota ja interaktiivisia dokumentteja. HTML5:stä käytetään myös lyhennettä XHTML, joka tulee sanoista Extensible Hypertext Markup Language. [11.]

3.4 CSS-ohjelmointikieli

Cascading Style Sheets eli CSS on ohjelmointikieli, joka kuvaa verkkosivuston ulkoasua. Sillä voi määritellä esimerkiksi taustakuvan, värivalinnat, fontit ja sivuston asettelun. CSS ei ole mitenkään kytköksissä HTML:n kanssa, vaan sitä voidaan käyttää lähes kaikkiin XML-pohjaisiin ohjelmointikieliin. HTML:n ja CSS:n erottelu helpottaa sivuston ylläpitoa ja mahdollistaa usean sivun tyylittelyt samannäköisiksi. [12.]

4 Tallennetun tiedon visualisointi

Tallennetun tiedon visualisoinnin lähtökohtana on, että jossain on suuri määrä tietoa, jonka tutkimisen helpottamiseksi se olisi saatava selkeämpään muotoon eli se tulisi visualisoida. Visualisointi itsessään tarkoittaa tiedon havainnollistamista erilaisin taulukoin ja kaavioin, jolloin tiedon ymmärtäminen ja tulkinta on helpompaa. Muita visualisoinnin muotoja ovat muun muassa kuvat ja animaatiot. Tiedon välittämistä visuaalisin keinoin on käytetty kauan: jo luolamaalaukset välittivät tietoa. Yleisiä käsitteitä visualisoinnissa ovat

- kartat, jotka visualisoivat maastoa ja paikkatietoja
- kaaviot, jotka kuvaavat laskelmien tuloksia
- käsitekartat, jotka kuvaavat käsitteiden välisiä yhteyksiä
- tietokonegrafiikka, joka käsittelee tietokoneen lukemaa tietoa, esimerkiksi tietokonepelit.

Edvard Tuften [15.] mukaan kuvaajan voi vahingossa tehdä siten, että sen esittämä tieto ei ole enää tarpeeksi informatiivista. Tällöin kuvaajaan Tuften mukaan tulee ”Lie factor” eli valehtelukerroin. Valehtelukerroin voi muodostua esimerkiksi siten, että ku-

vaaja alkaa tavanomaisesta poiketen jostain muusta arvosta kuin nollasta, tai siten, että kuvaajaan on lisätty jokin kerroin, joka saa erot näyttämään todellista pienemmiltä.

Tufte on luonut myös hyvän informaation visuaalisen esittämisen viisi teesiä:

- Ensisijaisesti esitä tietoa.
- Vältä tiedon vääristämistä.
- Tee isoista tiedostomääristä koherentteja.
- Silmää on kannustettava vertailemaan tiedon eri osia.
- Paljasta tiedosta useita kerroksia; yleisnäkymästä yksityiskohtiin.

Näiden teesien avuksi Tufte on luonut apuvälineitä, kuten tieto-mustesuhde, jossa lasketaan tiedon näyttämiseen ja muuhun visualiseen taustakohinaan käytettävän musteen määrää. [13; 14; 15.]

4.1 Google Charts -kaaviopalvelu

Insinöörityössä käytettiin Googlen tarjoamaa kaaviopalvelua, joka vähentää JavaScriptiin käytettävän ohjelmoinnin ja JavaScriptin käyttämien kirjastojen määrää. Google Charts on työkaluna helppokäyttöinen ja ilmainen, ja se tarjoaa laajan valikoiman taulukoita ja kaavioita käytettäväksi. Googlen tarjoamat kaaviot ovat hyvin interaktiivisia ja mahdollistavat erilaisten työpöytämäisten toimintojen upottamisen sivuille ohjaamaan kaavioita. Jos kaavioiden ulkonäkö ei miellytä silmää, niitä voi muokata halutun näköisiksi. Osaa kaavioista on myös mahdollista lähentää tai kääntää kaavio sivuttain. Kaaviot perustuvat HTML5:een ja Scalable Vector Graphicsiin (SVG), joka on XML:ää käyttävä vektorikuvaformaatti kaksiulotteisille kuville. Se tukee myös kuvien interaktiivisuutta ja animointia. HTML5:n ja SVG:n käyttö tarjoaa lähes täydellisen selaintuen, mikä tarkoittaa sitä, että eri tietokoneella käytettävien selainten lisäksi tuettuna ovat myös mobiililaitteiden verkkoselaimet (iPhone, iPad, Android, yms), kaikki ilman erillisiä selaimen laajennusosia (plugin).

Googlella on tarjolla kuvan 2 mukaisesti muun muassa seuraavia taulukointi- ja kaaviomalleja:

- maantieteelliset kaaviot
- pistekaavio
- pylväsdiagrammi
- histogrammi
- sarakekaavio
- yhdistelmäkaavio
- aluekaavio
- piirakkakaavio
- taulukkokaavio.

Kuva 2. Google Chartin tarjoamia kaavioita [16]

Tarjolla on myös hieman epätavallisempia kaavioita, kuten mittaristo, puukaavio, kuplataulukko ja lukuisia muita kaavioita. Etsittäessä täydellistä kaaviota halutun tiedon näyt-

tämistä varten on hyvä tietää, että kaikki kaaviotyypit käyttävät lähes samanlaista tiedonvälitysformaattia, mikä mahdollistaa eri kaavioiden testaamisen samalla lähdetiedostolla. [17.]

4.2 Muita visualisointiohjelmia

Google Charts ei tietenkään ole ainut tallennetun tiedon visualisointiin käytettävä ohjelma. Muita ohjelmia ovat muun muassa CanvasJs ja JQWidgets, jotka toimivat hyvin pitkälti samalla tavoin kuin Google Charts.

CanvasJs

CanvasJs on helppokäyttöinen HTML5:tä ja JavaScritpiä käyttävä taulukkokirjasto. Se tukee myös hyvin erilaisia selaimia ja mobiililaitteita, kuten Google Chartskin tekee. CanvasJs tarvitsee toimiakseen JavaScript-kirjastoja, jotka on ladattava käyttäjän palvelimelle ja joihin viitataan HTML-tiedoston pääosassa. [18.]

JQWidgets

JQWidgets on erikoistunut verkkosivuille rakennettavien sovellusten luomiseen. Sovellus voi olla mikä tahansa pieni toiminnallinen osa verkkosivuilla tai laitteissa. JQWidgetsillä on muiden toiminnallisuuksien joukossa myös reilu määrä kaavioita. Sen toiminta perustuu myös JavaScriptiin ja omiin JavaScript-kirjastoihin, jotka on ladattava käyttäjän omalle palvelimelle. [19.]

Visualisointiohjelmien ongelmat

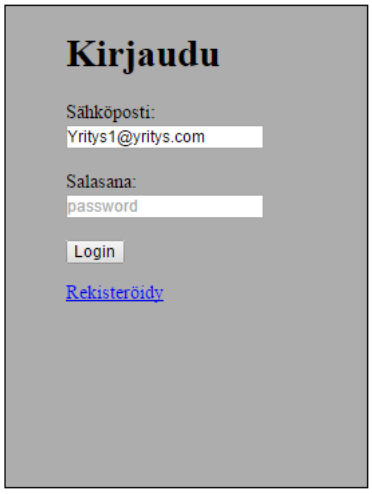
Kokeilin Google Chartsin lisäksi CanvasJs- ja JQWidgets-ohjelmia, mutta niiden käyttämät JavaScript-kirjastot ja hieman erilainen lähestymistapa tietokannasta haetun tiedon visualisoimiseen ei ollut sitä, mitä haettiin. JavaScript-kirjastojen toimiminen oli myös hieman kyseenalaista. Google Chartsin selkeä ohjeistus ohjelmoinnin kannalta, paremmannäköiset kaaviot ja suurempi valikoima saivat lopullisen valinnan kohdistumaan siihen.

5 Yritysasiakkaan sivusto ja kaaviot

Jotta kerätyn tiedon visualisoiminen internetsivuilla olisi mahdollista, insinöörityössä suunniteltiin yritysasiakkaiden sivustolle rakenne ja päätettiin, mitä julkaisualustaa käytetään. Tiedostojen kytköksiä selkeyttämään suunniteltiin tiedostorakenne, ja tietokantaa varten suunniteltiin tietokannan rakenne. Siihen, että esitettyjen kaavioiden tiedot ovat ajan tasalla, tarvitaan palvelintason ohjelmointia, ja siihen, että kaaviot näkyvät sivustolla, tarvitaan selaintason ohjelmointia.

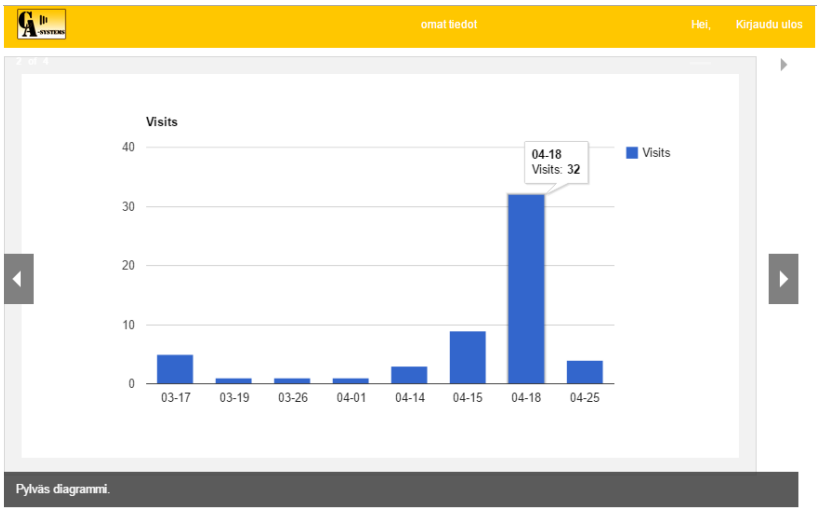
5.1 Sivuston rakenne

Yritysasiakkaiden sivuston rakenteesta tuli varsin yksinkertainen. Asiakas aloittaa sisäänkirjautumissivulta, jossa on mahdollista myös rekisteröidä uusi tili (kuva 3). Sivusto luotiin Foundationin julkaisujärjestelmällä, ja halutut kaaviot esitetään kuvakarusellissa.



Kuva 3. Yritysasiakkaiden sisäänkirjautumis- ja rekisteröitymissivu.

Kirjautumissivulta yritysasiakas siirtyy sivuille, jotka näyttävät erilaisin Googlen tarjoamin taulukoin haluttuja tietoja henkilöasiakkaista ja muusta kerätystä tiedosta (kuva 4).



Kuva 4. Yritysasiakkaiden sivuston ulkoasua.

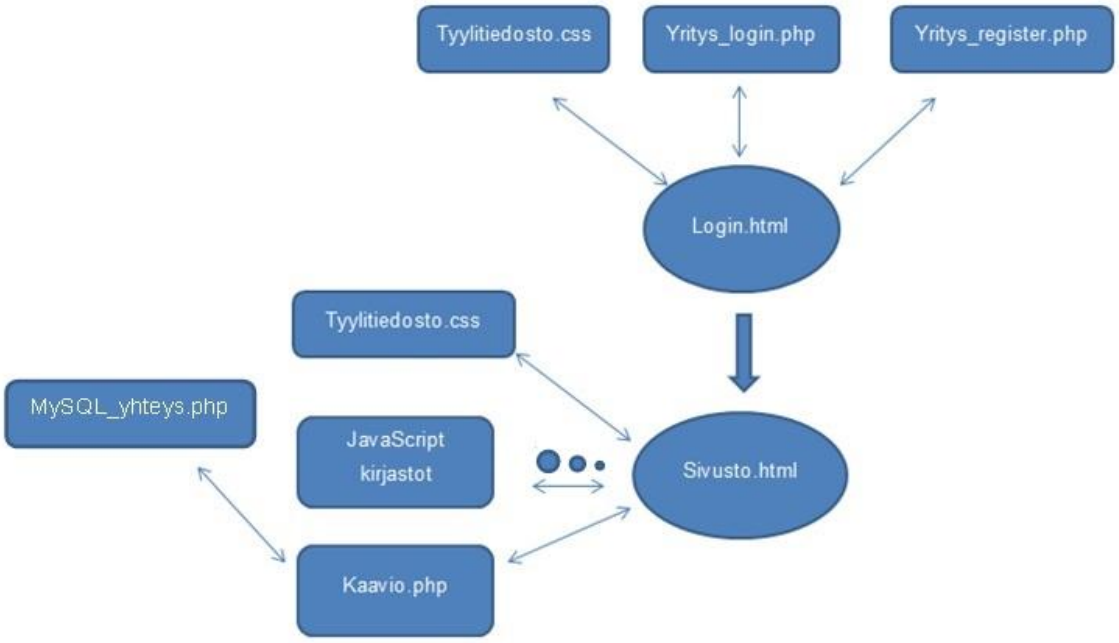
Foundation

Foundation on responsiiviseen toteutukseen painottunut julkaisualusta verkkosivuille. Foundationia tehtäessä on erityisesti pidetty mielessä kolme tärkeintä tekijää, kun mietitään verkkosivujen tekoa. Ensinnäkin sivuston on oltava nopea sen käyttäjälle. Foundationilla tehtyjen sivujen rakenne antaa ohjelmoijalle avaimet tehdä täydellisesti optimoituja sivustoja laitteesta tai näytön koosta riippumatta. Toiseksi, verkkosivujen ohjelmoiminen on tehty erittäin nopeaksi ruudukkomalliin pohjautuvan pohjan avulla. Foundationissa on myös uusi komentorivityökalu, jonka avulla voi aloittaa uuden Foundation-projektin erittäin nopeasti. Kolmas ajatus on opettelun helppous. Foundation on tehty siten, että se on helppo ja mukava oppia. Foundationin omilla sivuilla on myös kattava valikoima erilaisia ohjeistuksia, mikä auttaa ensikertalaista opettelussa. Näitä kolmea teemaa käyttäen Foundation on yltänyt Top15 avoimen lähdekoodin projektit -listalle. [20.]

Tiedostorakenne

Jokainen HTML-, PHP- ja CSS-tiedosto on kytköksissä toiseen tiedostoon. Tiedostorakenteen suunnittelu on tärkeä osa toimivan sivuston rakentamista. Kuva 5 antaa käsitystä siitä, millaisella tiedostorakenteella yritysasiakkaiden sivusto toimii. Kuvassa 5 ei ole eritelty kansiorakennetta, mutta pääpiirteittäin kansiorakenne menee siten, että

HTML-tiedostot ovat yhdessä kansiossa ja CSS- ja PHP-tiedostot HTML-tiedostojen alakansioissa.



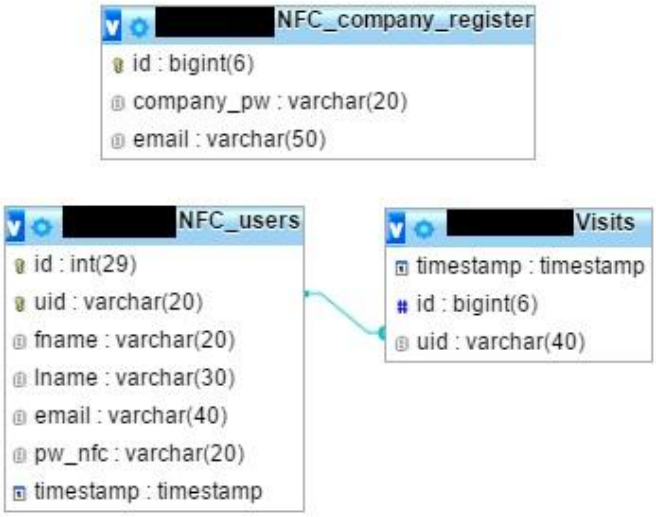
Kuva 5. Yritysasiakkaiden sivuston tiedostorakenne.

Sisäänkirjautumissivu on yhteydessä kirjautumista käsittelevään PHP-tiedostoon tai vaihtoehtoisesti, jos kyseessä on uusi asiakas, rekisteröintiä käsittelevään PHP-tiedostoon. Sisäänkirjautumissivulla on myös oma CSS-tiedosto, jonka kautta voidaan muokata sivun tyyliä fonteista taustaväreihin. Kirjautumissivu on yhteydessä pääsivulle, joka saa tyylinsä omasta CSS-tiedostosta. Pääsivun jokaisella kaaviolla on oma PHP-tiedosto, jotka kaikki käyttävät samaa MySQL-yhteyden luovaa PHP-tiedostoa. Osa pääsivun JavaScript-kirjastoista on pilvessä, johon voidaan ottaa suoraan yhteyttä JavaScriptistä, mutta esimerkiksi Foundationin käyttämät JavaScript-kirjastot ovat omassa kansiossaan ladattuna palvelimelle.

5.2 Tietokannan rakenne

Sivustolla käytettiin kolmea erillistä tietokantataulua. Taulut koostuvat kuvan 6 mukaisista yritysasiakas- (NFC_company_register), henkilöasiakas- (NFC_users) ja vierailut-taulusta (Visits). Yritysasiakkaan tauluun tallennetaan tiedot, joilla yritys voi kirjautua sisään sovellukseen, eli sähköpostiosoite ja salasana. Jokainen yritys saa oman id:n,

joka toimii taulussa myös yleisavaimena. Henkilöasiakkaan tauluun tallennetaan henkilökohtaisen identifiointinumeron (id) ja NFC-kortista saadun uniikin numeron (uid) lisäksi etunimi ja sukunimi, sähköpostiosoite, salasana henkilöasiakkaan sivuille kirjautumista varten ja aikaleima uuden käyttäjän luomisesta. Vierailu-tauluun tallentuvat aikaleima vierailusta, vierailun uniikkinumero ja vierailijan NFC-kortin numero.



Kuva 6. MySQL-tietokannan taulut ja rakenne.

Kuvasta 6 näkyy, että henkilöasiakastaulun perusavain ”uid” on yhdistetty tapahtumataulun viiteavaimeksi ”uid”. Yritysasiakkaan taulusta ei ole tehty suoria viittauksia muihin tauluihin tietokannassa. Viittaukset yritysasiakkaan tauluun tehdään PHP:n avulla siten, että jokaisella yritysasiakkaan sisäänkirjautumiskerralla haetaan yritysasiakkaan taulusta id kyseisen tapahtuman tapahtuma id:ksi. Visit-taulun id tulee siitä, että kun henkilöasiakkaan tapahtuma tallennetaan NFC-sovelluksen kautta tietokantaan, saa kyseinen tapahtuma sen yritysasiakkaan id:n, jonka sovelluksesta tapahtuma on tallennettu.

5.3 Ohjelmointi

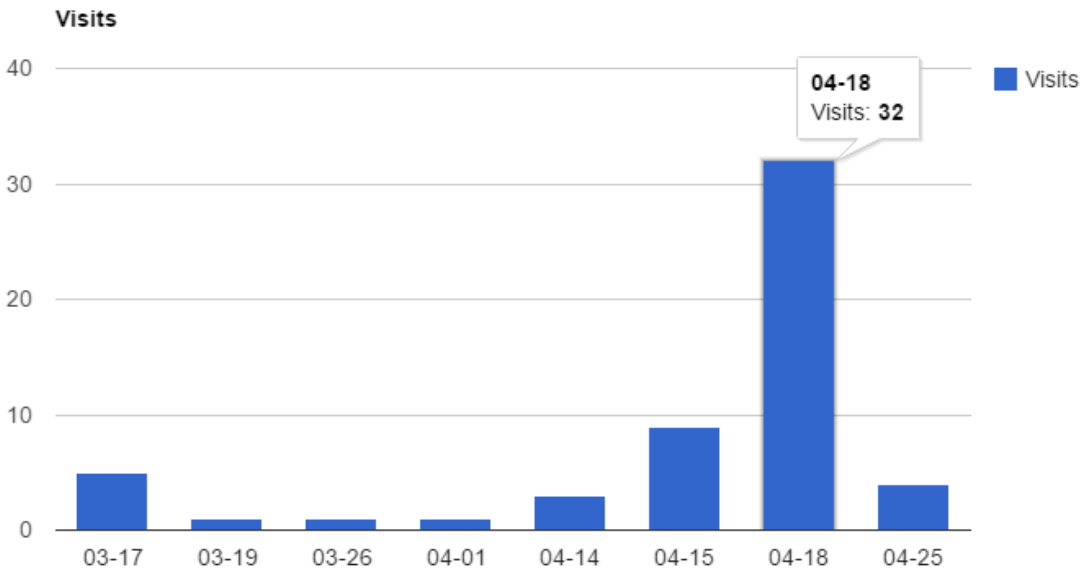
Aloittaessani projektia tutkin ja kokeilin erilaisia visuaaliseen datankäsittelyyn soveltuvia ohjelmia, ja testaamisen päätteeksi päädyin käyttämään Google Chartsia. Googlen sivuilla oli selaintason ohjelmointiin jonkin verran opastusta, mutta palvelintason ohjelmointi oli selvitettävä itse. Googlen internetsivuilta löytyi vain malli, minkänäköisellä merkkijonolla saadaan tieto näkymään. Sen jälkeen selvitin, miten saisin halutun tiedon

valmiista tietokannasta juuri oikealla tavalla siirrettyä PHP:n avulla lopulliselle sivustolle selaintason ohjelmoinnin käyttöön.

Kun tietokanta oli valmis, aloitin kerätyn tiedon käsittelyn ja reaaliaikaisen visualisoinnin ohjelmoinnin sivuille. Ohjelmointi voidaan jakaa viiteen osa-alueeseen. Ensimmäisessä osassa PHP-tiedosto hakee halutun tiedon tietokannasta. Tämän jälkeen haetulle tiedolle on tehtävä muutoksia siihen, minkälaisena se tulostuu, sillä kaavioiden käyttämä JavaScript osaa lukea vain tiettyjen kehyksien sisällä lähetettyä tietoa. Seuraavaksi PHP-tiedoston tieto käännetään muotoon, jota JavaScript osaa käsitellä. HTML-tiedostossa haetaan PHP-tiedostosta lähetetty tieto, minkä jälkeen JavaScriptin ja JavaScript-kirjastojen avulla tehdään kaavio valmiiksi verkkosivustolle näytettäväksi.

Kuvan 7 palkkikaavion eri vaiheet olivat seuraavat: Google Chartsin sivuilla on tarkka määritelmä siitä, minkälaiseen jonomuotoon näytettävä tieto on saatava, että kaavion JavaScript osaa käyttää sitä hyväkseen. Kuvan 7 mukaiseen palkkikaavioon tarvittavasta tiedosta on saatava tehtyä seuraavannäköinen jono PHP-tiedostossa, jotta se on käyttökelpoinen JavaScriptillä:

[[“x-akselin nimi”, “y-akselin nimi”],[”x-akselin arvo”, y-askelin arvo],[”x-akselin arvo”, y-askelin arvo], [”x-akselin arvo”, y-askelin arvo],[”x-akselin arvo”, y-askelin arvo]]



Kuva 7. Palkkikaavio asiakkaiden kävijämääristä.

PHP:n tiedonkäsittely

PHP-tiedoston alkuun aloitetaan uusi istunto ja tehdään kutsu tietokantaan yhdistämisestä require(”connect.php”):n avulla (kuva 8). Connect.php sisältää yhteydenottopyynnön haluttuun tietokantaan tarvittavien käyttäjätietojen kera.

```
<?php
session_start();
//load and connect to MySQL database
require("connect.php");
    #Connect to the database
```

Kuva 8. Istunnon aloitus ja yhteydenotto tietokantaan.

Kun tietokantaan on yhdistetty, voidaan tehdä pyyntö tietokantaan kuvan 9 mukaisesti.

```
//sessionid on sivulle kirjauduttaessa valikoitunut yrityksen id
$sessionid = $_SESSION['id'];

$SQLString = "SELECT DATE(timestamp) AS `Date`
    , COUNT(*) AS Visits
FROM    `Visits` WHERE id = $sessionid
GROUP BY
    DATE(timestamp)
    ";
```

Kuva 9. Istunnon yksilöllistäminen ja pyyntö tietokantaan.

Kuvasta 9 nähdään, että aluksi määritellään $sessionid. Istunnon id tulee siitä, että kun asiakas kirjautuu sisään sivustolle, haetaan istunnolle id:ksi yritysasiakkaan tietokantataulusta yrityksen id.

Ensin tietokannasta haetaan visits-taulusta kaikki aikamerkinnät eli timestampit, joiden id vastaa $sessionid:tä, jolloin saadaan jaoteltua pois kaikki ne tapahtumat, jotka eivät koske kyseistä yritystä. Samalla lasketaan tapahtumat yhteen COUNT-komennolla ja ryhmitellään päivämäärän mukaan. Kutsun alussa oleva DATE(timestamp) AS ’date’ muokkaa timestampin muotoa tavanomaisesta, joka näyttää päivän, kuukauden, vuoden, tunnin, minuutin ja sekunnin tapahtuman luomisesta tietokantaan, tarvittavaan pelkän kuukauden ja päivän näyttävään muotoon. Tämä kaikki tallennetaan $SQLString-instanssin sisään.

Seuraavaksi $SQLStringiin tallennettu tieto muokataan muotoon, jossa voidaan eritellä rivit ja laskea eri päivinä tapahtuvien tapahtumien määrä (kuva 10).

```
$result = mysql_query($SQLString);
$num = mysql_num_rows($result);



$data[0] = array('Date','Visits');
for ($i=1; $i<($num+1); $i++)
{
    $data[$i] = array(substr(mysql_result($result, $i-1, "Date"), 5, 5),
                      (int) mysql_result($result, $i-1, "Visits"));
}
```

Kuva 10. Tapahtumien erittely eri päiville.

Kuvan 10 alussa esiintyvään $result-parametriin tallennetaan kaikki tietokannasta haetut rivit yhdeksi paketiksi, ja $num järjestelee $resultin saaman paketin mysql_num_rowsin avulla tietynlaiseksi jonoksi, jota on helpompi käsitellä seuraavassa for-silmukkavaiheessa. For-silmukka on erittäin yleisesti käytetty ohjelmointitapa, jossa lasketaan yhteen tietoja niin kauan, kuin niitä riittää, minkä jälkeen tieto tallennetaan halutunnimiseen parametriin, tässä tapauksessa parametriin nimeltä $data. Tässä for-silmukassa samalla kun eri päivämäärille löytyviä tapahtumia lasketaan yhteen, niistä muodostetaan jono, joka on vielä tarvittavan aiemmin mainitun jonon mukainen. For-silmukan sisällä olevat ”Date” ja ”Visits” ovat staattisia ja tulevat jonon alkuun merkitsemään x- ja y-akseleiden nimiä.

Aiemmin avatulla koodilla saadaan aikaan kuvan 11 mukainen merkkijono, jossa on lueteltu päivämäärien mukaan vierailujen yhteenlaskettu määrä. Esimerkiksi 04-18 (kk-pv) on 32 asiakasta tehnyt vierailun yrityksen NFC-sovellukseen.

```
[["Date","Visits"],["03-17",5],["03-19",1],["03-26",1],["04-01",
1],["04-14",3],["04-15",9],["04-18",32],["04-25",4]]
```

Kuva 11. PHP-tiedoston tulostama merkkijono.

JavaScriptille sopiva merkkijono

Kerätyistä tiedoista muodostettu merkkijono ei ole suoraan luettuna PHP:stä sellaisessa muodossa, jota JavaScript ymmärtäisi. Tästä syystä merkkijono muutetaan JSON:n avulla sellaiseen muotoon, mikä sopii JavaScriptille. PHP:ssä tiedoston ulkopuolelle lähetettävä tieto lähetetään ”echo”-komennon avulla, ja tässä tapauksessa haluttu merkkijono oli tallennettu parametriin nimeltä $data, jolloin echolla lähetetään JSON:n muokkaama merkkijono $data kuvan 12 mukaisesti.

```
echo json_encode($data);
```

Kuva 12. Parametrin $data lähetys PHP-tiedostosta.

JavaScript ja HTML

Kun PHP:stä on saatu tiedot lähetettyä JavaScriptin ymmärtämässä muodossa, tieto haetaan HTML-tiedoston käyttöön. Ennen kuvassa 13 näkyvää koodia on ilmoitettu, mitä JavaScript-kirjastoja tarvitaan. JavaScript-kirjastojen ilmoittamisen jälkeen ladataan Googlen JavaScript-kirjastosta oikean kaavion tekemiseen tarvittava tietopaketti, tässä tapauksessa ”corechart”, ja kyseisestä paketista juuri ”drawChart”. Tämän jälkeen voidaan tehdä funktio, jossa ensin haetaan aiemmin tehty JSON- merkkijono halutusta tiedostosta kuvan 13 mukaisesti. Sitten valitaan muuttuja, tässä tapauksessa ”obj”, johon tallennetaan JSON-merkkijono purettuna JavaScriptin ymmärtämään muotoon, ja siitä suoraan tallennetaan muuttujaan ”data” Googlen visualisoimiseen käyttämän ohjelman mukaan muuttujasta ”obj” tehty versio. Lopuksi voidaan muuttujan ”options” avulla tehdä kaavioihin muokkauksia, ja tässä tapauksessa kaaviolle on annettu nimi ”Visits”.

```
<script type="text/javascript">
    google.load("visualization", "1", { packages: ["corechart"] });
    google.setOnLoadCallback(drawChart);

    function drawChart() {
        var jsonData = $.ajax({
            url: "php/pylvas.php",
            dataType: "json",
            async: false
        }).responseText;

        var obj = jQuery.parseJSON(jsonData);
        var data = google.visualization.arrayToDataTable(obj);

        var options = {
            title: 'Visits'
        };

        var chart = new google.visualization.ColumnChart(
                    document.getElementById('chart_div'));
        chart.draw(data, options);
```

Kuva 13. Pylväskaavion JavaScript-koodi.

Kun muuttujat ”data” ja ”options” ovat valmiita, tehdään uusi muuttuja, esimerkkitapauksessa ”chart”, jonka tietoihin tallennetaan pylväskaavion visuaalisen ilmeen tiedot ja sille oma nimi eli id. Lopuksi vielä piirretään kaavio käyttämällä muuttujia ”data” ja ”options”.

Vaikka JavaScriptin osalta kaavio on nyt täytetty oikeilla tiedoilla ja käsketty selaimen piirtää lopullinen versio, sivustolla se ei vielä näy. HTML-osuuden <body>-puolelle on tehtävä kaavioille omat paikat, jotta verkkoselain tietää, mihin kaavio sivustolla tulee. Kuvassa 14 näkyy, kuinka esimerkkisivuston kuvakaruselliin on tehty paikka palkkikaaviolle antamalla divin id:ksi sama nimi kuin kaavion id oli. Toiseen diviin on annettu vielä kuvakarusellin aihion nimike.

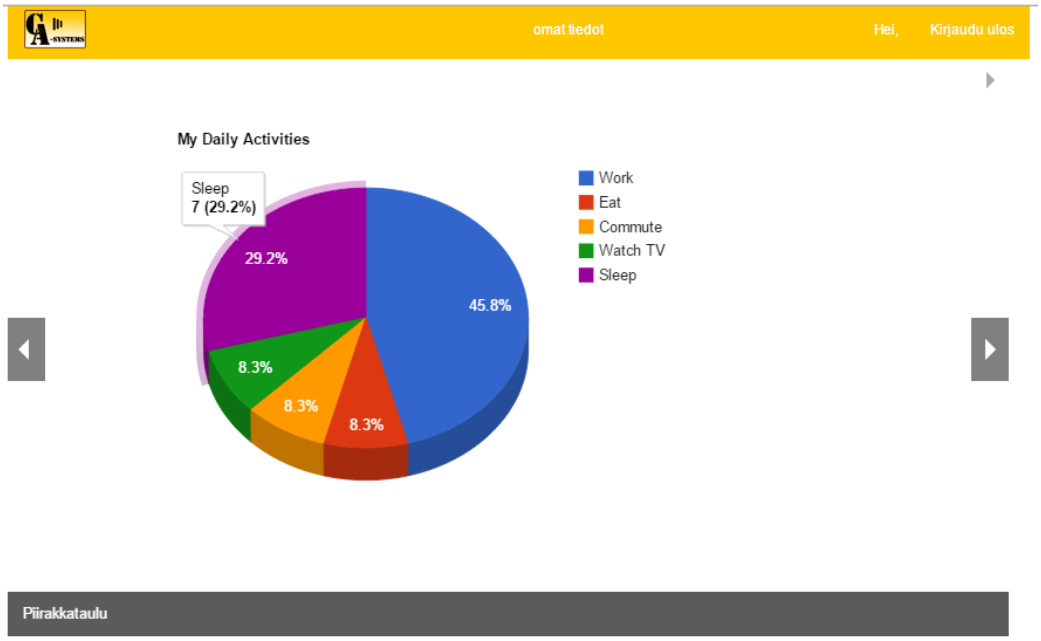
```
<li>
  <div class="panel" id="table_div" >

  <div class="orbit-caption">
    Table diagrammi.
  </div>
</li>
```

Kuva 14. HTML-koodi, joka tuo kaavion näkyviin sivuilla.

5.4 Kokeillut kaaviot

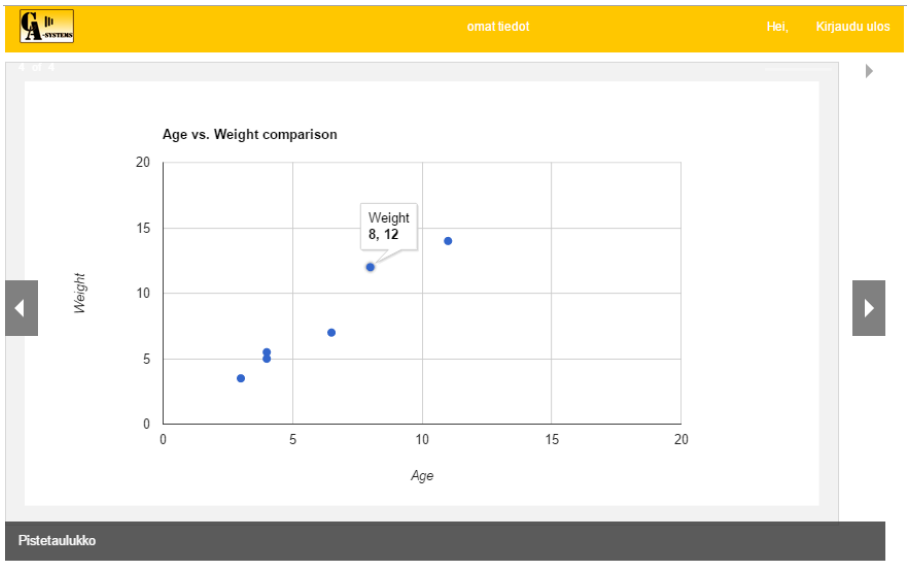
Lopuksi rakensin sivustolle palkkikaavion lisäksi muutaman testiluontoisen kaavion. Testikaavioiden tarkoituksena oli kokeilla kaavioihin tehtäviä visuaalisia efektejä ja toiminnallisuuksia sivustolla. Testikaavioiksi valitsin mielestäni kaikista tyypillisimpiä tai mielenkiintoisimpia kaavioita, joita Google Chartsilla oli tarjottavana. Kokeiltuja kaavioita aiemmin esitellyn palkkikaavion lisäksi ovat piirakkataulu, pistetaulukko ja taulukkokaavio.

Piirakkakaavioita (kuva 15) on parasta käyttää prosentuaalisten muutosten näyttämiseen, esimerkkinä miesten ja naisten osuus kävijöistä tai päivän aktiviteettien suhde toisiinsa.



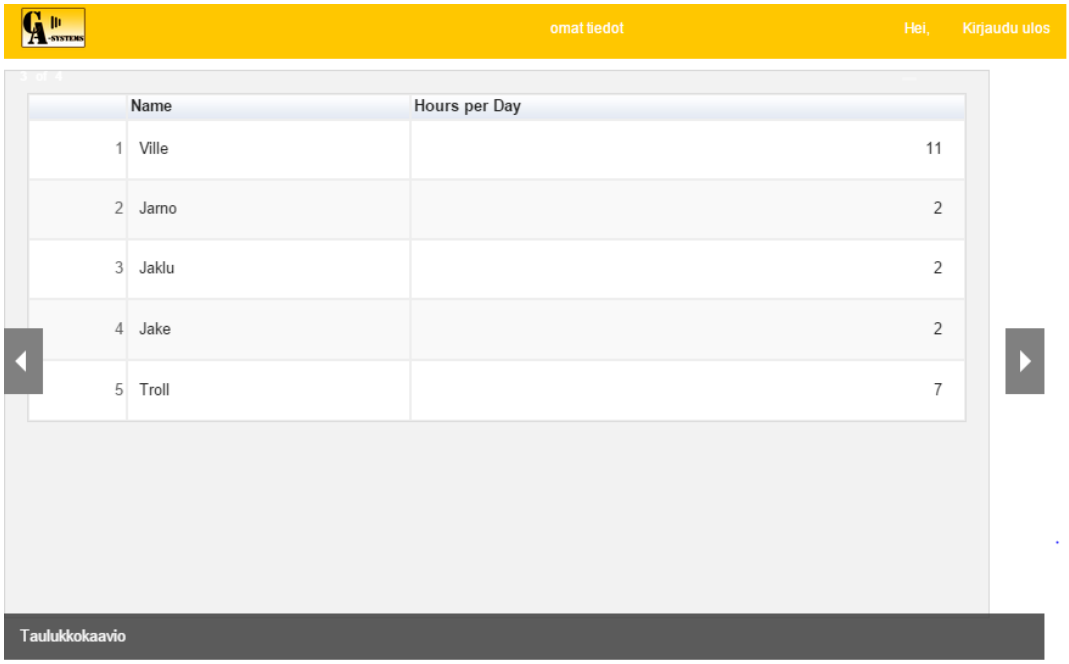
Kuva 15. Piirakkataulu sivustolla.

Pistetaulukko (kuva 16) soveltuu erittäin hyvin suuren tietomäärän käsittelemiseen, jolloin pienemmälle alueelle saadaan enemmän väkäsiä osoittamaan eri asioita.



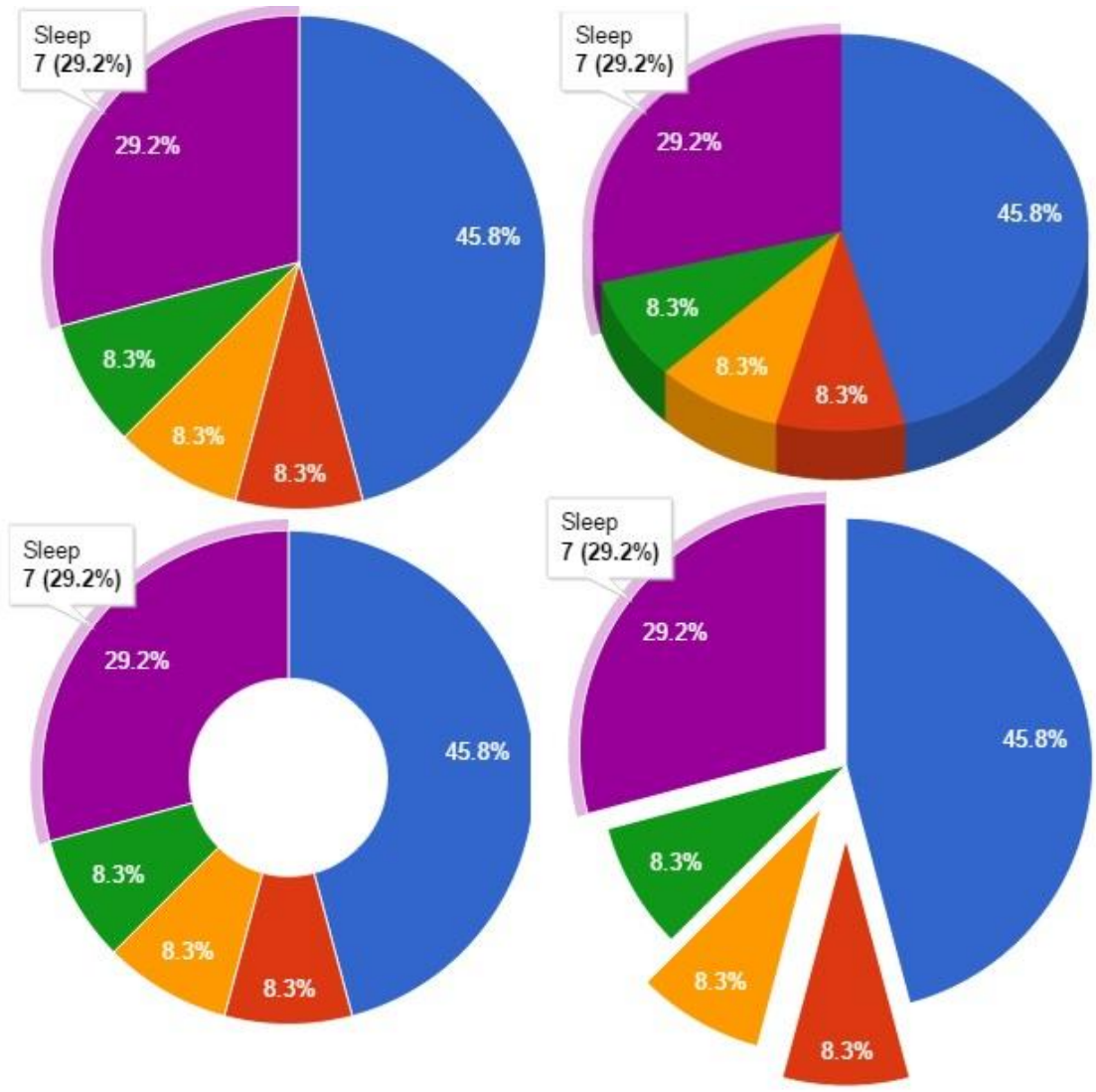
Kuva 16. Pistetaulukko.

Taulukkokaavioon (kuva 17) voi luetella hyvin esimerkiksi käyttäjien nimiä tai muuta tekstiä tietojen listaamiseen halutulla tavalla.



Kuva 17. Taulukkokaavio.

Kaavioita voidaan muokata tietyssä määrin kuvassa 13 näkyvän koodin ”options”-kohdasta. Esimerkkinä käytän piirakkakaaviota ja muokkaan siitä neljä erinäköistä versiota lisäämällä tai muuttamalla asetuksia. Piirakkakaavioon voidaan valita kuva 3D:llä tai ilman, kirjoittamalla asetuksiin ”is3D: true” tai vaihtoehtoisesti ”is3D:false” (kuva 18). Vaihtoehtoisesti, jos ei pidä piirakkamallista vaan haluaisi tehdä kaaviosta donitsimaisen, täytyy asetuksiin lisätä ”pieHole: 0.4”, jossa lukumääre vastaa donitsin halkaisijaa. Donitsi ei toimi samaan aikaan 3D:n kanssa. Piirakkakaaviosta saadaan myös eroteltu versio asetusten avulla käyttämällä ohjelmointiriviä ”slices: {1:{offset:0.3}, 2:{offset:0.2}, 4:{offset: 0.1}”, jossa ensimmäinen numero vastaa siivua ja toinen numero vastaa erittelyyn käytettyä matkaa. Kuvassa 18 näkyvät erilaiset piirakkakaaviot asetuksiin tehdyin muutoksin.



Kuva 18. Piirakkakaaviot ilman 3D:tä, 3D:n kanssa, donitsina ja eriteltynä.

Tietyissä kaavioissa on ”options”-kohdan lisäksi mahdollista tehdä muutoksia ulkonäköön muita tapoja käyttäen. Silloin muutokset lisätään jo PHP-tiedostossa tekemällä muutoksia lähetettävään merkkijonoon.

6 Yhteenveto

Tiedon jakamista verkkosivuilla visuaalisesti ja reaaliaikaisesti on kehitelty pitkään, ja viimeistään Googlen tuoman Charts-palvelun jälkeen erilaisten kaavioiden ja taulukkojen lisääminen sivuille on tehty erittäin helpoksi. Reaaliaikaisuuden aikaansaamiseksi tarvitaan tietokantasovellus, verkkosivut ja tietotaitoa yleisimmistä selain- ja palvelinta-

son ohjelmointitavoista. Tarvittavia ohjelmointikieliä ovat PHP, JavaScript ja HTML. Lisäksi tarvitaan tietotaitoa JavaScript-kirjastoista, JSON:sta ja CSS:stä. Googlen kaaviopalvelu on tällä hetkellä käyttäjäystävällisin käyttöönottaa, ja Googlen sivuilla on paljon alkuun pääsemiseen tarvittavia lähteitä. Google tarjoaa kaavioilleen myös huomattavan paljon muokkaamismahdollisuuksia.

Insinöörityössä tehtiin reaaliajassa muuttuva palkkikaavio, joka hakee henkilöasiakkaiden vierailut tietokannasta ja syöttää ne Foundation-julkaisualustaa käyttävälle verkkosivulle. Kaaviopalveluna käytettiin Google Charts -palvelua. Muitakin visualisointipalveluja kokeiltiin, mutta huonolla menestyksellä. Googlen sivuilla, kaavioiden ohjeistuksessa, mainittiin vain, millainen merkkijonon on oltava, jotta kaavio saadaan näkyviin. Tästä syystä käytin valmiiksi osaamiani ohjelmointikieliä ja tietokantapalvelua.

Lähteet

1 Tietokantojen perusteet. 2000. Verkkodokumentti. Helsingin yliopisto. < https://www.cs.helsinki.fi/u/laine/tikape/moniste/osa1.pdf > Luettu 6.4.2015.

2 Relaatiotietokannat, ER-kaavion muuntaminen relaatiotietokannaksi, Normalisointi, SQL ja ODBC - Luento 3. 2011. Verkkodokumentti. Jyväskylän yliopisto. < http://appro.mit.jyu.fi/tiedonhallinta/luennot/luento3 > Luettu 6.4.2015.

3 About MySQL. 2015. Verkkodokumentti. Oracle. < http://www.mysql.com/about/ > Luettu 6.4.2015.

4 NOSQL DATABASES EXPLAINED. 2015. Verkkodokumentti. Mongodb. < http://www.mongodb.com/nosql-explained > Luettu 6.4.2015.

5 Preface. 2015. Verkkodokumentti. The PHP Group. < http://php.net/manual/en/preface.php > Luettu 6.4.2015.

6 What can PHP do? 2015. Verkkodokumentti. The PHP Group. < http://php.net/manual/en/intro-whatcando.php > Luettu 6.4.2015.

7 JavaScript-perusopas: Osa 1 – Perusteet. 2007. Verkkodokumentti. Ohjelmointiputka. < http://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=js_01 > Luettu 6.4.2015.

8 JavaScript-kirjastot. 2010. Verkkodokumentti. Avkymppi. < http://www.avkymppi.net/joomla/vinkit/9-millions-of-smiles.html > Luettu 6.4.2015.

9 HTML (Hypertext Markup Language). 2005. Verkkodokumentti. Techtarget. < http://searchsoa.techtarget.com/definition/HTML > Luettu 6.4.2015.

10 HTML - HyperText Markup Language. Verkkodokumentti. Webopedia. < http://www.webopedia.com/TERM/H/HTML.html > Luettu 6.4.2015.

11 HTML5. Verkkodokumentti. Webopedia. < http://www.webopedia.com/TERM/H/HTML5.html > Luettu 6.4.2015.

12 What is CSS? Verkkodokumentti. W3.org. < http://www.w3.org/standards/webdesign/htmlcss#whatcss > Luettu 6.4.2015.

13 Computing Lie Factor by Dividing Percentages. Verkkodokumentti. Graphics Press LLC. < http://www.edwardtufte.com/bboard/q-and-a-fetch-msg?msg_id=00002o > Luettu 6.4.2015.

14 Siirtola, Harri. 2007. Interactive visualization of multidimensional data. Väitöskirja. Tampereen yliopisto.

15 Tufte, Edvard. 1983. The visual display of quantitative information. Graphics Press, LCCN.

16 Chart Gallery. 2015. Verkkodokumentti. Google.
< https://developers.google.com/chart/interactive/docs/gallery > Luettu 6.4.2015.

17 Using Google Charts. 2015. Verkkodokumentti. Google.
< https://developers.google.com/chart/interactive/docs/index > Luettu 6.4.2015.

18 Introduction. Verkkodokumentti. CanvasJS.
< http://canvasjs.com/docs/charts/intro/ > Luettu 6.4.2015.

19 About Us. Verkkodokumentti. JQWidgets.
< http://www.jqwidgets.com/about/ > Luettu 6.4.2015.

20 History of Foundation. Verkkodokumentti. Foundation.
< http://foundation.zurb.com/learn/about.html > Luettu 6.4.2015.