OULUN AMMATTIKORKEAKOULU

Timo Rautio

Mainosnäyttöjen etähallintaohjelmisto TosiMedia Ay:lle

Mainosnäyttöjen etähallintaohjelmisto TosiMedia Ay:lle

Timo Rautio
Opinnäytetyö
Kevät 2016
Tietojenkäsittelyn koulutusohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietojenkäsittely, Web-sovelluskehitys

Tekijä(t): Timo Rautio
Opinnäytetyön nimi: Mainosnäyttöjen etähallintaohjelmisto TosiMedia Ay:lle
Työn ohjaaja: Pekka Ojala
Työn valmistumislukukausi- ja vuosi: Kevät 2016 Sivumäärä: 19

Opinnäytetyön toimeksiantajana toimi Oululainen TosiMedia Ay. Opinnäytetyössä kehitettiin mainosnäyttöjen etähallintasovellus yrityksen Ideaparkissa sijaitsevalle Popup-center -toimipisteelle.

Työn tavoitteena oli suunnitella ja toteuttaa yrityksen käyttöön ohjelmisto, joka mahdollistaa mainosnäytöissä esitettävän sisällön hallinnan. Ohjelmistolla tulee pystyä lisäämään ja poistamaan esitettävää materiaalia, mikä tässä tapauksessa tarkoittaa JPEG- tai JPG -kuvasisältöä.

Opinnäytetyössä esitellään kehitettävän ohjelmiston suunnittelussa ja toteutuksessa käytetyt tekniikat ja menetelmät. Lopputuloksena syntyi ohjelmisto, jonka avulla käyttäjä pystyy kirjautumaan sivustolle sekä lisäämään ja poistamaan mainosnäytöissä esitettävää materiaalia.

Asiasanat: PHP, MySQL, Codeigniter

ABSTRACT

Oulu University of Applied Sciences
Degree Program in Business Information Systems, Web Application Development

---

Author(s): Timo Rautio
Title of thesis: Contens management software for TosiMedia Ay
Supervisor(s):Pekka Ojala
Term and year when the thesis was submitted: Spring 2016 Number of pages:19

The subject of this thesis was to create a web-based software for managing content of advertising displays to TosiMedia Ay.

The aim was to design and implement a software that could provide a remote access for user to manage content displayed in advertising displays. The user must be able to add and remove JPEG or JPG pictures with the software.

The thesis presents the techniques and methods used in the planning and implementation. The end result of thesis was a simple web-based software that has a user authentication component and a possibility to add and remove content from the advertising displays remotely.

Keywords: PHP, MySQL, Codeigniter

SISÄLLYS

1 JOHDANTO

Opinnäytetyön tarkoituksena oli suunnitella ja kehittää TosiMedia Ay:lle mainosnäyttöjen etähallinnan mahdollistava sovellus, jonka avulla yrityksen toimihenkilö pystyy lisäämään järjestelmään käyttäjiä ja käyttäjät puolestaan voivat lisätä näytöillä esitettäviä mainoskuvia käymättä lainkaan paikanpäällä. Ohjelmisto poistaa nykyisestä toimintaprosessista turhia työvaiheita ja säästää näin työaikaa ja polttoainekustannuksia.

Käyttäjien tallentamiseen käytettiin MySQL-tietokantaa ja ohjelmointikielenä PHP:ta. Kehitystyöhön käytettiin NetBeans 8.0.2 -ohjelmointiympäristöä ja CodeIgniter 3.0 -ohjelmistokehystä, koska siitä löytyy valmiiksi lukuisia työtä helpottavia funktioita. MVC-mallia hyödyntävä CodeIgniter mahdollistaa selkeän ja helppolukuisen ohjelmakoodin.

Sovelluksen toimivuus testattiin Google Chrome -selaimella, jonka versionumero opinnäytetyötä kirjoitettaessa oli 42.0.2311.90 m. Selainyhteensopivuutta ei testattu tai suunniteltu muille selaimille tämän opinnäytetyön puitteissa.

2 TOIMEKSIANNON TAUSTA JA TAVOITTEET

2.1 Toimeksiantaja

Työn toimeksiantaja on TosiMedia Ay -niminen oululainen vuonna 2014 perustettu markkinointialan pienyritys. Yritys lanseerasi vuoden 2015 ensimmäisellä neljänneksellä ainutlaatuisella konseptilla toimivan Pop Up-centerin Oulun Ideaparkkiin. Pop Up-centeristä yritykset voivat vuokrata erilaisia markkinointiratkaisuja kuten kokonaisen myymälätilan, tuotevitriinin, promootio- tai myyntipisteen. TosiMedia pystyy täten tarjoamaan uusille ja olemassa oleville yrityksille mahdollisuuden markkinoida tuotteitaan ilman pitkiä sitoutumisia tai kalliita alkuinvestointeja.

2.2 Opinnäytetyön tavoitteet

Työn tavoitteena on toteuttaa TosiMedia Ay:lle helppokäyttöinen ja nopeasti päivitettävissä oleva web-sovellus Pop Up-centerin mainosnäyttöihin. Näytöissä tulee voida esittää mainoksia asiakasyritysten tuotteista ja palveluista. Mainoksia on tarkoitus esittää web-selaimessa koko ruutu -tilassa siten, että esitys käynnistyy automaattisesti näyttöjen käynnistyessä ajastetusti ja näyttöihin haetaan aina ajantasainen sisältö.

Järjestelmä on tarkoitettu ainoastaan yrityksen työntekijöiden käyttöön, joten ainoastaan työntekijät pystyvät lisäämään ja poistamaan sisältöä järjestelmästä. Järjestelmä tulee etäkäyttöisyytensä vuoksi kuitenkin julkiseksi ja sitä varten siihen toteutetaan hallintatoimintoja varten erillinen kirjautumissivu sekä käyttäjien hallintatoiminto. Opinnäytetyössä henkilökohtainen tavoite on oppia hyödyntämään ohjelmointitaitoja reaalimaailman ongelmanratkaisussa.

3 TEKNIIKAT JA MENETELMÄT

Ohjelmoinnissa on käytetty CodeIgniter-sovelluskehystyä (Framework), joka perustuu MVC-arkkitehtuuriin. Sivuston toiminnallisuus, kuten kirjautuminen on toteutettu PHP-ohjelmointikielellä ja MySQL-tietokannalla. Ulkoasu on toteutettu käyttämällä HTML:ää sekä CSS-tyylitiedostoja.

3.1 Codeingniter

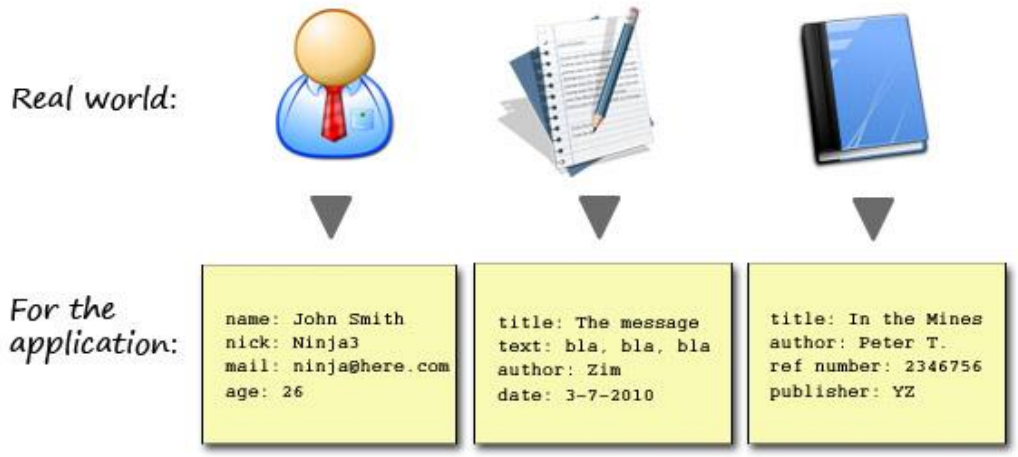
Codeigniter on Rick Ellisin vuonna 2006 EllisLab:lle kehittämä avoimeen lähdekoodiin perustuva MVC-arkkitehtuuria noudattava PHP-sovelluskehys, jonka tärkein päämäärä on helpottaa ohjelmistokehittäjien työtä (Ellislab, viitattu 20.4.2015). Codeigniter sisältää valmiiksi runsaasti kirjastoja useimmiten käytettäviä toimintoja varten, jotka nopeuttavat ja helpottavat ohjelmointityön valmistumista verrattuna siihen, että kehittäjä kirjoittaisi koodit alusta asti tyhjästä. Koska CodeIgniter on lisensoitu Apache/BSD-tyyppisellä avoimen lähdekoodin lisenssillä, on se täysin ilmainen. (Codeigniter, viitattu 4.5.2015.)

Aiemmin Codeigniterin kehittämisestä vastannut organisaatio EllisLab Inc. on vuoden 2014 viimeisellä neljänneksellä luopunut sovelluskehyksen kehittämisestä ja nykyään kehittämisestä vastaa Canadan Vancouverissa sijaitseva British Columbia Institute of Tecnology (Ellislab, Viitattu 21.4.2015).

3.2 MVC-arkkitehtuuri

MVC-arkkitehtuurissa sovellus jaotellaan kolmeen eri komponenttiin, joita ovat Model (malli), View (näkymä) ja Controller (käsittelijä) (kuvio 2) (w3school, Viitattu 20.4.2015).

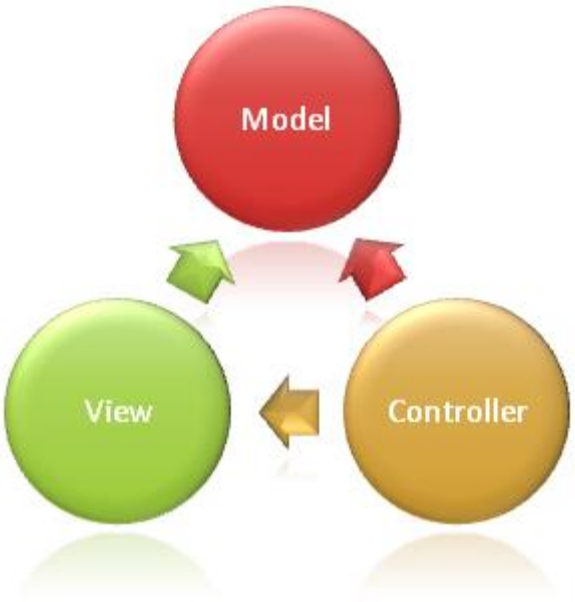
Malli sisältää tietoja ja tietoihin sovellettavia sääntöjä, joita sovellus hallitsee. Sovelluksissa kaikki on mallinnettu tietona, jota käsitellään tietyllä tavalla. Käsiteltävä tieto voi olla esimerkiksi käyttäjätietoja, kuten nimi, sähköposti ja ikä. Tietojen käsittelyä varten mallissa asetetaan tiettyjä sääntöjä, joita voivat olla esimerkiksi, että liittymispäivämäärä ei voi olla tulevaisuudessa, sähköpostiosoite täytyy olla oikeaa muotoa ja nimi ei voi sisältää kuin tietyn määrän merkkejä (kuvio 1). Mallin tehtävänä on toimia tietokantaoperaatioiden käsittelijänä ja välittää käsittelijälle käyttäjän tarvitsema tieto. (tuts+, Viitattu 4.5.2015.)



KUVIO 1. Tietojen kuvaus. (tuts+ 2016, viitattu 20.4.2016)

Näkymä käsittää sen osan sovelluksesta, joka näkyy käyttäjälle. Näkymä rakentuu yleensä mallissa olevista tiedoista, mutta sen luominen on mahdollista myös ilman mallia. Tässä työssä näkymään sijoitetaan kaikki ne sivut, jotka ladataan käyttäjälle. (tuts+, Viitattu 4.5.2015.)

Käsittelijän tehtävänä MVC-arkkitehtuurissa on vastaanottaa käyttäjän käyttöliittymässä aikaansaamat pyynnöt, jotka välittyvät HTTP GET- tai -POST -pyynnöillä ja tämän jälkeen käsittelijä järjestää pyyntöjen toteuttamiseen tarvittavat toimenpiteet. Yleensä tämä tarkoittaa sitä, että käsittelijä kutsuu tilanteeseen soveltuvaa mallia ja tämän jälkeen valitsee sopivan näkymän. (tuts+, Viitattu 4.5.2015.)



KUVIO 2. MVC-arkkitehtuurin komponentit. (W3schools.com 2015, Viitattu 20.4.2015)

3.3 PHP

PHP on palvelimella suoritettava ohjelmointikieli, joka on upotettu HTML-dokumenttien sisään. PHP:n rekursiivinen kirjainlyhenne tulee englanninkielisistä sanoista Hypertext Preprocessor ja se on laajasti käytössä oleva avoimen lähdekoodin ohjelmointikieli, joka on tarkoitettu käytettäväksi erityisesti web-sovelluskehityksessä. (w3schools, Viitattu 20.4.2015.)

Ammattilaisen käsissä PHP-ympäristö mahdollistaa nopean web-sovellusten kehittämisen, mutta on samaan aikaan myös helppo ohjelmointikieli omaksua vasta-alkajalle. PHP:n etuna on muun muassa se, että selaimelta saatava data, kuten lomakkeiden syötteet tai evästeet ovat helposti suoraan käytettävissä. Koska PHP on niin sanotusti löyhästi tyypitetty ohjelmointikieli, muuttujien tyyppiä ei tarvitse määrittää etukäteen, vaan niiden tyypit määräytyvät sen mukaan minkälaista dataa ja millaisin operaatioin muuttujaan sijoitetaan. (Rantala 2005, 9.)

Sovelluksen lähes koko toiminnallisuus on toteutettu PHP-ohjelmointikielellä. Sillä on toteutettu esimerkiksi sovelluksen kirjautumistoiminto, lomakkeiden syötteiden tarkastus ja uusien käyttäjien lisääminen tietokantaan.

3.4 MySQL

MySQL on tietokantajärjestelmä, joka on suosituin PHP:n kanssa käytössä oleva järjestelmä. MySQL-tietokantajärjestelmä on palvelimelta suoritettava tietokanta. Se on nopea, luotettava ja helppokäyttöinen ja se hyödyntää standardia SQL-kieltä. MySQL tukee lukuisia eri alustoja ja on lisäksi ilmainen. MySQL:n kehittämisestä, jakelusta ja tuesta vastaa Oracle Corporation. (w3schools, Viitattu 20.4.2015.)

MySQL:ssä data tallennetaan tauluihin. Taulu on kokoelma tietoja ja se koostuu sarakkeista ja riveistä. Tietokannat ovat käytännöllisiä, kun tietoa täytyy tallentaa kategorisesti. (w3schools, Viitattu 20.4.2015.)

3.5 JavaScript

JavaSript on Netscapen kehittämä oliopohjainen niin sanottu client-side scrpiting ohjelmointikieli jota käytetään selainskriptien tekemiseen. JavaScript mahdollistaa esimerkiksi lomakkeiden esitarkastamisen sekä hiireen reagoivat linkkipainikkeet. (Korpela & Linjama 2005 294-295.) Tässä opinnäytetyössä JavaScriptiä käytetään kuvasliderin toteutukseen.

JavaScript koodi voidaan liittää HTML-dokumenttiin useilla tavoilla. Koodi voidaan kirjoittaa suoraan HTML-dokumenttiin jolloin sen alku merkitään <script type=”text/javascript”>-tagilla ja se lopetaan </sript>-tagilla. JavaSript-koodi voidaan kirjoittaa myös erilliseen tiedostoon ja ottaa mukaan HTML-dokumenttiin linkittämällä se komennolla <script src=”esimerkkitkoodi.js” type=”text/javascript”>.(Ohjelmointiputka 2007, viitattu 22.8.2015.)

3.6 HTML

HTML (Hypertext markup language) on merkkauskieli, joka on tarkoitettu web-dokumenttien kuvaamiseen. HTML on tekstiä ja rakennetta. Rakenteisuudella tarkoitetaan sitä, että HTML-dokumentti sisältää tavallisen tekstisisällön lisäksi merkkauksen (markup), joka osoittaa dokumentin loogisen rakenteen. (Korpela & Linjama 2005 70-72.)

Elementti on yksi HTML:n peruskäsitteistä ja se koostuu alkutagista, sisällöstä ja lopputagista, kuten esimerkiksi sivun otsikko kirjoitetaan muotoon <h2>Otsikko</h2>. Alkutagi ilmaistaan siis <h2> ja lopputagi </h2>. Otsikon sisältö ilmaistaan aina tagien välissä. (Korpela & Linjama 2005 70-72.)

3.7 CSS

CSS on lyhenne sanoista Cascading Style Sheets ja sitä käytetään HTML-sivujen näkyvän ulkoasun määrittämiseen. CSS-tyylisäännöstö koostuu säännöistä (rule). Sääntöjen avulla voidaan asettaa tietylle elementille tiettyjä ominaisuuksia (properties) ja näille ominaisuuksille arvoja (value). Sääntö koostuu selektorista ja deklaraatiosta. Selektori ilmaisee mille elementille tyyli ollaan määrittämässä ja deklaraatio elementin ominaisuuden ja arvon. (Korpela & Linjama 2005 300-302.) Kuviossa 3 on esitetty opinnäytetyön sivun taustan värin määritys (kuvio 3).

Kuten JavaScript, niin myös CSS-tyylisäännöstö voidaan liittää HTML-dokumenttiin usealla tavalla. Suositeltavin tapa on linkittää tyylitiedosto HTML-dokumenttiin erillisenä tiedostona. HTML-sivu viittaa tyylitiedostoon seuraavaan tapaan: <link rel=”stylesheet” href=”style.css”>. Tyylisäännöt voidaan määrittää myös HTML-dokumentin Head-osaan jolloin deklaraatio sijoitetaan tagien <style type=text/css> ja </style> väliin. Tyylisääntö on mahdollista myös määrittää suoraan elementtiin sen alkutagissa style-määritteenä. Tällöin tyylisääntö käsittää vain kyseisen elementin esiintymän ja tällöin siitä jätetään pois selektori ja aaltosulut. (Korpela & Linjama 2005 305.) Esimerkiksi kuvion 3 taustavärin määritys tulisi silloin HTML-dokumenttiin muodossa <body style="background-color:#6E3F7D;">.

```
body {
    background-color:#6E3F7D;
}
```

SÄÄNTÖ						
		Deklaraatio (Declaration)				
Selektori		Ominaisuus(Property)		Arvo(value)		
body	**{**	**background-color**	**:**	**#6E3F7D**	**;**	**}**

KUVIO 3. CSS-sääntö

4 SUUNNITTELU

4.1 Tietokannan suunnittelu

Tietokannan hyvä suunnittelu ja rakentaminen on erityisen tärkeää, koska nykypäivänä tietokanta muodostaa modernien sovellusten perustan. Mikäli tietokannan suunnittelu on hoidettu huonosti, niin sovelluksesta ei tule kovinkaan onnistunutta, koska hankalia tietorakenteita joudutaan paikkaamaan tällöin sovellusohjelmilla. Mitä monimutkaisempia ja laajempia kokonaisuudet ovat, sitä enemmän korostuu tietokannan hyvä suunnittelu. Hovi, Huotari ja Lahdenmäki toteavatkin kirjassaan Tietokantojen suunnittelu & indeksointi, että ”koirankopin voi rakentaa ilman erityisiä piirustuksia, mutta mitä isompi ja monimutkaisempi rakennus on, sitä tärkeämpää on tehdä kunnon piirustukset ja tarkistaa esim. lämmitysjärjestelmän valinnan vaikutus arkkitehtisuunnitelmaan.” (Hovi, Huotari & Lahdenmäki 2005, 20.)

Ennen tietokannan suunnittelun aloittamista on pohdittava, että minkälainen olisi hyvä tietokannan rakenne. Rakenteen pohdinnassa tulee pitää mielessä, että sen keskeisiä ominaisuuksia ovat kattavuus, selkeys ja ymmärrettävyys, muutosjoustavuus, yleiskäytännöllisyys, eheys, ohjelmointimukavuus sekä suorituskyky. Ennen myös levytilan säästöä pidettiin yhtenä tavoitteena tietokannan suunnittelussa, mutta levytilan hintojen alentumisen vuoksi sen ei katsota olevan enää keskeisessä asemassa. (Hovi, Huotari & Lahdenmäki 2005, 20.)

Opinnäytetyössä toteutetaan niin sanottu räätälöity järjestelmä eli se tulee palvelemaan hyvin rajallista tarvetta. Suunniteltaessa räätälöityjä järjestelmiä on tärkeää, että tietokannan rakenne on selkeä sekä sovitettu tarkasti käyttötarpeisiin. Taulujen sekä tietojen nimeämisessä tulee käyttää itselleen tuttuja termejä ja lisäksi huolehdittava siitä, että sarakkeet tarkoittavat vain yhtä asiaa. (Hovi, Huotari & Lahdenmäki 2005, 20.)

Muutosjoustavuuden tulee olla hyvä, eli mahdollisten laajentamisten suorittaminen tulisi pystyä hoitamaan niin, että itse ohjelmaan tehtävät muutokset minimoidaan mahdollisimman tehokkaasti. Optimitilanteessa tietokannan ylläpitäjä pystyy suorittamaan muutos- ja ylläpitotyöt, kuten esimerkiksi lisäämään tauluja ilman, että olemassa oleviin ohjelmiin täytyisi tehdä muutoksia. Suunnitte-

lussa tulee pyrkiä välttämään taulujen pilkkomista ja yhdistämistä sekä aputauluja tai muita erikoisvirityksiä, jotka mahdollisesti voivat parantaa suorituskykyä paikallisesti joissakin kohdin, mutta samalla monimutkaistavat tietokannan rakennetta ja ohjelmointia. (Hovi, Huotari & Lahdenmäki 2005, 20.)

Muita tavoitteita tietokannalle ovat yhteensopivuus olemassa olevien tietojärjestelmien kanssa sekä skaalautuvuus siirryttäessä laiteympäristöstä tai hallintajärjestelmästä toiseen (Hovi, Huotari & Lahdenmäki 2005, 23). Tässä opinnäytetyössä ei kuitenkaan tulla ottamaan huomioon yhteensopivuutta tai skaalautuvuutta.

Opinnäytetyötä varten toteutettava tietokanta ei ole erityisen laaja tai monimutkainen, koska siinä käytetään vain yhtä taulua johon tallennetaan kirjautumista varten tarvittavat tiedot, joten tämän työn kannalta tietokannan suunnitteluun ei käytetä paljoa aikaa.

4.2 Ulkoasun suunnittelu

Ulkoasun suunnittelu lähtee liikkeelle sivuston käyttäjäkunnasta. Käyttäjäkunta tulee ottaa huomioon, koska se vaikuttaa muun muassa tekstityyppiin, kuvitukseen, tekstin määrään ja väreihin. Tämän lisäksi tulee myös tarkastella sivustolle tulevia kuvia sekä tekstejä ja pohtia yhtenäinen tyyli näille. (Korpela & Linjama 2005, 356.)

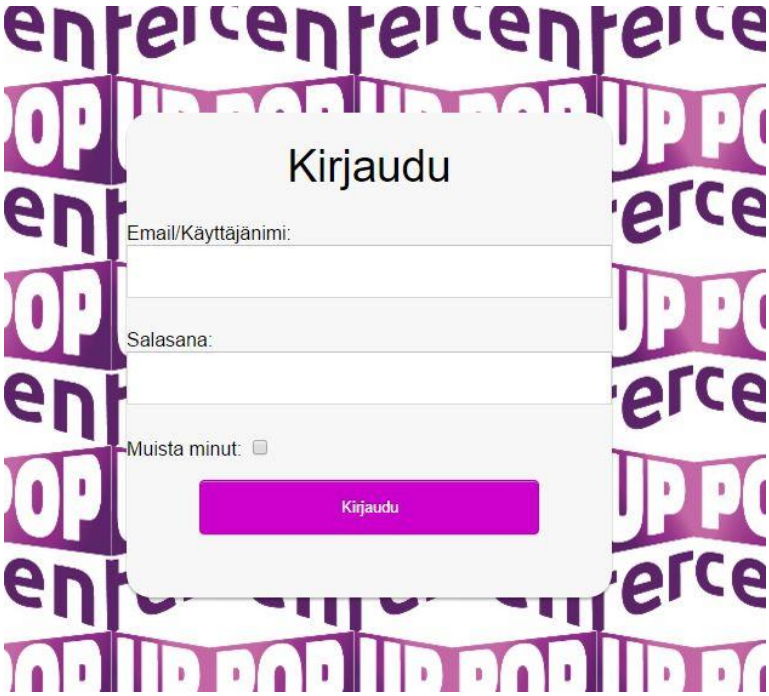
Verkkosivuston ulkoasusta haluttiin mahdollisimman yksinkertainen. Komponentteja ovat kirjautumissivu sekä median lisääminen/poistaminen ja esittäminen. Lisäksi ulkoasusta toivottiin mahdollisimman pelkistetty ja helppokäyttöinen.

5 TOTEUTUS

Käytännön osuuden toteutuksessa käytettiin apuna NetBeans IDE 8.1 sovelluskehitysalustaa sekä Codeigniter 3.0 -ohjelmistokehystä.

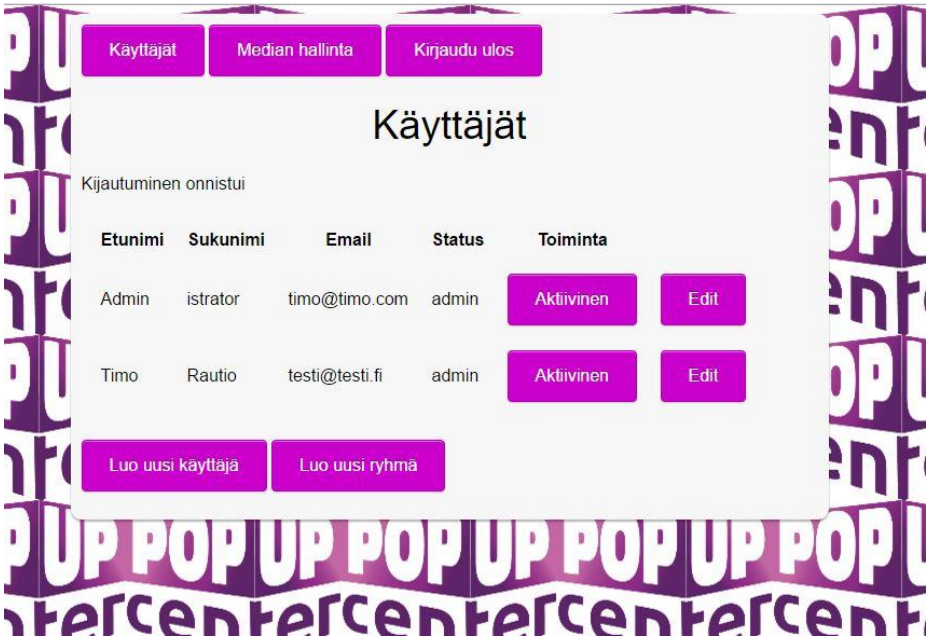
5.1 Käyttöliittymän toteutus

Käyttöliittymän toteutuksessa lähdettiin liikkeelle kirjautumissivusta, jonka käyttäjä näkee sivustosta ensimmäisenä (kuvio 4). Kirjautumissivun toteutus on hyvin yksinkertainen ja suoraviivainen toimenpide. Työskentelyn nopeuttamiseksi käytin apuna käyttäjän autentikoinnin toteutuksessa Codeigniter-ohjelmistokehykselle luotua Ion_Auth-autentikointikirjastoa. Pyrin noudattamaan sivuston ulkoasussa yrityksen värimaailmaa.

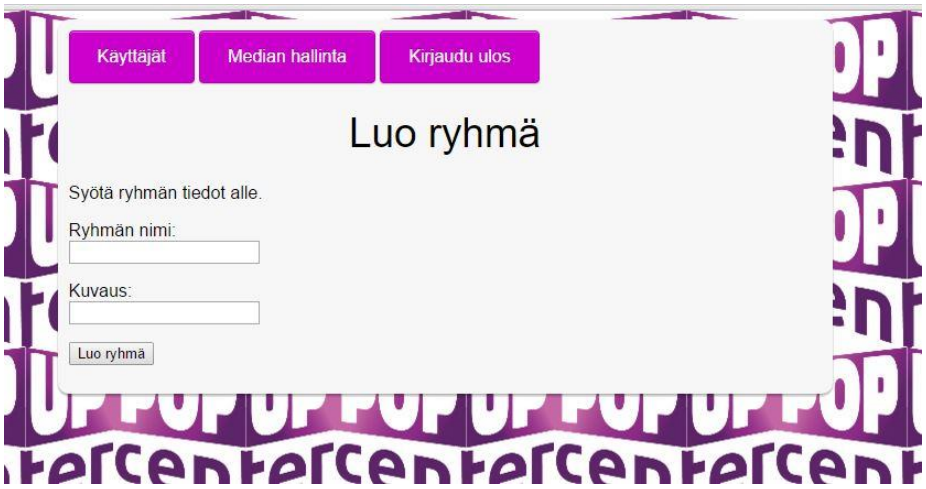


KUVIO 4. Kirjautumissivu

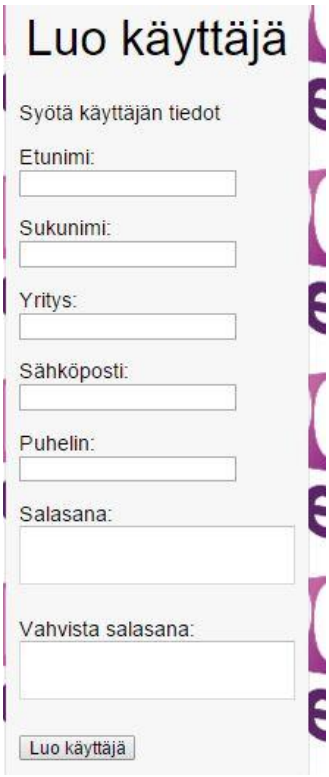
Seuraava käyttöliittymän toteutusvaihe oli käyttäjien hallinta –sivu (kuvio 5). Tältä sivulta käyttäjän on mahdollista lisätä uusia käyttäjiä sekä käyttäjäryhmiä (kuviot 6-7). Lisäksi sivulta löytyy linkit uloskirjautumiseen sekä median hallintaan ohjaava linkki.



KUVIO 5. Käyttäjien hallinta



KUVIO 6. Ryhmän luonti.



KUVIO 7. Käyttäjän lisääminen.

Kuvien lisäämistä varten luotiin sivu, josta käyttäjä pystyy lisäämään ja poistamaan kuvasisältöä sivustolta (kuvio 8). Kuvagallerian toteutuksessa käytin apuna image CRUD -kirjastoa työskentelyn nopeuttamiseksi. Sivulta löytyy linkit kuvakarusellin käynnistykseen sekä paluulinkki käyttäjät sivulle.

Kuvio 8. Kuvan lisääminen.

6 POHDINTA

Mielestäni sivuston toteutus onnistui asetettuun tavoitteeseen nähden kohtuullisen hyvin. Sovelluksen julkaiseminen ei ole tosin toistaiseksi ajankohtaista, mutta koen sen hyvänä asiana, koska pystyn lisäämään siihen omalla ajallani niitä komponentteja jotka rajautuivat tämän opinnäytetyön ulkopuolelle.

Opinnäytetyöni tekeminen alkoi melkoisen vauhdikkaasti, mutta puolivälin jälkeen työtahti koki pienen notkahduksen työkiireiden vuoksi. Suurimmat ongelmat kohtasin sovelluksen ohjelmoinnin parissa, mutta löysin ongelmiin ratkaisut pääasiassa internetistä löytämilläni ohjeilla sekä kokeilemalla.

Opinnäytetyön aikana opin käyttämään Codeigniter-sovelluskehystä paremmin sekä samalla palautin mieleeni aiemmin opittuja asioita etenkin tietokannoista. Jatkoa ajatellen voisi olla mielenkiintoista toteuttaa sama projekti jotain ajantasaisempaa menetelmää käyttämällä.

LÄHTEET

CodeIgniter. 2015. CodeIgniter at a Glance. Viitattu 4.5.2015, http://www.codeigniter.com/userguide3/overview/at_a_glance.html

Ellislab. 2015. A Brief History of CodeIgniter. Viitattu 20.4.2015, https://ellislab.com/codeigniter

Ellislab. 2014. Your Favorite PHP Framework, CodeIgniter, Has a New Home. Viitattu 21.4.2015, https://ellislab.com/blog/entry/your-favorite-php-framework-codeigniter-has-a-new-home

Hovi A, Huotari J, Lahdenmäki T. Tietokantojen suunnittelu & indexointi. . Jyväskylä: Docendo Finland Oy

Korpela K, Linjama T. Web-suunnittelu. Jyväskylä: Docendo Finland Oy

Ohjelmointiputka. JavaScript-perusopas: Osa 1 – Perusteet. Viitattu 22.8.2015, http://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=js_01

Rantala A. Web-ohjelmointi. Jyväskylä: Docendo Finland Oy

Tuts+. 2015. MVC for Noobs. Viitattu 4.5.2015, http://code.tutsplus.com/tutorials/mvc-for-noobs--net-10488

W3scools.com. 2015. ASP.NET MVC Tutorial. Viitattu 20.4.2015, http://www.w3schools.com/aspnet/mvc_intro.asp

W3schools.com. 2015. PHP 5 Introduction. Viitattu 20.4.2015, http://www.w3schools.com/php/php_intro.asp

W3schools.com. 2015. PHP MySQL Database. Viitattu 20.4.2015 http://www.w3schools.com/php/php_mysql_intro.asp