OULUN SEUDUN
AMMATTIKORKEAKOULU

Tuukka Kiiskinen & Roni Kokkonen

# Käyttäjähallinnan ja kirjautumisen toteutus web-sovellukseen

# Käyttäjähallinnan ja kirjautumisen toteutus web-sovellukseen

Tuukka Kiiskinen & Roni Kokkonen
Opinnäytetyö
Syksy 2013
Tietojenkäsittely
Oulun seudun ammattikorkeakoulu

# TIIVISTELMÄ

Oulun seudun ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma

---

Tekijät: Tuukka Kiiskinen & Roni Kokkonen
Opinnäytetyön nimi: Käyttäjähallinnan ja kirjautumisen toteutus web-sovellukseen
Työn ohjaaja: Pekka Ojala
Työn valmistumislukukausi ja -vuosi: Syksy 2013
Sivumäärä: 30

---

Opinnäytetyön toimeksiantajana toimi Oulun seudun ammattikorkeakoulun liiketalouden yksikkö. Opinnäytetyössä kehitettiin OpixManageria, joka on projektinhallintasovellus. Opix on koulun projekti, jonka kehitys on aloitettu vuoden 2011 maaliskuussa. OpixManageria on kehitetty tietojenkäsittelyn koulutusohjelman opiskelijoilla ja olemme osallistuneet kehitystyöhön työharjoittelussa.

Työn tavoitteena oli ohjelmoida OpixManageriin kirjautuminen ja käyttäjähallinta. OpixManagerissa oli paljon toimintoja ja ominaisuuksia ennen tämän opinnäytetyön aloittamista, joten kirjautumisen ja käyttäjähallinnan ohjelmointi oli seuraava looginen asia. Tämä aihe valittiin opinnäytetyöhön, koska olimme kehittäneet OpixManageria työharjoittelussa, joten kehitystyötä oli helppo jatkaa.

Opinnäytetyössä käytettiin NetBeans- ja Xampp-ohjelmistoja sekä CodeIgniter-ohjelmistokehystä. Kirjautuminen tehtiin yksinkertaisilla funktioilla, joissa verrataan käyttäjätunnusta ja salasanaa tietokantaan. Sovelluksessa oli valmiina kaksi käyttäjätilityyppiä ja niitä käytettiin apuna käyttäjähallinnan toteutuksessa. Lisäksi sovelluksen projektien henkilöillä on kolme erilaista roolia, joita käytettiin apuna projektisivujen käyttäjähallinnassa. Käyttäjähallinta toteutettiin ohjelmoimalla sovelluksen funktioihin if-lauseita, joissa tarkistettiin käyttäjätilityyppi tai käyttäjän rooli projektissa.

OpixMangeriin saatiin kehitettyä toimiva kirjautuminen sekä käyttäjähallintaa sen verran kuin opinnäytetyöhön oli rajattu. OpixManager aiotaan ottaa käyttöön tulevaisuudessa projektityökurssilla tietojenkäsittelyn koulutusohjelmassa. OpixManagerin kehitystyö jatkuu opinnäytetyön jälkeen ja kerromme omia kehitysideoitamme tämän opinnäytetyön pohdintaosuudessa.

---

Avainsanat: Kirjautuminen, Käyttäjähallinta, Autentikointi, Auktorisointi

# ABSTRACT

Oulu University of Applied Sciences
Degree Programme in Information Technology

Authors: Tuukka Kiiskinen & Roni Kokkonen
Title of thesis: Login and user management implementation for web application
Supervisor: Pekka Ojala
Term and year when the thesis was submitted: Autumn 2013
Number of pages: 30

This thesis was made for Oulu University of Applied Sciences. We developed OpixManager which is a project management web application. Opix is a school project and it was started in March 2011. OpixManager has been developed by students of Degree Programme in Information Technology and we have participated in the development in practical training.

Thesis goal was to implement login and user management for OpixManager. There was already a lot of features and functions in OpixManager before this thesis was started so implementing login and user management was the next logical thing to do. This subject was chosen, because we had developed OpixManager in practical training and it was easy to continue to develop OpixManager.

The programs that were used in this thesis were called NetBeans and Xampp and also CodeIgniter framework. Login was made with simple functions where username and password are compared to the database of the application. The application had two types of user accounts which were used in user management. In addition, the application had three types of roles for project members which were used in the user management of the project pages. User management was developed by programming if clauses in the application functions. The if clauses are used to check the account type or the user's role in a project.

We were able to develop a working login and user management. OpixManager will be used in the future in project work course in Degree Programme in Information Technology. The development of OpixManager will continue after this thesis and we will tell our ideas on how to develop this application at the end of this thesis.

Keywords: Login, User Management, Authorization, Authentication

**SISÄLLYS**

# 1 JOHDANTO

Opinnäytetyön tarkoituksena on kehittää OpixManageria, joka on projektinhallintatyökalu, jota Oulun seudun ammattikorkeakoulun liiketalouden yksikön oppilaat ovat kehittäneet vuoden 2011 maaliskuusta alkaen. OpixManager on websovellus, joka käyttää MySQL-tietokantaa. Ohjelmointikielenä käytetään PHP-ohjelmointikieltä. Päätimme kehittää OpixManageria NetBeans-ohjelmalla ja kehitystyöhön on valittu CodeIgniter-ohjelmistokehys, joka sisältää paljon valmiita PHP-funktioita. CodeIgniter lyhentää koodin määrää huomattavasti ja nopeuttaa ohjelmointia.

Kehitimme OpixManageria työharjoittelussa 5 kuukautta, joten meillä oli hyvät lähtökohdat opinnäytetyötä aloitettaessa. Ehdimme tehdä siihen harjoittelun aikana paljon uutta sisältöä. Kehitimme mm. projektinhallintasivuja, raporttisivuja sekä sivuston ulkoasua ja tietokantaa.

Kehitystehtävän tärkein asia on sisäänkirjautumisen ohjelmointi sovellukseen ja käyttäjähallinta. Koska OpixManagerissa on jo paljon erilaisia toimintoja, on kirjautumisen ohjelmointi työmäärältään merkittävä ja tärkeä asia. Käyttäjähallinnan avulla rajoitetaan käyttäjien oikeuksia tehdä muutoksia sovelluksessa oleviin tietoihin sekä niiden näkyvyyteen.

Aihe on ajankohtainen ja tärkeä, koska OpixManager aiotaan ottaa käyttöön oppilaitoksessa projektityökurssilla tietojenkäsittelyn koulutusohjelmassa. Tietojenkäsittelyn koulutusohjelman opiskelijat ovat testanneet OpixMangeria. He tarkastelivat sovelluksen toimintoja ja etsivät niistä virheitä.

Toisessa luvussa kerrotaan yleisesti käyttäjähallinnasta ja kirjautumisesta. Luvussa tarkastellaan myös erilaisia salasanan suojausmenetelmiä ja niiden tärkeyttä.

Kolmannessa luvussa kerrotaan OpixManagerista sekä työhön liittyvistä teknologioista ja ohjelmistoista. Luvussa tarkastellaan tarkemmin MVC-arkkitehtuuria.

Neljännessä luvussa käydään ensin läpi kehittämistehtävä. Tämän jälkeen kerrotaan, miten ne on tässä työssä toteutettu. Kehittämistehtävän toteutusta havainnollistetaan useilla kuvioilla.

# 2 KÄYTTÄJÄHALLINTA JA KIRJAUTUMINEN

Käyttäjähallinta on autentikointitoiminto, joka tarjoaa mahdollisuuden tunnistaa ja hallita kirjautuneiden käyttäjien tilaa verkossa. Sen avulla voidaan myös tarkastella ja valikoida käyttäjiä, jotka ovat kirjautuneena, kirjata käyttäjiä ulos ja hallita käyttäjien kirjautumisia ja kirjautumisajan pituuksia. (Bluecoat 2007, hakupäivä 30.10.2013.)

Useimmissa yrityksissä, joissa tietoturva on tärkeä asia, käytetään jonkinlaista autentikointia ja auktorisointia, jotta käyttäjät pääsevät käsiksi verkossa olevien tietojen ja sovellusten sisältöihin. Käyttäjien oikeudet voidaan varmistaa, ennen kuin heille annetaan pääsy sisältöön ja käyttäjien aktiivisuutta voidaan valvoa useiden lokitoimintojen avulla. Tavallisessa autentikointi- ja auktorisointitoteutuksessa ylläpitäjillä on useita vaihtoehtoja käytettävissä, miten käyttäjät autentikoidaan. (Bluecoat 2007, hakupäivä 30.10.2013.)

Autentikointi yhdistetään usein auktorisointiin. Autentikoinnilla tarkoitetaan käyttäjän tunnistamista ja auktorisoinnilla tarkoitetaan oikeuksien tarkastamista. Auktorisointia ei yleensä liitetä yksittäisiin käyttäjiin, vaan käyttäjät esimerkiksi ryhmitellään eri auktorisointitasojen mukaan. Esimerkiksi kun käyttäjä on tunnistettu, hänelle annetaan pääsy sovellukseen. Sen jälkeen riippuen käyttäjäryhmästä, käyttäjällä on oikeus suorittaa tiettyjä toimintoja, joita käyttäjäryhmälle on annettu. (Kent & Millett 2003, 36-41.)

Käyttäjähallinta antaa ylläpitäjille mahdollisuuden valvoa käyttäjien uloskirjaamista, esimerkiksi jos käyttäjä on passiivinen, tai käyttäjille annetaan pääsy uloskirjautumissivulle tai ylläpitäjä kirjaa manuaalisesti käyttäjän ulos. Yleensä uloskirjautuminen on käytettävissä esimerkiksi hallintapaneelissa. (Bluecoat 2007, hakupäivä 30.10.2013.)

Käyttäjähallinta perustuu siihen, että käyttäjät kirjautuvat sisään ja ulos. Käyttäjän tunnistaminen sisäänkirjautuneeksi antaa ylläpitäjille mahdollisuuden luoda

hallintokäytäntöjä, joilla hallitaan käyttäjiä ja oikeuksia. (Bluecoat 2007, hakupäivä 30.10.2013.)

Kirjautuminen on toiminto, jolla annetaan käyttäjälle pääsy esimerkiksi johonkin sovellukseen tai web-sivulle. Käyttäjä syöttää käyttäjätunnuksen ja salasanan esimerkiksi kirjautumislomakkeeseen. Käyttäjätunnusta ja salasanaa verrataan käyttäjän tiedoissa oleviin käyttäjätunnukseen ja salasanaan, ja jos ne vastaavat toisiaan niin käyttäjä kirjataan sisään.

Käyttäjätunnus on käyttäjän valitsema merkkijono, josta käyttäjä haluaa itsensä tunnistettavan. Käyttäjätunnuksena käytetään monesti myös sähköpostiosoitetta. Käyttäjä valitsee yleensä itselleen salasanan, jonka suositellaan olevan vähintään 8 merkkiä pitkä ja siinä olisi hyvä olla isoja ja pieniä kirjaimia, numeroita sekä erikoismerkkejä. (OWASP 2013, hakupäivä 21.11.2013).

Salasana suojataan yleensä hajauttamalla ja se on yksi tavallisista tietoturva-asioista, joka pitää ottaa huomioon kun suunnitellaan sovellusta, joka vastaanottaa salasanoja käyttäjiltä. Ilman hajautusta salasanat, jotka on tallennettu sovelluksen tietokantaan, on helppo varastaa. Hyvä tapa on lisätä hajautusalgoritmi käyttäjien salasanaan ennen kuin ne tallennetaan tietokantaan. Se tekee hyökkääjille vaikeaksi saada selville alkuperäistä salasanaa. (PHP 2013a, hakupäivä 13.11.2013.)

Yleisimmät hajautusmenetelmät ovat MD5, SHA1 ja SHA256. Näitä ei kuitenkaan suositella käytettävän, koska nykyään uudet tietokoneet kääntävät nämä hajautusmenetelmät nopeasti. Kaksi tärkeintä asiaa, jotka tulee ottaa huomioon kun hajautetaan salasana, ovat laskennallinen kulu ja suolaus. Mitä laskennallisempi hajautusalgoritmi on, sitä kauemmin kestää murtaa se. Suolaus tarkoittaa sitä, että hajautusprosessin aikana lisätään ylimääräistä dataa hajautukseen, joka tekee siitä paljon vaikeamman murrettavaksi. (PHP 2013a, hakupäivä 13.11.2013.)

# 3 OPIX-MANAGER

OpixManager on projektinhallintatyökalu, jota käytetään internet-selaimella eli se on web-sovellus. OpixManager on Oulun seudun ammattikorkeakoulun liiketalouden yksikön tietojenkäsittelyn koulutusohjelmassa aloitettu projekti, jossa toteutetaan opiskelijatöinä projektihallintaan soveltuva web-sovellus. Sen tekeminen on aloitettu vuonna 2011 ja siihen on osallistunut useita opiskelijoita. (Opix-Manager 2011, hakupäivä 6.5.2013.)

OpixManageria päätettiin työstää NetBeans-ohjelmalla, koska se on jo entuudestaan tuttu ja sitä on käytetty paljon koulussa. Ohjelmointityö tehdään PHP-ohjelmointikielellä ja HTML-sivunkuvauskielellä. Projektiin on valittu CodeIgniter-ohjelmistokehys, johon lähes kaikki kehitystyö perustuu. CodeIgniter sisältää valmiita PHP-funktioita, joita sovelluksessa käytetään. Tämä vähentää ohjelmoinnin määrää huomattavasti. Sovellukseen kuuluu myös MySQL-tietokanta, johon tallennetaan kaikki sovelluksessa käytettävät tiedot. (Opix-Manager 2011, hakupäivä 6.5.2013.)

Sovelluksessa on paljon ominaisuuksia, kuten projektien hallinta, henkilöstön lisääminen ja hallinta sekä asiakkaiden hallinta. Projektien hallinnassa käytetään tällä hetkellä kahta eri projektityyppiä, jotka ovat Scrum ja perinteinen. Sovellukseen on toteutettu myös kaikki käännökset englanniksi.

Sovellukseen kuuluu myös raporttisivut, joille tulostetaan tietoja useista projekteista ja ne voidaan tulostaa helposti myös paperille. Sovelluksessa on käytetty myös hieman Javascriptiä helpottamaan tiettyjä toimintoja. Esimerkiksi päivämäärän valintaruudussa aukeaa kalenteri, josta valitaan päivämäärä.

Syöttötietojen tarkistaminen on tärkeä osa sovellusta. Kaikki pakolliset tiedot, jotka sovellukseen syötetään, tarkistetaan erilaisilla funktioilla, jotta syötettävät tiedot ovat oikeassa muodossa.

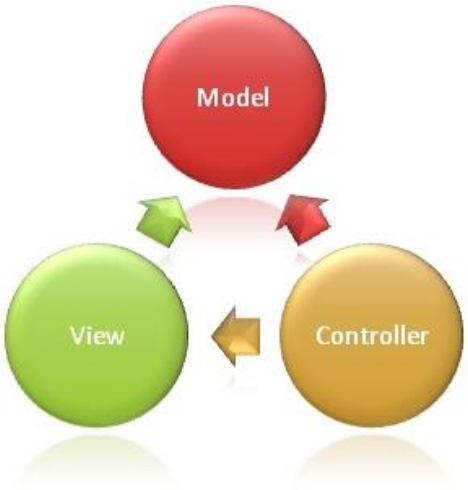
### 3.1 CodeIgniter

CodeIgniter on avoimen lähdekoodin ohjelmistokehys ja sitä käytetään dynaamisten web-sivujen tekemiseen käyttäen PHP-ohjelmointikieltä. Ensimmäinen julkinen versio on julkaistu helmikuussa 2006 ja viimeisin heinäkuussa 2013. CodeIgniterin lähdekoodia ylläpitää Ellislab ja sillä on avoimen lähdekoodin ohjelmistolisenssi. CodeIgniter sisältää paljon valmiita kirjastoja, joita tarvitaan usein käytetyissä tehtävissä, mikä nopeuttaa projektien tekemistä huomattavasti. (Ellislab 2013, hakupäivä 8.4.2013.)

CodeIgniter perustuu löyhästi suosittuun Model-View-Controller-malliin. Malli käsittelee tietokannan funktioita, näkymä näyttää datan ja ohjain käsittelee syötettävät tiedot ja muuntaa ne komennoiksi mallille tai näkymälle. (Ellislab 2013, hakupäivä 8.4.2013.)

### 3.2 MVC

MVC (Model-View-Controller) on yleinen suunnittelumalli, joka on tuttu monelle ohjelmistokehittäjälle. MVC-arkkitehtuurissa erotetaan sovellus kolmeen pääkomponenttiin: malliin, näkymään ja ohjaimeen. (Microsoft 2013, hakupäivä 17.10.2013.)




*KUVIO 1. MVC-komponentit*.

Mallin objektit ovat sovelluksen osia, jotka toteuttavat logiikan sovelluksen tietovarastoon. Usein mallin objektit hakevat ja tallentavat mallin tilan tietokantaan. Esimerkiksi tuoteobjekti voi hakea tietoa tietokannasta, käyttää sitä ja sitten kirjoittaa päivitetyn tiedon takasin tuotetauluun SQL-palvelimella. (Microsoft 2013, hakupäivä 17.10.2013.)

Näkymät ovat niitä komponentteja, jotka näyttävät sovelluksen käyttöliittymän. Yleensä käyttöliittymä on muodostettu mallin datasta. Esimerkiksi tuotetaulun edit-näkymässä voi olla tekstikenttiä, pudotuslistoja ja valintaruutuja, jotka riippuvat tuoteobjektin tilasta. (Microsoft 2013, hakupäivä 17.10.2013.)

Ohjainkomponentit käsittelevät käyttäjän vuorovaikutuksen, työskentelevät mallin kanssa ja valitsevat näkymän, joka näytetään käyttöliittymässä. MVC-sovelluksessa näkymä näyttää vain tiedot näytöllä ja ohjain käsittelee ja vastaa käyttäjän syöttämiin tietoihin ja vuorovaikutukseen. Esimerkiksi ohjain käsittelee hakuehtojen arvot ja välittää ne malliin. Tämän jälkeen malli suorittaa haun tietokantaan käyttäen välitettyjä arvoja. (Microsoft 2013, hakupäivä 17.10.2013.)

MVC-arkkitehtuuri helpottaa sovelluksen tekemistä, koska se pitää sovelluksen koodit erillään. Tämän ansiosta koodin lisääminen ja päivittäminen on yksinkertaisempaa ja helpompaa.

**NetBeans**

NetBeans on sovelluskehitysympäristö, jolla voi tehdä useita erilaisia ohjelmia kuten työpöytäsovelluksia, mobiilisovelluksia sekä web-sovelluksia. Se tukee useita ohjelmointikieliä, kuten Java, C/C+ ja PHP. Se sisältää paljon valmiita luonnoksia, joista voi aloittaa sovelluksen ohjelmoinnin, sekä useita toimintoja, jotka tukevat ohjelmointia, kuten virheiden tarkistus ja etsintä, automaattiset sisennykset ja ohjelmoinnin reaaliaikainen tarkistus. (NetBeans IDE 2013, hakupäivä 8.4.2013.)

NetBeansin avulla voi helposti työstää useita projekteja kerrallaan, sillä ne näkyvät oletuksena ohjelman vasemmassa laidassa. Projektien hallinta on help-

poa, sillä näkymässä näkyy kaikki projektiin sisältyvät kansiot ja tiedostot oikeassa hierarkiajärjestyksessä. (NetBeans IDE 2013, hakupäivä 8.4.2013.)

NetBeansin avulla voi helposti tehdä työpöytäsovelluksia, sillä siinä on editori, johon voi piirtää painikkeita sekä tekstikenttiä ja ohjelma tuottaa sovellukseen ohjelmoinnin automaattisesti. (NetBeans IDE 2013, hakupäivä 8.4.2013.)

Virheiden tarkistus on myös iso osa NetBeansia ja siinä on virheenjäljitin, jolla suoritetaan koodia ja etsitään virhekohtia. Ohjelmointiin voi merkitä tiettyjä kohtia, joiden kohdalla suoritus pysäytetään ja tarkastellaan ohjelmointia. Lisäksi voidaan tarkastella ohjelman perusfunktioita, joita sovelluksessa käytetään. (NetBeans IDE 2013, hakupäivä 8.4.2013.)

Netbeans tukee useita ohjelmointikieliä ja siihen voi asentaa lisäksi useita muita kieliä. Ohjelmaa voi käyttää myös kaikilla käyttöjärjestelmillä, jotka tukevat Javaa, kuten Windows, Linux ja Mac. NetBeansillä on iso käyttäjäryhmä, joka suunnittelee ohjelmaan paljon lisäosia, jotka voi lähettää ohjelmiston kotisivuille ja ne ovat kaikkien käyttäjien käytössä. (NetBeans IDE 2013, hakupäivä 8.4.2013.)

### Xampp

Xampp on sovellus, jota käytetään paikallisten web-palvelimien ylläpitoon ja sen avulla on helppo testata omia web-sovelluksia ilman internet-yhteyttä. Xampp koostuu Apache http –palvelimesta, MySQL-tietokannasta ja PHP-tulkista. Ohjelmaan on kirjoitettu skriptejä PHP- ja Perl-ohjelmointikielillä. (Xampp 2013, hakupäivä 8.4.2013.)

### PHP

PHP on tulee englanninkielisistä sanoista Hypertext Preprocessor ja se on laajalti käytetty avoimen lähdekoodin ohjelmointikieli, joka on erityisesti suunniteltu web-sovelluksia varten, mutta sillä voidaan tehdä myös työpöytäsovelluksia. PHP toimii kaikilla yleisimmillä käyttöjärjestelmillä, sekä se tukee useimpia ny-

kyisin käytössä olevia web-palvelimia. Sitä voidaan käyttää yhdessä HTML-sivunkuvauskielellä toteutettujen websivujen kanssa. PHP sisältää funktioita, joihin on sisällytetty HTML-elementtejä ja HTML-tageja ja se helpottaa web-sivujen luontia. PHP-ohjelmointikieltä pidetään yleisesti helppona ohjelmointikielenä aloittelijoille, mutta se tarjoaa myös paljon edistyneitä toimintoja kokeneille ohjelmoijille. (PHP 2013b, hakupäivä 8.4.2013.)

Suurin osa sovelluksesta on toteutettu PHP-ohjelmointikielellä, jolla on tehty funktiot, joita sovellus sisältää. Lisäksi apuna on CodeIgniterin valmiit PHP-funktiot, joita hyödynnetään sovelluksen funktioissa.

**HTML**

HTML eli HyperText Markup Language on www-sivujen rakenteen kuvaava sivunkuvauskieli. Se on kehitetty CERN-tutkimuskeskuksessa 1990-luvun alkupuolella. Www-sivuilla käytetään yleisesti HTML-sivunkuvauskieltä, koska sillä on helppo määritellä miten sivujen sisältö näkyy. Sen avulla määritellään esimerkiksi otsikot ja kappaleet tekstissä sekä kuvat ja linkit. HTML-sivunkuvauskielen apuna käytetään CSS-tiedostoja joihin määritellään esimerkiksi tekstikappaleiden ja otsikoiden fontit ja sijainnit sekä kuvien koot ja sijainnit. (HTML 2004, hakupäivä 2.10.2013.)

HTML-sivunkuvauskieltä ja PHP-ohjelmointikieltä voidaan käyttää rinnaikkain. OpixManageriin on tehty HTML-lomakkeita, joihin täytetään tietoja ja tiedot tallennetaan tietokantaan PHP-funktioiden avulla. OpixManagerin kaikilla sivuilla käytetään HTML-sivunkuvauskieltä ja sivujen ulkoasu on määritelty CSS-tiedostoon, jossa on eri HTML-elementtien ominaisuudet.

**MySQL**

MySQL on tietokantajärjestelmä, johon on helppo tallentaa www-sivujen muuttuvaa tietosisältöä kuten käyttäjien tietoja. Ensin tehdään tietokanta, johon luodaan tauluja, jotka koostuvat kentistä ja riveistä, joihin tallennetaan tietoa.

Tietokantaan tallennetaan ja sieltä haetaan tietoa kyselyillä, jotka kirjoitetaan SQL-kielellä. (MySQL 2009, hakupäivä 2.10.2013.)

# 4 KEHITTÄMISTEHTÄVÄ

Kehittämistehtävänä on kehittää OpixManageriin kirjautumisominaisuus ja käyttäjähallinta. Molemmat ovat tällä hetkellä tärkeitä ominaisuuksia, koska sovelluksessa on jo paljon muita ominaisuuksia ja tietoa. Tämän takia tarvitaan kirjautumista ja käyttäjähallintaa, jotta voidaan rajoittaa, mitä käyttäjät voivat tehdä sovelluksessa ja mitä tietoja käyttäjät näkevät.

Kirjautuminen on opinnäytetyön isoin ja tärkein asia. Kirjautuminen selkeyttää paljon sovelluksen käyttöä, koska sen avulla käyttäjä näkee vain omat projektit missä on mukana ja pystyy muokkaamaan vain kyseisiä projekteja. Samalla sillä estetään luvaton pääsy sovellukseen, ettei kuka vain pysty aukaisemaan sovellusta ja muokkaamaan, poistamaan tai lisäämään uutta tietoa.

Käyttäjähallinta on myös tärkeä asia, koska sen avulla rajoitetaan mitä käyttäjät voivat sovelluksessa tehdä. Sovellukseen tulee peruskäyttäjiä, jotka näkevät vain omat projektit, joissa he ovat on mukana ja pystyvät muokkaamaan vain kyseisiä projekteja. Lisäksi on käyttäjiä, joilla on järjestelmänhallitsijan oikeudet. He näkevät kaikki projektit ja pystyvät lisäämään uusia projekteja ja muokkaamaan kaikkia meneillään olevia projekteja.

Luvussa 4.1 kerrotaan, miten kirjautuminen on toteutettu tässä opinnäytetyössä ja luvussa 4.2 kerrotaan, miten käyttähallinta on toteutettu.

### 4.1 Kirjautuminen

```
class Login extends CI_Controller
{
    function __construct()
    {
        parent::__construct();
        $this->lang->load('login');
    }

    function index()
    {
        $data = array(
            'id' => '',
            'login_user_id' => '',
            'password' => '',
            'account_type' => ''
        );

        $data['main_content'] = 'login_view';
        $data['pagetitle'] = $this->lang->line('title_login');
        $this->load->helper(array('form'));
        $this->load->view('template', $data);
    }

}
```

*KUVIO 2. Sisäänkirjautumisohjain.*

Ylläolevassa kuviossa on sisäänkirjautumisohjain, jonka index-funktio luo tyhjän datataulun, jossa on paikat käyttäjän id:lle, käyttäjänimelle, salasanalle ja käyttäjätilin tyypille. Sen jälkeen funktio avaa sisäänkirjautumisnäkymän.


*KUVIO 3. Sisäänkirjautumisnäkymä.*

Ylläolevassa kuviossa on sisäänkirjautumisnäkymä OpixManagerista. Käyttäjä syöttää käyttäjänimen ja salasanan niille tarkoitettuihin kenttiin. Sen jälkeen käyttäjä painaa Login-painiketta.

```
function index()
{
    $this->load->library('form_validation');

    $this->form_validation->set_rules('txt_user_id', $this->lang->line('label_username'),
            'trim|required|xss_clean');
    $this->form_validation->set_rules('pwd_password', $this->lang->line('label_password2'),
            'trim|required|xss_clean|callback_check_database|md5');

    if ($this->form_validation->run() == FALSE)
    {
        $data = array(
            'id' => '',
            'user_id' => '',
            'password' => ''
        );

        $data['main_content'] = 'login_view';
        $data['pagetitle'] = $this->lang->line('title_login');
        $this->load->helper(array('form'));
        $this->load->view('template', $data);
    }
    else
    {
        redirect('home', 'refresh');
    }
}
```

*KUVIO 4. Sisäänkirjautumisfunktio.*

Kuviossa 4 on sisäänkirjautumisfunktio, joka suoritetaan sen jälkeen, kun käyttäjä on painanut Login-painiketta sisäänkirjautumisnäkymässä. Käyttäjänimelle ja salasanalle on asetettu tiettyjä vaatimuksia, jotka tarkistetaan ja jos tarkistus menee läpi, niin käyttäjä kirjataan sisään sovellukseen. Funktiossa tarkastetaan, että käyttäjä on syöttänyt käyttäjänimen ja salasanan required-funktiolla ja kentissä ei saa olla skriptejä, jonka xss_clean-funktio tarkistaa. Sen lisäksi syöttötiedoista poistetaan turhat välilyönnit alusta ja lopusta trim-funtiolla. Käyttäjän syöttämien salasanan ja käyttäjätunnuksen täytyy vastata tietokannassa oleviin käyttäjätunnukseen ja salasanaan ja ne tarkistetaan callback_check_database-funktiolla. Lisäksi salasana hajautetaan md5-funktiolla. Jos kaikki vaatimukset täyttyvät, käyttäjä ohjataan sovelluksen kotisivulle. Jos jokin vaatimus ei täyty, käyttäjä ohjataan takaisin kirjautumisnäkymään.

```
function check_database($password)
{
    $user_id = $this->input->post('txt_user_id');

    $result = $this->person_model->login($user_id,$password);

    if ($result)
    {
        $sess_array = array();
        foreach ($result as $row)
        {
            $sess_array = array(
                'id' => $row->id,
                'user_id' => $row->user_id,
            );
            $this->session->set_userdata('account_type', $row->account_type );
            $this->session->set_userdata('logged_in', $sess_array);
        }

        return TRUE;
    }
    else
    {
        $this->form_validation->set_message('check_database', 'Invalid username or password');
        return FALSE;
    }
}
```

*KUVIO 5. Käyttäjätunnuksen ja salasanan tarkistusfunktio.*

Ylläolevassa kuviossa on käyttäjätunnuksen ja salasanan tarkistusfunktio. Funktio lukee käyttäjätunnuksen sisäänkirjautumisnäkymään syötetystä kentästä ja salasana tulee funktioon parametrinä. Käyttäjätunnus ja salasana tarkistetaan Henkilö-mallissa olevalla sisäänkirjautumisfunktiolla, jonne ne välitetään parametreinä. Jos funktio palauttaa tietoja, niin luodaan istunto ja istuntotaulu.

Istuntotauluun tallennetaan käyttäjän id, sekä käyttäjätunnus. Istuntoon tallennetaan käyttäjätunnuksen tyyppi, sekä muuttuja, joka kertoo, että käyttäjä on kirjautunut sisään. Jos käyttäjätunnus tai salasana on syötetty väärin, käyttäjä ohjataan kirjautumisnäkymään, jossa näkyy virheilmoitus, että käyttäjätunnus tai salasana on väärin.

```
public function login($user_id, $password)
{
    $this->db->select('id, user_id, password, account_type');
    $this->db->from('person');
    $this->db->where('user_id', $user_id);
    $this->db->where('password', MD5($password));

    $query=$this->db->get();

    if ($query -> num_rows() === 1)
    {
        return $query->result();
    }
    else
    {
        return false;
    }
}
```

*KUVIO 6. Henkilö-mallissa oleva sisäänkirjautumisfunktio.*

Kuviossa 6 on funktio, joka saa parametreinä käyttäjätunnuksen ja salasanan niiden tarkistusfunktiolta. Funktio tarkistaa, että käyttäjätunnus ja salasana vastaavat tietokannassa olevia käyttäjätunnusta ja salasanaa ja palauttaa sen perusteella käyttäjän id:n käyttäjätunnuksen, salasanan ja käyttäjätunnuksen tyypin. Salasanat on hajautettu MD5-algoritmillä tietokantaan tallennettaessa ja syötetty salasana hajautetaan MD5-algoritmillä, ennen kuin sitä verrataan tietokantaan.

Kun kirjautumiseen vaadittavat funktiot on suoritettu, sisäänkirjautumisohjain ohjaa käyttäjän sovelluksen etusivulle. Kun etusivu avataan, niin kotiohjain aukaisee istunnon, jossa on kirjautumisfunktiosta saadut tiedot. Käyttäjän käyttäjätunnus näkyy sovelluksen oikeassa yläreunassa ja käyttäjätunnuksen alapuolella on uloskirjautumispainike.

```
function logout()
{
    $this->session->unset_userdata('logged_in');
    $this->session->unset_userdata('account_type');
    session_destroy();
    redirect('login', 'refresh');
}
```

*KUVIO 7. Uloskirjautumisfunktio*.

Ylläolevassa kuviossa on uloskirjautumisfunktio. Funktio suoritetaan, kun käyttäjä painaa uloskirjautumispainiketta. Funktio tyhjentää istuntoon tallennetut tiedot ja tuhoaa istunnon, sekä ohjaa uloskirjautujan sisäänkirjautumisnäkymään.

```
public function index()
{
    if ($this->session->userdata('logged_in'))
    {

    }
    else
    {
        redirect('login', 'refresh');
    }
}
```

*KUVIO 8. Esimerkkifunktio sovelluksesta.*

OpixManagerin jokaisessa funktiossa tarkistetaan, että sen suorittaja on kirjautunut sisään. Tarkistus suoritetaan lukemalla istuntomuuttuja, johon tallennetaan tieto, että käyttäjä on kirjautunut sisään. If-lauseen sisään tulee funktio, joka suoritetaan. Jos istuntomuuttuja on tyhjä, niin käyttäjä ohjataan sisäänkirjautumisnäkymään.

### 4.2 Käyttäjähallinta

Tässä opinnäytetyössä käyttäjähallinta on toteutettu jakamalla käyttäjät kahteen auktorisointiryhmään: ylläpitäjät ja peruskäyttäjät. Kun käyttäjä kirjautuu sovellukseen, istuntoon tallennetaan käyttäjätilityyppi, jotta sitä voidaan käyttää apuna sovelluksen käyttäjähallinnassa. Suurin osa toiminnoista on tarkoitettu vain

ylläpitäjille. Seuraavissa kappaleissa käydään läpi missä on käytetty käyttäjänhallintaan liittyviä tarkistuksia.

```
if ($this->session->userdata('account_type') == 1)
{
```

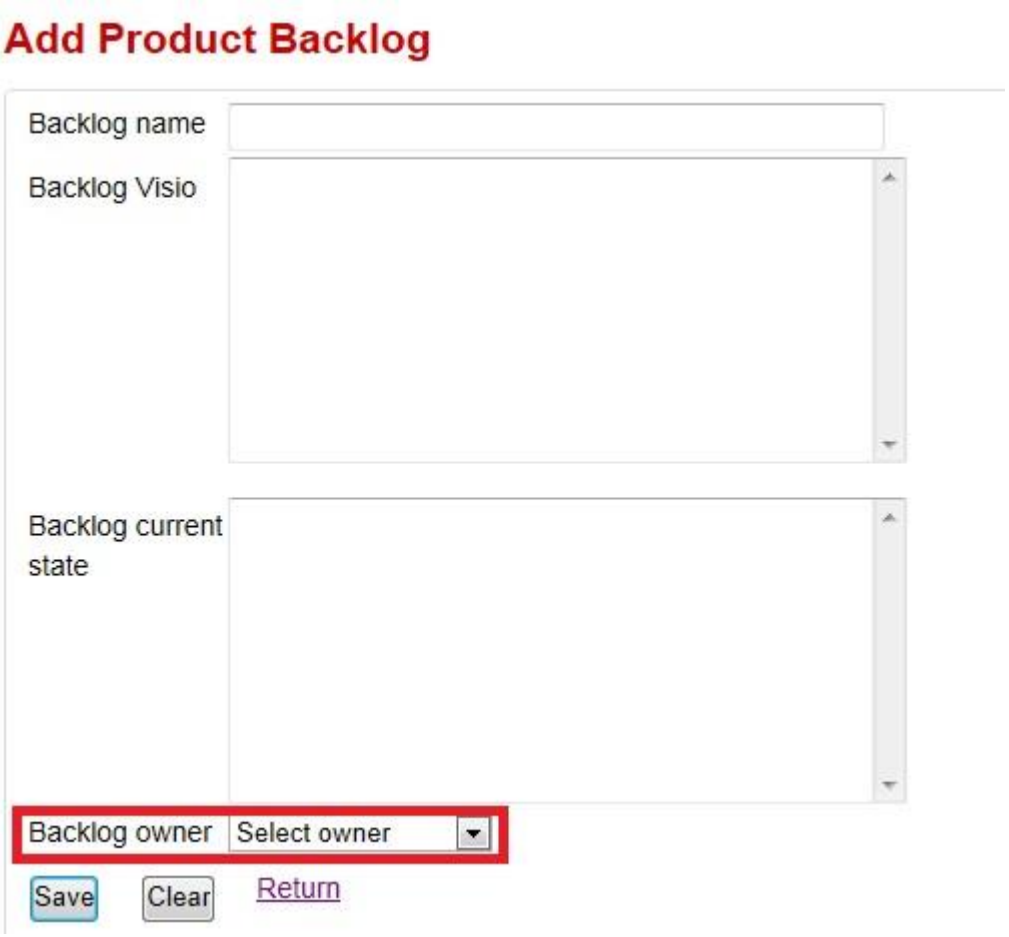
*KUVIO 9. Käyttäjätilityypin tarkistus.*

Ylläolevaa tarkistusta käytetään ylläpitäjille tarkoitetuissa sovelluksen funktioissa. If-lause tarkistaa, että kirjautunut käyttäjä on ylläpitäjä. Tietokannassa ylläpitäjän id on 1. Tällä tavalla rajoitetaan, että peruskäyttäjät eivät voi tehdä tiettyjä toimintoja.

```
if ($this->session->userdata('account_type') == 1 || $data['login_id'] === $id)
{
```

*KUVIO 10. Käyttäjätilityypin ja käyttäjän tarkistus*.

Kuviossa 10 tarkistetaan, että funktion suorittaja on ylläpitäjä tai kirjautuneena oleva henkilö. Tätä if-lausetta käytetään tietyn henkilön tietoja muokatessa. Ylläpitäjä voi muokata kaikkia henkilöitä, mutta kirjautunut käyttäjä voi muokata vain omia tietojaan.




*KUVIO 11. Tuotteen kehitysjonon lisäys.*

Tuotteen kehitysjonon voi luoda kuka tahansa, joka on mukana kyseisessä tuotteen kehitysjonoon liittyvässä projektissa ja samalla sille valitaan omistaja. Tämän jälkeen tuotteen kehitysjonoa voi muokata vain ylläpitäjä tai omistaja. Seuraavassa kuviossa on if-lause, jolla tämä saadaan toteutettua. Lause tarkistaa, että käyttäjä on admin tai tuotteen kehitysjonon omistaja.

```
if ($this->session->userdata('account_type') == 1 || $data['login_id'] == $product_backlog[0]->product_owner)
{
```

*KUVIO 12. Tuotteen kehitysjonon omistajan tarkistus.*

```
public function choose_person()
{
    if ($this->session->userdata('logged_in'))
    {
        $session_data = $this->session->userdata('logged_in');
        if ($this->session->userdata('account_type') == 1)
        {
            $data['selected_person'] = 0;
            $persons_from_db = $this->person_model->read_names();
            $this->load->helper("form_input_helper");
            $persons = convert_db_result_to_dropdown(
            $persons_from_db, 'id', 'name');
            $data['persons'] = $persons;

            $data['pagetitle'] = $this->lang->line('title_choose_person');
            $data['login_user_id'] = $session_data['user_id'];
            $data['login_id'] = $session_data['id'];
            $data['main_content'] = 'sprint_work/choose_person_view';
            $this->load->view('template', $data);
        }
        else
        {
            $data['login_user_id'] = $session_data['user_id'];
            $data['login_id'] = $session_data['id'];
            $error_message = $this->lang->line('not_allowed');
            $this->session->set_flashdata('$error_message', $error_message);
            redirect('sprint_work/index/' . $data['login_id']);
        }
    }
```

*KUVIO 13. Henkilön valinta.*

Ylläoleva funktio liittyy henkilön työtuntien lisäykseen. Jos käyttäjä on ylläpitäjä, näkyviin tulee pudotusvalikko, josta ylläpitäjä voi valita kenen käyttäjän työtunteja menee katsomaan. Jos käyttäjä on peruskäyttäjä, funktio ohjaa käyttäjän suoraan käyttäjän omalle tuntikirjaussivulle.

**Staff Roles**

Add Staff Role

| Role name | Description |
|---|---|
| Member | |
| Project Manager | |
| Scrum Master | |

*KUVIO 14. Projektityöntekijöiden roolit.*

Kuviossa 14 on kolme erilaista roolia, joista yksi lisätään henkilölle, kun tämä lisätään projektin jäseneksi. Jokaisella roolilla on oma id. Projektipäällikkö ja ylläpitäjä voi lisätä henkilöitä mukaan projektiin.


*KUVIO 15. Henkilöiden lisäys projektiin.*

Ylläolevassa kuviossa on GrafUx-projektin henkilöt. Esimerkiksi tähän projektiin henkilöitä voi lisätä ja muokata ylläpitäjä ja Paavo Päivänlahti, joka on projektipäällikkö. Kuviossa 16 on if-lause, mikä tarkistaa käyttäjätilityypin tai projektityöntekijän roolin id:n perusteella, voiko käyttäjä lisätä henkilöitä projektiin tai muokata jo mukana olevia henkilöitä. Tietokannassa projektipäällikön id on 3.

```
if ($data['person_role_id'] == 3 || $this->session->userdata('account_type') == 1)
{
```

*KUVIO 16. Käyttäjätilityypin ja projektityöntekijän roolin tarkistus.*

# 5 JOHTOPÄÄTÖKSET

Opinnäytetyön tavoitteena oli toteuttaa kirjautuminen ja käyttäjähallinta web-sovellukseen. Työ päätettiin aloittaa kirjautumisen toteuttamisesta. Etsimme internetistä esimerkkejä, miten kirjautuminen toteutetaan eri tavoin. Valitsimme tavan, jolla päätimme tehdä kirjautumisen OpixManageriin ja toteutus onnistui mielestämme hyvin. Saimme toteutettua OpixManageriin toimivan kirjautumisen, jossa käytetään hyväksi istuntoa sekä istuntomuuttujia. Niiden avulla sovellukseen luotiin tarkistuksia, jotta sovellusta ei pääse käyttämään, ellei kirjaudu sisään ensin.

Toinen osuus oli toteuttaa käyttäjähallinta. Sovelluksessa on kaksi käyttäjätilityyppiä, jotka ovat ylläpitäjät sekä peruskäyttäjät. Toteutimme sovellukseen tarkistuksia, joissa tutkitaan käyttäjän käyttäjätilityyppi ja sen perusteella käyttäjälle voidaan antaa oikeus suorittaa jokin toiminto sovelluksessa tai sitten toiminnon suoritus estetään. Iso osa sovelluksen toiminnoista on vain ylläpitäjiä varten. Onnistuimme tässä osuudessa hyvin ja saimme aikaan toimivia tarkistuksia.

Kirjautumisen avulla estetään sovelluksen luvaton käyttö ja rajoitetaan sovelluksen käyttöä. Sovellusta ei voi käyttää, jos ei ole tunnuksia ja tunnukset sovellukseen voi luoda vain ylläpitäjä. Käyttäjähallinnalla rajoitetaan peruskäyttäjien oikeuksia. Peruskäyttäjät voivat muokata vain projekteja, joissa he ovat mukana.

Tapa, jolla teimme kirjautumisen ja käyttäjähallinnan on yksinkertainen ja sitä on helppo hyödyntää jatkossa. Kirjautuminen ja käyttäjähallinta on helppo implementoida uusiin toimintoihin, joita OpixManageriin tullaan kehittämään tulevaisuudessa.

# 6 POHDINTA

Opinnäytetyön tavoitteena oli toteuttaa kirjautuminen ja käyttäjähallinta Opix-Manageriin. Opinnäytetyö on tehty käyttäen MVC-arkkitehtuuria ja CodeIgniter-ohjelmistokehystä. Ohjelmointi tehtiin NetBeans-ohjelmistolla ja sovelluksen toimivuus testattiin paikallisesti Xampp-ohjelman avulla.

Kirjautuminen toteutettiin funktioilla, joissa käyttäjän syöttämää käyttäjätunnusta ja salasanaa verrataan sovelluksen tietokantaan. Kun käyttäjä kirjautuu onnistuneesti sovellukseen, aloitetaan istunto. Istuntoon tallennetaan käyttäjän käyttäjätunnus, id ja käyttätilityyppi sekä muuttuja, joka kertoo, että käyttäjä on kirjautunut sisään.

Käyttäjähallinnassa käytettiin apuna istuntoon tallennettuja muuttujia sekä projektihenkilöiden eri rooleja. Käyttäjätilityypin perusteella peruskäyttäjiltä estettiin useita toimintoja, jotka ovat tarkoitettu vain ylläpitäjille. Projektihenkilön roolin perusteella annettiin eri rooleille oikeuksia suorittaa toimintoja projektisivuilla.

Menetelmät, joita käytimme opinnäytetyössä, olivat yksinkertaisia ja helppoja toteuttaa. Opinnäytetyö aloitettiin keväällä, mutta aikataulu muuttui työn aikana. Vaikka ohjelmointi oli kohtuullisen helppo toteuttaa, käytimme työhön paljon aikaa, koska OpixManagerissa oli jo ennen opinnäytetyön aloittamista paljon sellaisia toimintoja, jotka piti huomioida kirjautumisen ja käyttäjähallinnan toteutuksessa. Kirjautumisen ja käyttäjähallinnan ohjelmointi opetti meille paljon uutta asiaa, koska emme olleet aikaisemmin tehneet tämänkaltaisia töitä.

Koska olemme kehittäneet OpixManageria paljon aikaisemminkin, meillä on muutamia kehitysideoita, joita kerromme lyhyesti. Käyttäjän unohtaessa salasanan, ylläpitäjä joutuu alustamaan salasanan. Salasanan vaihto olisi hyvä toteuttaa automatisoidusti. Jos käyttäjä unohtaa salasanan, hän voi painaa painiketta kirjautumissivulla, jonka jälkeen sovellus lähettää uuden salasanan käyttäjän sähköpostiin. Tähän on CodeIgniterissä oma sähköpostikirjasto, jolla tämä voidaan toteuttaa. Tällä hetkellä salasana on suojattu MD5-hajautuksella, joka on

kohtuullisen nopea murtaa nykytekniikalla. Sen takia olisi hyvä tutkia parempia suojausmenetelmiä. Sovelluksen tietokantarakennetta kannattaisi miettiä vielä lisää, koska esimerkiksi, jos poistaa projektipäällikköroolin ja lisää sen uudestaan, rooli saa uuden id:n ja se sekoittaa nykyisen käyttäjähallinnan projektisivuilla. Tuotteen kehitysjonon omistaja tallennetaan tällä hetkellä nimellä tietokantaan. Parempi vaihtoehto olisi tallentaa tuotteen kehitysjonon omistaja tietokantaan henkilön id:n perusteella.

# LÄHTEET


Kent, S. & Millett, I. 2003. Who Goes There? Authentication Through the Lens of Privacy.United States, Washington, DC. National Academies Press

Bluecoat 2007. User management. Hakupäivä 30.10.2013,

https://www.bluecoat.com/sites/default/files/documents/files/User_Management.8.pdf

Ellislab 2013. CodeIgniter. Hakupäivä 8.4.2013,

http://ellislab.com/codeigniter

HTML 2004. HTML. Hakupäivä 2.10.2013,
http://www.sivut.org/html/oppaat/yleista.php

Microsoft 2013. MVC overview. Hakupäivä 17.10.2013,

http://www.asp.net/mvc/tutorials/older-versions/overview/asp-net-mvc-overview

MySql 2009. MySql. Hakupäivä 2.10.2013,
http://www.ohjelmointiputka.net/oppaat/opas.php?tunnus=mysqlphp01

NetBeans IDE 2013. Netbeans. Hakupäivä 8.4.2013,

https://netbeans.org/features/index.html

Opix 2013. Opix-Manager. Hakupäivä 6.5.2013,
http://opixproject.opiskelijaprojektit.net/index.php/opixproject/general

OWASP. Password length & complexity. Hakupäivä 21.11.2013,
https://www.owasp.org/index.php/Password_length_%26_complexity

PHP 2013a. Password. Hakupäivä 12.11.2013,
http://php.net/manual/en/faq.passwords.php

PHP 2013b. PHP. Hakupäivä 8.4.2013,
http://www.php.net/manual/en/intro-whatis.php

Xampp 2013. Xampp. Hakupäivä 8.4.2013,
http://www.apachefriends.org/en/xampp.html