VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Riku Siimesvaara

TYÖAJANSEURANTASOVELLUS

Case, Oy Merinova Ab

Tekniikka ja liikenne

2015

VAASAN AMMATTIKORKEAKOULU
Tietotekniikan koulutusohjelma

TIIVISTELMÄ

Tekijä	Riku Siimesvaara
Opinnäytetyön nimi	Työajanseurantasovellus
Vuosi	2015
Kieli	suomi
Sivumäärä	34
Ohjaaja	Santiago Chávez Vega

Tämän opinnäytetyön tarkoituksena oli kehittää uusi työajanseurantasovellus teknologiakeskus Merinova Oy:lle. Edellisen sovelluksen toiminnallisuus oli puutteellista ja käytetyt tekniikat vanhentuneita, joten uuden sovelluksen kehittäminen oli perusteltua. Uutta sovellusta kehittäessä otettiin erityisesti edellisen järjestelmän puutteet huomioon.

Web-sovelluksen käyttöliittymä toteutettiin käyttäen HTML5-kieltä ja jQuery-kirjastoa, sivuston ulkoasu tyyliteltiin Bootstrap CSS-komponenteilla. Palvelinpuolen kommunikaatio ja tietokantayhteydet toteutettiin PHP-ohjelmointikielellä. Tietokannaksi valittiin MySQL, johon vanhan Access-tietokannan tiedot täytyi siirtää.

Opinnäytetyön tuloksena kehitettiin sovellus, joka korvasi vanhan työajanseurantajärjestelmän.

Avainsanat HTML5, PHP, MySQL, jQuery, työajanseuranta, raportointi

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Tietotekniikan koulutusohjelma

ABSTRACT

Author	Riku Siimesvaara
Title	Attendance Tracking Software
Year	2015
Language	Finnish
Pages	34
Name of Supervisor	Santiago Chávez Vega

The goal of this thesis was to develop a new Attendance Tracking Software for technology centre Merinova Oy. The functionality of the previous application was inadequate and the techniques used in the development of the software outdated, so the development of a new application was needed. While developing the new application the shortcomings of the previous system were particularly taken into account.

The user interface of the Web application was implemented using HTML5 language and jQuery library, layout was styled with Bootstrap CSS components, server-side communication and database connection were implemented with PHP programming language. Old information from the MS Access Database was migrated to the new MySQL Database.

As a result of this thesis, an application that replaced the old Attendance Tracking Software was developed.

Keywords HTML5, PHP, MySQL, jQuery, working time attendance, reporting

SISÄLLYS

1 JOHDANTO....8
2 TYÖN TAUSTA....9
2.1 Teknologiakeskus Oy Merinova Ab....9
2.2 Käytössä oleva sovellus....9
3 KÄYTETYT TEKNIIKAT JA TYÖKALUT....10
3.1 HTML5....10
3.2 Javascript ja jQuery....10
3.3 DataTables, TableTools ja Editor....10
3.4 CSS3 ja Bootstrap....11
3.5 PHP....11
3.6 MySQL....11
3.7 LAMP....12
3.8 XAMPP....12
4 SOVELLUKSEN MÄÄRITTELY JA SUUNNITTELU....13
4.1 Vaatimusmäärittely....13
4.2 Käytettyjen tekniikoiden valinta....14
4.3 Tietokannan tietojen siirtäminen....14
4.4 Käyttöliittymän suunnittelu....15
4.4.1 Tuntikirjaus ja sovelluksen päänäkymä....15
4.4.2 Projektit....16
4.4.3 Hallintapaneeli....16
5 SOVELLUKSEN TOTEUTUS....17
5.1 Tietokantajärjestelmän muuntaminen....17
5.1.1 MS Access to MySQL-ohjelma....17
5.1.2 Vanhan tietokannan malli....18
5.2 PHP MySQL-yhteys....19
5.2.1 SQL-injektio....19
5.3 Sovelluksen ulkoasu....20
5.4 Käyttäjän tunnistautuminen....21
5.4.1 Tunnuksen rekisteröiminen....21
5.4.2 Salasanan tallentaminen tietokantaan....21
5.4.3 Salasanan palauttaminen....22
5.4.4 Sisäänkirjautuminen....22

5.5 Toiminnallisuus....23
5.5.1 DataTablesin käyttöönotto....23
5.5.2 Editorin käyttöönotto....24
5.5.3 Tuntikirjaus....26
5.5.4 Raportointityökalu....28
5.5.5 Käyttäjätasot....28
5.5.6 Hallintapaneeli....29
6 SOVELLUKSEN TESTAUS....30
6.1 Ensimmäinen vaihe....30
6.2 Toinen vaihe....31
6.3 Tuotanto....31
7 YHTEENVETO....32
LÄHTEET....33

LYHENTEET JA TERMIT

Access	Microsoft Access, Microsoftin relaatiotietokantaohjelma
AJAX	Asynchronous Javascript and XML, tekniikka vuorovaikutteiste verkkosivujen kehittämiseen
Bootstrap	Käyttöliittymäkomponenttikirjasto
CRUD	Create, Read, Update, Delete, kuvaa tietojen tellennuksen neljää perusfunktiota
CSS	Cascading Style Sheets, erityisesti verkkosivuilla käytetty tyyliohjeiden laji
HTML	HyperText Markup Language standardoitu kuvauskieli
JavaScript	Alustariippumaton ohjelmointikieli
Jquery	Javascript-kirjasto
LAMP	Linux, Apache, MySQL/MariaDB, PHP, Perl/Python, kokoelma avoimen lähdekoodin ohjelmia
PHP	Palvelinpuolen ohjelmointikieli
SQL	Structured Query Language, kyselykieli
W3C	World Wide Web Consortium, yhteenliittymä, joka ylläpitää ja kehittää WWW-standardeja

KUVIO- JA TAULUKKOLUETTELO

Kuvio 1.	Vanha sovellus	s. 9
Kuvio 2.	Käyttöliittymän suunniteltu etusivu	s. 15
Kuvio 3.	Raportti-välilehden suunniteltu ulkoasu	s. 16
Kuvio 4.	Tietokannan muuntaminen	s. 17
Kuvio 5.	Esimerkki EER-mallista	s. 18
Kuvio 6.	Tietokantayhteys	s. 19
Kuvio 7.	Tietokantakysely	s. 19
Kuvio 8.	Sovelluksen etusivu	s. 20
Kuvio 9.	Rekisteröintifunktion aktiviteettikaavio	s. 21
Kuvio 10.	Salasanan vaihtaminen sovelluksessa	s. 22
Kuvio 11.	Käyttäjän tila	s. 23
Kuvio 12.	Uloskirjautuminen	s. 23
Kuvio 13.	Datatables-taulun alustaminen	s. 24
Kuvio 14.	Editorin alustaminen ja konfigurointi	s. 24
Kuvio 15.	Editor PHP-kirjasto	s. 25
Kuvio 16.	Tuntikirjaus	s. 26
Kuvio 17.	Kysely suljinfunktioiden avulla	s. 26
Kuvio 18.	DrawCallBack-funktion käyttö	s. 27
Kuvio 19.	XMLHttpRequest-objekti	s. 27
Kuvio 20.	Ajax_response.php	s. 27
Kuvio 21.	Raportit-välilehti	s. 28
Kuvio 22.	Hallintapaneeli	s. 29
Taulukko 1.	Vaatimusmäärittely	s. 13

1 JOHDANTO

Suomen lain mukaan työnantajan on pidettävä kirjanpitoa kaikkien työntekijöiden tehdyistä tunneista, riippumatta siitä ovatko he tunti- tai kuukausipalkkaisia /7/. Työajanseuranta on niin työntekijän kuin työnantajan oikeusturva. Sen avulla tiedetään, mitä yrityksessä tehdään ja mihin työaikaa käytetään.

Hyvä työajanseurantajärjestelmä on osa kustannustehokasta yritystä. Työntekijän kannalta tärkeää on vaivaton tuntikirjanpito ja omien liukumasaldojen seuraamisen mahdollisuus. Työnantajan täytyy nähdä selkeä tilannekuva työntekijöiden tunneista ja työtuntien kertymisestä eri projekteille.

Markkinoilta löytyy useita työajanseurantajärjestelmiä, monet näistä ovat kuitenkin liian laajoja tai monimutkaisia erityiseen tarpeeseen. Tällöin räätälöidyn järjestelmän kehittäminen helpottaa sovelluksen käyttöä ja vähentää käyttäjän tekemää turhaa manuaalista työtä.

Tämä opinnäytetyö tehtiin teknologiakeskus Oy Merinova Ab:lle. Työn tarkoituksena oli kehittää uusi työajanseurantajärjestelmä vanhentuneen tilalle.

2 TYÖN TAUSTA

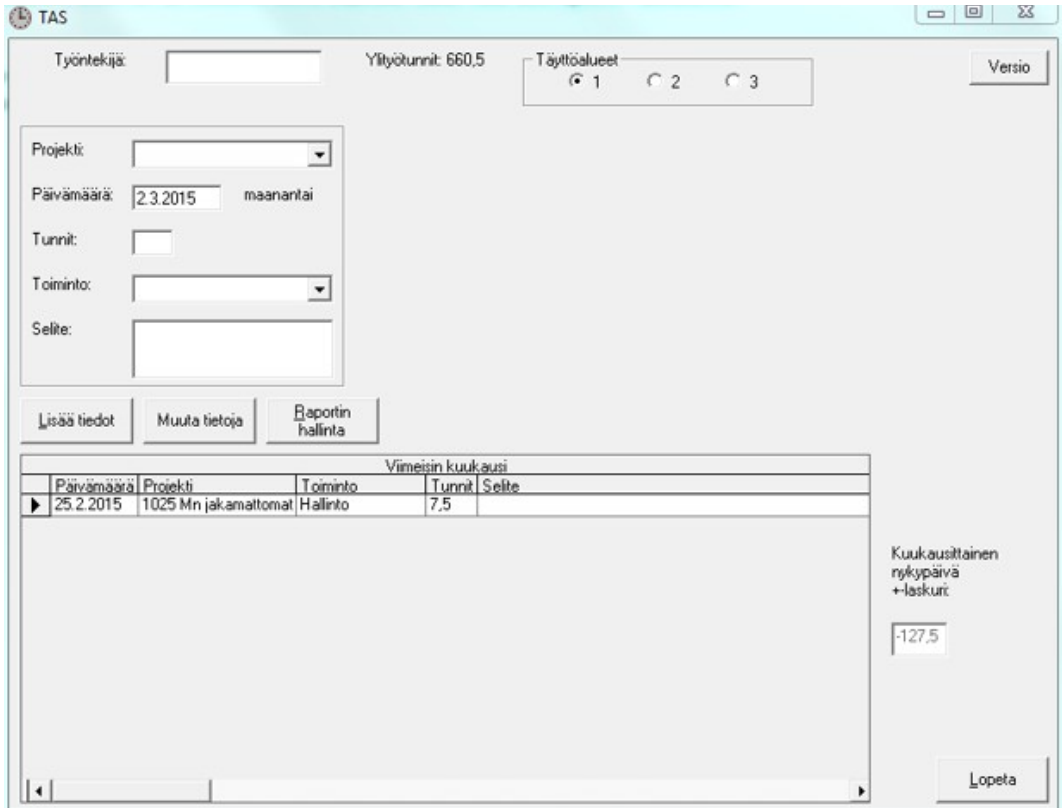
2.1 Teknologiakeskus Oy Merinova Ab

Merinova on erityisesti energiateknologia-alan yritysten toiminnan kehittämisen asiantuntija. Se vastaa Vaasan alueen teknologiakeskuspalveluista yhdessä osakkuusyhtiönsä Oy Vaasa Parks Ab:n kanssa. /10/

2.2 Käytössä oleva sovellus

Asiakkaalla oli käytössä heidän tarpeisiinsa räätälöity työajanseurantajärjestelmä **(Kuvio 1)**. Access-pohjainen työpöytäsovellus oli kuitenkin osoittanut ikääntymisen merkkejä, joten uuden sovelluksen kehittäminen todettiin tarpeelliseksi. Oli myös huomioitava, ettei vanhasta järjestelmästä ollut dokumentaatiota, joten uusi järjestelmä luotiin vanhan tietokannan, sovelluksen toimintojen ja esitettyjen toiveiden perusteella.

Järjestelmän siirtäminen verkkoon ja työntekijöille turhaa manuaalista työtä aiheuttavien ongelmien korjaus olivat tämän opinnäytetyön pääpainona.



Kuvio 1. Vanha sovellus

3 KÄYTETYT TEKNIIKAT JA TYÖKALUT

3.1 HTML5

HTML5-kieli on viimeisin versio verkkosivujen luontiin käytetystä HTML-merkintäkielestä. Se julkaistiin virallisesti 28. lokakuuta 2014 /19/. Terminä HTML5 on harhaanjohtava, koska sillä saatetaan tarkoittaa merkintäkielen lisäksi yleisesti nykyaikaisia web-tekniikoita (Javascript, CSS ja HTML) tai näillä tekniikoilla toteutettuja sovelluksia.

Työssä kehitetty web-sovellus on HTML5-sovellus. Sovelluksen käyttöliittymän rakenne on määritelty HTML-dokumenttina, ulkoasu on määritelty CSS-tyylitiedostoilla ja asiakaspään toiminnallisuus on toteutettu Javascript-ohjelmointikielellä.

3.2 Javascript ja jQuery

Javascript (JS) on oliopohjainen ohjelmointikieli, jota käytetään pääasiassa asiakaspään interaktiivisen toiminnallisuuden toteuttamiseen selaimissa.

Jquery on avoimen lähdekoodin Javascript-kirjasto, joka yksinkertaistaa Javascript ohjelmointia. Se on suunniteltu helpottamaan HTML-dokumentin ja DOM-elementtien hallintaa sekä Ajax-sovellusten kehittämistä. Lisäksi se mahdollistaa erilaisten liitännäisten käyttämisen. Tässä työssä jQuery-kirjastoa käytetään DataTables-liitännäisen hyödyntämiseksi.

3.3 DataTables, TableTools ja Editor

DataTables on jatkuvan kehityksen alla oleva jQuery-liitännäinen /5/. Se sisältää HTML-taulujen hallintaa helpottavia toimintoja, kuten tässä työssä käytetyn tietokantayhteyden ja suodatuksen.

TableTools on DataTables-liitännäinen, joka mahdollistaa taulukon tietojen tallentamisen CSV-, XLS- tai PDF-muotoon.

Editor on maksullinen lisäosa DataTablesiin, sen tärkein ominaisuus on muokkausmahdollisuuden lisääminen DataTables tauluihin /6/.

3.4 CSS3 ja Bootstrap

Cascading Style Sheets (CSS) on erityisesti verkkosivuille kehitetty tyyliohjeiden laji. Toisin kuin CSS2 joka on yksi iso standardi, CSS3 on jaettu moduuleihin /18/.

Tyyliohjeilla sivun ulkoasun määrittely ja yhtenäistäminen on helpompaa, ulkoasun päivittäminen onnistuu keskitetysti ja HTML-dokumentin rakenne pysyy yksinkertaisena. On kuitenkin huomioitava, että selaimet saattavat näyttää saman asian eri tavoin ja osa käyttäjistä saattaa yhä käyttää selainta, joka ei tue tyylisivuja.

Bootstrap on Twitterin työntekijöiden kehittämä, responsiivisten verkkosivujen ja web-sovellusten kehittämisessä avustava HTML-, CSS- ja Javascript-viitekehys. Sen viimeisin versio Bootstrap3 julkaistiin 19. elokuuta 2013 /2/.

3.5 PHP

PHP on erityisesti dynaamisten web-palveluiden tuottamiseen soveltuva palvelinpuolen komentosarjakieli. PHP-ympäristössä on valmiita luokkakirjastoja useille eri toiminta-alueille ja tietokannoille /14/.

Työssä hyödynnetään muun muassa mysqli-, mhash-, msession- ja mail-luokkakirjastoja.

3.6 MySQL

MySQL on maailman suosituin vapaan lähdekoodin relaatiotietokantaohjelmisto. Relaatiotietokannassa taulujen välille luodaan yhteys eli relaatio. Kantavana ajatuksena on se, että tieto tallennetaan vain yhteen paikkaan ja se on aina löydettävissä yhteyksien avulla /12/.

Työssä uudeksi tietokantajärjestelmäksi valittiin MySQL, koska se oli palveluntarjoajan käyttämä järjestelmä ja siirtyminen yhdestä relaatiotietokantajärjestelmästä (Microsoft Access) toiseen on helppoa.

3.7 LAMP

LAMP on kokoelma avoimen lähdekoodin ohjelmia, sen nimi on akronyymi neljästä alkuperäisestä komponentista: Linux-käyttöjärjestelmä, Apache HTTP-palvelin, MySQL-tietokanta ja PHP-ohjelmointikieli. Kokoelman voidaan suorittaa dynaamisia web-sivuja ja -sovelluksia /17/.

Asiakasyrityksen IT-palveluntarjoaja tarjoaa mahdollisuuden LAMP-alustan käyttöön.

3.8 XAMPP

XAMPP on täysin ilmainen paikallistasolla suoritettava testiympäristö, joka sisältää samat komponentit kuin LAMP /1/.

4 SOVELLUKSEN MÄÄRITTELY JA SUUNNITTELU

Tässä luvussa käydään läpi sovelluksen määrittelyn ja suunnittelun eri vaiheet ennen varsinaista toteutusvaihetta.

4.1 Vaatimusmäärittely

Alkuperäinen työajanseurantajärjestelmä oli räätälöity yrityksen tarpeisiin eikä dokumentaatiota ollut, joten määrittely aloitettiin tutustumalla nykyisen ohjelman toimintaan ja sen puutteisiin. Ensimmäisen palaverin jälkeen lähtökohdaksi asetettiin se, että työntekijä voi kirjata tuntejaan eri projekteihin, työntekijän ylityötunnit kirjataan automaattisesti liukumakassaan, kirjatuista tunneista on mahdollista saada raportti eikä käyttöliittymä saisi muuttua turhan paljon. Tämän lisäksi järjestelmään toivottiin helpompaa tapaa tuottaa raportteja, käyttäjätasojen mukaan rajoitettuja oikeuksia, työntekijälle mahdollisuutta muokata omia kirjauksiaan sekä tuntikirjauksen estämistä pyhäpäiviltä.

Päätettiin myös järjestelmän siirtämisestä työpöytäsovelluksesta verkkoon ja tietokannan muuntamisesta verkkolevyllä sijaitsevasta Access-tietokannasta palvelimella sijaitsevaan tietokantaan.

Taulukko 1. Vaatimusmäärittely

Vaaditut • Kolme eri käyttäjätasoa: projektityöntekijä, projektijohtaja ja admin • Tason 1 käyttäjä voi kirjata ja muokata omia tuntejaan eri projekteihin • Tason 1 käyttäjä näkee kuukauden aikana kirjatut tuntinsa • Tason 1 käyttäjä näkee kirjattujen tuntien summan, liukuman ja puuttuvat tunnit • Tason 2 käyttäjällä on raportointimahdollisuus kaikkien käyttäjien tunneista eri projekteissa • Järjestelmään voi lisätä uusia käyttäjiä • Järjestelmä laskee liukuman • Tietokannan siirto verkkolevyltä palvelimelle • Vanhojen tietojen täytyy olla saatavilla • Ohjelma estää tuntikirjauksen merkityiltä päiviltä
Odotetut • Selkeä ja helppokäyttöinen käyttöliittymä • Tason 2 käyttäjä voi hallita projektien työntekijöitä • Tason 3 (admin) käyttäjä voi lisätä projekteja
Lisätoiminnot • Ohjelma estää tuntikirjauksen automaattisesti pyhäpäivien perusteella

4.2 Käytettyjen tekniikoiden valinta

Tässä kappaleessa kuvataan toteutusvaihetta edeltävää sovelluksen suunnittelua. Suunnitteluvaiheessa pyrittiin miettimään teknologia, jolla sovellus olisi mahdollisimman pitkäikäinen ja helposti päivitettävä.

Luonteeltaan työajanseurantasovellukset ovat kevyitä, toiminnallisuus on lähinnä tietojen tallentamista tietokantaan ja tallennetun tiedon näyttämistä oikeassa muodossa (CRUD). Tästä syystä työ toteutettiin HTML5 web-sovelluksena.

Web-sovellukset toimivat laitteen internetselaimesta käsin, joten ohjelma ei vaadi erillistä asennusta, toimii kaikissa käyttöjärjestelmissä ja on haluttaessa käytettävissä kaikkialta internetyhteyden välityksellä. Web-sovelluksen päivittäminen ja ylläpitäminen on natiivisovellusta helpompaa. Natiivisovelluksen etuna on laitteiden sisäänrakennettujen ominaisuuksien helpompi hyödyntäminen ja parempi suoritusteho, mutta työajanseurantasovelluksessa näistä ei ole konkreettista hyötyä.

Asiakasyrityksen IT-palvelujentarjoajan palvelimella on LAMP-kokoelma, joten palvelinpuolen kieleksi on kokemuksen puutteesta huolimatta luonnollista valita PHP. PHP on helppokäyttöinen komentorivikieli, joka sisältää valmiit luokkakirjastot tarvittaville ominaisuuksille.

Jqueryn tarkoitus on helpottaa JavaScriptin käyttämistä ja vähentää turhan koodin kirjoittamista. Työssä käytettävä DataTables vaatii jQuery-kirjaston toimiakseen.

Osa toiminnosta vaatii palvelimelta haetun tiedon näyttämistä käyttäjän muutosten perusteella. Ajaxin avulla pystytään päivittynyt tieto näyttämään niin, ettei koko verkkosivua tarvitse ladata uudelleen. Näin sovelluksesta saadaan nopeampi ja sulavampi.

Projektin alussa päätettiin, että verkkolevyllä sijaitsevan Access-tietokannan tiedot on aika päivittää palvelimelle. Uudeksi tietokantajärjestelmäksi valitaan ilmainen MySQL, samasta syystä kuin palvelinpuolen kieleksi valittiin PHP.

4.3 Tietokannan tietojen siirtäminen

Tietokantajärjestelmän muuntaminen päätettiin toteuttaa ilmaisella Bullzip Access To MySQL-ohjelmalla.

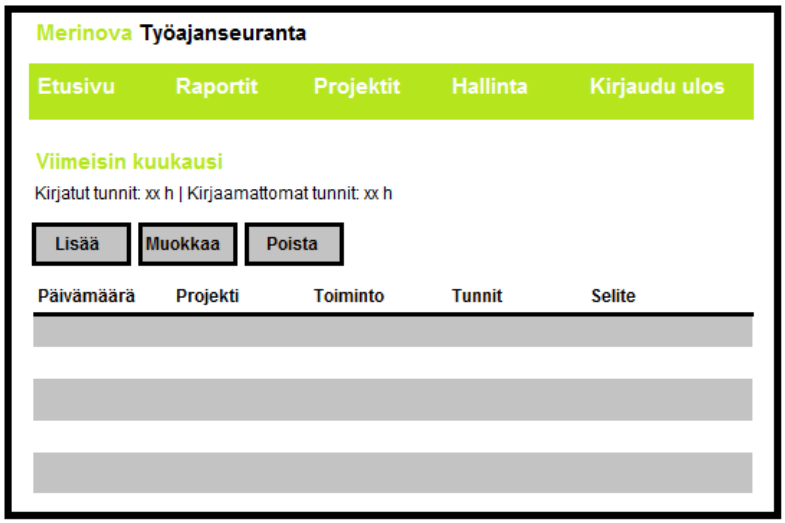
Valmiin tietokannan uudelleenkäyttäminen ei onnistunut ilman manuaalisia muutoksia. Työn aikana useita sarakkeita poistetaan ja uusia lisätään, tauluja nimetään uudelleen, sarakkeiden tyyppejä muutetaan ja uusia viite-eheyksiä luodaan.

4.4 Käyttöliittymän suunnittelu

Käyttöliittymä suunnitellaan vanhan sovelluksen pohjalta. Vanhentunutta ulkoasua piristettiin ja epäloogisia toimintoja selkeytettiin, mutta käyttöliittymän toiminnot pyrittiin pitämään samoina. Koska jokainen käyttäjä ei tarvitse sovelluksen kaikkia toimintoja, eri toiminnot päätettiin siirtää omille välilehdilleen. Näin ohjelman ulkoasu pysyy yhtenäisenä kaikille vaikka osa toiminnoista piilotetaan käyttäjätason mukaan.

4.4.1 Tuntikirjaus ja sovelluksen päänäkymä

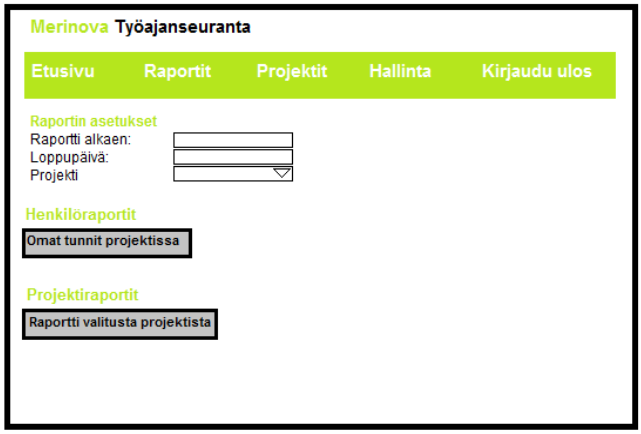
Sovelluksen käyttöliittymän pääpaino on helpossa ja nopeassa tuntien kirjauksessa. Tämä toiminto suunniteltiin niin, että se on ensimmäinen ja ainoa asia mitä sisäänkirjautunut käyttäjä näkee. Käyttäjän oman työajan seuraamisen helpottamiseksi etusivulla näkyy myös kuluneen kuukauden aikana kirjatut tunnit ja liukumasaldo.



Kuvio 2. Käyttöliittymän suunniteltu etusivu

Toinen tärkeä työajanseurantasovelluksen ominaisuus on raportointi. Alustava suunnitelma raportti-välilehden ulkoasusta **(Kuvio 3.)** tehtiin vanhan sovelluksen

perusteella, mutta myöhemmin ulkoasu muutettiin tukemaan uutta raportointijärjestelmää.



Kuvio 3. Raportti-välilehden suunniteltu ulkoasu

4.4.2 Projektit

Välilehdellä projektityöntekijä voi luoda raportin henkilökohtaisten työtuntien jakautumisesta eri projekteihin. Projektijohtaja näkee kaikkien projektiin töitä tehneiden työntekijöiden tunnit.

Työntekijä voi merkitä tunteja vain projekteihin, joissa on projektityöntekijänä. Projektienhallintaa varten kehitettiin uusi välilehti, jonka vain projektijohtaja voi nähdä. Tällä sivulla projektijohtaja voi lisätä työntekijöitä projekteihinsa.

4.4.3 Hallintapaneeli

Suurin ongelma vanhassa sovelluksessa oli hallintapaneelin puuttuminen. Tästä johtuen mitään muutoksia toiminnallisuuteen ei ollut mahdollista tehdä, joten uudet kyselyt tehtiin suoraan tietokantaa puukottamalla. Tahdoin kehittää järjestelmän, joka on helppokäyttöinen ja uusien tarpeiden mukaan muokattavissa, joten ohjelma tarvitsi hallintapaneelin.

Hallintapaneelista on rajattu pääsy tietokannan tauluihin, jotka tukevat ohjelman toiminnallisuutta. Sieltä käsin voidaan lisätä uusia työntekijöitä, projekteja ja hallita tuntikirjauksesta estettyjä päiviä. Välilehdellä näkyvä tieto järjestetään siten, että se on helpommin ymmärrettävissä kuin Access-tietokannan manuaalinen muokkaaminen.

5 SOVELLUKSEN TOTEUTUS

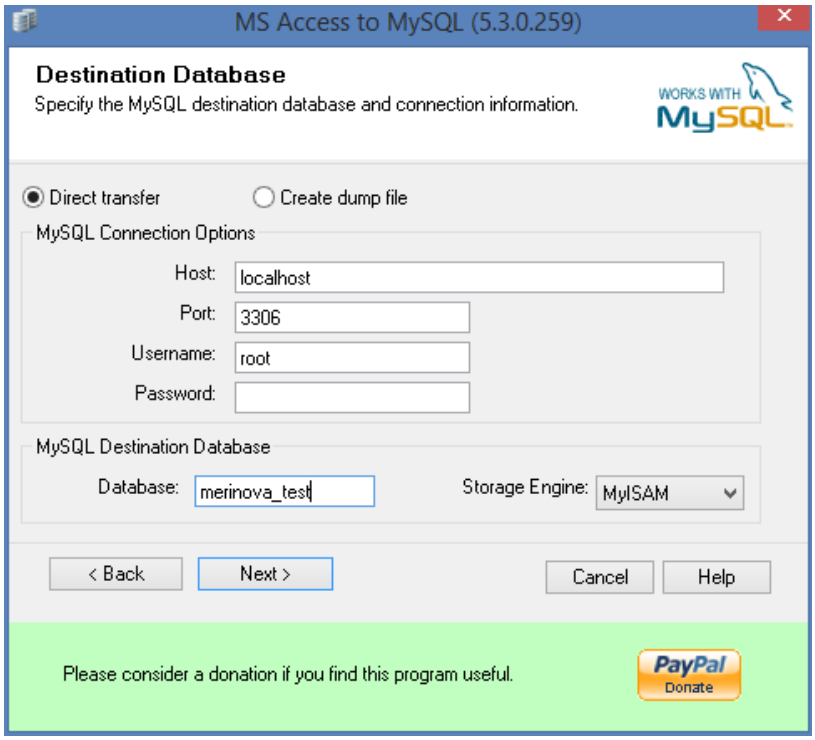
Tässä luvussa esitellään tietokantajärjestelmän muuntaminen, sovelluksen asiakaspään ulkoasun luonti sekä koko järjestelmän toiminnallisuuden toteutus. Tietoturvan takaamiseksi muuttujien nimet muutettu.

5.1 Tietokantajärjestelmän muuntaminen

Vanhan järjestelmän tiedot täytyi säilyttää, joten sovelluksen kehittäminen aloitettiin tietokannan siirtämisellä.

5.1.1 MS Access to MySQL-ohjelma

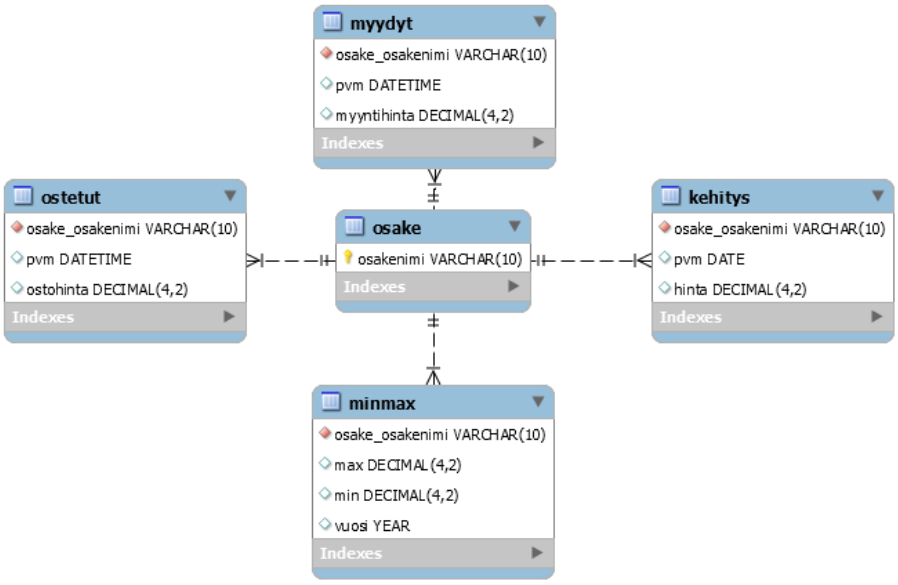
Tietokannan muuntamiseen valittiin ilmainen Bullzipin valmistama MS Access to MySQL-ohjelma**(Kuvio 4.)**. Ohjelman avustajan avulla valitaan muun muassa mitkä Access-tietokannan taulut, kentät, tiedot, indeksit ja viite-eheydet siirretään MySQL-tietokantaan. Kehitysvaiheessa kaikki tiedot kopioitiin uuteen testitietokantaan, mutta sovelluksen kehityksen aikana ilmenneiden ongelmien johdosta indeksejä ei kopioida tuotantoversioon siirryttäessä.



Kuvio 4. Tietokannan muuntaminen

5.1.2 Vanhan tietokannan malli

Kopioidun tietokannan rakenne on helpompi ymmärtää, jos sitä katselee laajennettuna ER-mallina (EER). EER-malli luotiin hyödyntämällä MySQL Workbench-ohjelman takaisinmallinnusominaisuutta(reverse engineering). /13/



Kuvio 5. Esimerkki EER-mallista

Tietokannan rakenteen paljastaminen on tietoturvan kannalta suuri riski, joten yllä **(Kuvio 5.)** on vain esimerkki EER-mallin rakenteesta.

Voidaan kuitenkin paljastaa, että mallista huomataan taulujen välillä olevien viittauksien(foreign key) puuttuminen. Toiseksi taulujen nimeämisperiaate on epäsäännöllinen ja tietokanta sisältää uuden sovelluksen kannalta hyödyttömät taulut *structure*, *prog_proj* ja *variables.*

Tutustumalla sovelluksen vaatimusmäärittelyyn ja tietokannan malliin voidaan todeta, että yhdellä työntekijällä voi olla monia työtunteja monessa eri projektissa . Projektilla voi olla yksi projektijohtaja ja useita projektityöntekijöitä. Työtunteja voi olla useita samana päivänä, samassa projektissa ja samalta työntekijältä.

Näillä alustavilla tiedoilla uuden sovelluksen kehittäminen aloitettiin.

5.2 PHP MySQL-yhteys

Työssä käytetään PHP 5:n MySQLi-luokkakirjastoa tietokantayhteyden luomiseksi. Luokkakirjaston avulla tietokantayhteyden- ja kyselyiden suorittaminen on helppoa. Esimerkki tietokantayhteyden avaamisesta MySQLi:n avulla**(Kuvio 6.)**:

```
1   <?php
2   $servername = "palvelin";
3   $username = "tunnus";
4   $password = "salasana";
5
6   // Luodaan yhteys
7   $conn = new mysqli($servername, $username, $password);
8
9   // tarkistetaan yhteys
10  if ($conn->connect_error) {
11      die("Yhteys epäonnistui";
12  }
13  echo "Yhteys onnistui";
14  ?>
```

Kuvio 6. Tietokantayhteys

Yhteyden avaaminen vaatii palvelimen osoitteen, käyttäjätunnuksen ja salasanan. Kun yhteys on avattu, tietokantaan on mahdollista tehdä muutoksia. Alla yksinkertainen esimerkki työssä käytetystä kyselystä **(Kuvio 7.)**:

```
$sql = "SELECT * FROM          WHERE          = 1";
$result = $mysqli->query($sql);
```

Kuvio 7. Tietokantakysely

Selkeyden vuoksi kyselyn sisältö muotoillaan muuttujaan ja muuttujaa lähetetään eteenpäin tietokantaan.

5.2.1 SQL-injektio

SQL-injektio on yksi tapa hyökätä tietokantapohjaisiin sovelluksiin. SQL-injektio tapahtuu syöttämällä haitallisia SQL-lausekkeita asiakassovelluksen kenttien kautta. Hyökkäyksen estämiseksi kaikki käyttäjältä tulevat syötteet pitää tarkistaa ja sanitoida injektion varalta /16/.

Onnistuneeseen hyökkäykseen on tunnettava edes osa tietokannan rakenteesta. Usein tarvittavan tiedon saaminen on kuitenkin helppoa hyödyntämällä ohjelman virheilmoituksia tai yleisiä taulujen nimeämisperiaatteita

SQL-injektion riskiä voidaan minimoida käyttämällä tietokantaa rajoitetuilla oikeuksilla, sanitoimalla käyttäjäsyötteet ja salaamalla tietokannan rakenteen.

5.3 Sovelluksen ulkoasu

Sovelluksen ulkoasu **(Kuvio 8.)** määriteltiin HTML5-kielellä ja tyyliteltiin Bootstrap CSS-komponenteilla. Bootstrapin alkuperäisen tyylitiedoston rinnalla käytettiin sovellukselle tehtyä mukautettua tyylitiedostoa /3/.

Bootstrapin käyttöön ei päädytty niinkään ulkoasun vaan toiminnallisuuden perusteella. Vaatimusmäärittelyn mukaan sovellus ei saa sallia tuntikirjausta pyhäpäivinä ja Bootstrapin laajennuksista löytyi vaatimuksien mukainen Bootstrap 3 Datepicker. Bootstrap 3 Datapicker lisää useita hyödyllisiä ominaisuuksia HTML5-kalenterisyötteeseen. Yksi näistä ominaisuuksista on valittujen päivien estäminen /8/.

Kuvio 8. Sovelluksen etusivu

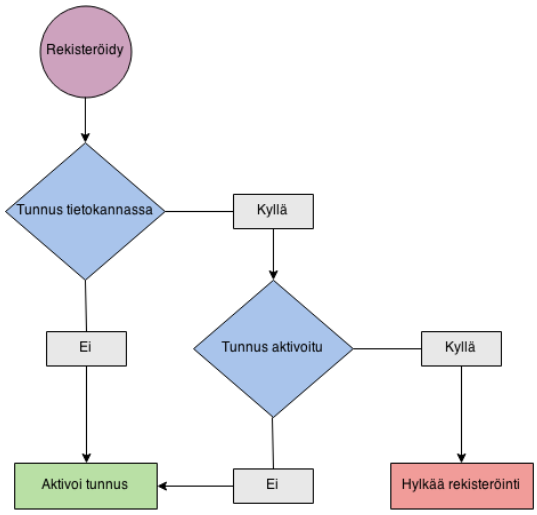
5.4 Käyttäjän tunnistautuminen

Vanhan sovelluksen käyttäjän tunnistautuminen perustui Visual Basicin luokkaan, joka tunnistaa Windowsiin kirjautuneen käyttäjän /11/. Vastaava toiminallisuus olisi mahdollista toteuttaa PHP:n LDAP luokkakirjastolla, mutta LDAP:n monimutkaisen luonteen vuoksi työssä käytetään yksinkertaisempaa tapaa: sovelluskohtaista käyttäjän tunnistautumista.

5.4.1 Tunnuksen rekisteröiminen

Tietokanta sisälsi tuntikirjanpidon vanhaa sovellusta käyttäneiden työntekijöiden osalta, joten oli kehitettävä tapa joka yhdistää vanhan järjestelmän tuntikirjanpidon uusille tunnuksille.

Rekisteröitymisfunktio **(Kuvio 9.)** kehitettiin niin, että ohjelma tarkastaa tietokannasta löytyvät tunnukset. Tunnuksien löytyessä tietokannasta tarkistetaan, onko käyttäjä aktivoinut ne jo aiemmin. Mikäli aktivointikenttä on tosi, hylätään rekisteröityminen. Aktivointi mahdollistaa vanhan tunnuksen käytön uudessa järjestelmässä. Uudet tunnukset hyväksytään ja aktivoidaan automaattisesti.



Kuvio 9. Rekisteröintifunktion aktiviteettikaavio

5.4.2 Salasanan tallentaminen tietokantaan

Vaikka sisäverkossa toimivan sovelluksen tietoturvariskit ovat pienemmät, ei käyttäjien salasanoja koskaan tulisi tallentaa tietokantaan selväkielisenä. Yleinen tapa tallentaa salasana tietokantaan on sen tiivistäminen.

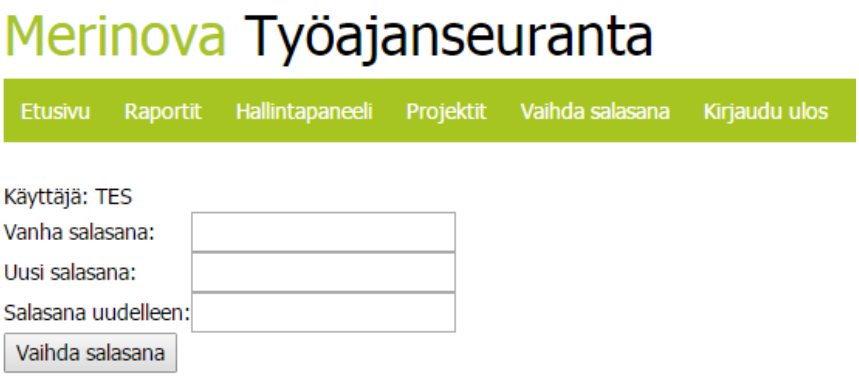
Tiivisteen tarkoitus on, ettei siitä pysty selvittämään alkuperäistä selväkielistä salasanaa helposti. Tiivisteen turvallisuutta voidaan parantaa lisäämällä salasanan perään ”suola” ennen tiivisteen laskemista. Suola pidentää salasanaa ja lisää näin sen turvallisuutta, niin ettei käyttäjän tarvitse itse kirjoittaa erittäin pitkää salasanaa.

Salasanan tiivistämiseen käytetty algoritmi ja suolaus vaikuttavat murtamisnopeuteen suuresti. Liian yksinkertaisesti tiivistetty salasana on helppo murtaa vertaamalla sitä esilaskettuihin tietokantoihin (Rainbow table).

5.4.3 Salasanan palauttaminen

Salasanan tiivistämisestä johtuen käyttäjän oman salasanan palauttaminen ei onnistu. Tästä syystä salasanan palauttamisfunktio kehitettiin niin, että uusi satunnaisesti luotu salasana lähetetään käyttäjätunnukselle kuuluvaan sähköpostiin ja salasanasta luotu tiiviste tallennetaan tietokantaan.

Käyttäjäkokemuksen parantamiseksi salasanan vaihtaminen on kuitenkin mahdollista hallintapaneelissa **(Kuvio 10.)**.



Kuvio 10. Salasanan vaihtaminen sovelluksessa

5.4.4 Sisäänkirjautuminen

Sovellus on monisivuinen, joten käyttäjän tunnistautumisen tilan on säilyttävä sivun vaihtuessa. Tämä ratkaistaan luomalla sisäänkirjautumisen yhteydessä istunto, joka sisältää käyttäjätunnuksen oikeudet ja sisäänkirjautumisen tilan.

PHP istunto (session) on tapa tallentaa tietoa, jota voidaan käyttää usealla sivulla. Toisin kuin keksi, tämä tieto ei tallennu käyttäjän koneelle /15/.

Sovelluksessa käyttäjän tila tarkistetaan yksinkertaisella funktiolla (muuttujan nimi muutettu). Kun muuttujalla on asetettu, funktion arvo on tosi **(Kuvio 11.)**.

```
function logged_in () {
    if (isset($_SESSION['muuttuja'])) {
        return true;
    } else {
        return false;
    }
}
```

Kuvio 11. Käyttäjän tila

Käyttäjän kirjautuessa ulos istunto tuhotaan ja kaikki istunnon muuttujat alustetaan tyhjiksi. Istuntomuuttuja ei ole enää asetettu, joten *logged_in()* funktion arvo on epätosi **(Kuvio 12.)**.

```
// unset all session variables
$_SESSION = array();
// destroy the session cookie
if(isset($_COOKIE[session_name()])) {
    setcookie(session_name(), '', time()-50000, '/');
}
// destroy the session
session_destroy();
```

Kuvio 12. Uloskirjautuminen

5.5 Toiminnallisuus

Työajanseurantasovelluksen perustoiminnallisuus on CRUD-periaatteen mukainen. Tietokannan tietoja luodaan (Create), luetaan (Read), päivitetään (Update) ja tuhotaan (Delete). Näiden toimintojen toteuttamiseen valittiin ilmainen DataTables jQuery liitännäinen ja sen maksullinen Editor lisäosa.

5.5.1 DataTablesin käyttöönotto

DataTables on joustava työkalu, joka helpottaa HTML taulujen hallintaa. Sen ominaisuuksiin kuuluu muun muassa palvelinpuolen prosessointi sekä taulun tietojen järjestäminen ja suodatus. Alla oleva kuvio **(Kuvio 13.)** kuvastaa Datables-taulun alustamista yhdellä funktiolla.

```
$(document).ready(function(){
$('#taulu').DataTable();
});
```

Kuvio 13. DataTables-taulun alustaminen

Alustuksen jälkeen HTML-taulu, jonka uniikki *id* on *#taulu*, toimii DataTables tauluna.

5.5.2 Editorin käyttöönotto

Editor tekee DataTables tauluista muokattavia. Sen valmiin PHP-kirjaston avulla käyttäjän tekemien muutosten päivittäminen tietokantaan on helppoa.

Editorin alustaminen tapahtuu funktiolla**(Kuvio 14)**:

```
var editor = new $.fn.dataTable.Editor( {
    "ajax": "php/test_hours.php",
    "table": "#test_hours",
    "fields": [
    { label: 'First name', name: 'first_name' },
    { label: 'Last name',  name: 'last_name'  }
]
```

Kuvio 14. Editorin alustaminen ja konfigurointi

Editorin konfigurointi tapahtuu syöttämällä parametrejä objektille. Yllä *ajax* viittaa URL osoitteeseen, minkä kautta taulussa oleva tieto lähetetään kun submit()-metodia kutsutaan. *Table* on taulu, jossa Editor on käytössä ja *fields* sisältää taulun muokattavat kentät.

Editorin PHP-kirjasto otetaan käyttöön konfiguroimalla *config.php* tiedosto. Tiedosto sisältää tiedot, joiden avulla kirjasto yhdistää tietokantaan.

Tietokantayhteyden konfiguroinnin jälkeen kirjasto on valmis käytettäväksi. Käyttö aloitetaan lisäämällä *DataTables.php* kirjasto.

Tämän jälkeen luodaan PHP Editor instanssi *Editor::inst($db, 'test_hours')*. Ensimmäinen parametri $db saa arvonsa konfiguraatiotiedostosta, toinen parametri viittaa operoitavan tietokannan taulun nimeen.

Fields()-metodin kautta kerrotaan Editor instanssille, mitä taulukon tietoja muokataan.

Data prosessoidaan process()-metodilla. Tässä vaiheessa Editor instanssi tekee asiakaspään pyytämät muutokset palvelimelle. Lopuksi data täytyy lähettää takaisin asiakkaalle, jotta taulu voidaan päivittää vastaamaan muutoksia. Se tapahtuu

json()-metodilla, joka lähetyksen lisäksi muotoilee datan automaattisesti JSON muotoon.

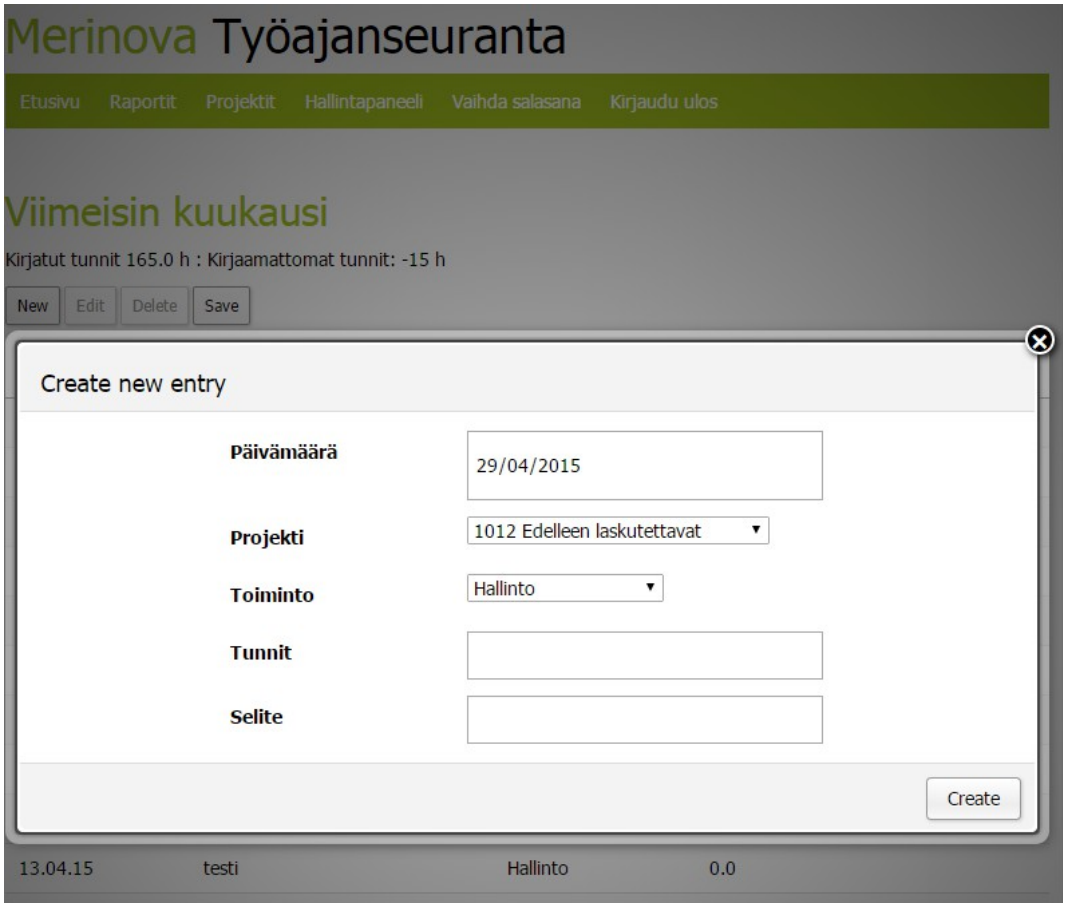
Alla esimerkki Editor PHP-kirjaston käytöstä **(Kuvio 15.)**.

```
<?php

// DataTables PHP library
include( "/php/DataTables.php" );

// Alias Editor classes
use
    DataTables\Editor,
    DataTables\Editor\Field,
    DataTables\Editor\Format,
    DataTables\Editor\Join,
    DataTables\Editor\Upload,
    DataTables\Editor\Validate;

// Build Editor instance and process the data coming from _POST
Editor::inst( $db, 'test_hours' )
    ->fields(
        Field::inst( 'first_name' ),
        Field::inst( 'last_name' )
    )
    ->process( $_POST )
    ->json();

?>
```

Kuvio 15. Editor PHP-kirjasto

5.5.3 Tuntikirjaus

Tuntikirjaus toteutettiin pääosin DataTables Editorilla. Sivulla käyttäjä pystyy kirjaamaan, muokkaamaan ja poistamaan senhetkisen kuukauden tunteja **(Kuvio 16.)**.



Kuvio 16. Tuntikirjaus

Sivulla näkymä rajoitettiin vain kirjautuneen käyttäjän sen hetkisen kuukauden tietoihin. Editorin PHP-kirjastot tarjoavat where()-metodin **(Kuvio 17.)**, jonka toiminta vastaa SQL WHERE-kyselyä. Where()-metodi ei toimi kuitenkaan yhtä helposti kuin suora kysely ja monimutkaisemmat pyynnöt vaativatkin usean kutsun yhdistämistä PHP:n suljinfunktioiden avulla.

```
$editor->where( function ( $q ) {
$q
    ->where('     työntekijä     ',$_SESSION['  tt_id  '])
    ->and_where(function ( $r ){
        $r->where( 'MONTH(   pvmsarake     )', 'MONTH(CURRENT_DATE)', '=', false );
        $r->where( 'YEAR(   pvmsarake     )', 'YEAR(CURRENT_DATE)', '=', false );
    });
} );
```

Kuvio 17. Kysely suljinfunktioiden avulla

Sivulla näkyy myös työntekijän sen hetkisen kuukauden kirjattujen tuntien summa. Summa lasketaan palvelinpuolella tietokannasta, joten muutoksen näyttäminen asiakaspuolella vaatii tavallisesti sivun uudelleenlatauksen. Sivun uudelleenlataus heikentäisi käyttäjäkokemusta, joten uusi summa haetaan asynkronisesti Ajaxilla käyttäjän muokatessa taulukkoa.

DataTables sisältää funktion drawCallBack **(Kuvio 18.)**, jota kutsutaan aina kun DataTables taulukon sisältö piirretään. Piirto-funktioa kutsutaan aina kun taulukkoon tehdään muutos, joten Ajax-pyynnön lähettäminen drawCallBack-funktiolla toimii hyvin vuorovaikutuksessa käyttäjän tekemien muutosten kanssa.

```
"drawCallback": function( settings ) {
loadXMLDoc();
},
```

Kuvio 18. DrawCallBack-funktion käyttö

Ajax käyttää XMLHttpRequest-objektia **(Kuvio 19.)** datan vaihtamiseen asynkronisesti palvelimen kanssa. Vaihdon onnistuessa *ajax_response.php* ohjelman tuloste näytetään HTML-elementissä, jonka *id* on *testing*.

```
function loadXMLDoc(){

    var xmlhttp;

    xmlhttp=new XMLHttpRequest();
    xmlhttp.onreadystatechange=function(){

      if (xmlhttp.readyState==4 && xmlhttp.status==200){
        document.getElementById("testing").innerHTML=xmlhttp.responseText;
      }
    }
    xmlhttp.open("GET","/php/ajax_response.php",true);
    xmlhttp.send();
}
```

Kuvio 19. XMLHttpRequest-objekti

Ajax_response.php sisältää tietokantayhteyden, SQL kyselyn ja laskutoimituksen. Se tulostaa Ajax-pyyntöön vastauksen **(Kuvio 20.)**:

```
echo " <h2>Viimeisin kuukausi</h2> Kirjatut tunnit $sum h : Kirjaamattomat tunnit: $lack h";
```

Kuvio 20. Ajax_response.php

5.5.4 Raportointityökalu

Tietokannasta on saatava työntekijä- ja projektikohtaisia raportteja määrätyltä aikaväliltä.

Tieto esitetään DataTablesilla ja rajataan käyttäjätason mukaan tietokannasta täytetyillä HTML-elementeillä **(Kuvio 21.)**.

Kuvio 21. Raportit-välilehti

Tabletools liitännäinen mahdollistaa taulukossa olevan tiedon kopioimisen leikepöydälle sekä tallentamisen CSV-, XLS- ja PDF-tiedostoon. Puutteena mainittakoon, ettei liitännäinen tue PDF-tiedoston ulkoasun muotoilua

5.5.5 Käyttäjätasot

Jokaiselle työajanseurantasovelluksen käyttäjälle on määritelty käyttäjätaso. Eri tason käyttäjät pystyvät tekemään sovelluksessa eri asioita.

Sovelluksen käyttäjätasot ovat projektityöntekijä, projektijohtaja ja ylläpitäjä.

Projektityöntekijä voi kirjata, muokata ja poistaa omia tuntejaan projekteihin, johon projektijohtaja on heidät lisännyt. Projektityöntekijä voi tehdä raportin omista tunneistaan.

Projektijohtaja voi edellä mainittujen asioiden lisäksi hallita omia projektejaan ja tehdä niistä raportteja.

Ylläpitäjä voi edellä mainittujen asioiden lisäksi hallita käyttäjiä, nimetä uusia projektijohtajia, hallinnoida projekteja ja estää tuntikirjauksen tietyiltä päiviltä.

Käyttäjätaso tarkistetaan ja asetetaan istuntomuuttujaan sisäänkirjautumisen yhteydessä. Osa valintapalkeista näkyy vain käyttäjille, joilla on oikeus käyttää kyseisen välilehden toimintoja.

5.5.6 Hallintapaneeli

Sovelluksen pitkäikäisyyden takaamiseksi on asiakkaan pystyttävä muokkaamaan sitä tarpeidensa mukaan. Hallintapaneelin kautta ylläpitäjällä on pääsy sovelluksen toiminnallisuutta tukeviin tietokannan tauluihin.

Hallintapaneelista **(Kuvio 22.)** voidaan muokata käyttäjien tasoja, hallita projekteja ja estää tuntikirjauksesta päiviä. Välilehdellä tehdään muutoksia tauluihin, jotka vaikuttavat jokaisen käyttäjän näkymään.

Kuvio 22. Hallintapaneeli

Jotta ulkoasu pysyisi selkeänä, toimintojen piilottaminen toteutettiin Simple-expand jQuery-liitännäisellä /9/. Liitännäisen avulla on helppo luoda avattavia ja suljettavia HTML-elementtejä.

6 SOVELLUKSEN TESTAUS

Web-sovellusta testattaessa on otettava huomioon natiivijärjestelmistä poikkeavia seikkoja. Toiminnallisuuden ja tietoturvan lisäksi on testattava muun muassa yhteensopivuutta, käytettävyyttä ja suorituskykyä eri selaimilla /4/.

Web-sivuilta- ja sovelluksista kannattaa yleisesti tarkastaa miten sivun navigointi ja päivittäminen toimii, mitä tapahtuu jos useampi selaininstanssi avataan sekä miten sivu käyttäytyy eri selaimilla ja resoluutioilla. Tietokantoja käyttävien sovellusten kanssa syötteiden tarkistus on erittäin tärkeää, oletetaan että käyttäjä saattaa käyttää järjestelmää tahattomasti tai tahallisesti väärin.

Tietoturvaa testattaessa on otettava huomioon web-sovelluksen tunnetut haavoittuvuudet: SQL-injektio (Katso sivu 19), Cross Site Scripting (XSS) ja Cross Site Request Forgery (XSRF).

Cross Site Scripting-haavoittuvuus perustuu asiakaspään syötteistä tulostettavan HTML-koodin väärinkäyttöön. Paras tapa välttää haavoittuvuus on käyttää vaihtoehtoisia tapoja HTML-koodin luomiseen, mutta joskus sivuston tarpeet pakottavat tähän. Tällöin turvallisin menetelmä on siistiä HTML koodi ennen tulostusta.

Cross site-pyynnön väärennöksessä (XSRF) hyödynnetään sivuston luottamusta käyttäjän selainohjelmaan. Selaimen kautta lähetetään palvelimelle väärennetty pyyntö, jonka palvelin luulee tulevan todennetulta käyttäjältä.

Opinnäytetyössä kehitettävä sovellus tulee käyttöön pienen yrityksen sisäverkkoon, joten ulkopuolisten hyökkäysten riski on pieni. Tämä ei kuitenkaan ole syy jättää tietoturvaa huomioimatta.

6.1 Ensimmäinen vaihe

Sovellus testattiin kahdessa vaiheessa ennen tuotantoa. Ensimmäinen vaihe koostui paikallisesta kehitysympäristössä, jossa aktiivista testausta suoritettiin kehityksen aikana. Tässä vaiheessa huomioitiin erityisesti sovelluksen yleinen toiminnallisuus eri selaimilla ja resoluutioilla sekä käyttäjän syötteiden tarkastus mahdollisten SQL-injektioiden varalta.

6.2 Toinen vaihe

Kehittäjä tekee usein oletuksia, miten sovellusta tulisi käyttää ja mitä syötteitä sille tulisi antaa. Tästä syystä seuraavassa vaiheessa luotiin uusi paikallinen testiympäristö kannettavalle tietokoneelle, joka vietiin yrityksen työntekijöiden testattavaksi. Tässä vaiheessa pyrittiin kitkemään viimeiset virheet järjestelmän toiminnallisuudesta ja syötteistä. Lisäksi toisessa vaiheessa tarkistettiin, kuinka hyvin järjestelmä vastasi vaatimusmäärittelyä ja varauduttiin tekemään muutoksia käyttäjien uusien toiveiden perusteella.

6.3 Tuotanto

Sovelluksen läpäistyä testauksen molemmat vaiheet, oli aika siirtää järjestelmä yrityksen sisäverkkoon. Sovellus ja Access-tietokannan viimeisin kopio siirrettiin yrityksen palveluntarjoajan palvelimelle.

Tuotantoon siirtymisen raskain vaihe oli tietokannan muuntaminen. Kehitysvaiheessa tietokannan taulujen ja sarakkeiden nimiin tehtiin useita muutoksia sekä viite-eheyksiä lisättiin, joten samat muutokset oli toistettava käsin. Koska toimenpide toteutetaan vain kerran, totesin manuaalisen työn olevan nopeampaa kuin automatisoivan skriptin kehittämisen.

7 YHTEENVETO

Olen erityisen kiinnostunut yritysten kustannustehokkuuden parantamisesta, joten opinnäytetyön aiheena työajanseurantajärjestelmä oli mielenkiintoinen ja palkitseva. Sovelluksen toteutus sujui mielestäni hyvin. Vaatimusmäärittelyn mukainen toiminnallisuus täyttyi ja vanha sovellus voitiin korvata.

Opin paljon uutta työhön valituista tekniikoista ja sovelluskehityksestä työn edetessä, tästä syystä esimerkiksi osa suunnittelusta tapahtui kehityksen lomassa. Tulevaisuudessa hyödyntäisin ehdottomasti avoimen lähdekoodin ohjelmistokehystä, joka tukee MVC-arkkitehtuuria. Lisäksi tehokkaan ohjelmointiympäristön pystyttäminen projektin alkuvaiheessa olisi säästänyt hermoja ja aikaa.

Järjestelmä luo hyvän pohjan yrityksen sähköisen toiminnan kehittämiseen. Sovellukseen voisi helposti lisätä esimerkiksi matka- ja kululaskujen käsittelyn.

LÄHTEET

/1/ Apache Friends, XAMP. Viitattu 24.4.2015.
https://www.apachefriends.org/index.html

/2/ Bootstrap, Bootstrap 3 released. Viitattu 23.4.2015.
http://blog.getbootstrap.com/2013/08/4/bootstrap-3-released/

/3/ Bootstrapbay, How to customize the Default Bootstrap Style, the Right Way. Viitattu 29.4.2015
https://bootstrapbay.com/blog/customize-bootstrap/

/4/ Cybercom Group, Web-sovelluksen testaus. Viitattu 30.4.2015.
http://www.cs.tut.fi/~testaus/s2011/luennot/Web-sovellusten%20testaus.pdf

/5/ DataTables, DataTables plug-in. Viitattu 23.4.2015.
https://DataTables.net/

/6/ Editor, Editing for DataTables. Viitattu 23.4.2015.
https://editor.DataTables.net/

/7/ Finlex, Työaikalaki. Viitattu 22.4.2015.
http://www.finlex.fi/fi/laki/ajantasa/1996/19960605

/8/ Github, Bootstrap 3 Datepicker. Viitattu 29.4.2015.
http://eonasdan.github.io/bootstrap-datetimepicker/Installing

/9/ Github, Sylvain Hamel Simple-expand. Viitattu 30.4.2015.
http://sylvain-hamel.github.io/simple-expand/

/10/ Merinova, Teknologiakeskus Oy Merinova Ab. Viitattu 22.4.2015.
http://www.merinova.fi/merinova/

/11/ MSDN Microsoft, Visual Basic User Authentication. Viitattu 28.4.2015.
https://msdn.microsoft.com/en-us/library/vstudio/3sd5bddt%28v=vs.100%29.aspx

/12/ MySQL, MySQL Documentation. Viitattu 24.4.2015.
http://dev.mysql.com/doc/

/13/ MySQL, MySQL Workbench reverse engineering. Viitattu 27.4.2015.
https://dev.mysql.com/doc/workbench/en/wb-reverse-engineering.html

/14/ PHP, PHP Documentation. Viitattu 24.4.2015.
http://php.net/docs.php

/15/ PHP, Sessions. Viitattu 28.4.2015.
http://php.net/manual/en/book.session.php

/16/ PHP, SQL injection. Viitattu 28.4.2015.
http://php.net/manual/en/security.database.sql-injection.php

/17/ Ubuntu documentation, ApacheMySQLPHP. Viitattu 24.4.2015.
https://help.ubuntu.com/community/ApacheMySQLPHP

/18/ W3C, CSS Specifications. Viitattu 23.4.2015.
http://www.w3.org/Style/CSS/current-work.en.html

/19/ W3C, HTML5. Viitattu 23.4.2015.
http://www.w3.org/TR/html/