TOIMINNANOHJAUSJÄRJESTELMÄN KEHITTÄMINEN

Tommi Halonen

Opinnäytetyö

Ammattikorkeakoulututkinto

Savonia
ammattikorkeakoulu

SAVONIA-AMMATTIKORKEAKOULU **OPINNÄYTETYÖ**
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Koulutusohjelma Tietotekniikan koulutusohjelma	
Työn tekijä(t) Tommi Halonen	
Työn nimi Toiminnanohjausjärjestelmän kehittäminen	
Päiväys 16.1.2011	Sivumäärä/Liitteet 29
Ohjaaja(t) Lehtori Jussi Koistinen, toimitusjohtaja Antti Puntanen	
Toimeksiantaja/Yhteistyökumppani(t) Dream On Networks Oy	

Tiivistelmä

Opinnäytetyöni aihe on toiminnanohjausjärjestelmän kehittäminen. Toteutin työn kehittämällä Markkinointitoimisto Johdin Oy:n toiminnanohjausjärjestelmää. Loin toiminnanohjausjärjestelmään kolme uutta lisäosaa: kalenterin, tehtävienhallinnan ja työajanseurannan. Dream On Networks Oy ja Markkinointitoimisto Johdin Oy olivat määritelleet kehitettävät osiot ja listanneet halutut ominaisuudet, joiden pohjalta toteutin opinnäytetyöni toiminnallisen osion.

Toiminnanohjausjärjestelmän valmis pohja oli luotu käyttäen PHP-ohjelmointikieltä ja järjestelmän tietokantamoottorina toimi MySQL. Tekemissäni osioissa käytin lisäksi JavaScriptiä, jQueryä sekä Ajax-tekniikkaa.

Kalenteri on tarkoitettu Johdin Oy:n työntekijöiden käyttöön, josta he voivat sopia tapaamisia, varata neuvottelutiloja ja seurata työtovereidensa menoja. Tehtävienhallinta on tarkoitettu työntekijän omien tehtävien seurantaan ikään kuin muistilistana, jonne voi lisätä kaikki omat tehtävät sekä merkitä ne tehdyiksi. Tehtävienhallinta on suoraan yhteydessä toiminnanohjausjärjestelmään tekemääni kolmanteen osioon, työajanseurantaan. Työajanseurannassa työntekijät voivat merkitä tehdyt työtunnit jokaiselle päivälle ja merkitä tunteja projekteille, tehtäville tai muille menoille, kuten koulutukselle tai talvilomalle. Työajanseurannassa on myös mahdollisuus luoda raportteja työntekijöiden työajoista projektipäälliköille ja palkanlaskijalle.

Kaikki kehittämäni toiminnanohjausjärjestelmän osiot ovat käytössä Markkinointitoimisto Johdin Oy:llä ja ne ovat helpottaneet työntekoa ja osoittautuneet erittäin käyttökelpoisiksi työkaluiksi, joka olikin tämän projektin tavoite.

Avainsanat
Toiminnanohjausjärjestelmä, PHP, JavaScript, jQuery, Ajax

SAVONIA UNIVERSITY OF APPLIED SCIENCES **THESIS**
Abstract

Field of Study Technology, Communication and Transport	
DegreeProgramme Degree Programme in Information Technology	
Author(s) Tommi Halonen	
Title of Thesis Upgrading an Enterprise Resource Planning System	
Date January 16, 2011	Pages/Appendices 29
Supervisor(s) Lecturer Jussi Koistinen, CEO Antti Puntanen	
Project/Partners Dream On Networks Oy	

Abstract

The subject of this project was to upgrade an existing ERP system with three new sections: calendar, assignment control and work time monitoring. Dream On Networks Oy and Markkinointitoimisto Johdin Oy had already specified these sections and therefore it was possible to start with made-up specifications.

The ERP system was created with a very popular PHP programming language and database engine, the widely used MySQL. When making these modules I also used JavaScript, jQuery and Ajax techniques. The calendar was made for the employees of Johdin Oy for controlling the schedules, making appointments and booking conference rooms. The assignment control section was made for supervising own work assignments of the employees who should mark their assignments and also mark them completed. This section was connected to the third section, work time monitoring where employees mark their working hours each day. The employee can mark hours for projects, assignments or some other reasons like trainings or winter holidays. Work time monitoring also enable reporting to project managers and for salary payment.

The project was finished as a functional package. All sections are in use and have made working easier and especially proved to be very useful tools.

Keywords
Enterprise Resource Planning System, ERP, PHP, JavaScript, jQuery, Ajax

SISÄLTÖ

1 JOHDANTO

Dream On Networks Oy on pieni internetohjelmointiin suuntautunut yritys, jossa olen työskennellyt syyskuusta 2009 lähtien tehden erilaisia ohjelmointitöitä. Töistä noin puolet on ollut ulkoasujen taittoa graafikon tekemistä suunnitelmista XHTML-muotoon ja toinen puoli internetpohjaisisten järjestelmien ohjelmoimista, kuten tässä opinnäytetyössä tekemäni toiminnanohjausjärjestelmän moduulit.

Markkinointitoimisto Johdin Oy on nimensä mukaisesti markkinointitoimisto, joka on erikoistunut kuluttajien aktivoimiseen näkyvillä toimintatavoilla, kuten kampanjoilla ja promootioilla erilaisissa tapahtumissa ja messuilla. Markkinointitoimisto Johdin Oy on perustettu vuonna 2002 ja sillä on noin 20 työntekijää sekä satoja kenttätyöntekijää ympäri Suomea.

Työntekijöiden, tapahtumien, kampanjoiden sekä monien muiden toimintojen hallinta ja valvonta vaatii tehokkaan ja nykyaikaisen ohjelmiston. Ennen aloittamaani kehitystyötä, Markkinointitoimisto Johdin Oy:llä käytössä ollut toiminnanohjausjärjestelmä sisälsi toimintoja ja tietoja, jotka olivat tarpeellisia markkinointiprojekteissa, tapahtumien luomisesta kenttätyöntekijöiden palkanmaksuun ja laskutukseen sekä raportointiin. Johdin Oy tarvitsi näiden lisäksi uusia toimintoja. Oma osuuteni oli luoda jo käytössä olevaan toiminnanohjausjärjestelmään uusiksi toiminnoiksi kalenteri, työajanseuranta ja tehtävienhallinta.

Ohjelmoimani kalenteri on viikkopohjainen järjestelmä ja jokaiselle käyttäjälle henkilökohtainen työväline. Sen ominaisuuksiin kuuluu oman ja muiden työntekijöiden kalenterien selaaminen, merkintöjen luominen ja muiden käyttäjien kutsuminen tapaamisiin. Kehittämäni työajanseurannan toimintoja ovat projektille tai työtehtävälle käytetyn ajan merkitseminen sekä oman työajan seuraaminen. Työajanseurannassa on myös mahdollista katsella raportteja esimerkiksi tietyssä projektissa mukana olevien henkilöiden ajankäytön osalta.

Ohjelmoimassani tehtävienhallinnassa lisätään tehtäviä itselle tai muille työntekijöille sekä aikataulutetaan ne. Tehtävienhallinnasta on helppoa seurata kunkin oman tehtävän etenemistä erilaisilla listaustavoilla. Tehtävienhallinnan tehtävien ajankäyttöä hallitaan työajanseurannasta, jonne lisätyt tehtävät tulevat näkyviin automaattisesti.

Uusi järjestelmä on huomattava parannus vanhempaan. Minkäänlaista yhteistä kalenteria ei Johdin Oy:llä ole ollut aiemmin käytössä. Jokainen työntekijä on huolehtinut omasta kalenteristaan ja sen merkinnöistä itse, olipa kalenteri sitten paperilla, matkapuhelimessa tai internetissä toimiva. Uudella kalenterilla on mahdollista tarkastella vaikka kaikkien työntekijöiden ajankäyttöä samasta paikasta.

Vanha työajanseuranta ja tehtävienhallinta ovat olleet jonkun muun kuin Dream On Networks Oy:n luomia internetpohjaisia järjestelmiä. Tekemässäni järjestelmässä myös näistä ominaisuuksista tuli toimivampia, kun esimerkiksi työtehtävistä ja työntekijöiden ajankäytöstä on saatavilla raportteja.

2 KÄYTETYT TEKNIIKAT

2.1 PHP

PHP on tehokas työkalu dynaamisten ja interaktiivisten web-sivujen luomiseen. PHP on todella suosittu, ilmainen ja tehokas vaihtoehto esimerkiksi Microsoftin ASP:lle. [1] PHP:n etuna on helppo syntaksi sekä vahva tyypittömyys muuttujissa, jolloin muuttujien alustusta ei välttämättä tarvita.

2.2 MySQL

MySQL on suosittu SQL-tietokannan hallintajärjestelmä, jonka on omistanut vuoden 2008 tammikuusta lähtien Sun Microsystems. Sun Microsystemsin taasen omistaa nykyisin Oracle Corporation, toinen tietokantajättiläinen. MySQL on saatavilla sekä vapaalla GNU GPL-lisenssillä sekä kaupallisella lisenssillä. [2]

2.3 JavaScript

JavaScript on pääosin käyttäjän selaimessa toimiva komentosarjakieli. Yksi JavaScriptin käytetyimpiä sovellutuksia on dynaamisen toiminnallisuuden lisääminen Web-sivuille. JavaScript mahdollistaa esimerkiksi onClick-tapahtumankäsittelijän, jolla voidaan poimia käyttäjän tekemiä hiiren painalluksia. [3]

2.4 Ajax

Ajax (Asynchronous JavaScript and XML) on tekniikka, jonka avulla voidaan siirtää tietoa selaimen ja palvelimen välillä ilman, että koko www-sivua täytyy ladata uudelleen. Ajaxin avulla sivun eri osia on mahdollista päivittää

itsenäisesti ja elementti kerrallaan, esimerkiksi jonkin uutiskommentin lähettämisessä ei tarvita uutta sivulatausta, pelkkä Ajax-kutsu riittää lisäämään kommentin tietovarastoon. [4]

2.5 jQuery

jQuery antaa web-kehittäjille helpon tavan luoda näyttäviä efektejä hyvin vähällä määrällä itse ohjelmointikoodin kirjoitusta. Koska jQuery on helppoa integroida omille sivuille, kasvaa sen suosio kokoajan. jQueryä on jokapuolella: muunmuassa Facebook ja Twitter käyttävät monia jQueryn toimintoja ja efektejä. [5]

jQueryn hyvä ystävä on jQueryUI, jossa on erilaisia käyttöliittymäkomponentteja, kuten dialogeja, välilehtiä tai päivämäärän valinta –komponentti. jQueryUI sisältää myös valmiita efektejä elementtien näyttämiseen tai piilottamiseen, kuten häivytys ja liu'uttaminen.

2.6 Ohjelmointiympäristö

Kehityspalvelimen ohjelmiston oli LAMP, eli Linux, Apache, MySQL sekä PHP. Palvelimen .htaccess-tiedosto on konfiguroitu käsittelemään palvelimen 404-virheet ja ohjaamaan näiden liikenteen index.php –sivulle. Eli kaikki sivut joita ei löydy, ohjautuvat index.php –sivulle. Index.php:ssa voidaan ottaa url-osoite talteen, ja ladata tarvittava sisältö sen avulla, lähes samalla tavalla kuin käyttämällä PHP:n $_GET-muuttujaa, mutta käyttäjän näkemä url-osoite on mahdollista näyttää huomattavasti siistimpänä, esimerkiksi www.esimerkki.fi/index.php?sivu=tehtavat&toiminto=lisaa voidaan näyttää muodossa www.esimerkki.fi/tehtavat/lisaa.

3 MÄÄRITTELY

Toiminnanohjausjärjestelmän pohja oli jo luotu aiemmin. Systeemiin on helppo kehittää uusia ominaisuuksia ja toimintoja sen modulaaripohjaisuuden vuoksi. Järjestelmän toiminta perustuu uudelleenohjaukseen Apache-palvelinohjelmiston asetustiedoston avulla, .htaccess-tiedosto tarkistaa löytyykö käyttäjän syöttämää sivua vai ei. Jos sivua ei löydy, ladataan palvelimelta index.php -sivu, joka lataa url-osoitteessa mainitut tiedostot. Mikäli osoitteessa on tiettyjä sanoja, kuten tallenna tai hae, ladataan palvelimelta pelkkä tiedosto. Muissa tapauksissa ladataan myös järjestelmän ulkoasu, tarvittavat PHP-luokat sekä muut mahdolliset skriptit, kuten JavaScript-tiedostot. Järjestelmän osoitteet ovat muotoa toiminnanohjausjärjestelmä.fi/moduuli/toiminto/. Tässä tapauksessa index.php lataisi ulkoasun sekä muut tarpeelliset tiedostot. Kesken ulkoasun luomisen ladataan moduuli.php -niminen tiedosto. moduuli.php -tiedostosta löytyy moduuli-niminen luokka ja toiminto olisi tämän luokan metodi.

```
moduuli.php
classmoduuli
{
          function toiminto()
          {
          // suoritettava koodi
          }
}
```

Esimerkiksi tekemääni tehtävienhallinnan tehtävän lisäämissivun osoite olisi toiminnanohjausjärjestelmä.fi/tehtavat/lisaa/.

```
tehtavat.php
class tehtavat
{
      ...
      function lisaa()
      {
                ...
      }
      ...
}
```

Tällainen tekniikka mahdollistaa helpon kehityksen ja ominaisuuksien lisäämisen järjestelmään. PHP-luokkien metodeihin voi myös kirjoittaa esimerkiksi JavaScript- tai jQuery-koodia, joilla saadaan luotua näyttävämpiä ja toimivampia käyttöliittymiä, koska sivuja ei välttämättä tarvitse aina ladata uudestaan.

Käytössä olevaa tietokantakoneistoa hallitaan MySQL-ohjelmiston avulla ja sille on tehty valmiiksi toimintoja, jotka kattavat kaiken tarpeellisen: hakemisen lisäyksen, päivityksen ja poistamisen. Lisäykselle ja päivittämiselle on valmiit metodit, jotka ottavat parametreina taulukon sekä tietokannan taulun nimen. Molemmat funktiot vertailevat parametrina tulleen taulukon avaimia toisena parametrina annetun taulun sarakkeiden nimiin. Eli aina kun taulukosta löytyvä avain löytyy myös tietokannan taulun sarakkeiden nimistä, tapauksesta riippuen lisätään sinne uusi rivi tai päivitetään vanhaa riviä.

Lisäämisen tai päivittämisen ratkaisee, onko parametrina tulevassa taulukossa id-nimistä avainta. Jos avain löytyy ja sillä on jokin numeerinen arvo, on kyseessä päivitys. Jos taas avainta ei löydy tai se on jotain muuta kuin numero, tehdään tauluun uusi rivi. Tietokantajärjestelmän tavoitteena on luoda jokaiseen tauluun id-niminen sarake, joka on taulussa perusavain (primary key). Näin ollen lomakkeen käsittelijä voi helposti tutkia lomakkeelta tulevasta datasta, onko kyseessä uuden rivin lisäys vai vanhan päivitys.

```
//$_POST-taulukko
Array
{
      [nimi] => Teppo Testaaja
      [osoite] => Testikuja 8
      [kengannumero] => 43
}

//taulun “tiedot” sarakkeiden nimet tietokannassa
//id | nimi | osoite

//Lisäyksen koodi
lisays($_POST, “tiedot”);
```

Ylläolevassa esimerkissä lisättäisiin tiedot-tauluun uusi rivi, koska id-nimistä avainta ei ole. Tauluun lisättäisiin vain nimi ja osoite, koska kengannumero-nimistä saraketta ei tietokannan taulusta löydy.

3.1 Kalenteri

Markkinointitoimisto Johdin Oy halusi viikkopohjaisen, puolen tunnin tarkkuudella toimivan kalenterin seuraavilla ominaisuuksilla:

- merkinnän lisäys itselle tai valituille henkilöille
- merkintöjen kategoriat (kokous, muu, yksityinen)
- oman tai valittujen henkilöiden kalenterien selaaminen
- omien tai valittujen henkilöiden kalenterien merkintöjen selaaminen ja tarkasteleminen
- merkintää lisättäessä valituille henkilöille, lähetetään kaikille sähköposti, josta merkinnän voi joko hyväksyä tai hylätä (vain kaikkien osallistujien hyväksymät merkinnät näytetään, muutoin merkintä näytetään hyväksymättömänä)
- pieni navigointikalenteri, jonka avulla selataan isompaa kalenteria.

3.2 Tehtävienhallinta

Johdin Oy toivoi tehtävienhallintaan mahdollisuutta lisätä tehtäviä järjestelmässä jo oleviin projekteihin. Uudelle tehtävälle annettiin vaaditut tiedot:

- projekti, jolle esimerkiksi Googlesta tuttu haku, joka näyttää hakukentän alla hakua vastaavat tulokset
- otsikko eli tehtävän nimi ja selite, jos tehtävään liittyy jotain mikä ei välttämättä otsikosta käy ilmi tai vaatii tarkennusta
- määräaika (deadline) sekä arvio tehtävään kuluvista työtunneista
- tehtävän tekijä, pudotusvalikossa listattuna kaikki työntekijät ja oletuksena on valittuna kirjautunut työntekijä, eli tehtävän luoja (työntekijöille siis mahdollisuus delegoida tehtäviä esimerkiksi kollegoille)
- lisäksi näkymättömänä ominaisuutena tehtävän status (uusi, kesken, valmis) sekä näille kaikille oma aikaleima

Tehtävienhallintaan kuuluu myös tehtävien selaaminen ja muokkaaminen. Tehtävien listaukseen kuului muutama erilainen listaustyyli:

- puunäkymä, jossa näkyy projektit listattuna, ja jokaisen projektin alla kuhunkin projektiin liitetyt tehtävät deadlinejärjestyksessä, jotka on määritelty kirjautuneelle käyttäjälle.
- projektin otsikkoon tehtävän lisäys -nappi, jolloin ylempänä listatussa uuden tehtävän lisäyksessä on valittu projekti jo valmiina.
- listaus myöhässä olevista tehtävistä
- listaus tehtävistä deadlinejärjestyksessä
- valmiit tehtävät, jossa myös mahdollisuus hakea halutun aikavälin mukaan.

Lisäksi kolmessa ensimmäisessä listaustyylissä on mahdollisuus merkitä tehtävä aloitetuksi, jolloin sen status muuttuu keskeneräiseksi tai merkitä keskeneräinen tehtävä valmiiksi. Näistä toiminnoista otetaan myös sen hetkinen aikaleima tietokantaan.

3.3 Työajanseuranta

Työajanseurantaan Johdin Oy halusi seuraavat ominaisuudet:

- mahdollisuus hakea projekteja samalla tavalla kuin tehtävienhallinnassa, eli hakusanaa vastaavat tulokset tulee näkyville kirjoitettaessa hakukentän alapuolelle
- valitulle projektille on mahdollista syöttää työtunteja valitulle päivälle
- projektihakukentän alapuolelle listaus käyttäjän omista tehtävistä, listassa vasemmalla tehtävä ja oikealla kenttä, johon syötetään tunnit valitulle päivälle
- näiden kahden alapuolelle oletuksena suljettu listaus (eli vain linkki näkyvillä, josta lista aukenee), jossa on näkyvillä kaikkia muut tekemiset esimerkiksi vuosiloma, koulutus tai jokin henkilökohtainen syy
- syöttömahdollisuuksien viereen seurantapalkki, jossa ylimpänä näkyvillä valitun viikon, kuukauden ja vuoden tehdyt tunnit.

- tehtyjen tuntien alla pieni kuukausikalenteri, jossa näkyy myös tehdyt tunnit visuaalisemmin. (Vihreä taustaväri päivän kohdalla tarkoittaa, että tunteja on sille päivälle jo vaadittu määrä, keltainen taustaväri taasen tarkoittaa että työpäivän tunnit ovat puutteelliset, lisäksi kalenteri toimii myöskin valitsimena)
- kalenterin alapuolelle näkyviin valitun päivän tehdyt tunnit yksinkertaisessa listausmuodossa; vasemmalla projekti, tehtävä tai muu syy ja sen vieressä tehdyt tunnit (tästä listauksesta on myös mahdollista poistaa tehtyjä tunteja)

3.4 Yleisesti

Markkinointitoimisto Johdin Oy:n asettamat määritykset toiminnanohjausjärjestelmän suhteen olivat selkeät ja yksinkertaiset, joten päätin toteuttaa ne kaikki. Mitään erityisen vaikealta tuntuvaa ominaisuutta ei itselleni tässä joukossa ollut. Ulkoasun määrittely oli pääosin tarpeetonta, koska kaikki nämä moduulit tulivat aiemmin mainuttuun osittain valmiiseen toiminnanohjausjärjestelmään, jonka ulkoasu oli jo tehty. Minun tehtäväkseni tuli ratkaista ulkoasun osalta esimerkiksi kalenterin merkintöjen tyylit ja eri komponenttien, kuten listojen ja lomakkeiden koot ja sijainnit.

Teknisen totetuksen suhteen asiakas oli avoin, joten mitään määrityksiä ei siltä osin tehty. Sain itse päättää, käytänkö pelkkää PHP:ta vai toteutanko toiminnot käyttäen PHP:n apuna JavaScriptiä. Koska kyseessä oli rajoitettu käyttäjäkunta, jossa kaikilla oli nykyaikainen internetselain (Firefox tai Safari), joten JavaScriptin käyttö ei ollut ongelma.

Ulkoasuja ja käyttöliittymiä kehittäessä on ikävää huomata, kuinka jokin elementin reunus tai siirtoefekti ei toimikaan Internet Explorer 7:lla tai 8:lla, puhumattakaan Internet Explorer 6:sta, jota monet yritykset (onneksi lähinnä valtion ja kunnan yritykset) käyttävät vieläkin. [6 ja 7] Yleensä Internet Exploreria varten on ohjelmoitava omat CSS- sekä javaScript-tiedostot. Tässä projektissa ohjelmointiprosessi helpottui, koska toimintojen ei tarvinnutkaan toimia millään Internet Explorerin versiolla.

4 SUUNNITTELU

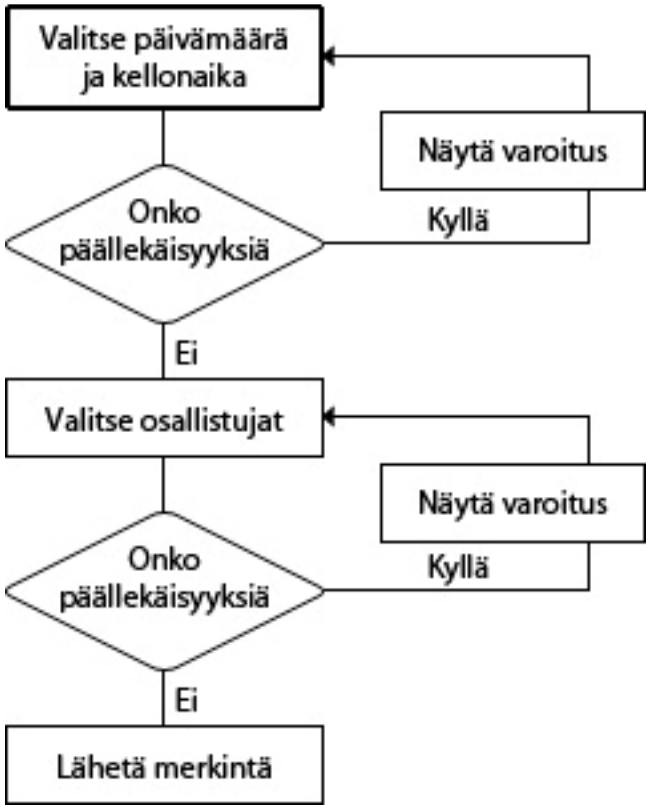
4.1 Kalenteri

Jokaisen moduulin suunnittelun aloitin miettimällä ulkoasua eri komponenttien ja elementtien sijaintien suhteen. Kalenterin suhteen ratkaisu oli helppo, koska viikko- ja kuukausikalenterin perusilme on pysynyt samana pitkään ja käynyt jokaiselle tutuksi. Hyvää ja toimivaa kaavaa on turhaa lähteä keksimään uudestaan. Koska pääasiallinen käyttö kohdistuu viikkokalenteriin, oli siitä saatava mahdollisimman suuri ja selkeä. Pientä kuukausikalenteria käytetään vain suuremmassa kalenterissa näytettävän aikavälin ohjaamiseen sekä yleistarkasteluun valitun kuukauden merkintöjen osalta.

Päätin toteuttaa kalenterin toiminnan täysin JavaScriptillä, jotta siitä saataisiin mahdollisimman nopeakäyttöinen, näin vältyttäisiin turhilta sivulatauksilta esimerkiksi kalenterin aikavälin vaihtamisessa tai uuden merkinnän luomisessa. Erilaisia valmiita kalenteriratkaisuja tutkiessa, törmäsin jälleen kerran jQueryyn, jota en ollut tuolloin vielä juurikaan käyttänyt. jQuerylle on tarjolla suuri määrä kolmannen osapuolen kirjoittamia lisäosia (plugin). Lisäosia tutkittuani löysinkin jQuerylle tehdyn viikkokalenterin. Sovellus oli sellainen mitä olin itsekin suunnitellut, mutta se ei tukenut usean käyttäjän merkintöjen yhtäaikaista näyttämistä. Muita sopivia valmiita ratkaisuja en löytänyt, joten kalenteri oli luotava itse alusta alkaen.

Asetuin asiakkaan asemaan miettiäkseni mahdollisimman käyttäjäystävällistä ratkaisua. Merkintöjen osalta päädyin ratkaisuun, jossa kalenterissa on mahdollista maalata haluttu aikaväli, jolloin merkinnän alku- ja loppuaika päivämäärän kera tulee automaattisesti merkinnänluomislomakkeeseen. Googlen kalenteria käyttäessäni olen huomannut siinä käytettävän samanlaista ajankohdan valintatapaa. jQueryn todella selkeää ja kattavaa dokumentaatiota tutkittuani huomasin, että samankaltainen maalaustekniikka ei ole vaikeaa toteuttaa, joten päätin toteuttaa merkinnän luomisen maalaamalla.

Kun haluttu ajankohta on maalattu, avautuu pienen navigointikalenterin alle lomake, jossa kysytään merkinnän tiedot: nimi, kuvaus, paikka, tyyppi ja osallistujat. Jos osallistujia valitaan, heille lähetetään sähköposti, jonka kautta he pystyvät joko hyväksymään tai hylkäämään merkinnän.



Kuva 1 Merkinnän tekemisen kuvaus.

Myöhemmin kalenterimerkintöihin haluttiin lisää ominaisuuksia, joten muutin lomakkeen aukeamaan kalenterin päälle modaalisena dialogina, jolloin sivun valmista ulkoasua ei tarvinnut muokata. Tällöin pääosin kaikki kalenterinäkymän komponentit mahtuivat kerralla ruudulle.

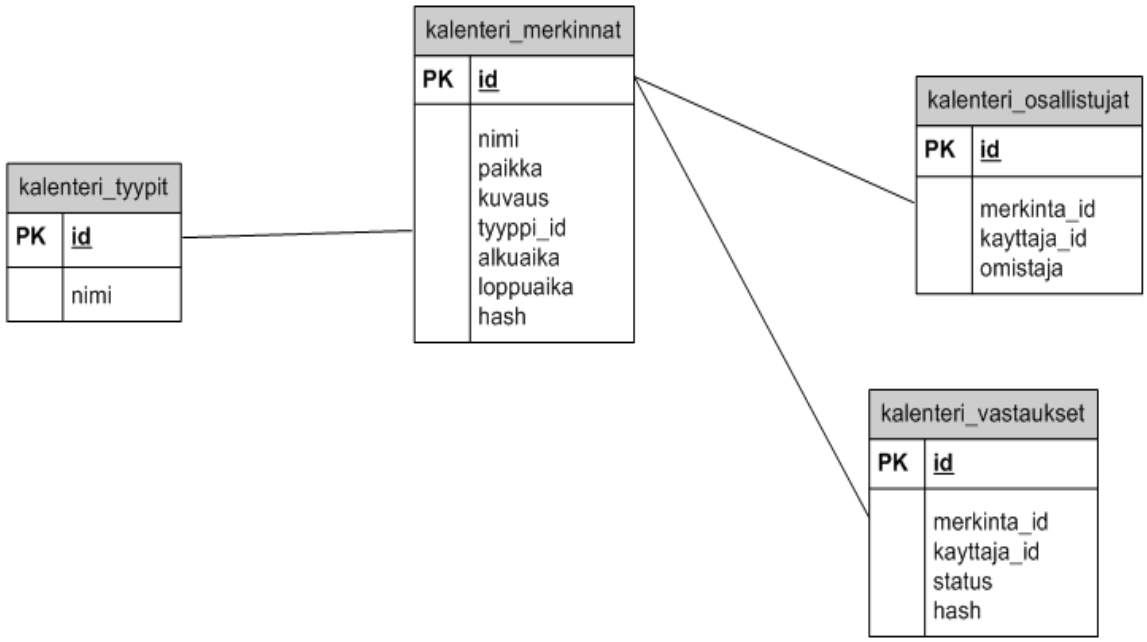
Kalenterin tietokantaan suunnittelin tarvitsevani neljä uutta taulua: merkinnöille, merkintöjen tyypeille, osallistujille sekä sähköpostitse lähetettäville pyynnöille.

```
kalenteri_merkinnat:
- id, INT(11) PRIMARY KEY AUTO_INCREMENT
- nimi, TEXT
- paikka, TEXT
- kuvaus, TEXT
- tyyppi_id, INT(11)
- alkuaika, INT(11)
- loppuaika, INT(11)
- hash, VARCHAR(128)

kalenteri_tyypit:
- id, INT(11) PRIMARY KEY AUTO_INCREMENT
- nimi, TEXT

kalenteri_osallistujat:
- id, INT(11) PRIMARY KEY AUTO_INCREMENT
- merkinta_id, INT(11)
- kayttaja_id, INT(11)
- omistaja, TINYINT(1)

kalenteri_vastaukset
- id, INT(11) PRIMARY KEY AUTO_INCREMENT
- merkinta_id, INT(11)
- kayttaja_id, INT(11)
- status, TEXT
- hash, TEXT
```

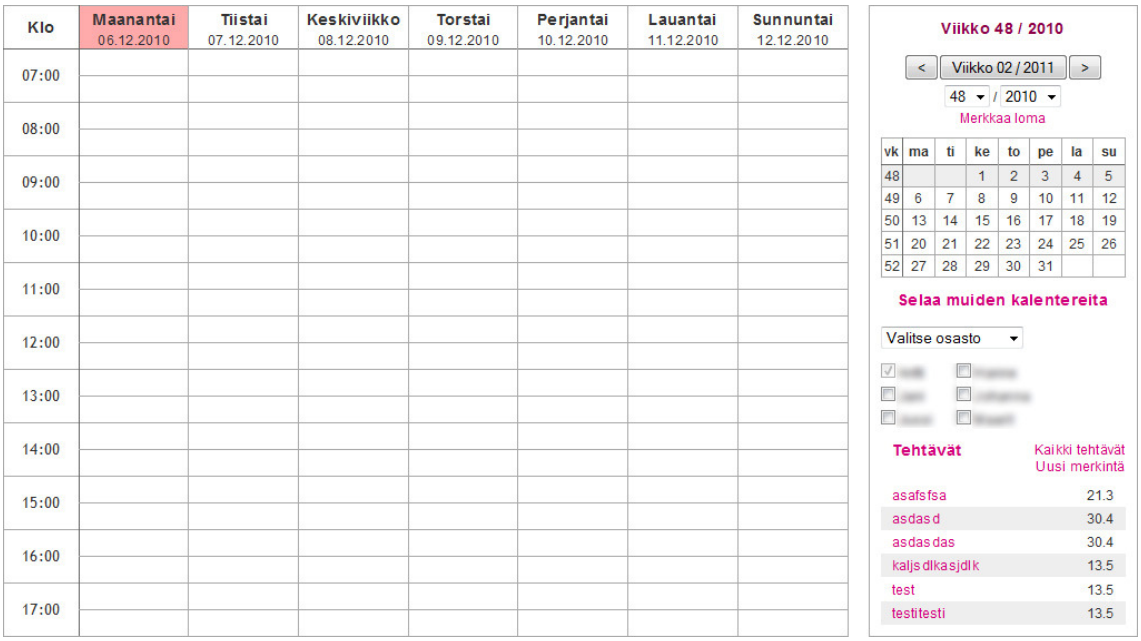


Kuva 2 Kalenterin tietokantakuvaus

Tietokannan kalenteri_osallistujat-taulun omistaja-sarake määrittää merkinnän omistajan, periaatteessa merkinnällä voisi olla monta omistajaa, mutta

käytännössä näin ei ole. Mitään erikoista syytä ei tosin ole, miksei omistajaa merkitä kalenteri_merkinnat-tauluun.

Kun kalenteriin tehdään merkintä, lähetetään kaikille osallistujille - lukuunottamatta omistajalle - sähköposti, josta käyttäjä voi joko hyväksyä tai hylätä kalenteripyynnön. Käyttäjän hyväksyessä pyynnön tarkistetaan kalenteri_vastaukset-taulusta muiden käyttäjien vastaukset. Kun merkintä tehdään, lisätään tauluun kaikki käyttäjät sekä niiden statukseksi ”odottaa”. Jos käyttäjä hyväksyy pyynnön ja kyseisellä merkinnällä on vielä statuksia, jotka ovat odotustilassa, jätetään merkintä keskeneräiseksi. Jos taas käyttäjä on kalenteripyyntöä hyväksyessä viimeinen hyväksyjä, muutetaan merkintä hyväksytyksi. Sen jälkeen hyväksytty merkintä näytetään kaikille kyseisen kalenterimerkinnän käyttäjille heidän henkilökohtaisessa kalenterissaan.



Kuva 3 Kalenterin päänäkymä, ilman merkintöjä. Myös pyhäpäivät haluttiin kalenteriin, näkyvillä on onkin itsenäisyyspäivä, hiiren mennessä pyhäpäivän päälle, näkyy kyseisen pyhän nimi.

4.2 Tehtävienhallinta

Koska tehtävien listaukseen annettiin hyvät ja kuvaavat määritelmät, oli niiden suunnittelukin helppoa. Käytännössä kaikki erilaiset näkymät olivat ulkoasullisesti toistensa kopioita, vain listauksen järjestys vaihteli.

Uuden tehtävän lisäämisessä tuli vastaan minulle ennestään tuntematonta ohjelmoitavaa: nykyisin useissa internetpalveluiden hakukentissä oleva itseään täydentävä haku. Tällöin hakukenttään kirjoitettaessa haettiin mahdolliset löytyneet tulokset hakukentän alle ilmestyvään listaukseen. Tähän toteutukseen löytyi puolivalmis koodi, johon täytyi muokata tietokantahaku, joka onnistui helposti käyttämällä SELECT-lausekkeen LIKE-ehtoa. Hakukentän alle ilmestyvään tuloslistaan tarvittiin lisäksi ominaisuus, jolla listasta sai valittua halutun tuloksen. Tähän tarpeeseen sopiva oli jQueryllä toteutettava click-tapahtumankäsittelijä, jolla saadaan poimittua käyttäjän hiiren painallus sekä html-elementti, jota painetaan.

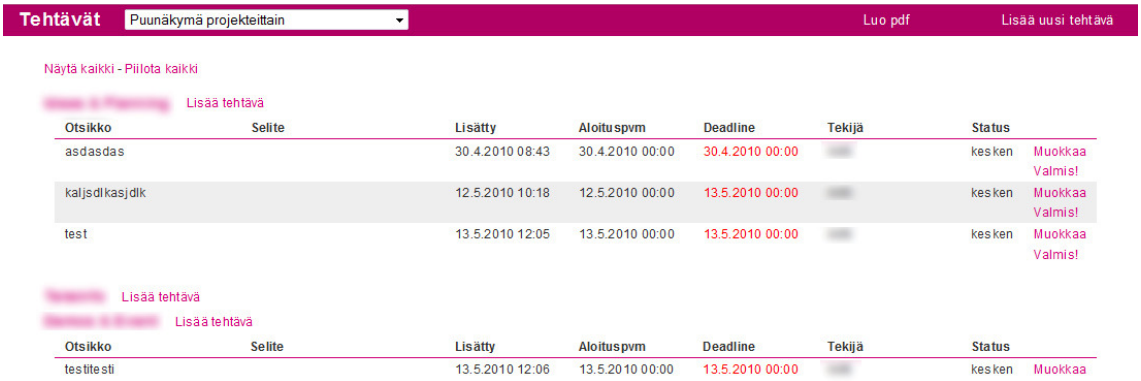
Projektin valinnan lisäksi lisäsin dynaamista toiminnallisuutta lisäämällä päivämääräkenttiin jQueryUI:n valmiin datepicker-lisäosan. Datepicker liitetään tekstikenttään ja sen aktivoituessa avautuu pienikokoinen kuukausikalenteri, josta valitaan haluttu päivämäärä. Näin annettu päivämäärä saadaan aina oikeaan muotoon. Lisäksi käyttäjien tekemien mahdollisten virhepainallusten määrä pienenee, joka auttaa esimerkiksi lomakkeiden tarkistamisessa. Tässä tapauksessa tarvitsee vain tarkistaa, että päivämääräkenttä ei ole tyhjä.

Tehtävienhallinta tarvitsi vain yhden uuden taulu tietokantaan, nimeltään tehtavat:

```
tehtavat
- id, INT(11) PRIMARY KEY AUTO_INCREMENT
- projekti_id, INT(11)
- otsikko, TEXT
- selite, TEXT
- aloitusaika, INT(11)
- deadline, INT(11)
- tuntiarvio, DECIMAL(10,2)
- tekija_id, INT(11)
- status, TEXT
- laittoaika, INT(11)
- valmisaika, INT(11)
```

Uuden tehtävän lisäämisessä tallennettiin kaikki tiedot, mikäli dataa oli syötetty. Poikkeuksena kuitenkin valmisaika-kenttä, johon päivitettiin aikaleima vasta kun tehtävä merkittiin valmiiksi.

Listaustapoja oli alun perin neljä erilaista. Ensimmäinen listaustapa on ”puunäkymä projekteittain”, jossa listataan kaikki halutun työntekijän projektit ja projektin tehtävät. Tähän näkymään myös haluttiin kunkin projektin kohdalle linkki, josta pääsee lisäämään uuden tehtävän kyseiselle projektille. Eli normaali tehtävänlisäyslomake, jossa projekti on valmiiksi valittu.



Kuva 4 Tehtävienhallinnan puunäkymä.

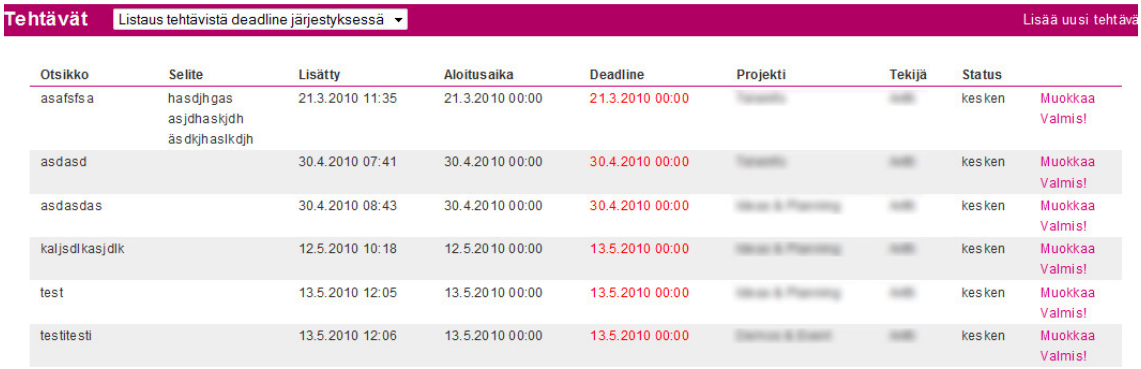
Loput kolme listausta olivat lähes samanlaisia ulkoasultaan, vain järjestystapa ja hakuehto muuttuivat. Näistä kolmesta yksi oli listaus myöhässä olevista projekteista, jossa näytetään vain tehtävät joiden määräaika on jo mennyt. Toinen listaus oli tehtävistä deadlinejärjestyksessä, jolla haetaan kaikki tehtävät ja listataan ne deadlinen mukaan. Viimeinen listaus oli valmiiden tehtävien listaus, jossa haetaan statukseltaan valmiiksi merkityt tehtävät. Valmiiden tehtävien listaukseen haluttiin myös aikavälihaku valitsemalla alku- ja loppupäivämäärä. Päivämäärien valintaan sopi jo tehtävän lisäyksessä käytetty jQueryUI:n datepicker.

Markkinointitoimisto Johdin Oy:n työntekijöiden jo käyttäessä tehtävienhallintaa, ilmeni tarve uudelle listausnäkymälle. Projektipäälliköille tarvittiin mahdollisuus nähdä kaikki projektiin osallistuvien työntekijöiden tehtävät. Käytännössä tämä oli laajennettu puunäkymä. Päätin tehdä tästä uuden kokonaan uuden näkymän, joka näkyisi vain projektipäälliköille.

Uudessa näkymässä listataan kaikki projektit, joissa järjestelmään kirjautunut projektipäällikkö on projektin vetäjänä. Listauksen väliotsikoiksi tulee projektien nimet ja näiden väliotsikoiden alle listataan kunkin kyseiseen projektiin kuuluvan työntekijän tehtävät. Listaus oli siis hyvinkin samantapainen kuin aiemmin kertoman ”puunäkymä projekteittain” –listaus.

Lisäksi uuteen listaukseen tarvittiin yksi hakuoptio, jolla voidaan hakea vain kesken olevat tehtävät. Jos optiota ei ole valittu, haetaan sekä kesken olevat että valmiit tehtävät. Tällaisen valinnan tekemiseen sopi parhaiten checkbox-nappi, jota painamalla listaus päivittyy suoraan ilman sivun uudelleen latausta. jQueryn .is-metodilla checkbox-napin tilan selvittäminen oli helppoa:

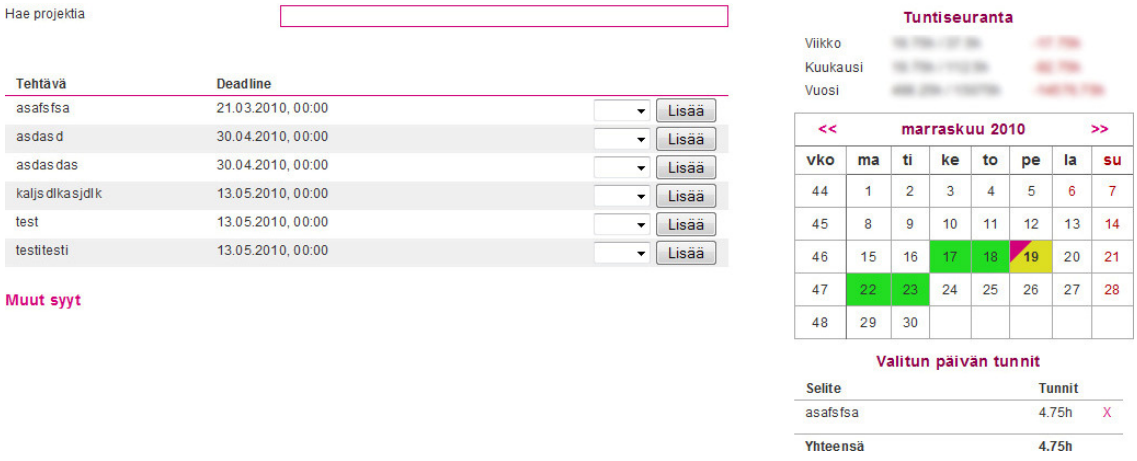
```
if($("checkbox").is(":checked"))
{
      // nappi on valittu
}
```



Kuva 5 Tehtävienhallinnan deadline-näkymä. Muut näkymät ovat samalla tyylillä tehty, vain osa sarakkeista muuttuu.

4.3 Työajanseuranta

Työajanseurannan suunnitteluun käytin enemmän aikaa kuin kalenterin ja tehtävienhallinnan suunnitteluun. Tiesin jo suunnitteluvaiheessa, että ruudulle tulisi paljon näytettävää, joten selkeä ja tarkasti jaoteltu ulkoasu oli tarpeen. Koska työajanseurannan tärkeimmän osan, kalenterin, värimaailma oli muuhun ulkoasuun verrattuna selvästi erottuva, oli mahdollista jättää kalenterin ja sen ylä- ja alapuolelle tulevat tuntiseurantalistat noin puolta pienemmäksi kuin sivun vasemman puolen eli työtuntien lisäämisen.



Kuva 6 Työajanseurannan päänäkymä

Projektien työtuntien lisäämiseen haluttiin samanlainen projektinhakemistoiminto kuin tehtävienhallinnan uuden tehtävän lisäyksessä, tarvittava koodi oli siis jo valmiina. Muutoksia täytyi kuitenkin tehdä projektin valitsemiseen: valittu projekti täytyi saada näkyviin erilaiseen muotoon ja täysin eri nimisen html-elementin sisällöksi. Ratkaisin ongelman muotoilemalla mahdollisten hakutulosten listausta. Hakutulosten listauksessa, projektin nimeä hiirellä painettaessa, kopioidaan projektin uniikki id-numero sekä projektin nimi jQueryllä talteen. Id-numero laitetaan piilotettuun input-kenttään, projektin nimi laitetaan näkyville sekä lisätään sen perään tuntimääräkenttä ja tallennuspainike.

Projektihaun alapuolella on listattuna käyttäjän tehtävät samalla tavalla esitettynä, kuten mahdollisesti haettu projekti. Tehtävien listauksen alla on listattu muut mahdolliset syyt, joiden takia työtunteja voi mahdollisesti tulla. Esimerkiksi talviloma, koulutus tai yrityksen oma tapahtuma. Myös nämä muut syyt on listattu samalla tavalla kuin projektit ja tehtävät. Syyn nimen vierellä on tuntimääräkenttä sekä tallennuspainike.

Tuntien lisääminen toimii Ajax-kyselyllä, jolloin vältetään turha sivunlataus. JavaSriptillä Ajax-kysely aloitetaan luomalla selaimesta riippuen joko ActiveXObject (Internet Explorer)- tai XMLHttpRequest-tyyppinen (kaikki muut selaimet) olio, jonka jälkeen valmistellaan olio laittamalla sille parametrit sekä kutsuttavan sivun palauttaman vastauksen käsittelemistoiminnot. Tällainen vie useita rivejä. jQuery hoitaa asian lyhyemmin; helpoimmillaan se vie vain yhden rivin. jQuerylle annetaan elementin tunniste johon tiedot haetaan ja

kutsutaan load-metodia, jolle ainoa pakollinen parametri on url-osoite, josta tieto halutaan hakea.

```
$(elementti).load(”taalta_ladataan.php”);
```

Lisäparametreinä load-metodille voi antaa PHP:lle menevän $_POST-taulukon sisällön ja callback-metodin, jota kutsutaan kun tiedot on onnistuneesti noudettu. Tämän callback-metodin parametriksi voi laittaa muuttujan, johon kutsutun sivun tulostamat tiedot palautuu.

Koska load-metodi korvaa valitun elementin sisällön haetulla tiedolla, se ei toimi jokaisessa tilanteessa. Post-metodi toimii samalla tavalla, mutta sitä ei liitetä mihinkään elementtiin, joten palautuva tieto on vapaammin käsiteltävissä. Käytännössä metodi toimii myös siten, että palautettavaa tietoa ei käsitellä mitenkään. Tällaisesta toiminnosta on hyötyä, kun halutaan esimerkiksi vain päivittää tietokannan tietoja.

Työajanseurannan tapauksessa käytin metodia kuitenkin molempiin suuntiin. PHP-sivu, jota kutsuin, hoiti ensin tietojen päivityksen, jonka jälkeen se palautti päivitetyt tiedot takaisin pyynnön lähettäneelle jQuery-metodille. jQueryn post-metodi otti palautuneet tiedot ja päivitti sivulla olevia html-elementtejä.

```
$.post(”taalta_ladataan.php”, { id: 2, nimi: ”Matti”
}, function(d) {
     // muuttuja d sisältää palautuneen datan.
     $(elementti).text(d);
});
```

Yllä oleva esimerkki hakee dataa ”taalta_ladataan.php” –sivulta ja vie sinne $_POST-taulukon, jossa on sarakkeet id sekä nimi sekä niiden arvot ”2” ja ”Matti”. PHP:n puolella tietoja voidaan esimerkiksi päivittää tietokantaan. Näiden jälkeen jQuerylle palautuu kaikki tiedot, mitä PHP:lla tulostetaan, jokseenkin kankeassa tekstimuodossa.

5 OHJELMOINTIVAIHE

Ohjelmointivaihe oli sujuvaa ja helppoa jQueryn ominaisuuksien ansiosta. jQueryllä lähes kaiken datan lataaminen hoituu helposti ilman erillisiä lisäys- tai muokkaussivuja. Asian kääntöpuolena on kuitenkin PHP:lla kirjoitettavat datanhakemismetodit. Koska jQuery hakee tiedot käyttäen Ajaxin kautta HTTP-pyyntöjä, oli kirjoitettava useita erilaisia php-skriptejä lataamaan tietoja tietokannasta sekä palauttamaan ne oikeassa muodossa takaisin jQuerylle. Kalenteria varten tein erillisen tiedoston, jossa on lähes tuhat riviä ohjelmointikoodia pelkkään tiedon hakemiseen ja esittämiseen.

Koska jQuery on JavaScript-kirjasto, on sen lähdekoodi nähtävillä kaikille selaimen avulla, toisin kuin palvelimella suoritettavat esimerkiksi PHP- tai ASP-koodit. Tietoturvan kannalta olisi siis erittäin suuri riski tehdä yleiskäyttöinen JavaScript- tai jQuery-metodi, johon voisi parametrilla antaa suoraan tietokannan taulun nimen sekä lisättävät, päivitettävät tai poistettavat.

Kalenterin ohjelmoinnin jälkeen huomasin, että jQueryllä voi hakea myös JSON-muotoista (JavaScript Object Notation) dataa. JSON on nimensä mukaisesti olio-tyyppistä dataa, jolloin sitä voidaan käydä läpi yksinkertaisemmin, verrattuna pelkkään tekstimuotoiseen dataan. Myös PHP sisältää valmiit metodit taulukon kääntämiseksi JSON-muotoon.

```
PHP:
$taulukko[0]["nimi"] = "matti";
$taulukko[0]["pituus"] = 180;
$taulukko[1]["nimi"] = "Liisa";
$taulukko[1]["pituus"] = 165;
json_encode($taulukko);
...

jQuery:
$.getJSON("testi.php", { parametri: hae_taulukko },
function(taulukko) {
     for(var i = 0; i <taulukko.length; i++) {
               var nimi = taulukko[i].nimi;
               var pituus = taulukko[i].pituus;
               alert("Nimi:      "+nimi+",      pituus:
"+pituus);
     }
});
```

JSON-muotoa käytettäessä pystyin yksinkertaistamaan datan tulostuksen muotoilua PHP:n puolella. Esimerkiksi lomakkeen tietojen hakeminen ei muuttunut, mutta tietojen tulostaminen lomakkeelle helpottui. Aiemmin palautin PHP:lla tiedon tekstimuodossa, jossa tiedot oli eroteltu jollain erikoismerkillä, esimerkiksi putkella tai alaviivalla. jQueryllä teksti täytyi pilkkoa osiin ja lisätä saadut osat paikoilleen esimerkiksi lomakkeeseen. JSON:n avulla tietojen tulostaminen oli huomattavasti yksinkertaisempaa.

Suurimmaksi ongelmaksi muodostui järjestelmän ominaisuuksien lisääntyminen kesken ja myös projektin jälkeen. Koin koodin muokkaaminen ja ominaisuuksien lisäämisen hankalaksi, mikä saattaa johtua siitä ettei minulla ollut juurikaan aiempaa kokemusta näin laajoista ohjelmointikokonaisuuksista. Erityisesti kalenterin muokkaaminen on erittäin haastavaa. Ominaisuuksia on paljon, eikä esimerkiksi lomakkeissa ole juuri yhtenäisyyttä. Ohjelmointiprosessia helpottaisi jos kaikki halutut ominaisuudet olisivat tiedossa enne työn aloittamista. Käytännössä tämä on kuitenkin harvoin mahdollista.

6 TESTAUS JA KÄYTTÖ

Testasin toiminnanohjausjärjestelmää samaan aikaan sitä tehdessäni. Kehitin järjestelmää eri palvelimelle kuin missä varsinainen käytössä oleva toiminnanohjausjärjestelmä oli. Kehityspalvelin oli konfiguroitu vastaamaan varsinaista käyttöpalvelinta. Myös tietokanta oli kopioituna, joten tietokantavahingot oli minimoitu, minkä lisäksi tietokannoista oli varmuuskopiointi kerran päivässä. Erillisen kehitysympäristön käyttö on erityisen kannattavaa, koska vahingon sattuessa mitään tärkeää ei pääse hajoamaan tai häviämään.

Kun sain järjestelmän uudet osat määrittelyn mukaiseksi, siirrettiin ne varsinaiselle käyttöpalvelimelle. Markkinointitoimisto Johdin Oy:n työntekijät alkoivat vähitellen käyttämään uusia moduuleita. Käyttöönotossa ei sattunut suuria virheitä. Koska toiminnanohjausjärjestelmä on moduulipohjainen, ei uusien ominaisuuksien lisääminen aiheuttanut muuta, kuin uusien linkkien lisäämisen ulkoasuun.

Pyysin Markkinointitoimisto Johdin Oy:n työtekijöiltä palautetta tekemistäni moduuleista sen jälkeen, kun heillä oli jo jonkin verran kokemusta niiden toiminnasta. Tein yksinkertaisen lomakkeen, jossa kysyin kaikkien moduulien hyviä ja huonoja puolia. Pyysin kaikkia paikalla olevia järjestelmää käyttäviä työntekijöitä täyttämään laatimani lomakkeen. Palaute oli pääosin hyvää, kaikista tekemistäni moduuleista. Varsinkin kalenterin palaute oli erinomaista; kalenteri on helpottanut palavereiden sopimista sekä jakanut tehokkaasti tietoa siitä, missä kollegat liikkuvat.

Eniten vastauksissa toivottiin kalenteriin toimintoa, joka linkkaisi kalenterin käyttäjän matkapuhelimeen tai henkilökohtaiseen Google-tiliin. Matkapuhelimia varten tarvitsisi todennäköisesti jonkin taustaprosessin, joka skannaisi uusia kalenterimerkintöjä kokoajan, ja uuden merkinnän ilmaantuessa, lisäisi sen puhelimen omaan kalenteriin. Tällaisten toimintojen kehittäminen voisi olla hyvä tapa kehittää ohjelmoimiani moduuleita edelleen.

7 YHTEENVETO

Toiminnanohjausjärjestelmästä syntyi toimiva kokonaisuus ja se on osoittanut toimivuutensa myös käytännössä. Kehittämieni moduulien nopeus on normaalien internetjärjestelmien tasoa, sillä niiden lataus ei kestä muutamaa sekuntia pidempään. Ajaxilla toteutetut kyselyt eivät jää pyörimään ympyrää tai kaada selainta. Aina kun JavaScript on laajalti käytössä, se aiheuttaa tietokoneen prosessille kuormaa, mutta tässä tapauksessa prosessin kuorma ei kasva montaa prosenttia, vaikka kalenteria käyttäisikin tehokkaasti.

Ohjelmointiprosessin aikana huomasin, että haasteellisinta JavaScriptin ja Ajaxin käytössä on selaimen takaisin-napin käyttäminen. Selaimen url-osoite ei vaihdu, koska latauksen voi tehdä ilman koko sivun lataamista uudelleen. Näin ollen takaisin-napin painallus vie edelliselle sivulle eikä edelliselle Ajax-lataukselle. Luulen, että tämä on tottumuskysymys ja muutaman virhepainalluksen jälkeen käyttäjä muistaa varmasti välttää takaisin-napin painamista.

Opinnäytetyön myötä opin jQueryn käytöstä, ominaisuuksista ja mahdollisuuksista. Nykyiset internetsivustot sisältävät hyvin paljon käyttöliittymän visuaalisia ja toiminnallisia muokkausmahdollisuuksia, mitkä mahdollistavat yhä monipuolisemmat ja dynaamisemmat ulkoasut. Ajaxin avulla sivulatausten määrä vähenee. Vaikutuksensa on myös verrattain vasta kehitetyllä jQueryllä, jolla esimerkiksi modaalisen dialogin näyttäminen erilaisilla animoinneilla onnistuu muutamalla ohjelmointikoodirivillä.

Ennen liittymistäni projektiin, oli aiemmissa palavereissa jo käsitelty ja päätetty toiminnanohjausjärjestelmän uusista osista, joten en päässyt kokemaan tällaisen projektin alkumetrejä, kuten esimäärittelyä. Valitettavasti en myöskään päässyt ideoimaan ja vaikuttamaan tulevien moduulien ominaisuuksiin tai toimintoihin. Olisin halunnut voida vaikuttaa esimerkiksi paljon muokatun kalenterin ominaisuuksiin jo alusta alkaen. Asiakkailla ei yleensä, ymmärrettävästä syystä ole juurikaan tietoa itse toteutuksesta, joten olisi erittäin tärkeää, että järjestelmän ohjelmoija saisi mahdollisuuden vaikuttaa suunnitteluprosessissa.

Jos voisin nyt palata vuoden 2010 helmikuuhun ja aloittaa kalenterin uudestaan, kehittäisin sitä varmaankin eri tavoin kuin nyt. Ulkoasullisesti olen siihen tyytyväinen, mutta ohjelmointikoodi on hieman vaikeaselkoista. Nyt osaisin jakaa toimintoja paremmin eri metodeiksi, sekä käyttää Ajax-kyselyissä JSON-muotoa. Kehittäisin myös lomakkeille jonkinlaisen yhtenäisen systeemin.

Yleinen ongelma monilla ohjelmoijilla on varmasti dokumentoiminen. Olisi hyvä tottua kirjoittamaan jokaiseen metodiin lyhyt kommentti ohjelmointikoodia koskien, joka helpottaisi koodin muokkaamista myöhemmin. Se selventäisi ja palauttaisi mieleen kehittäjällekin metodin tarkoituksen ja toiminnan.

Opinnäytetyöprosessin ja toiminnanohjausjärjestelmän kehittämisen onnistumisesta kertoo kuitenkin parhaiten se, että asiakas on tyytyväinen saamaansa toimivaan järjestelmään. Opinnäytetyöprosessin myötä sain kerättyä kokemusta, joka auttaa kehittämään aina parempia, yksinkertaisempia ja toimivampia ratkaisuja.

8 LÄHTEET

1. http://www.w3schools.com/php/default.asp, viitattu 12.01.2011
2. http://fi.wikipedia.org/wiki/MySQL, viitattu 12.01.2011
3. http://fi.wikipedia.org/wiki/JavaScript, viitattu 12.01.2011
4. Heilmann, Christian, 2006 – Beginning Javascript with DOM Scripting and Ajax: From Novice to Professional, Appress®, Berkeley, California
5. Beighley, Lynn, 2010 - jQuery For Dummies®, Wiley Publishing, Inc., Indianapolis, Indiana
6. http://www.oindex.fi/listing/stats/, viitattu 16.01.2011
7. http://fin.afterdawn.com/uutiset/artikkeli.cfm/2010/05/04/net_applicationsin_mukaan_ie_n_markkinaosuus_on_jo_alle_60, viitattu 16.01.2011

www.savonia.fi

Savonia
ammattikorkeakoulu