Henrik Hakala

TIETOJÄRJESTELMÄ TYÖN VAATIVUUDEN JA HENKILÖKOHTAISEN TYÖSUORITUKSEN ARVIOINTIIN

Liiketalouden koulutusohjelma
Tietojenkäsittelyn suuntautumisvaihtoehto
2011

samk
Satakunnan ammattikorkeakoulu
Satakunta University of Applied Sciences

TIETOJÄRJESTELMÄ TYÖN VAATIVUUDEN JA HENKILÖKOHTAISEN TYÖSUORITUKSEN ARVIOINTIIN

Hakala, Henrik
Satakunnan ammattikorkeakoulu
Tietojenkäsittelyn koulutusohjelma
Marraskuu 2011
Ohjaaja: Grönholm, Jukka
Sivumäärä: 46
Liitteitä: 3

Asiasanat: tietojärjestelmä, arviointi

Opinnäytetyön aiheena oli tietojärjestelmän laatiminen Keski-Satakunnan Terveydenhuollon kuntayhtymälle. Tietojärjestelmän laatimisen tavoitteena oli työn vaativuuden arvioinnin, henkilökohtaisen työsuorituksen arvioinnin, sekä niistä johdettujen tietojen kyselyn, tallentamisen ja käsittelyn koostaminen web-pohjaiseksi palveluksi..

Työssä pyrittiin kertomaan tietojärjestelmän valmistamisen vaiheista sekä työkaluista. Normaalit työvaiheet käsittävät määrittelyn, suunnittelun, toteutuksen ja testauksen. Tuloksena syntyy järjestelmä johon sekä työntekijät että esimiehet syöttävät tietoja, joiden avulla työntekijän palkkaa voidaan muuttaa verrattaessa työn vaativuutta henkilökohtaisiin työsuorituksiin.

Opinnäytetyö sisälsi järjestelmän laatimisen palvelimesta loppukäyttöön asti. Työstä nähdään myös kuinka laajoiksi järjestelmät saattavat paisua, jos niiden annetaan itsenäisesti kehittyä.

INFORMATION SYSTEM FOR EVALUATION OF WORK REQUIREMENTS AND PERSONAL PERFORMANCE

Hakala, Henrik
Satakunnan ammattikorkeakoulu, Satakunta University of Applied Sciences
Degree Programme in Business Information Systems
November 2011
Supervisor: Grönholm, Jukka
Number of pages: 46
Appendices: 3

Keywords: information system, evaluation

Topic of this thesis was to create an information system to Keski-Satakunnan terveydenhuollon kuntayhtymä. Systems primary goal was to compile data from requirement evaluations of work and personal performance evaluations, then to create a web based service where that data is queried, saved and processed.

Thesis was meant to present how to create information system and what tools you can create it with. Normal stages of creating such system consists definition, planning, execution and testing. The result is a system in which employees and managers are feeding the data, which can be used to change employees wage based on a comparison of work requirements and employees personal performance.

The thesis included information about the system creation from configuring the server to final usage. The thesis also shows how big these systems can expand if they are let grow independently.

SISÄLLYS

1 JOHDANTO 6
2 MÄÄRITTELY 7
2.1 Tiedot ja tietokanta 7
2.2 Toiminnot 7
2.3 Ulkoiset liittymät 7
2.4 Muut ominaisuudet 8
2.5 Suunnitteluraajoitteet 8
3 JÄRJESTELMÄALUSTA 9
3.1 Palvelimen ominaisuudet 9
3.2 Asennuksien eheys ja turvallisuus 11
3.3 Apache-asennus 12
3.4 PHP-asennus 12
3.5 MySQL-asennus 13
3.6 phpMyAdmin-asennus 14
4 SUUNNITTELU 15
4.1 Kokouskäytäntö 16
4.2 Käyttöliittymän esisuunnittelu ja suunnittelupalaveri 16
4.2.1 Käyttöliittymä 17
4.2.2 Aloituspalaveri 18
4.3 Kehitysympäristö 18
4.4 Prosessimalleista 19
5 TOTEUTUS 21
5.1 Esikoodaus 21
5.1.1 Html-koodaus 21
5.1.2 css-koodaus 23
5.1.3 PHP-koodaus 24
5.2 Palaveri 2 28
5.3 Työnkuvauksien hallinta 29
5.4 Henkilökohtaisen työsuorituksen arviointi 30
5.5 Työn vaativuuden arviointi 32
5.6 Palaveri 3 34
5.7 Tietokanta 36
6 TESTAUS 38
6.1 Testiprosessi 38
6.2 Motiivit testaukseen 39
6.3 Testauksen toteutus 40

6.4 Hyväksymistestaus 40
7 KÄYTTÖ JA YLLÄPITO 41
7.1 käyttökoulutus 41
7.2 Päivitykset 41
7.2.1 Järjestelmäalusta 41
7.2.2 Palvelimen sovellusasennuksien päivitys 42
7.2.3 Tietokanta 42
7.2.4 Käyttöliittymä 43
7.2.5 Käsittelijät 43
8 LOPPUSANAT 43
LÄHTEET 46
LIITTEET

1 JOHDANTO

Opinnäytetyöni on tarkoitus tarjota Keski-Satakunnan Terveydenhuollon kuntayhtymälle selainpohjainen arviointijärjestelmä palkkausjärjestelmän tueksi nykyisten paperiversioiden ja manuaalisen tiedon tallennuksen ja laskutoimituksien tilalle. Nykyisin järjestelmä koostuu paperilomakkeista. Tiedon tallennus vaatii luonnollisesti fyysistä tilaa sekä tieto saattaa olla vanhentunutta tai puutteellista. Myös data saattaa vaihdella sekä mahdollinen tilastointi on vaikeaa. Jos tietoa halutaan ylipäätään tarkastella tai muutoin tilastollisesti tutkia, joudutaan data joka tapauksessa siirtämään sähköiseen muotoon esim. excel-tauluihin.

Kuntayhtymälle oli esitelty tarjousta kyseisestä järjestelmästä erään ohjelmistoyrityksen taholta. Tarve järjestelmälle ei kuitenkaan ollut välitön, koska nykyinen järjestelmä toimii omalla tavallaan. Olin suorittanut pakollisen työharjoitteluni kuntayhtymän ATK-huollossa ja minua pyydettiin auttamaan kertyneissä töissä sekä muussa työtaakassa. Samalla myös syntyi ehdotus että voisin valmistaa arviointijärjestelmän opinnäytetyönäni, joten päästäisiin tilanteeseen jossa itse saisin opinnäytetyöni kirjoitettua sekä kuntayhtymä saisi web-pohjaisen arviointijärjestelmän varsin edullisesti verrattuna muihin tarjolla oleviin vaihtoehtoihin. Myös projektin työmäärä on jokseenkin yhden henkilön toteutettavissa.

Tavoitteena on siis tarjoilla web-pohjainen arviointijärjestelmä, jonka ansiosta kaikki paperidokumentit voidaan poistaa kierrosta sekä tiedon tutkiminen sekä arviointiprosessi tulevat käytettävimmiksi. Kun järjestelmä on vaatimuksiinsa verraten valmis, voidaan jatkossa siihen lisätä mahdollisesti lisäkomponentteja esim. tietokannan datan erimuotoiseen tarkasteluun. Koska järjestelmä käsittelee keskitason luottamuksellista tietoa, tietoturva nostetaan suurelle prioriteetille. Järjestelmän käyttäjämäärä tulee olemaan noin 250 yksikköä. Johtuen tekijän kokemuksesta ja aikataulusta, järjestelmän optimointi ei ole ensisijaisen tärkeää, mutta sujuvan käytön varmistaminen kuitenkin pyritään varmistamaan niin pitkälle kuin mahdollista.

2 MÄÄRITTELY

2.1 Tiedot ja tietokanta

Tietokannan tulee pystyä tallentamaan vähintään ~250 yksikköä käyttäjätietoja. Tiedot ovat luonnollisesti hajautettuna relaatiomallin mukaiseen tietokantajärjestelmään. Käyttäjien perustietojen lisäksi tietokantaan tallennetaan kuvaukset tehtävistä, työntekijöiden henkilökohtaiset arvioinnit työstään, henkilökohtaisesta tavoitelomakkeesta ja tavoitekeskustelusta sekä kehityskeskustelusta. Laitteistotasolla mahdollinen muutaman sadan kyselyn määrä tulee ottaa huomioon, ettei palvelimen suorituskyky lopu kesken. Itse kyselyjen optimointia tulisi tehdä mutta arviointijärjestelmän hajakäytön kannalta, lukuun ottamatta kaikkien käyttäjien samanaikaista käyttöä, se ei ole ensisijaisen tärkeää. Tietokannan määrittelyä suoritetaan lomakkeiden vaatimusten mukaisesti toteutuksen aikana.

2.2 Toiminnot

Toiminnot alaistasolla koostuvat vain lomakkeiden täytöstä sekä mahdollisesta jatkokeskustelusta. Esimiesaseman toiminnot taasen käsittävät edellä mainittujen lisäksi käyttäjien sekä tietojen hallintaa. Tähän kuuluvat muun muassa pisteytyksien hallinta, käyttäjien oikeuksien hallinta, dokumenttien hallinta. Hallintakomponentteja pyritään toteuttamaan ennen projektin testausvaiheen alkamista mahdollisimman monta. Hallintatoiminnot siirtävät tiedon hallinnan pois varsinaisesta järjestelmäkoodauksesta.

2.3 Ulkoiset liittymät

Käyttöliittymältään järjestelmä tulee noudattamaan niin sanottua perusmallia. Perusmalli yleensä koostuu harvasta komponentista, joihin kuuluu esimerkiksi menu ja sisältö. Ulkoisiin laitteistoihin arviointijärjestelmästä ei tule kytkentöjä. Ohjelmistoliittymiä varten voidaan jatkossa koodata rajapinta, jos sille esiintyy tarvetta. Tietoliikenneliittymiä ei tehdä.

2.4 Muut ominaisuudet

Arviointijärjestelmän suorituskyky pyritään ainakin alustavasti pitämään perustasolla. Suurempia optimointeja ei tehdä, mutta ne ovat mahdollisia projektin jatkotoimenpiteinä. Kuitenkin pyritään varmistamaan käytön sujuvuus perustasolla niin pitkälle kuin mahdollista. Käytettävyyden varmistamiseksi, järjestelmä valmistetaan mahdollisimman yksinkertaiseksi ja vain vastaamaan tarkoituksenmukaista käyttötarkoitusta varten. Järjestelmän toipumiseen kiinnitetään huomiota lomakkeen tarkistajien ja tietokannan osalta, jolloin tietokannan käsittelijöihin koodataan varmistusmekanismit. Suurimmat suojaukset tehdään lomakkeen täyttöihin sekä tietokannan syöttöjä varten. Tiedon suojaus ulkopuolisia haittoja vastaan on erittäin tärkeää. Arviointijärjestelmä tullaan suojaamaan käyttäjätunnistuksien, sessiotunnisteiden (ja mahdollisien evästeiden) sekä SQL-injektioiden varalta. Arviointijärjestelmän palvelin tullaan varmistamaan erilliselle varmuuskopiointilaitteistolle automaattisesti, ilman järjestelmän valmistajalta vaadittavia toimenpiteitä.

2.5 Suunnittelurajoitteet

Vaikka standardeja pyritään käyttämään mahdollisimman tarkasti, ei esimerkiksi php-muotoisten html-dokumenttien validointiin ole omaa työkalua. Kuitenkin jokainen html-sivu koodataan php-muotoiseksi, jotta järjestelmä saadaan toimivaksi. Html-validointia suoritetaan, mikäli suurempia puutteita ilmenee käyttöä testattaessa. Laitteistolla ei teoriassa ole rajoitteita, mutta laitteiston muokkaus eli tässä tapauksessa virtuaaliympäristön muutokset saattavat vaatia hieman laajempia toimenpiteitä, jos muutoksia halutaan tehdä esimerkiksi järjestelmän jo ollessa käytössä. Tällöin palvelu luonnollisesti joudutaan keskeyttämään laitteiston muutoksien ajaksi. järjestelmän koodaus, ympäristö, ylläpito sekä muut toiminnot suoritetaan käyttämällä ilmaisohjelmia, joista valitaan käyttötarkoituksiin sopivimmat. Varsinaisia ohjelmarajoitteita ei näin ollen synny.

3 JÄRJESTELMÄALUSTA

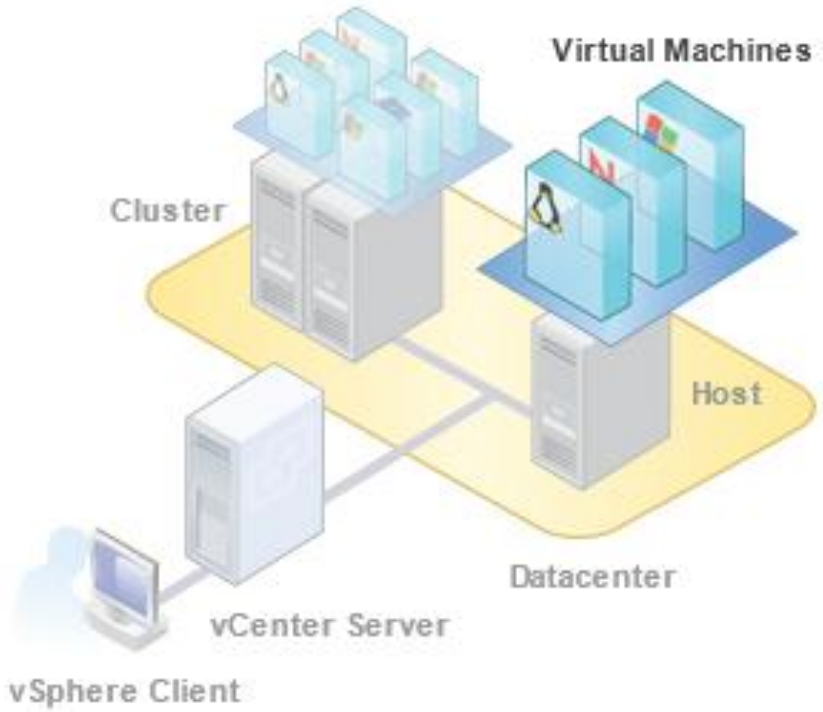
Järjestelmä rakennetaan käytettäväksi verkkoselaimella. Käytettävät selaimet käsittävät Internet Explorer, Opera, Mozilla Firefox, Google Chrome. Pääasiallisena työympäristön selaimena kuitenkin käytetään Internet Exploreria, joten järjestelmä testataan järjestelmällisesti tällä selaimella. Vaikka Internet Explorer on yksi käytetyimmistä selaimista, ei kyseinen selain kuitenkaan toimi aivan standardien mukaisesti. IE vaatii hieman html- ja css-specifikoodia näyttääkseen sivut sellaisina kuin ne on suunniteltu näkymään.

Arviointijärjestelmää ei tulla liittämään kuntayhtymän olemassa olevaan käyttäjätietokantaan. Käyttäjätietokannan haku, käyttö sekä hallinta todettiin liian monimutkaiseksi liittää arviointijärjestelmään. Kuntayhtymän käyttäjäkanta käsittää myös paljon tietoa, joka olisi karsittava pois ennen käyttöönottoa mm. oikeuksien ristiriidat, vanhentuneet käyttäjät. Arviointijärjestelmä voidaan kuitenkin liittää jo voimassa oleviin järjestelmiin, mutta vasta itse järjestelmäprojektin päätyttyä. Rajapinnan teko tulevaisuudessa ei todennäköisesti tule olemaan kovin monimutkaista.

Arviointijärjestelmä tarjotaan ainakin aluksi vain intranet-palveluna eli lähiverkon ulkopuolelta järjestelmää ei voida käyttää. Ulkoverkon käyttö esim. kuntayhtymän verkkosivujen kautta mahdollistetaan, mikäli siihen syntyy tarvetta. Näin ollen kaikki suojausperiaatteet on pidettävä mahdollisimman ajan tasalla. Itse lähiverkko on suojattu kolmannen osapuolen hallinnoimalla palomuurilla sekä kuntayhtymän virus-ja vakoilusuojauksella (F-secure).

3.1 Palvelimen ominaisuudet

Järjestelmä asennetaan virtuaalipalvelimelle Windows Server 2008 R2 64-bit käyttöjärjestelmään. Virtuaalipalvelimella tarkoitetaan käytännössä palvelinta, jossa voidaan suorittaa useita palvelimia samalla laitteistolla. Virtuaalipalvelimelle voidaan tehdä niin monta palvelinta kuin laitteiston resurssit antavat myöden.



Kuva 1. KSTHKY:n virtuaaliympäristö.

Kuvassa 1 olevan KSTHKY:n virtuaaliympäristön hallinnoinnista vastaa tässä tapauksessa Vmwaren ohjelmistot vCenter server ja vSphere client, jolla palvelua käytetään. Datacenter eli tietokeskus on ensisijainen tiedon säiliö, jossa säilötään objekteja, kuten isäntiä eli hosteja, virtuaalikoneita ja clustereita eli ryppäitä. Rypäs koostuu monesta isännästä ja kun isäntiä luodaan, siirtyvät isännän resurssit osaksi ryppään resursseja. Isäntä eli host on tietokone joka käyttää virtualisointisovellusta kuten esimerkiksi ESX tai ESXi, joilla luodaan virtuaalikoneita ja luodaan näille resursseja. Koska palvelin on virtuaalipalvelin, voidaan sen keskusmuistia tai massamuistia lisätä ilman, että palvelimen käyttöjärjestelmää tai muita osia joudutaan poistamaan. Sen sijaan prosessointitehon muutokset, esimerkiksi prosessorin \ prosessoreiden lisääminen, vaativat palvelimen täydellisen uudelleenasennuksen. Palvelimen nimeksi asetettiin ARVOSERVER, joka toimii domainissa THKY.DOM.

Palvelimen ominaisuudet:

- Prosessori: Intel Xeon X5550 @ 2,67GHz

- Keskusmuisti: 2,0GB EDO @ 557Mhz (3-3-3-?)
- Emolevy: Intel Corporation 440BX
- Näytönohjain: Standardi VGA adapteri, resoluutiolla 1680x1050@60Hz
- Kiintolevy 100GB VMware Virtual disk SCSI Disk Device (SAS(Serial Attached SCSI))
- Optinen asema: NECVMWar VMware IDE CDR10 ATA Device

Palvelimen massamuistista ohjelmien käytössä ~15%, joten suorituksiin sekä tietokantoihin varattu ~85% eli ~85GB. Koska palvelimeen asennettavan järjestelmän käyttäjäkunta on ~260 yksikköä, massamuistin määrää ei jouduta kasvattamaan. Palvelin omistetaan täysin arviointijärjestelmän sekä muiden sen tarjoamien palveluiden käyttöön.

Koska järjestelmän tekoon pyritään luonnollisesti käyttämään mahdollisimman vähän rahallisia resursseja, pyritään ohjelmistokehitykseen, käyttöönottoon sekä varsinaiseen suoritukseen käyttämään ilmaisia ohjelmia, mikäli niitä on saatavilla.

3.2 Asennuksien eheys ja turvallisuus

Jokainen asennettava paketti tulee varmistaa jollakin tavalla niin, että sitä ei ole muokattu millään tavalla poikkeavasti valmistajan ilmoittamiin pakettisisältöihin verraten. Vaikka ohjelmistot tulisi pääsääntöisesti ladata valmistajien omien linkkien kautta, useasti valmistaja ei itse tarjoa latausta tiedostoistaan, vaan se on ulkoistettu muulle yritykselle. Valmistajan sivuilta tulee aina nähdä esimerkiksi joko MD5 checksum, GnuPG tai RPM varmenteet tai allekirjoitukset. Mikäli tunnisteet tai allekirjoitukset eroavat valmistajien ilmoittamista, kyseessä on väärennös tai muokattu ohjelma, jotka usein sisältävät esimerkiksi haittaohjelmia.

Arviointijärjestelmän tiedostot varmennettiin käyttäen MD5-summaa. Käytössä oli helppokäyttöinen ohjelma MD5SUMS. Jokainen ladattu paketti täsmäsi valmistajan ilmoittamiin summiin. Täten voidaan olla varmoja että alusta, jolle koko järjestelmän varsinainen tiedonkulku ja -hallinta, perustuu, ei ole vaarassa.

3.3 Apache-asennus

Palvelimeen asennetaan Apache http-palvelinohjelmisto. Apache tarjoilee siis tiedostojen jakamista HTTP-protokollan ylitse. Apache on avoimeen lähdekoodiin perustuva, eli se on ilmainen. Apache on myös suosionsa ansiosta erittäin päivitetty mm. tietoturvan sekä suorituskyvyn kannalta (Apache HTTP server project 2011). Apacheen on myös mahdollista integroida lisäjärjestelmiä esimerkiksi PERL-scripteille, Urlien muokkaamiselle tai MySQL:n lokien muodostamiselle, jota voidaan mahdollisesti käyttää myös arviointijärjestelmään.

Apachesta asennetaan versio 2.2.17, jossa ei ole ssl-tukea. Asennettava ohjelma on 32-bittinen, jotka kuitenkin suoriutuu 64-bittisessä käyttöjärjestelmässä. Asennuksen aikana ohjelmaan syötettiin domain- ja palvelinnimi sekä järjestelmänvalvojan sähköpostiosoite, joka on pakollinen. Sähköpostiosoitetta ei kuitenkaan välttämättä tarvitse käyttää tai sen voi varsinaisessa järjestelmän koodissa ylikirjoittaa, mikäli tähän tarvetta tulee. Apachen konfigurointi kirjoitetaan httpd.conf-tiedostoon. Samassa tiedostossa on myös mahdollisuus asettaa käyttöön lisäkomponentteja, mutta perusasetuksilla konfiguraatio-tiedostossa ei ole kytkentöjä muihin, kuin kriittisiin palveluihin. Tiedostosta muutetaan arviointijärjestelmää varten tiedostojuuri sekä index-sivut, eli järjestelmän aloitussivut, joita voi mahdollisesti olla useita (mm. index.php-tiedosto) alkukäynnistystä varten.

Apachen toimivuus testataan syöttämällä selaimen osoiteriville osoite ”localhost”, jolloin ohjelmisto ilmoittaa selaimeen toimivansa. Apache on käynnissä kokoajan, mutta sen voi kuitenkin sammuttaa tai uudelleenkäynnistää vaivattomasti hallintatyökalun avulla.

3.4 PHP-asennus

Palvelimeen asennetaan ilmainen yleisscript-kieli PHP, jota voidaan käyttää esimerkiksi siirtämään dataa tietokannan ja websovelluksen välillä. PHP on varta vasten suunniteltu web-ympäristöön ja sen voikin täten helposti upottaa esimerkiksi html-koodin mukaan. Arviointijärjestelmässä pyritään kuitenkin pitämään PHP-koodi

mahdollisimman paljon eri tiedostoissa mm. tietoturvan vuoksi, koska PHP-koodia ei näytetä suorituksen aikana sovellusten käyttäjälle. Näin ollen PHP-koodiin voidaan myös koodausta ja testausta helpottamaan kirjoittaa aluksi staattiset ja loppuvaiheessa dynaamiset kirjautumistiedot MySQL-tietokantaa varten, jotta näitä tietoja ei tarvitsisi kirjoittaa joka tiedostoon, tai funktioon erikseen. Kun PHP-koodi on eri tiedostossa, se on tällöin myös helppolukuisempaa.

PHP:sta asennetaan versio 5.3.4. PHP versiot on spesifioitu eri alustoille, ja koska arviointijärjestelmää varten asennettiin Apache versio 2, tulee PHP:stä käyttää VC6 versiota. PHP asennetaan oletuslaajennuksilla, joista kuitenkin valtaosa on poistettu käytöstä, jonka ansioista koodaaja voi tarpeensa mukaan kytkeä käyntiin vain tarvitsemansa laajennukset. Useat laajennukset ovat myös riippuvaisia muista lisälaajennuksista, ja niiden asentaminen usein aluksi johtaa järjestelmän toimimattomuuteen. PHP asentaa Apachen configurointi-tiedostoon tarvitsemansa tiedot; esimerkiksi php.ini-käynnistystiedoston sijainnin.

3.5 MySQL-asennus

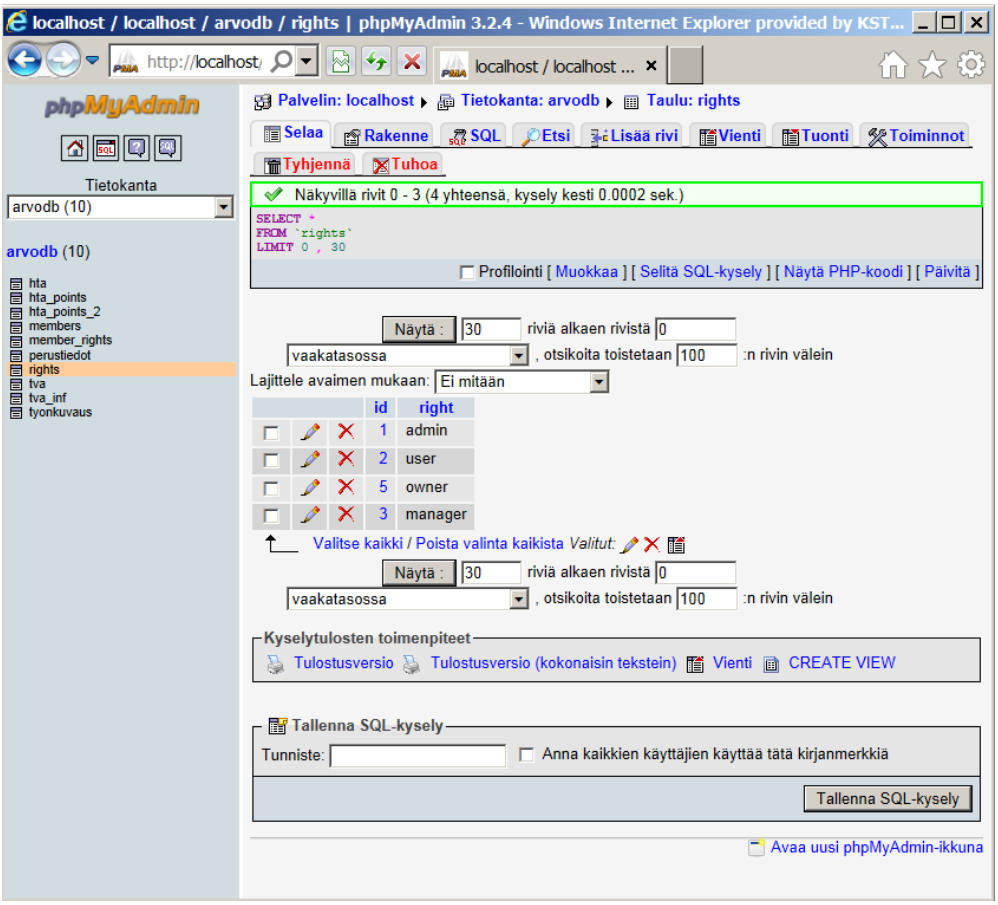
Palvelimeen asennetaan MySQL-tietokannnan hallintajärjestelmästä versio 5.5.8 32bit. MySQL:n hallinta tapahtuu normaalisti komentoriviltä. Hallintaan kuitenkin arviointijärjestelmässä käytetään phpMyAdmin-hallintatyökalua, jonka käyttöliittymä helpottaa huomattavasti tietokannan tarkastelua ja muokkaamista. MySQL-tietokantaan tallennetaan tiedot käyttäen PHP-ohjelmalogiikkaa. MySQL:n asennus perusasetuksilla ei vaadi varsinaisen asennuksen lisäksi muita kuin juurikäyttäjän ja sen salasanan luomisen.

MySQL:n tarjoilemat tallennusmoottorit:
MyISAM – oletusmoottori, hyvä suorituskyky
CSV – kirjoittaa datan tekstitiedostoon pilkulla eroteltuna
MRG_MYISAM – kokoelma identtisiä MyISAM-tauluja
BLACKHOLE – /dev/null-moottori (kaikki tieto häviää)
InnoDB – vuorovaikutustuki, rivilukitus, viiteavaimet (varsinainen MySql:n tarjoilema relaatiomoottori)

ARCHIVE	Arkistointimoottori
MEMORY	nopea, muistipohjainen, tilapäisten taulujen käsittelyyn

3.6 phpMyAdmin-asennus

Palvelimelle asennetaan phpMyAdmin-tietokannanhallintatyökalun versio 3.3.9. PHPMyAdmin-ohjelma ei ole varsinaisesti asennettava ohjelma. Kun ohjelma on ladattu, tarvitsee se vain purkaa kansioon, Apachessa määritellyn document-juuren eli sovellusten pääkansion alle. PHPMyadminin omasta konfiguraatio-tiedostosta tulee tehdä kopio, joka uudelleennimetään käyttöönottoa varten ja kopioon eli varsinaiseen käytettävään konfiguraatio-tiedostoon lisätään blowfish-tunnus mahdollisia evästeitä varten. Hallintatyökalu kysyy käynnistettäessä edellä luotuja juuritunnuksia päästääkseen hallinnoimaan MySQL-tietokantoja. Hallintatyökalun kuvassa 2 näkyvän graafisen käyttöliittymän ansiosta tietokannan hallinta on helppoa.

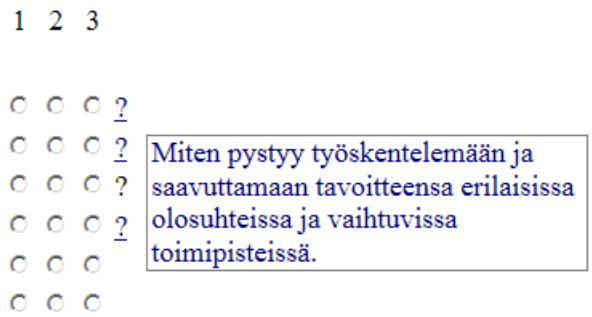


Kuva 2. PhpMyAdmin graafinen käyttöliittymä kehitysympäristössä.

4 SUUNNITTELU

Käyttöliittymässä tulee ottaa huomioon monia seikkoja. Varsinaisien standardien mukaan käyttöliittymäsuunnittelu sekä sen toteutus todettiin melko tiukaksi. Vaikka tiukasti standardin mukaan koodattavaa sivuston ulkoasua sekä käytettävyysseikkoja ei arviointijärjestelmään koodata, tulee silti muistaa muutamia peruskäsitteitä ja ohjenuoria, joiden avulla voidaan varmistaa sujuva ja mielekäs käyttökokemus.
(Wikipedia 2011/a)

Käyttöliittymän tulee olla käyttötarkoituksen mukainen. Tähän pyritään karsimalla ja välttämällä kaikkea tietoa, joka ei ole järjestelmän käytön mukaista tai vaadittavaa. Toisin sanoen, se tarkoitus, joka järjestelmällä on, tulee näkyä käyttöliittymässä; Järjestelmän ei tule näyttää käyttäjälle muuta tietoa. Arviointijärjestelmän käyttöliittymän tulee tiedottaa käyttäjälle varsinkin suorien informaatioiden eli linkkitekstien, otsikoiden, alustuspuheiden yms. kautta tarvittava alustustieto järjestelmän käyttöä varten. Käytön tueksi käytetään myös mm. kuvassa 3 näkyviä kuplaohjeita ja lisäinformaatioikkunoita, jos käyttökokemus tulevissa testitapauksissa osoittautuu epäselväksi.



Kuva 3. Esimerkki kuplaohjeesta kun osoitin viedään kysymysmerkin päälle.

Käyttäjän tulee saada tietyn asteinen kontrolli käyttöliittymään. Käyttäjän tulee siis saada vaikuttaa käytön aloittamiseen, suuntaan ja tahtiin, jolla hän järjestelmää käyttää, kunnes tarkoituksenmukaiset toimet on tehty. Käyttöliittymän tulee vastata käyttäjien ennakko-odotuksia sekä sen tulisi olla mahdollisimman riippumaton käyttäjän tietotaitoon, koulutukseen yms. verrattuna. Järjestelmän tulee antaa käyttäjälle rajattua informaatiota, mikäli syntyy ongelmatilanteita. Liian tarkkoja ongelmaraportteja

ei tule näyttää käyttäjälle, vaan tyydytään tarjoilemaan vain kriittiset virheilmoitukset mm. lomakekenttien väärintäyttäminen, puuttuvat tiedot.
(Wikipedia 2011/a)

Käyttöliittymän tulisi standardin mukaisesti olla yksilöitävä jokaisen käyttäjän ominaisuuksien sekä käyttökokemuksien mukaan. Koska arviointijärjestelmä pyritään rakentamaan vain kriittisten toimintojen mukaiseksi, ei käyttöliittymän muokattavuuteen tulla puuttumaan. Näkymät ja muu rakenne ovat kaikille käyttäjille samat, lukuun ottamatta esimiestason näkymiä. Järjestelmään koodataan kuitenkin monitasoinen käyttäjätietokanta, jonka avulla voidaan vaadittavia lisäominaisuuksia antaa vain niitä tarvitseville käyttäjille, jos tarve tällaiseen rakenteeseen syntyy. Käyttöliittymän tulisi myös olla sellainen, jonka avulla voidaan myös oppia ja muistaa käytön perustoimenpiteet. Arviointijärjestelmän käytössä oppimisella ei kuitenkaan ole suurta vaikutusta, koska informaatio pyritään tarjoamaan varsin yksiselitteisesti.

4.1 Kokouskäytäntö

Kokouskäytäntöä on syytä käyttää projektiympäristössä ja vähänkään pidemmissä tai virallisemmissa tapaamisissa, joissa asioista sovitaan. Käytäntö käsittää aikataulun, roolit, asiat sekä dokumentoinnin. Kokouksen järjestää projektipäällikkö. Projektipäällikön tehtävänä on näin ollen kutsua osallistujat, laatia kokouksen esityslista, lähettää etukäteismateriaali sekä aikataulu esitettäville asioille. Kokoukselle tulee valita sihteeri ja puheenjohtaja. Roolit voivat vaihdella. Sihteeri kirjaa muistioon läpikäydyt ja sovitut asiat. Puheenjohtajan rooli on jakaa puheenvuoroja, jotta keskustelu pysyy johdonmukaisena. Puheenjohtaja koostaa päätökset jotka kirjataan muistioon. Muistiot jaetaan kokouksen päätyttyä osallistujille. Hyvä tehokas kokous\palaveri tulee olla mahdollisimman lyhyt eli mielellään muutamia kymmeniä minuutteja. (OK-opintokeskus 2011.)

4.2 Käyttöliittymän esisuunnittelu ja suunnittelupalaveri

Järjestelmän suunnittelu aloitetaan hahmottelemalla yksinkertainen eli raaka versio käyttöliittymästä sekä palvelun sisällöstä. Tässä projektissa tehdään suoraan käyttö-

liittymäkuvaus word-tiedostoon, joka esitellään tilaajalle paperimuotoisena. Kuvauksessa ei kuitenkaan esitelty kaikkia ominaisuuksia perusteellisesti, jotta ylimääräisen työn määrä tulee minimoitua, jos liittymä ei olekaan sitä, mitä tilaaja odottaa. Dukumenttiin kirjattiin pääpiirteittäin liitteen 1 mukaisesti, miltä käyttöliittymän tulisi näyttää .

Kuvaus on hahmoteltu pienehkön esimateriaalin avulla. Esimateriaalina on saatu henkilökohtaisen työsuorituksen arviointilomake, työkuvauslomake, taustatietolomake sekä palkkauksen ja tehtäväkohtaisen palkan merkityksiä ja lakimääräyksiä sisältäviä dokumentteja. Dokumenttien sisältöön ei kuitenkaan syvennytä liiallisesti, jotta järjestelmän pääpiirteet saadaan ensin kuvattua mahdollisimman tehokkaasti. Kuvaus esitellään ensimmäisessä suunnittelupalaverissa ja sitä täsmennetään käyttöliittymän sekä tarjottavan palvelun osalta niin että turhat ominaisuudet jätetään pois sekä vaadittavat lisäratkaisut lisätään suunnitelmaan. Järjestelmään ei kuitenkaan tule lisätä mitään, joka poikkeaa järjestelmän kokonaistarkoituksen mukaisista määrityksistä ts. arviointijärjestelmästä.

4.2.1 Käyttöliittymä

Käyttöliittymän perusrakenne suunniteltiin mahdollisimman yksinkertaiseksi. Liittymä koostuu linkkilistasta, ylä- sekä alatunnisteesta, sisältöalueesta ja taustasta. Linkkilista asetetaan sivun vasempaan reunaan. Linkkien avulla käyttäjä saa navigoitua haluamaansa paikkaan (Kuva 4). Liittymä skaalataan keskelle ruutua ja reunoista rajataan tilaajan haluama tila pois, mutta kuitenkin niin, että sisällön hahmottaminen ja käyttö säilyy järkevällä tasolla.

Esimiehellä linkkilista laajentuu kuvan 4 mukaisesti sisältöalueeseen kun tarkastellaan hänen alaisiaan. Sisältöalueeseen näytetään eri linkeistä erityylisiä rakenteita mm. erityylisiä lomakkeita ja listoja. Käyttöliittymään ei tehty aluksi mitään muotoiluja, vaan liittymä pidettiin vain hahmotelmana. Liittymä suunniteltiin taulukoksi niin, että sivuston koodaus ja asemointi css-määrityksissä on helpompaa.

Kuva 4. Hahmotelma käyttöliittymästä.

4.2.2 Aloituspalaveri

Koska tilaajan aikataulu oli rajallinen, annettiin tilaajalle esisuunnitelma käyttöliittymästä ja linkityksistä ensin omaan tarkasteluun. Näin ollen tilaaja voi etukäteen miettiä listan pohjalta mahdollisia kysymyksiä ja ratkaisuja. Jotta projektin aikatauluun ei tule mainittavia aukkoja, käytetään neuvotteluun pääsyn sekä muu jäävä odotusaika dokumenttien täydentämiseen, lisäsuunnitteluun ja raa'an html-sivupohjan koodaukseen.

4.3 Kehitysympäristö

Testausjärjestelmän koodaus tehdään pääasiassa kehitysympäristössä. Näin voidaan taata, että varsinaiseen palvelimeen ei suunnata hyökkäyksiä tai muuta väärinkäyttöä, jos puutteellinen järjestelmä olisi sinne asennettu. Kehitysympäristönä käytetään Xammp-ohjelmistoa, joka mahdollistaa Apache-webpalvelimen emuloinnin omalla työasemalla. Xammp pitää sisällään kaikki web-palvelimen ja sen tarvitsemien komponenttien asennustiedostot mm. Apachen, MySQL:n, sekä php-tuen. Ohjelmisto

tarjoilee myös muita web-järjestelmien ohjelmistoja esimerkiksi ftp-palveuita ja sähköpostipalvelun sekä yksinkertaisen ja luontevan käyttöliittymän palveluiden tarkastelemiseen ja hallinnointiin.

Xammp-ohjelmistoa ei kuitenkaan missään nimessä saa käyttää varsinaisen järjestelmän alustana, koska Xammp tarjoilee oletuksena hyvin monia ominaisuuksia, lisäosia sekä palveluita suoraan piittaamatta tietoturvasta. Xammp on kolmannen osapuolen koostama ohjelmisto, jossa ei ole otettu huomioon kaikkia mahdollisia haittatekijöitä. Kun järjestelmä on valmis, voidaan tiedostot siirtää manuaalisesti kopioimalla varsinaiseen palvelimeen. Yksittäistä tietokantatiedostoa ei suoraan ole vaan se täytyy erikseen luoda PhpMyAdminin työkalulla, jonka jälkeen tiedoston voi siirtää palvelimelle.

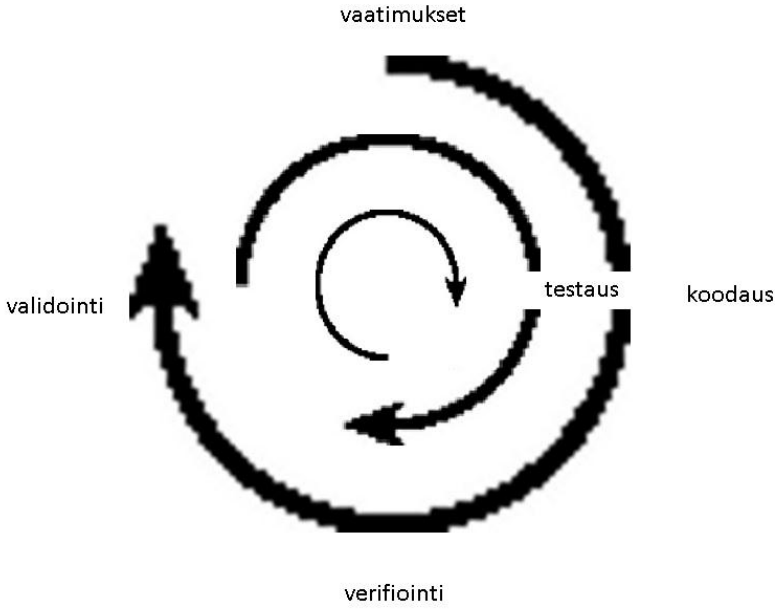
4.4 Prosessimalleista

Ohjelmistotuotannon tarpeisiin hyödynnetään monia erilaisia prosessimalleja tai niiden variaatioita. Kyseisien mallien on tarkoitus tarjota prosessin avuksi yleisiä kuvauksia miten prosessi viedään läpi. Mallien tulee olla niin korkealla tasolla, että niitä voidaan tarvittaessa varioida muille niitä tarvitseville tahoille tai sovelluksille sekä niiden kehittämiseen.

Prosessimallit koostuvat vaiheista, jotka ns. kommunikoivat keskenään. Vaiheet käsittävät kaikki tuotannon tärkeät vaiheet. Vaihejaossa kiinnitetään huomiota suoritusjärjestyksiin, vaikutussuhteisiin ja toimenpiteisiin, joita tarvitaan rajapintojen vuoksi. Koska prosessi jaetaan vaiheisiin, tulee jokaisen vaiheen saada jonkin syötteen sekä tulostaa jonkin syötteen. Näin ollen syötteitä voidaan käsittää osatuloksina, joita voidaan arvostella. Niin kuin projektilla tai prosessilla, myös osavaiheella on oltava selvä alku sekä loppu. Yleensä osavaiheita käytetään hyväksi toistamalla niiden tuloksia edellä tehtyjen vaiheiden syötteinä. Tällöin saadaan tarkennettua tulostetta ja päästään parempaan ratkaisuun. Osavaiheille on myös prosessin sisällä muita aputoimenpiteitä mm. dokumentointi, laadunvarmennus. (Wikipedia 2011/b)

Käytettyjä yleismalleja ovat mm. lineaarinen malli (ts. vesiputousmalli), prototyyppimalli, RAD-malli, erilaiset evoluutiomallit, spiraalimalli, komponenttimalli, rinnakkaisen kehitystyön malli, formaalien menetelmien malli, ja muut neljännen sukupolven tekniikoita hyödyntävät mallit. (Wikipedia 2011/b)

Arviointijärjestelmän kehityksessä hyödynnetään Spiraalimallia, johon liitetään prototyyppimallin ominaisuuksia. Tällöin päästään kuvassa 5 hahmoteltuun ns. vapaasti kokeiltavaan prosessien ja projektin läpiviemiseen tarvittavaan tekniikkaan, jossa järjestelmän laatijalle annetaan suuri hallinnan mahdollisuus, mutta kuitenkin sillä tasolla että järjestelmän tulee toimia halutulla tasolla ja tarkoituksiensa mukaisesti. Osavaiheista syntynyttä dataa tullaan varmentamaan asiakastasolla. Varmistettua dataa peilataan uudestaan kaikilla tasoilla kunnes saavutetaan taso joka vastaa määriteltyjä kriteereitä kaikilla asiakkaan vaatimilla tasoilla. Tärkeimpinä osavaiheina ovat suunnittelu, toteutus ja testaus. Riskianalyysiä tehdään vain järjestelmän laatijan toimesta, mikäli jo olemassa olevien pienten riskien lisäksi ei muuta löydetä.



Kuva 5. Yksinkertainen spiraalimalli

5 TOTEUTUS

5.1 Esikoodaus

Koska projektin edistyminen viivästyi omien työkiireiden, mutta ennen kaikkea tilaajan kiireiden vuoksi, voidaan aika, joka neuvotteluun odottamisesta ja omien työtauoista jää, käyttää sivuston esikoodaamiseen. Esikoodaamiseen voidaan sisällyttää mm. sivuston tiedostorakenteen luontia, html-koodausta, css-tyylitiedostojen sekä php-tiedostojen esiluontia. Koska mm. tietokantayhteydet ja muut perusasetukset tulee joka tapauksessa luoda, voidaan ne luoda jo ennen varsinaisen järjestelmän aloittamista.

5.1.1 Html-koodaus

Html-lyhenne tulee sanoista Hyper Text markup Language. Html:n kuvaa tagimuotoisesti verkkosivun. Tagit käännetään verkkoselaimessa verkkosivuksi, joka näytetään käyttäjälle. Html-tagi muodostuu yleensä pareittain eli aloittavasta ja lopettavasta tagista. Myös yksittäisen tagin käyttö on mahdollista. Html tiedosto sisältää myös normaalin tekstin. Html-dokumentti on näin ollen verkkosivu. Html-tiedostosta voi etsiä virheitä käyttämällä se validaattorin lävitse. (w3scools.com 2011)

Html-koodaukseen voidaan yksinkertaisimmillaan käyttää esimerkiksi mitä tahansa tekstinkäsittelyohjelmaa. Saatavilla on kuitenkin monia enemmän ja vähemmän kehittyneitä editoreita html-koodaukseen, jotka tarjoilevat mm. monia kirjoitusta helpottavia ominaisuuksia esimerkiksi automaattisen tagin täydennyksen. Projektiin otettiin alustavaan käyttöön Bluefish-editori, joka kuitenkin vaihdettiin huonon käyttökokemuksen jälkeen selkeämpään Notepad++ -editoriin.

Järjestelmän html-sivu ei oletusarvoisesti osaa tarjoilla ä- tai ö-kirjaimia asiakkaalle, joten ne täytyy manuaalisesti tarjoilla. Ä- sekä ö-kirjaimet saadaan käyttöön koodaamalla html head-tagin sisään metatieto-tagi johon sisällytetään UTF-8-merkistön määritys: ”<meta http-equiv="Content-Type" content="text/html;charset=UTF-8"/>”. Metatiedon kautta koodattuna merkistömääritys ei ole kovin toimiva kaikilla

alustoilla. Määritys on mahdollista muuttaa myös editoreilla, jotka koodaavat tunnisteen itse tiedoston ominaisuuksiin.

Vaikka järjestelmän käyttöliittymä ei olekaan saanut niin sanottua virallista hyväksyntää, voidaan sen runko kuitenkin tehdä jo valmiiksi. Runkoon eli kuvan 6 mukaiseen näkymään ei tule vielä koodata mitään monipuolisempaa ja jos tilaaja ei olekaan tyytyväinen ehdotettuun malliin, sen muokkaaminen onnistuu varsin helposti. Kuitenkin, jos rungon teko jätettäisiin vahvistukseen asti, aikaresurssi kuitenkin kuluu, ilman että mitään tehtäisiin.

Arviointijärjestelmän ulkoasu koostuu neljästä pääelementistä: yläsarake, alasarake, menu eli linkkilista/navigointi ja sisältöelementti. Sisältöelementtiä tullaan laajentamaan vielä syvemmälle, mutta pääasiallisesti tämä elementti pitää sisällään kaiken muokattavan sisällön esimerkiksi lomakkeet. Elementit koodataan käyttämällä div-tageja, jotka tunnistetaan ”id”-tunnisteella mm. css-tyylitiedostoja varten.

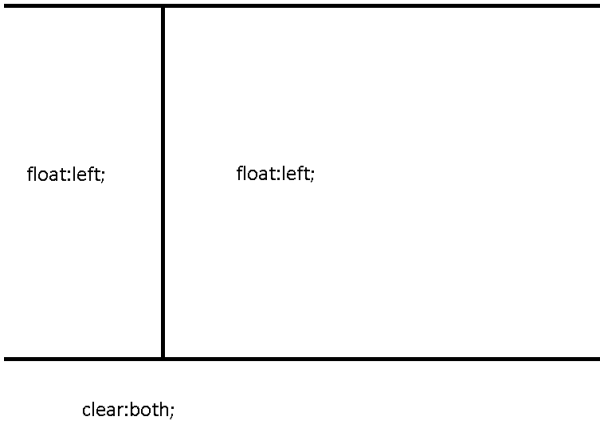
Kuva 6. Arviointijärjestelmän käyttöliittymä.

Standardien käyttö on ensisijaisen tärkeää. Näin voidaan huolehtia siitä, että palvelu palvellaan oikeanmuotoisena kaikkia niitä asiakkaita eli selaimia varten, jotka standardeja osaavat käyttää. Näin ollen pystytään minimoimaan eri selaimia varten tehtyä erityiskoodia, jota ennen standardointia piti kirjoittaa melkoisesti. Standardointi tapahtuu käyttämällä W3C-yhteisön määrityksiä ja validaattoreita. Mm. html- ja css-tiedostot voidaan tarkistaa käyttämällä validaattoreita. Validaattorit ilmoittavat virheistä tai puutteista, mikäli sellaisia dokumenteissa esiintyy. Standardointivahvistus voidaan näyttää käyttäjille sovelluksessa, mutta se ei ole ensisijaisen tärkeää.

5.1.2 css-koodaus

Css-lyhenne tulee sanoista Cascading Style Sheets. Css:n tarkoituksena on määritellä miltä verkkosivun ulkoasu näyttää, miten komponentit esitetään, minkä värisinä jne. Css mahdollistaa ulkoasumäärityksien eriyttämisen varsinaisen html-dokumentin ulkopuolelle. Näin ollen voidaan helpottaa ulkoasun muokkausta koska esimerkiksi yhtä css-tiedostoa voidaan käyttää monessa eri html-sivussa. Sivuston ulkoasu voidaan täten myös koostaa monesta eri css-tiedostosta, jolloin mahdollistetaan erittäin monipuolinen ulkoasu sekä joustavuus. Css mahdollistaa myös joidenkin toiminnallisuuksien tekemisen verkkosivulle esim. reaaliaikaisesti muuttuvat ulkoasu. Css-määritykset voidaan kirjoittaa eri tavoilla esimerkiksi tagi-kohtaisesti tai tageille määritettyjen nimien/tunnistekohtaisesti, jolloin päästään entistä tarkempaan ulkoasumääritysten hallintaan. (Wikipedia 2011/c)

Järjestelmään koodataan asemoinnit. Css-määrityksissä käytetään kuvan 7 mukaista asettelumallia. Tähän voidaan helposti käyttämällä kellutusmääritystä ”float – clear” eli kelluttamalla komponenttia vasemmalle ja nollaamalla määritys viimeisessä komponentissa. Koska menu ja sisältöalueet ovat vierekkäin ja yläpalkin ollessa staattinen, voidaan alapalkkia puskea alaspäin, jos sisällön ja/tai menun sisältö kasvaa. Tekniikan ongelmaksi kuitenkin tulee palkkien eri korkeudet, jolloin lyhyemmän, tässä tapauksessa ”staattisen” menun, ominaisuudet mm. taustaväri ei kopioidu tyhjälle ylimääräiselle alueelle. Korjaus vaatii kuitenkin suuremman luokan panostusta koodiin, eikä korjausta tehdä, mikäli siitä ei mainintaa saada asiakkaan toimesta.



Kuva 7. Kellutus.

Koska järjestelmän käyttäjän käyttävän erilaisia näyttölaitteita ja erilaisia resoluutioita, täytyy myös järjestelmän olla selvälukuinen ja mahtua näytölle kokonaisuudessaan. Poikkeus tietenkin aiheutuu, jos mennään liian pienille alueille. Sivusto koodataan 1024:n pikselin leveyteen, vanhojen näyttöjen kuvasuhteen ollessa 4:3. Laajemmilla resoluutioilla sivun pohja jää näkyviin. Sivun minimiarvo on siis 1024 pikseliä näkyäkseen kokonaisuudessaan. Pohjalle asetetaan alustavasti taustaväriksi valkoinen.

5.1.3 PHP-koodaus

Personal Home Page, nykyisin tunnettu Hypertext Preprocessor eli PHP-kieli on script-kieli, jonka tarkoitus on tuottaa dynaamisia verkkosivuja. Koodi kirjoitetaan joko html-koodin sekaan tai omiksi script-tiedostoiksi. PHP-koodi tulkitaan palvelimen PHP-muduulissa, joka tuottaa pyydetyn sivun. PHP on saatavilla ilmaiseksi lähes kaikille käyttöjärjestelmille ja alustoille. (Wikipedia 2011/d)

PHP oli koodaajalle ennestään melko tuntematon alusta, joten taustatutkimukseen ja opiskelemiseen meni valtava määrä aikaa. Pääasiassa ongelmia tuotti php:n vaatimukset sivuston tiedostorakenteen vuoksi. Tiedostorakenteen tulisi olla mahdollisimman yksinkertainen, mutta kuitenkin niin eroteltu, että valtavilta tiedostomääriltä yhdessä kansiossa vältyttäisiin. Rakenteen tulee myös olla sellainen, jolla pystytään

välttämään saman tiedon kirjoittamista uudelleen. Sivuston rakenteeseen ei ole tiettyä oikeata mallia, mutta kun koodausta suoritettaan enemmän, päädytään mahdollisesti tilanteisiin, joista on työlästä tai mahdotonta suorittaa vaadittavia toimenpiteitä halutun toiminnallisuuden saavuttamiseksi. Rakenne päädyttiin rakentamaan aluksi niin, näkymät sekä logiikka ovat eri tiedostoissa: pages, scripts, error, styles yms. Rakennetta voidaan tietenkin muokata kesken kaiken, mutta mitä enemmän kansioihin sisällytetään tietoa ja linkityksiä, sitä enemmän korjausvaatimukset muutettujen rakenteiden vuoksi tiedostoissa kasvavat.

Kaiken toiminnallisuuden keskipisteenä toimii index.php-tiedosto, joka sijaitsee sivuston juuressa ja joka tulee ladatuksi aina ensimmäisenä. Koska index-tiedosto toimii kaiken keskipisteenä, tulisi siihen massiivinen määrä koodia. Koodi voidaan kuitenkin purkaa tiedostorakenteeseen eri kansioihin, jolloin index-php voi suorittaa esimerkiksi include-metodilla tiettyjen sivujen tarjoilemisen tai luokkien ja scriptien käyttöä. Puumaisen sivustorakenteen ansiosta voidaan helposti varmistaa juuritason koodilla kriittiset toiminnot esimerkiksi sessiokäytäntö. Mikäli kyseessä olisi laajempi sivusto, voitaisiin rakenne hajottaa vielä ns. sivukohtaisesti kansioihin, joissa jokaisessa tarjoiltaisiin oma index.php. Kun koodauksessa edetään pidemmälle, kaikki alikansiot piilotetaan, niin että loppukäyttäjä ei pysty suoraan näkemään tai muokkaamaan niiden sisältämiä tiedostoja, ainakaan ilman autentikointia.

Kun Html-runko on koodattu, voidaan se hajottaa elementteihin, ja käyttämällä php:n include-metodia, saadaan elementit koostettua halutuksi näkymäksi eli sivuksi. Html-runko paloiteltiin neljään osaan, jotka mainittiin kohdassa 3.3.1. Paloittelulla saadaan aikaiseksi komponentteja, joita voidaan käyttää eri tavoilla eri näkymissä, mikäli se on tarpeellista. Ja koska komponentit on eritelty, voidaan niitä käyttää mm. muissa sivustoissa ilman itse koodin muuttamista. Lopuksi siis päästään rakenteeseen, jossa index.php sisällyttää näkymän, joka sisällyttää komponentit. Komponenteista mm. menu-komponentti, joka sisältää linkit, joista sisältökomponentin sisältöä vaihdetaan, kommunikoi index-php:n kanssa. Index.php vaihtaa sisällytetyn näkymän sisältökomponentin, saadessaan php:n GET-metodin avulla menu-linkeistä lähetetyn atribuutin. GET-metodi käyttää url-kentän tietoa. Näin saavutetaan rakenne, joka näyttää loppukäyttäjälle kuin mitään erikoista ei olisi tapahtunut, mutta koodi itsessään tulee

joustavammaksi. Url-kenttään lähetetyt php-atribuutit eivät näy ilman osoiterivin valitsemista. Toimintalogiikka kirjoitetaan eri tiedostoihin.

Toimintalogiikka koostuu monesta eri osasta. Ensimmäisenä ja ehkä tärkeimpänä seikkana on kuitenkin tietoturva. Jotta järjestelmästä saadaan tuvallinen, jokainen scripti, luokka tai muu toiminnan osa tulee suojata. Suojaus yleensä toteutetaan joko pääsyn estämisellä tai koodin sekoittamisella tai salakirjoituksella. Jokainen SQL-kysely tulee koota niin, että sitä ei saa käsiteltyä muutoin kuin tarkoitetulla tavalla, eikä etenkään sellaisen käyttäjän toimesta, kenellä ei ole oikeuksia edes käyttää järjestelmää.

PHP tuo mukanaan myös session ja evästeet (cookies), joita käyttämällä käyttäjät voidaan autentikoida. Järjestelmään rakennetaan käyttäjätunnus/salasana tunnistus, joka liitetään sessio- ja evästekäytäntöihin. Koska sivuston näkymät tarjoillaan index.php:n kautta, voidaan sessiokäytäntö kirjoittaa vain index.php-tiedostoon (tai include-metodilla). Eväste-käytäntö otetaan käyttöön, mikäli aloituspalaverin vaatimusmäärittely sen vaatii. Mikäli sivustoon lisätään toiminnallisuutta, joka poikkeaa normaalinäytöistä, on jokainen uusi näyttö suojattava sessiokäytännöllä erikseen. Käyttäjä tarkistetaan tietokannasta. Koska eväste ladataan etäkoneelle, siihen ei tule sisällyttää mitään arkaluontoista tietoa. Sessio pitää sisällään sessio-id:n, jota voidaan mm. käyttää suoraan osoitekentässä tai sisällyttää se evästeeseen. Evästeen pituus tulee rajata tiettyyn aikarajaan.

Käyttäjältä vaaditaan kirjautumiseen käyttäjätunnus sekä salasana. Salasanan uudelleenlähettäminen rajoitetaan kolmeen kertaan, minkä jälkeen käyttäjätunnus lukitaan. Salasanan pituus tulee olla vähintään viisi merkkiä. Lukituksen avaamisen mahdollisuus lisätään joko esimiehen tai järjestelmänvalvojan oikeuksiin tai molempiin. Järjestelmään lisättiin myös uuden käyttäjätunnuksen luominen. Uuden käyttäjän luonnissa haetaan tietokannasta jo valmiiksi saman käyttäjätunnuksen omaavaa käyttäjää. Mikäli tällainen löytyy, järjestelmä estää tunnuksen luonnin tietokantaan sekä tarjoilee käyttäjälle virheilmoituksen. Käyttäjältä vaaditaan salasanan uudelleenkirjoitus, minkä ansiosta voidaan karsia salasanan väärinkirjoittamisen mahdollisuutta. Sekä kirjautumisen sekä rekisteröitymisen hyväksyminen (submit) lähettää tiedot lomakkeenkäsittelijöille, jotka luovat varsinaisen session kaiken muun tarkistuksen jälkeen,

sekä uudelleen ohjaavat käyttäjän järjestelmän varsinaiselle etusivulle. Etusivulla tarjoillaan logout-navigointi, jota käyttämällä käyttäjä kirjataan ulos eli toisin sanoen sessio tuhotaan. Sessioille annetaan alustavasti parametrit username eli käyttäjätunnus sekä status eli session voimassaoloasetus. Sessiolla on oletuksena parametrinaan pelkkä id eli tunnistetieto.

Koska sessiokäytäntö otettiin käyttöön, ei käyttäjä tai mahdollinen haittatekijä, voi käyttää osoiterivin tietoa navigoidakseen järjestelmässä, mikäli hänellä ei ole voimassa olevaa sessiotunnistetta. Session parametrit eivät saa sisältää mitään arkaluontoista tietoa käyttäjästä, vaan käyttäjä liitetään tietokannassa omiin tietoihinsa myöhemmin. Lomakkeiden suojauskäytäntönä käydään läpi kaikki tulevat POST- ja GET-metodien lomakkeilta tuomat arvot. Arvot käydään läpi erikoismerkkien tai mahdollisten muiden yritysten estämiseksi, mm. pelkkä tyhjä rivi. Uuden käyttäjän luonnissa tullaan tunnuksien pituudet määrittämään vaatimusmäärittelyn mukaisiin pituuksiin. Salasanakäytäntö tullaan varmentamaan niin, että kaikki salasanat käännetään md5-koodauksen taakse ja salataan käännöksen jälkeen. Mahdollinen lisäsuojaus koodataan käsittelijöihin myöhemmin.

Koska järjestelmä käsittää esimies- sekä alaissuhteita käsittäviä tietoja, tulee järjestelmään tehdä käyttäjätasot. Käyttäjätasoiksi toteutukseen laaditaan admin- ja user-käyttäjätasot eli esimies- ja käyttäjätasot. Tasoja tehdään oman vaatimusmäärittelyn mukaisesti vain kaksi tasoa, jotka vaaditaan kriittisten toimintojen suorittamiseen. Esimerkiksi vain lukutasoa tms. ei tarvita. Koska järjestelmältä vaaditaan tietynasteista dynaamisuutta, käyttäjätasot toteutetaan kuitenkin niin, että mahdollinen monitasotunnistautuminen mahdollistetaan. Tähän tarkoitukseen tietokantaan tulee rakentaa kolme taulua käyttäjätunnuksien tasoihin liittämistä varten, yksinkertaisen oikeusmallin kahden taulun sijaan. Edellä mainittu rakenne mahdollistaa siis monen käyttäjän liittämisen moneen oikeuteen. Tietokannan rakenteeseen koodattiin tunnistekenttä, jonka ominaisuus auto-increment eli automaattinen arvon nostaminen takaa ettei samaa tunnistetta ole, eikä tule olemaan, eri käyttäjällä. Mikäli käyttäjä esimerkiksi tuhotaan, samaa tunnistetta ei oteta enää käyttöön. Automaattinen arvon nostaminen siis nostaa arvoa käyttäjiä lisättäessä äärettömyyteen asti, mutta koska käyttäjäryhmä koostuu alustavasti noin 250 yksilöstä, ei laajuus tule täyttymään milloinkaan järjestelmän käyttöiän aikana vaikka käyttäjien välisiä tyhjiä arvoja tulee lisää.

5.2 Palaveri 2

Jatkopalaverissa esikoodauksessa huomattiin muutamia virheitä ja puutteita. Työnkuvaus linkki todettiin harhaanjohtavaksi kyseisellä tasolla, joten se muutettiin nimelle ”Työn vaativuuden arviointi”. Työnkuvauslomakkeet näytetään vasta kyseisen sivun auettua. Työnkuvaus lomakkeet ovat ennalta määrättyjä/suunniteltuja sekä niiden lukumäärä on alustavasti 34 kappaletta. Koska järjestelmä ei voi ennalta arvata kullekin työntekijälle tarkoitettua työnkuvausta, tulee työnkuvauslomake valita jostakin erikseen. Työnkuvaus avataan alustavasti ”työn vaativuuden arviointi”-linkin jälkeen avattavasta pudotusvalikosta. Työnkuvauslomake voitaisiin myös päätellä työntekijän ammattinimikkeestä, joka täytettäisiin perustietoihin, mutta koska ammattinimikkeiden suuri määrä estää niiden listauksen järkevällä tasolla, on nimikkeet jätettävä vapaaseen kirjoitusasuun ja työnkuvaukset eriytettävä valittavaksi erikseen.

Etukäteen laadittuja työnkuvauksia oli edellä mainittu määrä. Tiedostot olivat word-tiedostoja eli .doc-tiedostoja. Tästä johtuen päädyttiin ongelmaan, jossa word-dokumentin koodaus ei näy oikeanlaisena html-muotoisena. Vaihtoehtoina ongelmanratkaisuun oli kaksi vaihtoehtoa. Joko tiedostot konvertoidaan oikeaan muotoon tai tiedoille tehdään oma tietokanta, josta teksti haetaan sivulle. Konvertoinnissa oli kuitenkin se ongelma, että tiedostojen oikean muotoinen ulkoasu ei pysy hallinnassa sekä tiedostojen lisäys, poisto tai muokkaus ovat painajaismaisia toteuttaa, koska tällöin jouduttaisiin muokkaamaan itse tiedostoa.

Mikäli tiedot tallennetaan tietokantaan ja ne määrätään ennalta määrättyihin paikkoihin html-dokumenttiin tuodessa, voidaan esimiehille koostaa hallintatyökalu, jolla voidaan muokata tietokannan tauluja turvallisesti suoraan sovelluksesta. Tällöin järjestelmän ylläpitäjän ei tarvitse koostaa tauluja alkukappaleiden jälkeen vaan vastuu voidaan siirtää esimiehille. Jo toteutettua käyttäjätaso-ominaisuutta voidaan hyödyntää tässä tapauksessa antamalla vain tietyille henkilöille järjestelmänvalvojan toimesta valtuudet muokata työkuvauksia. Näin ollen edes kaikki esimiehet eivät ole velvoitettuja pitämään tietoja ajan tasalla, vaan ainoastaan ne käyttäjät, joille oikeudet annetaan. Kyseinen käyttäjä voi järjestelmän juostavuuden ansiosta luonnollisesti olla myös työntekijä. Työnkuvaukset päätettiin tehdä tietokantaan.

Henkilökohtaisen työsuorituksen arviointi oli väärin ymmärretty järjestelmän laatijan toimesta. Arviointia ei tee työntekijä itse, vaan sen tekevät esimies ja yksi ulkopuolinen taho. Ulkopuolisella taholla viitataan organisaation työtekijään joka ei ole suoraan sidoksissa kyseisiin henkilöihin tai tehtäviin. Henkilökohtaisen työsuorituksen arvioinnin lomakkeen muokkaaminen rajoitetaan siis esimiehen käyttöoikeuksiin, mutta data itsessään näytetään työntekijälle. Lomakkeesta puuttui myös sarakkeet: kuka työsuorituksen on arvioinut ja koska kehityskeskustelu on käyty. Kyseiseen lomakkeeseen tulee mahdollisesti vielä muuta dataa tai viittauksia, mutta näistä päätetään vasta tulevissa palavereissa.

Käyttöliittymässä ei ollut mainittavia puutteita sekä käyttäjätunnistautuminen esikoodauksessa määritetyillä tasoilla vastasi tarpeita riittävästi. Käyttäjätunnistautumiseen sekä kirjautumiseen tullaan vielä tekemään tietoturvaa edistäviä muokkauksia.

5.3 Työnkuvauksien hallinta

Työnkuvauksien hallinta päätettiin koodata erilliseksi komponentiksi esimiesten tai muiden tätä ominaisuutta käyttävien tahojen piiriin. Käyttöliittymää pyöriteltiin mahdollisten toimivien vaihtoehtojen kannalta. Aluksi Käyttöliittymä koodattiin yksinkertaiseen muotoon, jonka jälkeen tätä muotoa jatkojalostettiin toimintamallien mukaisesti mm. testauksen kautta, käytettävimpään kuvan 8 mukaiseen malliin.

Kuva 8. Työnkuvauslomake.

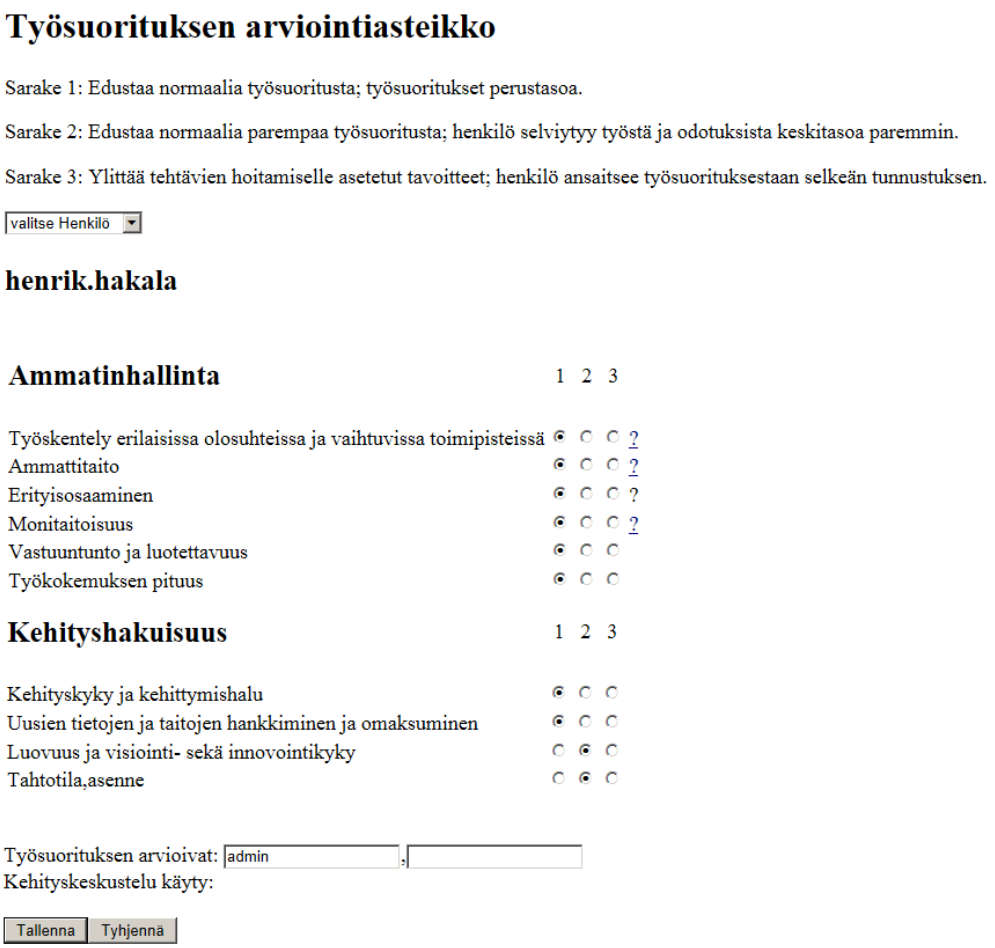
Lomakemalli koostuu työnkuvauksen valintakomponentista, uuden kuvauksen luontilinkistä, täytettävistä tekstikentistä sekä tallennus- ja tuhoamispainikkeesta. Lomakkeen tuhoamispainikkeelle luodaan varmistuskysely, jolla varmistetaan, ettei käyttäjä ole vahingossa pyytänyt poistoa. Valintakomponentti on alaspudotusvalikko-komponentti, joka populoidaan tietokannassa olevilla työkuvauksilla. Valikosta avautuvasta listasta valittaessa kyseisen kuvauksen tiedot päivittyvät suoraan lomakekenttiin. Jos valitaan ”uusi kuvaus”, kentät tyhjennetään, mutta mitään ei vielä kuitenkaan kirjoiteta tietokantaan. Muutokset jo olemassa oleviin kuvauksiin voidaan tehdä suoraan muokkaamalla tekstiä. Kun uusi tai muokattu työkuvaus on valmis, tallennetaan se tietokantaan painamalla tallenna-nappia.

Työnkuvauslomake koostuu kahdesta php-tiedostosta, jotka kommunikoivat keskenään eli itse hallinnan käyttöliitymäkäsittelytiedosto sekä tiedon varsinaisen käsittelyn ja lähettämisen tiedosto (ts. script-tiedosto). Näin ollen kirjoitettava koodi muodostaa hieman kaoottiselta vaikuttavan tiedon edestakaisen siirron. Koska työnkuvauksen valintalista populoidaan suoraan tietokannasta, jouduttiin listan html-lauseet eli option-kentät koodaamaan php-toistorakenteen sisään. Toistorakenteessa voidaan näin ollen tulostaa kuvauksen nimi noutamalla se sql-tietokannasta. Script-tiedosto käsittää myös muut työkuvauksen toimintojen käsittelyn metodit. Lomakkeen nimen valinta joudutaan uudelleenlähettämään html-tiedostosta script-tiedostoon, jotta kyseisen kuvauksen tiedot voidaan tulostaa tekstikenttiin tarkastelua ja muokkausta varten. Myös muissa lomakkeissa, joissa käytetään esitäyttöä, tullaan tekemään koodi samalla tavalla, mutta on muistettava tietoturvaseikat sekä tiedon eheys.

5.4 Henkilökohtaisen työsuorituksen arviointi

Henkilökohtaisen työsuorituksen arviointi-lomake koostuu neljästä elementistä. Elementit ovat henkilön vaihto, valintaruudut, pisteytystaulut sekä arvostelijakentät. Koska arvioinnin tekee yhtenä osapuolena esimies ja toisena kolmas osapuoli, on arviointilomakkeen käsittely rajoitettu esimiesoikeuden piiriin. Käytännössä tämä tarkoittaa kolmannen osapuolen fyysistä läsnäoloa esimiehen kanssa. Esimiehen nimi sekä muut tarvittavat tiedot lomakkeeseen tuodaan suoraan tietokannasta valmiiksi.

Kolmas osapuoli täyttää tietonsa lomakkeeseen erikseen. Henkilön vaihto on alaspudotusvalikko, jossa henkilöt listataan, ja josta valitsemalla henkilön tiedot tulostetaan kenttiin mahdollisia muutoksia varten. Listaus laaditaan järjestykseen, mikäli koodi ei näin oletuksena tee. Tietoihin lisätään mahdollinen päiväyskenttä vasta jatkopalaverin jälkeen.



Kuva 9. Työsuorituksen arvioinnin lomake.

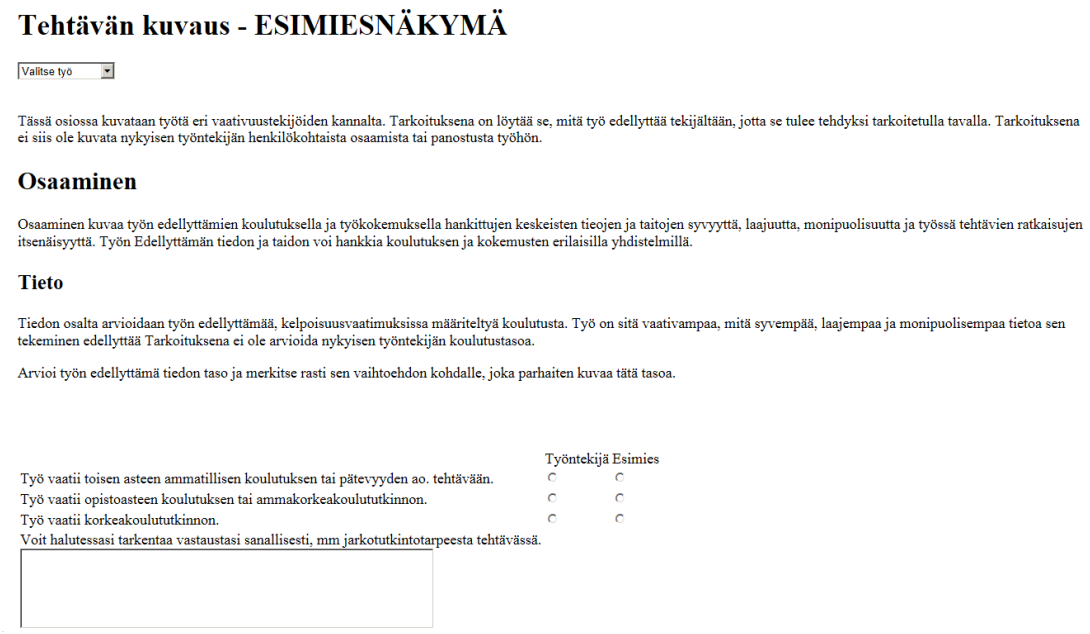
Kuvan 9 mukainen lomake koostuu lukuisista radio-painikkeista, joiden laatiminen html-sivuksi sekä näiden script-tiedostoon koodaaminen tietokantaa varten muodostui suureksi työksi. Koska tietokantaan tallennetaan yksiselitteistä dataa, on jokaisesta komponentista tuleva tieto tallennettava erikseen taulun eri kenttiin. Samat nimikkeet edellä mainitun tilanteen mukaisesti oli siis kirjoitettava niin tietokantaan, lomakkeenkäsittelijään kuin itse lomakkeeseenkin. Johtuen lomakkeen rakenteesta, jouduttiin data kirjoittamaan tietokannan tauluun niin että jokaiselle sarakkeelle tuli oma tunniste, eikä yhden sarakkeen periaatteella kuin mallien mukaisesti olisi pitänyt

tehdä. Tästä syystä myös data kerääminen mm. käsittelijässä silmukoimalla muodostui ongelmalliseksi. Tietueet kirjoitettiin purkamaan tieto sarake kerrallaan.

Pisteytystaulut ovat osittain kyseisestä lomakkeesta irrallisia komponentteja. Koska pisteytykset saattavat vaihdella eri henkilöittäin tai aloittain, sekä relaatiomallin mukaisen yksiselitteisyyden mukaan, ei niitä voi kirjoittaa samaan tauluun. Pisteet kuitenkin tullaan kirjoittamaan eri tietokantaan tai niille laaditaan parempi käyttöliittymä/hallintatyökalu, mutta työmäärän ollessa edelleen kohtalaisen suuri, priorisoitiin kyseinen seikka alhaiselle tasolle vaikka kyseessä on yksi perustoiminnallisuuden osista. Pisteiden normaali laskeminen kuitenkin koodataan käsittelijään normaalisti. Laskun loppusumma näytetään laatijalle ja tieto tallennetaan tietokantaan. Pisteytys näytetään oletettavasti kohdistetulle henkilölle, mutta näyttäminen vahvistetaan jatkopalaverissa, jossa myös esitetään pisteytyksen käyttäytyminen ja näyttäminen, sekä mahdolliset korjausvaatimukset. Lomakesivu käyttää jo mainittuja tietokantayhteyksiä sekä suunniteltuja käsittelijärunkoja datan siirtoon.

5.5 Työn vaativuuden arviointi

Työn vaativuuden arvioinnin ensimmäinen osuus koostuu kuvan 10 laisesta alaspudotusvalikosta, josta valitaan oikeanlainen työnkuvaus esimiesten esitäytettyjen nimikkeiden avulla. Kun työnkuvaus valitaan, tulostetaan näytölle kaikki taulun tieto eli tekstikentät sekä nimi. Php ei kuitenkaan osannut suoraan tuoda tekstiä tietokannasta oikean muotoisena, koska tuotava data käyttää järjestelmätason rivinvaihtoja. Php tarjoilee kuitenkin asiaa helpottavan funktion nl2br, jolla saadaan kivuttomasti data oikean muotoiseksi tulostukseen (PHP.net 2011). Tulostetta ei siis tässä lomakkeessa voida muokata.



Kuva 10. Alaspudotusvalikko ja täyttökentät.

Työn vaativuuden arviointi-lomake oli jo hahmoteltu koodiin paperilomakkeen laiseksi, kunnes törmättiin suuriin suunnitteluvirheisiin. Koska työn vaativuuden arviointi vaatii työn valitsemisen pudotusvalikosta, näkyvät kaikki irrelevantitkin työt työntekijälle. Koska lomakkeenkäsittelijä tullaan sitomaan näkyvyyteen, työntekijöillä olisi mahdollisuus arvioida kaikkia töitä. Tätä ei tietenkään voi sallia jo pelkästään turhan datan syntymisen takia. Kaikkien töiden arviointi voidaan toki sallia esimerkiksi ryhmätunnukselle tai luoda erikoisoikeus, jos tarvetta syntyy. Itse paperilomake koostuu ”kysymystauluista”, joihin tulee sekä esimiehen, että työntekijän merkata yksi kolmesta vaihtoehdosta, minkä jälkeen voidaan tehdä vertailua. Koska esimiehen valintaa ei voi ennen työntekijän valintaa näyttää, on sovellukseen koodatussa taulussa myös turhia sarakkeita. Kaikki sarakkeet tulisi näyttää vasta jatkokeskusteluun siirryttäessä, jotta osapuolien kantaa ei voi prosessin aikana muuttaa. Paperiversion huonona puolena on, että esimies näkee työntekijän vastauksen ennen kuin kirjoittaa omansa. Arviointisovellus taasen antaa mahdollisuuden piilottaa kaiken datan, kunnes kaikki arviot on laadittu. Järjestelmän laatijan kannalta esimiehen ei tule nähdä alaisensa vastauksia ennen kuin on tehnyt omansa, jotta vaikuttamisen mahdollisuus voidaan eliminoida. Web-lomakkeen tulisi näyttää työntekijälle vain hänen oma työnsä, joka on joko määrätty tunnuksenluonnin jälkeen työnantajan toimesta, tai jonka työntekijä valitsee valikosta täyttäessään muita tietojaan perustietoihin. Lomakkeen tulee myös tallentaa moniulotteinen vastaus tietokantaan, jonka avulla voi-

daan laskea pisteitä. Moniulotteisuus syntyy koska vastauksille voidaan antaa eri pisteitä eri kysymyksien ja valintojen perusteella.

Ongelman korjauksen ensimmäinen vaihe käsittää työn valinnan siirtämisen muualle. Koodi päätettiin vaihtaa niin että sekä työntekijälle sekä esimiehelle (tai arvosteluoikeuden saaneelle) laaditaan oma näkymä. Työntekijä ilman erikoisoikeuksia ei näe, eikä voi arvioida muita kuin oman työnsä. Mutta koska työt on kuitenkin kaikki listattava, on otettava huomioon että käyttäjä voi tahattomastikkin valita väärän työn tai ei edes tiedä mitä on tarjolla tms. Tästä johtuen valintaan on koodattava käsittelijä, joka poistaa työarvion sekä muut mahdolliset kyseiseen työhön liittyvät tietokantaan talletetut arvot, mikäli perustietojen työtehtävää muutetaan. Esimiehet näkevät kaikki työt ja voivat arvioida kaikkia töitä kuvan 10 kenttien mukaisesti. Esimiehien näkymä voidaan rajata työntekijöidensä töihin, mutta rajaus tehdään vain jos asiakas sitä vaatii. Mahdolliset esimiesten esimiehet näkevät kaikki työt. Esimiehen lomakkeenkäsittelijöinä voidaan käyttää samaa käsittelijää kuin työntekijälläkin. Lomakkeiden rakenne kuitenkin estää turhan tiedon näyttämisen edes esimiehen esimiehelle, toisin sanoen kaikki käyttäjät toimivat omina yksiköinään, joille annetaan erikoisoikeuksia. Myös vertailu tehdään koostetun datan pisteytyksen perusteella, joten ristiriitoja ei pitäisi syntyä.

5.6 Palaveri 3

Palaveri käytiin johtavan hoitajan kanssa. Tarkoituksena oli selventää arviointiprosessin loppuosa koodausta varten jotta suurimmilta mahdollisilta suunnitteluvirheiltä vältyttäisiin. Suuri osa arvioinnin kulusta on johtavan hoitajan käsittelyalueella. Järjestelmän sivut esiteltiin komponenteittain ja vaikka läheskään kaikki toiminnallisuus ei ollut valmiina, voitiin käyttöliittymästä nähdä miten järjestelmä toimii tulevaisuudessa. Muutamia puutteita löydettiin.

Suurimpana lisättävänä seikkana tuli ilmi arkistoinnin tarve. Koska järjestelmää arviointiprosessi toistuu vuosittain, tulee jokainen lomake tallentaa tietokantaan. Arkiston pohjalta voidaan myös tehdä tilastointia. Koska nykyinen tietokanta on laadittu pitämään vain yhtä lomakkeenlähetysdataa sisällään kerralla, tulee siihen kirjoittaan

joko lisätunnisteita tai luoda uusia aputauluja. Tieto voidaan mahdollisesti siirtää kokonaan myös arkistointitauluihin. Tietokantamuutokset määritellään ja suunnitellaan myöhemmin, koska sovelluksen runko ja käyttöliittymä pitää ensin saada kokonaan valmiiksi. Nämä pitää myös merkitä tunnisteilla mm. id, aika. Arkistoitavat tiedot tulisi näyttää niin esimiehelle kuin työntekijällekin. Suurin huoli syntyy, jos tietokannan tauluihin joudutaan lisäämään kenttiä. Tällöin lomakkeenkäsittelijän koodi on muutettava vastaavanlaiseksi tai sovellus lakkaa toimimasta odotetulla tavalla.

Perustietoihin tulee lisätä sosiaaliturvatunnus kokonaisuudessaan käyttäjien tunnistusta varten. Vaikka käyttäjätunnus erottelee arvioitavat käyttäjät, tulee henkilön tunnistaminen tehdä jatkokäsittelyä varten. Koska henkilötunnuksen tarkistaja on erittäin työlästä valmistaa ja se välttämättä tarvitaan, kopioidaan se suoraan valmiina funktiona järjestelmään. Mikäli tunnus on väärä, menevät tietokannan rivit sekaisin. Käyttäjätunnus on jo tässä vaiheessa liitettynä erillisenä tietona perustietoihin tietoturvasyistä. Perustietoihin lisätään myös työyksikkö hallintaa varten. Näin ollen voidaan erotella työntekijät myös yksikön perusteella, koska kaikki kunkin työtehtävän suorittajat eivät toimi samassa paikassa. Paperidokumentoinnissa kehityskeskustelun liitteenä oleva kehityskeskustelun yhteenvetolomake (Liite 2) koodataan kehityssuunnitelmaotsikolle.

Lomakkeille on mietittävä mahdollinen ratkaisu automaattista tallennusta varten, mikäli selain suljetaan tai muu mahdollinen odottamaton käyttäjän toimi suoritetaan. Automaattitallennusta mietittiin kuitenkin vaadittavaksi lähinnä työsuorituksen arvioinnissa. Jokainen lomake on koodattu niin että tiedon tallennus vaatii käyttäjävahvistuksen. Toisaalta taas esimerkiksi selaimen sulkeminen tai sivunvaihto ei automaattisesti tarkoita sitä että käyttäjä olisi tietonsa halunnut tallentaa. Ongelmaa tarkastellaan myöhemmässä vaiheessa. Selaimen sulkeutuessa tulee käyttäjän sessio tuhota kokonaisuudessaan, jotta sen avulla ei enää päästä automaattisesti järjestelmään sisään vaan käyttäjän tulee syöttää tunnuksensa ja salasanansa. Työntekijän tulee nähdä hänestä esimiehensä tekemät työsuorituksen arvioinnit. Myös tähän tarvitaan arkistointiominaisuutta.

5.7 Tietokanta

Tietokanta on tietyllä tavalla järjestetty joukko dataa. Data on yleensä järjestetty käytännöllisen mallin mukaan, niin että rakenne palvelee varsinaisen järjestelmän prosesseja, jotka tietoa hakevat ja käsittelevät. Kaikkien käytettävien sovellusten ja järjestelmien tietovarastona on tietokanta, jos tietoa aiotaan säilyttää ohjelman suorituksen päätyttyä. Tietokantaa käsitellään tietokannanhallintasovelluksella ts. DBMS, jonka tarkoituksena on hallita normaalistikin erittäin monimutkaisia tietorakenteita. Tietokannanhallintasovellukset on yleensä koottu sovelluspaketeiksi, joista kuuluisimpina Oracle, SQL Server, MySQL ja DB2. Tallennettu data ei yleensä ole suoraan käytettävissä toisessa tietokannanhallintasovelluksessa, mutta dataa voi halutessaan käsitellä ja liikuttaa standardeja, mm. SQL tai ODBC, käyttäen, mutta kuitenkin jokaisen tietokannanhallintasovelluksen hallitessa omaa tietokantaansa. Hallintasovellusta tarvitaan myös tarjoilemaan tehokkaan saatavuuden, hakuajan ja tietoturvan niin monelle käyttäjälle kuin mahdollista. (Wikipedia 2011/e)

Tuetuin tietokantakieli on SQL, jonka tarkoituksena on relaatiomallin mukaisien tietokantojen käyttö. SQL yhdistelee datamäärityskielen, datakäsittelykielen ja datakyselyjen rooleja. Relaatiomalli pohjautuu predikaattilogiikkaan. Relaatiomallin tarkoituksena on kuvailla datan käyttöä mutta ei varsinaisesti näyttää miten jokin asia tehdään. Käyttäjät määrittelevät mitä dataa tallennetaan ja mitä tietoa siitä halutaan. Tämän jälkeen asia annetaan tietokannanhallintasovellukseen, joka huolehtii tiedon rakenteiden kuvauksesta ja hakumenettelyistä. (Wikipedia 2011/e)

Arviointijärjestelmän perustana toimii tietokannan hallintajärjestelmä MySQL sekä sen ns. käyttöliittymä phpMyAdmin. MySQL:n oletustietokantamoottori MyISAM ei vaadi relaatioita tai relaatiomallin mukaista muuta rakennetta toimiakseen. Arviointijärjestelmän perustalle kuitenkin otettiin MyISAM-moottori sen suorituskyvyn takia. MySQL:n avulla voidaan rakentaa täysiä relaatiotietokantoja, mutta tämä toimenpide vaatii tietokantakantamoottorin vaihdoksen esim. InnoDB:hen. Arviointijärjestelmän tietokantaa rakenneltiin ja suunniteltiin jo varhaisessa vaiheessa, jotta päästiin testaamaan tarvittavia kyselyitä lomakkeiden toimivuutta testattaessa. Tietokantamoottorin vaihto edellyttää tietokannan rakenteiden tarkistuksen, jonka vuoksi toisen tietokantamoottorin vaihto saattaa koitua kohtalokkaaksi.

Suurin uhka tietokannan perustamisessa huomattiin kuitenkin olevan sen eheys. Arviointijärjestelmän tietokannan rakenteen laajetessa huomattiin että määritys ja suunnittelu oli laiminlyöty monella taholla. Tietokannan rakenne on lähes suoraan yhteydessä lomakkeisiin ja sitä kautta näyttöihin. Vaikka tietokantaa ei tarvitse käytännössä reaalitietokannaksi laatia, tulee sen täyttää tietyt rakenteelliset vaatimukset. Taulujen nimeäminen olisi tullut suorittaa perustasolla. Nykyinen tietokanta käsittää jokaiselle lomakkeelle omat taulunsa, mutta kuitenkin niin että relaatiomallin mukaiset suhteet ovat kunnossa (esim. ei monen suhde moneen yhteyksiä vaan käytetään aputauluja). Tieto on oltava yleisessä muodossa ja saatavilla helposti hauissa. Vaikka osa tauluista voidaan nähdä aputauluina eli niitä ei välttämättä suoraan käytetä, on perustaulujen oltava nimettynä asianmukaisesti. Arviointijärjestelmän tietokannan taulujen nimeämiskäytäntö alkoi huolestuttaa kun uudelle taululle täytyi antaa numeerisia lisäarvoja.

Kun ensimmäiset taulut oli laadittu, linkitetty ja testattu eli siis sisälsivät jo dataa, huomattiin että seuraavat taulut ja niiden tiedot saattoivatkin sisältää tietoa, jonka olisi ollut tarkoitus olla toisessa taulussa. Tämä aiheutui mm. paperiversion kolminkertaisesta tiedonkyselystä (Liite 3), jossa pisteytyksenkin tulisi olla dynaaminen, jolloin tietokantaan on laadittava useita tauluja tämän tiedon tallentamiseen. Kun tiedot tallennetaan tällaisenaan tauluihin, on järkevä arkistointi vaikeaa. Tieto tarvitsisi kirjoittaa uudelleen joko arkistointimoottorilla käytettäväksi tai paremmin käytössä oleviin tauluihin. Koska taulujen muutos arvioitiin todella työlääksi ja virheiden määrä olemaan suuri, on tietokanta määritettävä, suunniteltava sekä kirjoitettava kokonaisuudessaan uudelleen, mistä johtuen osa lomakkeenkäsittelijöistä joudutaan korjaamaan. Vanhoista tauluista otetaan luonnollisesti kaikki tarvittava tieto talteen. Arviointijärjestelmän toteuttaminen keskeytettiin opinnäytetyön aikataulun täydellisen myöhästymisen vuoksi. Pelkästään tietokannan korjaukseen sijoitettava panostus olisi ylittänyt aikaresurssin, joten päätettiin viivästyttää sovelluksen toteuttamista.

6 TESTAUS

Testauksen tarkoituksena on löytää ohjelman häiriöitä. Häiriöt johtuvat vioista ja virheistä. Järjestelmän tulee toimia. Testausta tehdessä panostus tarkoituksenmukaiseen ja päättäväiseen määrittelyyn ja suunnitteluun voidaan havaita. Tuntemattomien tekijöiden, yleensä vaatimuksien, testaamattomuus aiheuttaa jälkikäteen suuria kustannuksia virheidenetsinnässä ja -korjauksessa jos vian häiriöt vyöryvät muuhun järjestelmään. Testaus järjestelmään tehdään joko sen ollessa valmis tai kaikkien sovelluksen kehityksen vaiheiden aikana. Testaamisen yleisesti käytetyt tasot ovat yksikkötestaus, integraatiotestaus, järjestelmätestaus, järjestelmän integraatiotestaus sekä alpha- ja betatestaus, joissa testaus tuodaan jo lähelle järjestelmän loppukäyttäjiä. Kun testaus on suoritettu edellä mainituilla tasoilla, tehdään vielä regressiotestausta, jossa suoritetaan korjatut, testeissä virheitä antaneet kohdat. (Wikipedia 2011/f)

6.1 Testiprosessi

Perinteinen tapa testata on järjestelmän valmistuttua valita itsenäinen joukko käyttäjiä, jotka testaavat järjestelmän ennen sen toimittamista asiakkaalle. Testausvaihetta valitettavasti usein käytetään puskurivyöhykkeenä varsinaisen toteutuksen viivästymisen jatkeena. Tästä johtuen itse testausvaihe jää usein erittäinkin lyhyeksi tai laiminlyödään kokonaisuudessaan. Testaamista voidaan aloitella tekemään kuitenkin jo projektin alettua, ja jatkaa kunnes projekti on valmis. Testausta voidaan tehdä myös hieman uudemmilla tavoilla, joissa ohjelmiston testaus on ohjelmistolähtöistä. Tässä tavassa testitapahtumat kirjoitetaan ensin, mutta kuitenkin yleensä varsinaisen sovelluksen rinnalla. Tämän kaltaiset testit aiheuttavat jo oletuksena virheitä. Kun järjestelmä kehittyy ja laajenee, testitapaukset muotoutuvat monipuolisimmiksi niihin lisättäessä lisää mahdollisia virhetilanteita ja reunaehtoja. Yksikkötestit pidetään osana järjestelmää ja käännetään varsinaisen sovelluksen mukana. Tarkoituksena on saada aikaan jatkuvaa kehitystä, jonka ansiosta päivityksiä voidaan julkaista niin nopeasti kuin mahdollista ja näin ollen taata turvallinen ja luotettava järjestelmä. (Wikipedia 2011/g)

Useimmiten eri yrityksillä on eritasoiset vaatimukset ja toteuttamistavat testaukseen. Voidaan kuitenkin väittää että tyypillinen testaus tapahtuu yleensä tiettyjen normien, joita ilman hyvää testaamista ei voi tehdä, puitteissa. Testaaminen tulisi aloittaa vaatimusmäärittelyllä, jonka aikana testaajat työskentelevät sovelluskehittäjien kanssa päätettäessä mitä testataan ja millä tavoin. Testille tulee laatia suunnitelma, josta ilmenee mm. strategia, itse suunnitelma ja testiympäristö. Testejä tulee kehittää prosessien, tapahtumien sekä scriptien osalta. Testejä suoritetaan suorittamalla järjestelmän toimintoja ja näitä raportoidaan eteenpäin ohjelmistokehittäjille. Testauksesta laaditaan raportteja, joiden pohjalta päätetään onko sovellus valmis julkaistavaksi sellaisenaan. Kehitystiimi analysoi raportit ja tekee varsinaisen päätöksen. Päätöksenteossa on yleensä mukana myös asiakkaan edustaja. Jos järjestelmä ei raporttien mukaan suoriudu testeistä kehitystiimi hoitaa järjestelmän korjauksen, jonka jälkeen se uudelleen testataan. Yleensä sovelluksien testaamiseen käytetään pientä sovellusta, jonka on käytännössä sama kuin edellisellä testauskerralla. Tällöin voidaan olla varmoja, etteivät tehdyt korjaukset ole aiheuttaneet vääriä muutoksia tai lisää virhetilanteita. Mikäli testit eivät mene läpi, kehitystiimi ottaa järjestelmän uudelleentarkistettavaksi. Kun testit suoriutuvat hyväksyttävien kriteerien rajoissa sovellus toimitetaan asiakkaalle sekä avainsyötöt, opitut asiat, lokit sekä muut projektiin liittyvät dokumentit arkistoidaan tulevaa käyttöä varten. (Wikipedia 2011/g)

6.2 Motiivit testaukseen

KSTHKY:llä ei käytännössä ole mitään standardeja ohjelmistojen testaukseen. Ohjelmistotestauksen tärkeyttä ei kuitenkaan tule aliarvioida. Pieninkin vika saattaa eskaloituessaan aiheuttaa suuren rahallisen tappion. Testauksessa löydettyjen vikojen korjaus saattaa aiheuttaa totaalisen viivästymisen. Vaikka arviointijärjestelmään ei periaatteessa sijoiteta ollenkaan rahaa, tulee testaus läpäistä kohtuulisin normein jo pelkästään käytettävyyden ja tietoturvan tähden. Käytännössähän jo pelkän laitteiston käyttäminen maksaa, ja mikäli ohjelmistoa ei esimerkiksi jostakin virheestä johtuen päästä käyttämään, laitteisto istuu tyhjän panttina. Myös henkilöstöaineiston viivästyttäminen ohjelmistoa varten aiheuttaa suman, jos arviointijärjestelmä ei tulekaan valmiiksi, tämä sumautunut työ täytyy purkaa. Näistä seikoista johtuen, ei testeistä läpäisemättömiin seikkoihin käytännössä reagoida heti, vaan ohjelmisto julkaistaan,

elleivät viat ole peruskäytön kannalta vakavia. Testaustiede on pääosin erittäin tarkkaa ja systemaattista toimintaa. Arviointijärjestelmän ominaisuuksien takia pyrittiin kuitenkin olemaan hieman suurpiirteisempiä. Vikojen korjaus tulee suorittaa julkaisun jälkeen vaikka se tarkoittaisikin sovelluksien uudelleenkäynnistämisiä. Tämä aika on kuitenkin pienempi kuin aika, joka venähtäisi pidemmäksi jos virheiden korjaukseen ja järjestelmän julkaisun pitkittämiseen käytetään koko ajan suureneva määrä työaikaa koska halutaan korjata aina vain enemmän vikoja.

6.3 Testauksen toteutus

Testejä varten valitaan noin viisi henkilöä. Testihenkilöt ovat tässä tapauksessa asiakkaan työntekijöitä koska projekti tehdään yhden hengen voimin. Toki omaa testaamista on suoritettu koko sovelluskehityksen ajan, mutta koska sovelluskehittäjä itse tulee usein hyvin sokeaksi omille virheilleen ja osaa käyttää ohjelmaa niin kuin on tarkoitettu, useat virheet jäävät huomaamatta. Testejä varten testiryhmän olisi saatava koulutus käyttöä varten, mutta koska halutaan testata myös opastavaa käyttöliittymää, on testiryhmä pidettävä epätietoisuudessa. Näin ollen testaajat voivat antaa palautetta käytettävyydestä, luettavuudesta, luotettavuudesta yms. Testattavat kohteet arvotaan, jonka jälkeen jokainen testaaja testaa koko järjestelmän niin perusteellisesti kuin suinkin osaa. Lopuksi testaajille opastetaan kaikki järjestelmän toiminnallisuudet, jotka saattavat jäädä varsinaisista testeistä ulos esimerkiksi huonon käytettävyyden tai väärän informaation vuoksi. Testitapauksia eli esim. lomakekenttiin syöttöjä voitaisiin myös generoida ohjelmallisesti lähes kaikki mahdolliset tapaukset, jolloin päästäisiin lähelle suurempaan kattavuutta mutta koodimäärän ollessa jo varsin suuri, ei kyseiseen toimenpiteeseen ryhdytä.

6.4 Hyväksymistestaus

Kun järjestelmän testaus läpäisee vaatimukset, voidaan se hyväksyttää asiakkaalla. Hyväksymistestaus käsittää käyttöliittymän, muutaman testitapauksen sekä muun toiminnallisuuden esittämisen johdolle. Mikäli johdon edustaja hyväksyy perustoiminnallisuuden sekä muut ominaisuudet, voidaan siirtyä koulutukseen. Jos järjestel-

mä ei läpäise asiakkaan vaatimuksia, järjestelmän viat tai puutteet kirjataan ylös sekä tehdään uudelleen aikataulutus mahdollisimman nopealle syklille esimerkiksi kuukaudelle.

7 KÄYTTÖ JA YLLÄPITO

7.1 käyttökoulutus

Koulutukseen varataan järjestelmän pääkäyttäjille eli esimiehille tunnin mittainen koulutus liittyen järjestelmän käyttöön. Pääkäyttäjät opastavat tarvittaessa alikäyttäjiä eli työntekijöitä järjestelmän käytössä. Koska järjestelmä on verrattain pieni sekä siihen implementoitu ohjeistus on varsin kattava, voidaan koulutuksen tarve arvioida minimaaliseksi. Koulutuksen järjestää sovelluksen laatija.

7.2 Päivitykset

Järjestelmä tulee päivittää kaikkien komponenttiensa osalta viimeisimpiin mahdollisiin versioihin ennen testauksen alkua. Päivitykset on myös mahdollistettava järjestelmän valmistuttua.

7.2.1 Järjestelmäalusta

Järjestelmäalustan päivitys sisältää sekä laitteistopäivitykset että sovelluspäivitykset. Mahdollinen laitteistopäivitys voidaan tehdä helposti laajentamalla virtuaalipalvelimen kapasiteettia mm. muisti, laskentateho. Päivitys kuitenkin vaatii vähintään virtuaalikoneen sammutuksen eli toisin sanoen käyttöjärjestelmä sammutetaan. Itse käyttöjärjestelmän päivitys voidaan järjestää käyttämällä kuntayhtymän voimassa olevia lisenssejä, joita joka tapauksessa tulee olemaan tarvittava määrä. Vaikka koko virtuaalipalvelin varmistetaan toisaalle, joten se on tätä myöden palautettavissa, ei tällaiseen tilanteeseen kuitenkaan suoraan haluta päätyä. Arviointijärjestelmän komponentit kopioidaan tilapäiseen tietovarastoon ennen kuin päivitys suoritetaan. Tietovarasto

eli tiekanta kopioidaan aina ensimmäisenä, jotta varmistutaan että data ei häviä. Jokainen asetus pyritään dokumentoimaan mahdollisimman tarkasti. Järjestelmän käyttämät ohjelmistot päivitetään viimeisimpiin versioihinsa käyttöjärjestelmän päivityksen jälkeen eli on muistettava tuhota vanhat asennusmediat. Samalla on kiinnitettävä edelleen huomiota asennuksien eheyteen ja turvallisuuteen.

7.2.2 Palvelimen sovellusasennuksien päivitys

Arviointijärjestelmää valmistettaessa todettiin sovelluksista jo julkaistun päivityksiä. Palvelimelle asennettuun Apache http-palvelimeen saatavilla oli 2.2.21 päivitys. PHP-kielitukeen oli julkaistu 5.3.8 sekä 5.4beta päivitykset, joista yleisesti ottaen beta-versioita ei kannata asentaa mahdollisten virheiden varalta. phpMyAdmin-hallintasovellukseen tarjolla olivat 3.4.7.1 ja 3.3.10.5 päivitykset. Voidaan huomata että lähes jokaiseen sovellukseen julkaistiin päivityksiä työn aikana. Tästä syystä palvelimen asennus tulisi tehdä vasta järjestelmän lähes ollessa valmis. Alkutestauksen voi käytännössä suorittaa kehitysympäristössä. Kun sovellukset päivitetään, voidaan ne asentaa kokonaisuudessaan uudestaan eli poistamalla ja asentamalla uudestaan. Sovelluksien config- eli asetustiedostot sekä mahdolliset omat tietokannat on otettava talteen. Muun muassa php:n asennus käsittävää valtavan määrän eri asetuksia, joiden uudelleenasettaminen saattaa olla työlästä.

7.2.3 Tietokanta

Arviointijärjestelmän tietokannan päivitys voidaan tehdä järjestelmän ollessa käytössä tai muun päivityksen ohessa. Tietokannan tietojen muuttaminen, siirtäminen, luominen, tuhoaminen tulisi testata ennen suoritusta, jottei mahdollisia kadonneita tietoja jouduta palauttamaan varmuuskopioista. Palvelun suoritus eli ts. kyselyjen suoritus kuitenkin joudutaan keskeyttämään päivityksen ajaksi, minkä jälkeen aineisto pakotetaan uudelleenlähettämään varsinkin jos käyttöliittymään eli lähetettävän datan muotoon tms. tulee muutoksia. Käyttökatko päivitettäessä tulisi ilmoittaa jo noin tuntia ennen varsinasta katkoa. Sovelluksen sessio-ominaisuudet kuitenkin mahdollista-

vat pitkien käyttämättömien sessioiden käytön. Session vanhentumista voidaan muuttaa asiakkaan tarpeiden mukaan.

7.2.4 Käyttöliittymä

Jos käyttöliittymän lomakkeiden täyttöön tehdään muutoksia, tulee tietokantaan tehdä myös lomakkeen edellyttämät muutokset. Visuaalista liittymää voidaan kuitenkin muokata ajonaikaisesti niin kauan kun scriptit tai muut käsittelijät eivät muutoksesta välitä. Päivitys voidaan tehdä myös sivustopäivityksenä, jolloin koko järjestelmän kaikki php- eli käytännössä html-sivut päivitetään.

7.2.5 Käsittelijät

Datan käsittelijöiden päivittäminen on yksi välttämättömimmistä toimenpiteistä. Todennäköisyys sille että käsittelijöissä on tietoturvaukko tai muu virhe on suuri. Virheisiin kuuluu mm. saapuvan datan tietotyyppi, pituus ja muoto. Päivitys on tehtävä siis joka tapauksessa, toimisi järjestelmä muuten kokonaisuudessaan hyvin, tai ei. Suurimmat tietoturvaominaisuudet koodataan käsittelijöihin. Käsittelijöihin saapuva data voi olla minkälaista tahansa. Tämän data on vastaanotettava, käsiteltävä, lähetettävä eteenpäin, turvallisesti. Data voi saapua järjestelmään myös eri nopeudella, jonka vuoksi nopea uudelleenlähetys saattaa aiheuttaa ruuhkaa palvelussa. Pahimmillaan suuri määrä lähetyksiä ilmenee palvelinestohyökkäyksenä, jossa käytetään koko palvelun resurssit niin että käyttö tulee mahdottomaksi.

8 LOPPUSANAT

Vaikka tietojärjestelmää ei vielä saatu valmiiksi, tulee Keski-Satakunnan terveydenhuollon kuntayhtymä saamaan selainpohjaisen arviointijärjestelmän palkkausjärjestelmänsä tueksi. Tuki ei kuitenkaan tarkoita suoraa integrointia järjestelmään vaan arviointijärjestelmä pyritään ainakin aluksi tarkoituksenmukaisesti pitämään omana osittain suljettuna järjestelmänään, johon kuitenkin mahdollistetaan ulkoisten tieto-

lähteiden liittäminen laajempaa jatkokokonaisuutta ajatellen. Muun muassa käyttäjienhallinta voidaan jatkossa ajatella siirrettäväksi Microsoft Windowsin toimialueen käyttäjätietokannasta ja hakemistopalvelusta. Näin ollen käyttäjät voivat käyttää työasematunnuksia joko automatisoidusti arviointijärjestelmän tunnistaessa käyttäjä tai samoilla tunnuksilla kirjautumalla.

Kaikki paperiversioiden tieto tallennetaan sähköisesti tietojärjestelmään. Paperidokumentteja tulee syntymään vain sopimusten muodossa. Kuntayhtymälle kuitenkin opinnäytetyön kirjoitushetkellä suunnitteilla olevat Väestörekisterikeskuksen varmennepalvelut saattaisivat tulevaisuudessa mahdollistaa kaikkien käyttäjätunnistuksien sisällyttämisen fyysisille korteille jolloin järjestelmän käyttäminen voisi tulla entistäkin vaivattomammaksi. Itse toimialuekirjautumiseen korttitunnistusta ei kuitenkaan tulla ottamaan käyttöön. Arviointijärjestelmän ensisijaiset käyttäjät eli esimiehet sekä muu johto tulee pakottaa uuteen järjestelmään vaikka vanha tapa saattaisi aikataulun puitteissa olla nopeampi laatia. Tietojen hakeminen sekä tutkiminen helpottuvat ja tietokannan hakuihin voidaan jatkossa tehdä lisää ominaisuuksia joilla saadaan kaikki juuri se tieto mitä he tarvitsevat niin kuin haluavat. Kun tiedon määrä kannassa lisääntyy, lisääntyvät myös sen käyttöaika ja muut intressit käyttöä kohtaan. Jos Järjestelmä saavuttaa pisteen, jossa voidaan todeta käyttömäärä riittävän suureksi ja järjestelmän käytöstä saadaan palautetta, voidaan jatkokehitykseen sijoittaa aikaa sekä luoda mahdollisia lisäosia. Jatkossa voisi myös ajatella vanhan aineiston osittaista siirtämistä esimerkiksi työharjoittelijoiden voimin tietokantaan jolloin voitaisiin tehdä pidemmän ajan tarkastelua.

Johdon raportoinnin helppolukuisuuteen tulisi aina kiinnittää laajasti huomiota. Opinnäytetyön kirjoitushetkellä kuntayhtymään tuleva Microsoftin Sharepoint-ympäristön osatarkoitus on tarjota johdon raportoinnin ominaisuuksia ja tietokantojen datojen näyttöä. Tämä mahdollistaisi myös arviointijärjestelmän datan näyttämisen kyseisellä alustalla. Myös itse sovellus voitaisiin jatkossa ajatella käytettäväksi Sharepoint-ympäristössä. Tietojen tarkastelu tulisi tapahtua yhdellä silmäyksellä.

Vaikka tietojärjestelmän määrittelyn ja suunnittelun tärkeyttä korostetaan, ei näihin seikkoihin ymmärretty panostaa tarpeeksi aikaa. Varsinkin tietokannan määrittelyn ja

suunnittelun laiminlyönti, sekä relaatiomallin epätarkka sisällyttäminen, ja tätä myöden niiden korjaaminen pitkittävät merkittävästi tietojärjestelmän aikataulua.

LÄHTEET

Apache HTTP server project. 2011. About. Viitattu 10.11.2011
http://httpd.apache.org/ABOUT_APACHE.html

OK-opintokeskus. 2011. Kokouskäytännöt. Viitattu 17.8.2011.
http://ok-opintokeskus.fi/kokous/index.php?k=8125

PHP.net. 2011. PHP-manual nl2br. Viitattu 1.10.2011.
http://php.net/manual/en/function.nl2br.php

Wikipedia. 2011/a. User interface design. Viitattu 6.7.2011.
http://en.wikipedia.org/wiki/User_interface_design

Wikipedia. 2011/b. Ohjelmistotuotanto. Viitattu 21.10.2011.
http://fi.wikipedia.org/wiki/Ohjelmistotuotanto

Wikipedia. 2011/c. Cascading_Style_Sheets. Viitattu 16.7.2011.
http://en.wikipedia.org/wiki/Cascading_Style_Sheets

Wikipedia. 2011/d. PHP. Viitattu 1.11.2011.
http://en.wikipedia.org/wiki/PHP

Wikipedia. 2011/e. Viitattu 1.11.2011.
http://en.wikipedia.org/wiki/Database

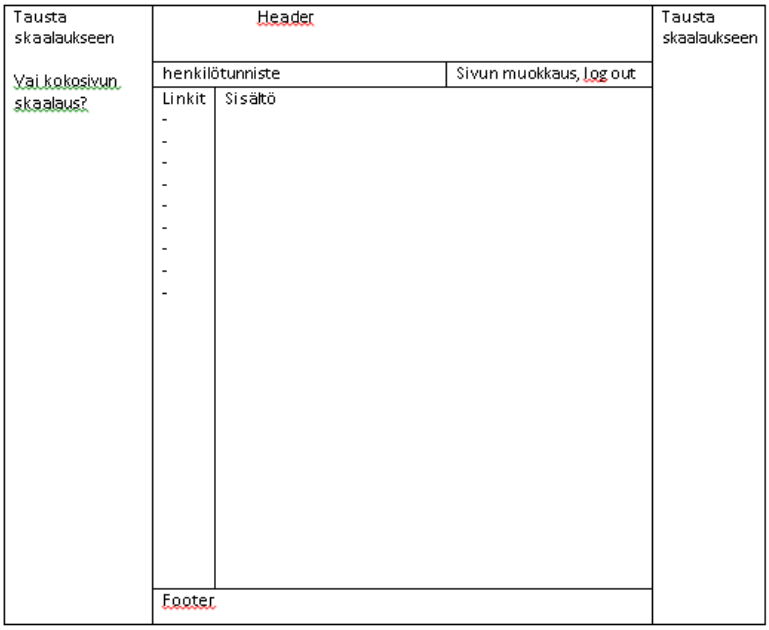
Wikipedia. 2011/f. Viitattu 5.11.2011.
http://fi.wikipedia.org/wiki/Ohjelmiston_testaaminen

Wikipedia. 2011/g. Software_testing. Viitattu 1.11.2011.
http://en.wikipedia.org/wiki/Software_testing

w3scools.com. 2011. HTML-Introduction. Viitattu 3.8.2011.
http://www.w3schools.com/html/html_intro.asp

Käyttöliittymän hahmotteleminen.

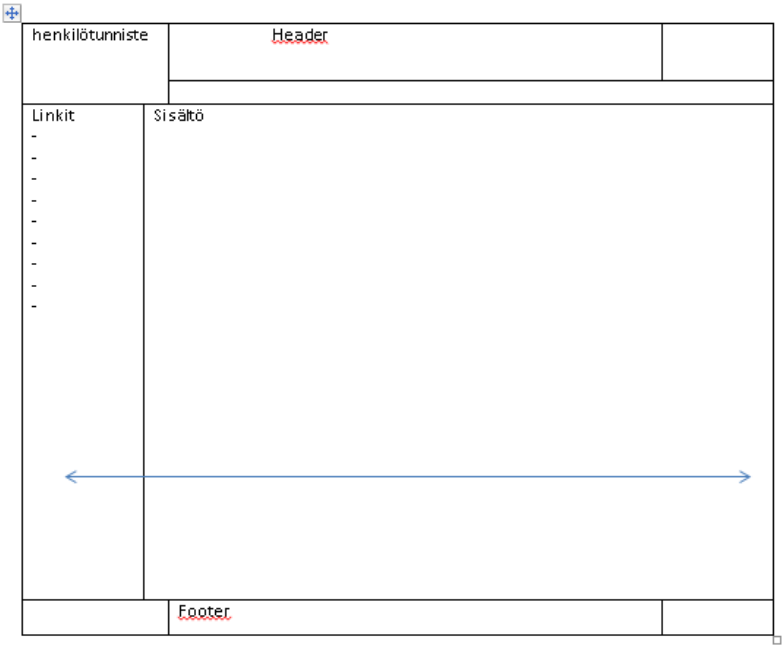
Malli1



Sivun muokkaus, log out vasemmalle linkeiksi? – esitelty esimerkkikuvissa.

Henkilötunniste vasempaan yläkulmaan? - esitelty esimerkkikuvissa.

Malli2



Rekisteröinti

- Kaikki peruskäyttäjiksi

Esimiehille oikeudet adminin kautta

Kirjautuminen

- Käyttäjätunnus
- Salasana
- (esimiestunnistus)

Linkit (puuratkaisu vai vanhan piilottaminen+takaisin?)

Peruskäyttäjä

- Perustiedot
- Tehtävän kuvaus
- (Tavoitteet)
- Henkilökohtaisen työsuorituksen arvionti
- Kehityskeskustelu
- (Kehityssuunnnitelma)
- Palaute

Esimies

- + Alaislista
- + Pisteytystaulukot sekä niiden muokkaus

Syvemmin

Perustiedot:

- Nimi
- Koulutus
- Tehtävänimike
- Tehtäväalue / tulosyksikkö
- Sopimusryhmä → työnkuvauslomakkeen ja sen pisteytyksien valinta
- Muut tiedot

Tehtävänkuvaus: -> työn vaativuuden arvionti

- Työn muodolliset kelpoisuusvaatimukset
    - Koulutus
    - Kielitaito
    - Kokemus
- Tarkka työnkuvauslomake
    - Sopimuksien mukaan

(Tavoitteet:)

Henkilökohtaisen työsuorituksen arvionti:

- Linkitys vai peilaus tehtävänkuvaukseen vai ollenkaan?
- Arviointilomake
- Ketä on arvioinnin tehnyt?

Keshityskeskustelu:

- Vapaa tekstimuotoinen keskustelu?
- Palaverikirjaus?
- Lomake?
- Jotakin muuta?

(Kehityssuunnnitelma:)

Palaute:

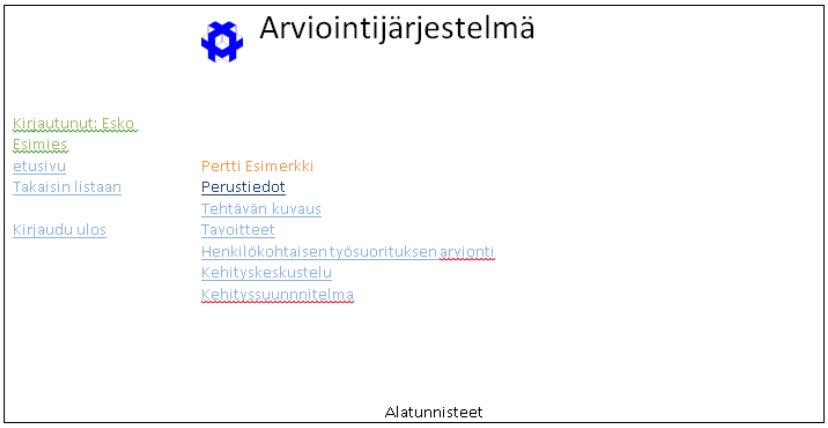
- Esimiehelle
- Adminposti

Alaislista:

- Alaisten henkilökohtaisien työsuorituksien arvionti sekä muut tiedot

Pisteytystaulukot sekä niiden muokkaus:

- Pistetaulukko



- Jotakin muuta hallittavaa?

Muuta

Kuplaohjeet?

Onko esimiehen esimiestä?

LIITE 2

Kehityskeskustelun yhteenveto.

Keski-Satakunnan
Terveydenhuollon
Kuntayhtymä

Kehityskeskustelun yhteenveto

Nimi:_____

Aika ja paikka:_____

Kehityskeskustelussa olemme käsitelleet seuraavat asiat:

1. Työntekijän henkilökohtaiset tavoitteet

   _____

   _____

   _____

2. Koulutussuunnitelma

   _____

   _____

   _____

3. Yhteistyössä kehitettävät alueet

   _____

   _____

   _____

4. Sovitut toimenpiteet.

   _____

   _____

   _____

Keskustelijoiden allekirjoitukset

_____ _____

Postiosoite:
Koulukatu 2
29200 HARJAVALTA

Puhelin:
02-6773 111

Sähköposti:
etunimi.sukunimi@ksthky.fi

Faksi:
02-6773 760

Internet:
www.ksthky.fi

LIITE 3

Henkilökohtaisen työsuorituksen lomake.

KESKI-SATAKUNNAN TERVEYDENHUOLLON KY

HENKILÖKOHTAISEN TYÖSUORITUKSEN ARVIOINTI (KVTES)

Tehtäväalue/tulosyksikkö:_____

ARVIOINTIPERUSTEET	Pisteytys 1	2	3
AMMATINHALLINTA • työskentely erilaisissa olosuhteissa ja vaihtuvissa toimipisteissä • ammattitaito (tiedot, taidot, menetelmät) • erityisosaaminen • monitaitoisuus • vastuuntunto ja luotettavuus • työkokemuksen pituus	5	10	15
TYÖTULOKSET • tavoitteiden saavuttaminen • aikaansaavuus • toiminnan ja työtulosten laatu • taloudellisuus • resurssien tehokas käyttö	5	10	15
YHTEISTYÖKYKY • ihmissuhdetaidot • vuorovaikutustaidot • yhteistyötaidot • kommunikointitaidot • neuvottelutaidot • joustavuus • erilaisuuden hyväksyminen • asiakaspalvelutaidot	3	6	10
KEHITYSHAKUISUUS • kehityskyky ja kehittymishalu • uusien tietojen ja taitojen hankkiminen ja omaksuminen • luovuus ja visiointi- sekä innovointikyky • tahtotila, asenne	3	6	10
Pisteet yhteensä			

Työsuorituksen arvioivat _____kuun ____päivänä 200__ _____ _____

Kehityskeskustelu käyty _____kuun ____päivänä 200__ _____ _____

TYÖSUORITUKSEN ARVIOINTITASOT

Arviointiasteikko

Sarake 1: edustaa normaalia työsuoritusta; työsuoritukset perustasoa

Sarake 2: edustaa normaalia parempaa työsuoritusta; henkilö selviytyy työstä ja odotuksista keskitasoa paremmin

Sarake 3: ylittää tehtävien hoitamiselle asetetut tavoitteet; henkilö ansaitsee työsuorituksestaan selkeän tunnustuksen