Lappeenrannan teknillinen yliopisto
Tuotantotalouden ja tietotekniikan tiedekunta
Tietotekniikan osasto

Mielipidekyselyjärjestelmän suunnittelu ja toteutus

Tarkastaja: Professori, KTT Jouni Lampinen

24.1.2007

Teemu Karvonen
Ojapolku 11 A
45150 Kouvola
040-5927262

TIIVISTELMÄ

Lappeenrannan teknillinen yliopisto
Tietotekniikan osasto

Teemu Karvonen

Mielipidekyselyjärjestelmän suunnittelu ja toteutus

Kandidaatintyö

2007

39 sivua ja 4 kuvaa

Tarkastaja: Professori, KTT Jouni Lampinen

Hakusanat: kyselyjärjestelmä, Internet-pohjaisen järjestelmän suunnittelu ja toteutus, toteutusmenetelmät

Keywords: survey system, Internet-based system design and implementation, implementation methods and practices

Tämän työn tavoitteena on suunnitella yksinkertainen Internet-pohjainen mielipidekyselyjärjestelmä sekä esitellä yksityiskohtaisesti järjestelmän toteutus ja siihen liittyvät menetelmät. Menetelmistä esitellään ainoastaan ennalta valitut menetelmät järjestelmän toteutukseen, tietojen esittämiseen, esitystavan muotoiluun sekä tietojen varastointiin. Järjestelmä toteutetaan HTML- ja PHP-kielillä sekä käyttämällä CSS-menetelmän tyylimuotoiluja ja XML-kielen muotoiluun perustuvia tiedostoja tietovarastoina. Järjestelmän suunnitteluun liittyen työssä pyritään kuvaamaan järjestelmään toteutettavat kaksi erillistä käyttöliittymää, pääkäyttäjän käyttöliittymä ja normaalin käyttäjän käyttöliittymä, sekä näihin toteutettavat toiminnot. Pääkäyttäjän tärkeimmät toiminnot ovat mielipidekyselyiden luominen, käyttäjien lisääminen kyselyihin sekä kyselyiden tulosten seuranta. Normaalin käyttäjän toiminnot taas rajoittuvat kirjautumiseen ja kyselyyn vastaamiseen.

Järjestelmän toteutuksen kuvauksessa kuvataan tarkasti edellä mainittujen kahden käyttöliittymän toiminnot sekä näiden toimintojen toteutustavat. Lisäksi toteutuksen kuvauksen yhteydessä määritellään tarkasti järjestelmän tietovarastoina toimivien tiedostojen sisällön muoto. Työn lopputuloksena syntyi valituilla toteutustavoilla toteutettu toimiva mielipidekyselyjärjestelmä sekä tämä järjestelmän suunnitteluun ja toteutuksen selvittämiseen keskittynyt dokumentti. Toteutetusta järjestelmästä ei tullut täydellinen vaan jatkokehityksessä voidaan harkita esimerkiksi tietokannan käyttämistä järjestelmän tietovarastoina sekä joidenkin lisäominaisuuksien toteuttamista. Tavoitteeseen päästiin kuitenkin, sillä toteutettu järjestelmä on toimiva ja käyttötarkoitukseensa sopiva.

ABSTRACT

Lappeenranta University of Technology
Department of Information Technology

Teemu Karvonen

Designing and implementing a survey system

Thesis for the Degree of Bachelor of Science in Technology

2007

39 pages and 4 figures

Examiner: Professor Jouni Lampinen

Keywords: survey system, Internet-based system design and implementation, implementation methods and practices

The goal of this work is to design a simple Internet-based survey system to collect information on chosen subjects and to give a detailed description of the implementation. Also some predefined methods for implementing the system, storing and presenting the data, and formatting the representation of the data are described. These predefined methods are HTML- and PHP-languages for implementation, CSS specification for representing the data and XML-language for formatting the stored data. System design phase aims to present the two separate system interfaces – administrator interface and normal user interface – and the functions that will be implemented to these interfaces. The most important functions of the administrator interface are creating queries, adding users to queries, and viewing the results of the queries. The functions included in the normal user interface are restricted to logging in to the system and answering to a query.

Presenting the implementation of the designed system is focused on detailed descriptions of the two mentioned interfaces. In addition the format of the data that is stored in the data storage files is described accurately. The final results of this project were a working system for registering users' opinions implemented using the above-mentioned methods and this document describing the design and implementation the system. The implemented application was not perfect and further development should be considered in for example providing a database system for storing data and in developing some new features. The outcome was still at least satisfying since a fully working system was developed.

SISÄLLYSLUETTELO

1 JOHDANTO

Tämän dokumentin tarkoituksena on esitellä ja kuvata menetelmiä, joita käytetään toteuttamaan mielipidekyselyjärjestelmä, sekä suunnitella ja toteuttaa kyseinen järjestelmä. Aiheen laajuuden vuoksi sitä on rajattu siten, että tässä dokumentissa kuvataan ainoastaan niitä menetelmiä, joita käytetään varsinaisen järjestelmän toteuttamiseen. Lisäksi dokumentissa kuvataan periaatteellisella tasolla suunniteltavan järjestelmän rakenne ja tarkasti lopullisen toteutetun järjestelmän rakenne ja toteutustavat. Tämän dokumentin lisäksi järjestelmän toteutukseen liittyen laaditaan järjestelmän käyttöohje.

Järjestelmän toteutuksessa käytetään internet-pohjaisten järjestelmien luomiseen tarkoitettuja ohjelmointikieliä ja standardeja. Käyttöliittymän ja toiminnallisuuden toteutukseen liittyviä teknologioita ovat PHP (Hypertext Preprocessor), HTML (HyperText Markup Language) ja CSS (Cascading Style Sheets), joista jokaisella on oma osa-alueensa toteutuksessa. Tiedon varastointiin ja hallintaan liittyviä teknologioita taas ovat XML (Extensible Markup Language) ja SQL (Standard Query Language)-kieleen perustuva MySQL-tietokanta. Näistä menetelmistä, niiden käyttötarkoituksesta liittyen aiheena olevaan projektiin sekä niiden aiheuttamista rajoituksista on kerrottu lisää myöhemmin tässä dokumentissa. Lisäksi lopulta havainnollistetaan jokaisen teknologian roolia lopullisessa järjestelmän toteutuksessa sekä pohditaan järjestelmän lopulliseen toteutukseen liittyvien tekniikoiden soveltuvuutta eri osa-alueiden toteutukseen ja toteutetun järjestelmän mahdollisia kehittämismahdollisuuksia.

2 KÄYTTÖLITTYMÄN JA TOIMINNALLISUUDEN TOTEUTUSTEKNIIKAT

Käyttöliittymän ja sovelluksen toiminnallisuuden toteuttamiseen käytetään kolmea menetelmää, jotka ovat PHP, HTML ja CSS. Näistä PHP liittyy lähinnä toiminnallisuuden ja käyttöliittymän dynaamisesti luotavien osien toteutukseen. HTML puolestaan tarjoaa mahdollisuuden käyttöliittymän rakenteen ja sisällön luomiseen ja CSS mahdollisuuden sisällön esitystavan muokkaamiseen. Yhdessä menetelmät siis tarjoavat hyvät mahdollisuudet helppokäyttöisen ja monipuolisen käyttöliittymän ja sen taustalla toimivan järjestelmän toteuttamiseen.

2.1 HTML

HTML on Internetissä käytetty selaimen välityksellä toimiva tekstin ja grafiikan rakenteen ilmaisemiseen kehitetty merkkauskieli. Se perustuu SGML (Standard Generalized Markup Language)-kieleen, joka on HTML-kieltä laajempi järjestelmä asiakirjojen hallitsemiseen. Merkkauskielenä HTML perustuu niin sanottuihin tageihin, jotka toimivat kielen tärkeimpinä elementteinä. Tageja käyttämällä HTML-kieli määrittelee joukon tyylejä ja merkkimuotoiluja, joita voidaan käyttää dokumenttien laatimiseen. Peruselementteihin voidaan liittää attribuutteja täydentämään elementtien määrittelyä. Attribuuttien avulla voidaan tarkentaa elementtien sisältöä tai niiden rakennetta. [10, 25]

2.1.1 Dokumentin rakenne

HTML-kielessä tagit tulee tyhjiä tageja lukuun ottamatta aloittaa ja lopettaa erikseen. Tyhjällä tagilla tarkoitetaan elementtiä, jolla ei ole sisältöä. Periaatteessa aloitustagilla ilmaistaan että jokin dokumentin osio alkaa ja lopetustagilla että se loppuu. Kaikki tagit on ympäröity kulmasuluilla ja sisältävät ennalta kieliopissa määritellyn avainsanan. Tämän lisäksi lopetusstagit erotetaan aloitustageista `/`-merkin avulla. Mikäli tagilla ei ole sisältöä, se voidaan myös lopettaa kirjoittamalla `/`-merkki ennen lopettavan kulmasulun kirjoittamista. Tyhjissä tageissa, kuten esimerkiksi rivinvaihdossa `<br>`, ei kuitenkaan käytetä /-merkkiä. Esimerkkinä HTML-dokumentin varsinaisen sisällön aina aloittava aloitustagi kirjoitetaan `<body>` ja dokumentin sisällön päättävä lopetustagi kirjoitetaan `</body>`. Näiden tagien yhdistelmä voidaan periaatteessa kirjoittaa myös muodossa

<body/>, mikäli dokumentilla ei ole ollenkaan sisältöä. Jokaisen HTML-dokumentin tulisi kuitenkin sisältää vähintään HTML-dokumentin aloittava html-elementti, otsikkoalueen määrittävä head-elementti, tämän sisällä sijaitseva otsikon määrittävä title-elementti sekä dokumentin sisällön määrittävä body-elementti. [9, 25]

Attribuutteja käytetään kuvaamaan ja tarkentamaan tagien sisältämää tietoa ja sen muotoilua. Attribuutit sijaitsevat tagien sisällä ja ne koostuvat attribuutin nimestä ja attribuutille annetusta arvosta. Arvo annetaan tietylle attribuutille =-merkin avulla ja arvo kirjoitetaan aina heittomerkkien sisään. Esimerkiksi taulukko, jonka leveydeksi halutaan 200 pikseliä, saadaan aikaan tageilla <table width=”200”></table>. [25]

2.1.2 Ominaisuudet

Varsinainen HTML-dokumentti on puhdas tekstidokumentti, joka sisältää edellä mainittuja tageja, niiden attribuutteja, sekä tagien avulla ilmaistua tietoa. Kieltä ei ole sellaisenaan tarkoitettu kuvaamaan sivujen muotoilua, eli sitä miltä sivut näyttävät selaimessa, vaan ainoastaan sivun rakennetta. Eri selaimet tulkitsevatkin samanlaisia HTML-dokumentteja eri tavoilla riippuen selainten tavasta käsitellä tageja ja niiden attribuutteja. Edellä mainitusta syystä HTML-kieli sopii hyvin tiedon ja sen rakenteen, muttei esitystavan ilmaisemiseen. [9, 10, 25]

Yksi HMTL-kielen tärkeimmistä perusominaisuuksista on kyky linkittää dokumentteja toisiinsa. Tämä mahdollistaa laajojen HTML-sivustojen laatimisen ja sivujen liittämisen toisiinsa. Lisäksi linkkien avulla samalla sivulla voidaan siirtyä paikasta toiseen ja sivustoille voidaan liittää esimerkiksi verkossa olevia tiedostoja ja kuvia. [9]

Yhdessä selainten ja muiden tekniikoiden kanssa HTML mahdollistaa erilaisten dynaamisten Internet-sivustojen laatimisen. Dynaamisuudella tarkoitetaan sitä, että sivuston dokumenttien sisältö tai ulkoasu voi muuttua käyttäjän toimenpiteiden mukaisesti tai jopa käyttäjäkohtaisesti. Dynaamisuuden mahdollistavia tekniikoita ovat esimerkiksi sivustojen ulkonäköä ohjaavat CSS-tyylisivut sekä PHP:n ja javascriptin kaltaiset komentokielet. [5, 9, 28]

2.2 CSS

CSS on HTML- ja XML-dokumentteja varten kehitetty menetelmä, jonka avulla voidaan laatia näkyvää ulkoasua ohjaavia tyylimäärittelyjä kyseisille dokumenteille. CSS-dokumenttien sisältämien tyylimäärittelyjen avulla voidaan kontrolloida dokumenttien Internet-selainten välityksellä näkyviä esitystyylejä, dokumenttien tulostustyylejä tai vaikka eri elementtien ääntämistyylejä. CSS mahdollistaa ulkonäkö- ja muotoiluasetusten erottamisen varsinaisesta dokumentista, joten tyylimäärittelyjen avulla voidaan luoda dynaamisia Internet-sivuja, joiden ulkonäköä voidaan muokata muokkaamatta varsinaisen sivun sisältöä ja elementtejä. [6, 9, 28]

CSS-tyylidokumentteja on pääsääntöisesti kahdenlaisia, sivuntekijän tyylejä ja käyttäjän tyylejä. Sivuntekijän tyylidokumentilla tarkoitetaan CSS-dokumenttia, jonka tekijä on laatinut tiettyä sivustoa varten. Sivun käyttäjän tyylimäärittely taas tarkoittaa yleistä tyylimäärittelyä, jonka käyttäjä on laajentanut koskemaan kaikkia käyttämiään sivuja. CSS-tyylidokumenttien avulla voidaan kontrolloida esimerkiksi HTML- ja XML-dokumenteissa esiintyviä fontteja, niiden kokoa, väriä ja muotoiluja. Lisäksi tyyleillä voidaan säädellä muun muassa tekstirivin korkeutta, eri elementeille asetettavia marginaaleja ja täytteitä, dokumenttien taustavärejä ja kuvia sekä linkkien muotoiluja. Alkuperäiseen edellä luetellut perusmäärittelyt sisältävään CSS1-spesifikaatioon on tullut laajennuksia CSS2-spesifikaation muodossa. CSS2 lisää tuen muun muassa absoluuttisesti sijoitettaville elementeille, automaattiselle numeroinnille ja sivun vaihdoille. Lisäksi kehitteillä on CSS3, joka tulee laajentamaan tyylimäärittelyjä entisestään. [2, 6, 9, 28]

2.2.1 Dokumentin rakenne

CSS-dokumentti koostuu säännöistä, jotka puolestaan koostuvat selektoreista, ominaisuuksista ja ominaisuuksille asetetuista arvoista. Säännöllä voidaan asettaa joidenkin elementtien, eli HTML- tai XML-tagien, halutuille ominaisuuksille arvoja. Samassa CSS-dokumentissa voi olla rajaton määrä sääntöjä. CSS-kielen sääntö koostuu selektorista ja yhdestä tai useammasta deklaraatiosta, joka on kirjoitettu selektorin perään aaltosulkeiden sisään. Deklaraatio taas koostuu ominaisuuden nimestä ja ominaisuudelle annetusta arvosta, jotka on erotettu toisistaan kaksoispisteellä. Deklaraatio loppuu puolipisteeseen. Esimerkiksi yhtenä sääntönä voisi olla `h1 {font-size: 20px; text-`

`aling: center;}`, jolloin ensimmäisessä deklaraatiossa ykköstason otsikon fontin kooksi asetettaisiin 20 pikseliä ja toisessa deklaraatiossa ykköstason otsikon teksti keskitettäisiin. Saman selektorin, tässä tapauksessa `h1`, sisällä olevat deklaraatiot koskevat siis kaikki selektorin viittaamaa elementtiä. Deklaraatioissa ominaisuuksille annettujen arvojen tyyppi ja usein myös arvoalue on ennalta määritelty. Esimerkiksi ominaisuuden `text-align` arvo voi olla ainoastaan `left`, `right`, `center` tai `justify`. [9, 28]

Selektoria voidaan käyttää usealla eri tavalla sekä CSS-dokumentissa että HTML-dokumentissa. Selektori voi olla luokka-, tunniste-, konteksti-, pseudoluokka-, määrite-, tai peräkkäisyysselektori. Tärkeimmät näistä ovat neljä ensimmäisenä mainittua. Luokkaselektori ilmaistaan CSS-säännössä pisteen (.) avulla ja HMTL-dokumentissa siihen viitataan attribuutilla `class`. Esimerkiksi mikäli selektori on `h1.suuri` niin siihen viitataan HTML-dokumentissa tagilla `<h1 class=″suuri″>`. Tunnisteselektori merkitään CSS-sääntöön risuaitamerkin (#) avulla ja siihen viitataan HTML-dokumentissa attribuutilla `id`. Kontekstiselektori merkitään CSS-säännössä välilyönnin avulla ja se edellyttää että jälkimmäinen tyyli on edellä mainitun tyylin sisällä HTML-dokumentissa. Esimerkiksi `div.suuri a` viittaa HTML-dokumentin linkkielementtiin (a), joka sijaitsee elementin `<div class=″suuri″>` sisällä. Lisäksi pseudoluokkaselektoria käytetään viittaamaan eri tilassa oleviin linkkielementteihin. [9, 28]

2.2.2 Ominaisuudet

CSS-menetelmän tärkein ominaisuus on, että se mahdollistaa HTML- ja XML-dokumenttien ulkoasun muuntamisen ilman itse dokumenttien muokkaamista. Tämä tekee dokumenteista dynaamisia ja mahdollisesti jopa käyttäjäkohtaisesti täysin erilaisia. Lisäksi kaikkien muotoilujen keskittäminen samaan tiedostoon mahdollistaa helpon ylläpitämisen ja haluttaessa dokumenttien ulkonäön muuntamisen. Tämä mahdollistaa myös kaikkien muotoiluun liittyvien attribuuttien karsimisen varsinaisista dokumenteista, jolloin niistä tulee lyhyempiä, selkeämpiä ja nopeampia ladata. [9, 28]

CSS-kielen avulla dokumenttien muotoilua ei voida niin sanotusti pakottaa tyylimuutoksiin. CSS-säännöt ovat enemmän ehdotuksia, joita dokumentissa noudatetaan, mikäli se on mahdollista. CSS-sääntöjen tehokkuutta rajoittavat muun muassa joistain

selaimista puuttuva tai rajoitettu CSS-tuki, virheellinen bugeja sisältävä CSS-tuki, mahdollisuus ottaa CSS-tuki pois käytöstä, sekä HTML-tagien alkuperäiset muotoilut. Näiden vuoksi laadittavat dokumentit ja CSS-tiedostot tulisi yhdessä testata halutuissa ympäristöissä ennen varsinaista käyttöönottoa tai vaihtoehtoisesti CSS-tuen puutteeseen tulisi varautua muilla tavoin. Hyvänä ominaisuutena CSS-tuen puuttuminen ei kuitenkaan aiheuta dokumentin lataamisessa kuin ulkoasumuotoilujen hylkäämisen. Dokumenttien varsinainen sisältö voidaan siis esittää ilman muotoilujakin. [9, 28]

2.3 PHP

PHP on vapaaseen lähdekoodiin perustuva komentokieli, jota käytetään lähinnä Internet-sovellusten dynaamisten ominaisuuksien toteuttamiseen. Sen tärkein ominaisuus on juuri mahdollisuus luoda dynaamista sisältöä ja liittää tämä sisältö HTML-sivuille. PHP on kehittynyt nopeasti CGI (Common Gateway Interface)-ohjelmien komentokokoelmasta täysin itsenäiseksi ja suosituksi ohjelmointikieleksi. [3, 8]

2.3.1 Ominaisuudet

Vapaan lähdekoodin tulkattavana ohjelmointi kielenä PHP on yleistynyt nopeasti. Tulkattavan kielen selkeitä etuja ovat mahdollisuus toimia ilman kääntäjää pelkän tulkin avulla sekä dynaamisen sisällön luomisen helpottuminen. Lisäksi PHP-kielelle on saatavissa ilmainen WWW-palvelimen sisäisenä moduulina toimiva tulkki, joka mahdollistaa PHP-komentojen ajamisen suoraan www-palvelimella välityksellä. Tulkin suora yhteys WWW-palvelimeen mahdollistaa myös PHP-koodia sisältävien sivustojen nopeamman ajamisen, sillä tulkkia ei tarvitse käynnistää erikseen. Kaikki PHP-kielellä ohjelmoidut toiminnot ajetaan aina palvelimella ennen sivujen lähettämistä käyttäjälle selaimen välityksellä. [3, 8]

Edellä kuvattu PHP-kielen käyttötapa, palvelimella tapahtuva operaatioiden suorittaminen, on juuri sen suurin vahvuus. Kyseiseen käyttötapaan tarvitaan WWW-palvelin, sille asennettava PHP-tulkki sekä Internet-selain, jonka välityksellä ohjelma voidaan suorittaa ja tulokset näyttää käyttäjälle. PHP-kielisiä ohjelmia voidaan kuitenkin käyttää myös komentoriviltä. Tällöin ohjelmien ajamiseen ei tarvita WWW-palvelinta tai selainta, vaan ainoastaan ajamisen mahdollistava tulkki. Tarvittaessa PHP-kieltä voidaan käyttää

edellisten tapojen lisäksi myös luomaan graafisten käyttöliittymien avulla toimivia sovelluksia. Tämä vaatii kuitenkin PHP-kielen vahvaa tuntemusta ja kehittyneempien ominaisuuksien hallintaa. [7]

PHP-kielen avulla on periaatteessa mahdollista toteuttaa Internet-pohjaisiin järjestelmiin HTTP-autentikaatio. Tämä ominaisuus on kuitenkin käytössä vain, mikäli PHP on asennettu Apache-palvelimen moduuliksi. Toiminnon ollessa käytössä voidaan PHP-skriptissä lähettää selaimelle autentikaatiopyyntö, jolloin selain kysyy käyttäjältä käyttäjätunnusta ja salasanaa. Kyseiset tiedot tallennetaan muuttujiin ja mikäli tiedot täsmäävät, tavoiteltu sivusto ladataan uudelleen. Hyvinä ominaisuuksina PHP-kieli tukee myös evästeitä ja sessioita. Evästeillä tarkoitetaan ominaisuutta, joka mahdollistaa tietojen tallentamisen väliaikaisesti käyttäjän selaimeen. Evästeiden käyttäminen mahdollistaa esimerkiksi eri käyttäjien liikkeiden seuraamisen. Sessiot taas mahdollistavat käyttäjästä saadun tiedon säilyttämisen esimerkiksi useiden käyttökertojen ajan. Tällöin eri käyttäjiä voidaan tunnistaa ja mahdollisesti joitain käyttäjäkohtaisia asetuksia voidaan säilyttää useiden käyttökertojen ajan. Sessioiden sisältämän informaation tietoturvaa ei kuitenkaan voida taata, mikä saattaa tehdä niiden käyttämisestä ongelmallista joissain tilanteissa. [14, 23, 24]
PHP-kieli omaa samankaltaiset rakenteet kuin muut yleiset ohjelmointikielet, kuten C, Perl ja Java. Näistä rakenteista esimerkkeinä voivat olla `while` ja `for` –silmukat, `if`-ehtolauseet sekä muuttujien ja funktioiden käyttäminen. PHP-kieli on rakenteeltaan yksinkertainen, mutta se omaa useita tehokkaita ominaisuuksia. PHP-kieli toimii joustavasti yhteistyössä HTML-kielen kanssa. Mihin tahansa HTML-sivun kohtaan voidaan lisätä PHP-koodia kirjoittamalla se suoraan HTML-sivulle skriptin aloitus- ja lopetustagien väliin. Tagipareja on useita vaihtoehtoisia, mutta HTML-kielessä parhaiten aloitustageina toimivat `<?php` tai `<script language=”php”>` sekä vastaavina lopetustageina `?>` tai `</script>`. Luotaessa HTML-sivulle sisältöä PHP-kielen avulla generoitavan HTML-sivun lähdekoodista ei paljastu, että kysymyksessä on ohjelmointikielellä generoitu sisältö, vaan lähdekoodissa, samoin kuin sivun normaalissa ulkoasussa, näkyy ainoastaan PHP-koodin tuottama sisältö. [3, 8]

Useimmista ohjelmointikielistä poiketen PHP-kielessä on tuki sekä proseduraaliselle että oliopohjaiselle ohjelmoinnille. Kyseistä olio-ohjelmoinnin mahdollistavaa ominaisuutta ei

ollut vielä PHP 3:ssa, mutta sitä on tuettu neljännestä versiosta lähtien. PHP 5 lisäksi täydentää edeltäjänsä olio-ohjelmointiominaisuuksia sekä korjaa niissä esiintyneitä puutteita. [7]

PHP-kieli on yhteensopiva yleisimpien käyttöjärjestelmien, kuten Windowsin, Linuxin, Unixin ja MAC OS X:n kanssa. Lisäksi se tukee useita erilaisia WWW-palvelimia ja PHP-kielellä kirjoitettuja ohjelmia voidaan tarvittaessa ajaa myös CGI-ohjelmina. PHP tukee laajasti myös erilaisia tietokantoja, joista merkittävimpinä voidaan mainita MySQL, Direct MS-SQL sekä IBM:n kehittämä DB2. Laajan tietokantatuen lisäksi PHP mahdollistaa jatkuvien tietokantayhteyksien muodostamisen. Jatkuvilla yhteyksillä tarkoitetaan yhteyksiä, jotka säilyttävät yhteyden tietokantaan, vaikka tietyn ohjelmiston osan suoritus päättyy. Jatkuvat yhteydet lisäävät tietokantojen käytön tehokkuutta, sillä samaan tietokantaan ei luoda useita samanaikaisia yhteyksiä eikä kyseistä yhteyttä tarvitse luoda jokaisella käyttökerralla erikseen. [7, 11]

2.3.2 Tietoturva

Monet PHP-kielen perusominaisuudet, kuten kyky avata tiedostoja ja suorittaa toimintoja palvelimella, luovat kielelle useita tietoturvariskejä. PHP on kuitenkin suunniteltu nimenomaan esimerkiksi C- ja Perl-kieliä turvallisemmaksi tavaksi luoda CGI-ohjelmia. Paras tapa kontrolloida PHP-koodin tietoturvaa onkin käyttää oikeanlaisia PHP-tulkin konfiguraatio-optioita sekä noudattaa tietoturvallisia ohjelmointikäytäntöjä. Koska täysin turvallinen järjestelmä on käytännössä mahdoton, paras tapa on löytää käytäntö, jossa yhdistyvät mahdollisimman pienet riskit sekä hyvä käytettävyys. [19, 22]

PHP voidaan asentaa palvelimelle joko Apache-palvelimen moduulina tai CGI-rajapintaa käyttävänä ohjelmistona. CGI-rajapintaa käyttävänä ohjelmistona PHP takaa tietoturvan paremmin kuin muuta ohjelmointikielet. PHP tarjoaa muun muassa mahdollisuuden estää tiedostojen ja hakemistojen suorittaminen ja avaaminen käyttäjän antamien argumenttien perusteella sekä käyttöoikeustarkistusten kiertämisen kutsumalla suoraan `cgi-bin` –hakemistossa olevaa ohjelmistoa. Jos taas PHP asennetaan Apache-palvelimen moduuliksi, se perii palvelimen käyttäjäoikeudet. Tämä aiheuttaa tietoturvariskin, mikäli käyttäjän, jolla ei ole oikeuksia palvelimelle, pitää päästä käyttämään esimerkiksi palvelimella

sijaitsevaa tietokantaa. Tällöin tietokannalle tulee asettaa omat käyttäjätunnukset ja salasanat sen käyttämisen turvaamiseksi. Helpoin tapa tähän on käyttää esimerkiksi LDAP (Lightweight Directory Access Protocol)- tai htaccess (Hypertext Access)- menetelmiin perustuvaa autentikointia. [20, 21]

Suurimmat PHP-kielen tietoturvariskit liittyvät palvelimen tiedostojärjestelmän muokkaamiseen sekä palvelimella sijaitsevien tietokantojen muokkaamiseen. Tiedostojärjestelmän muokkaamisessa, eli esimerkiksi uusien tiedostojen tai kansioiden luomisessa tai vanhojen muokkaamisessa, PHP luottaa palvelimella määriteltyihin kansio- ja tiedosto-oikeuksiin, joilla voidaan kontrolloida muun muassa kirjoitus-, luku- ja suorittamisoikeuksia. Nämä oikeudet myös varmistavat tiedostojen ja kansioiden sisältämien tietojen turvallisuuden. Suurimpina riskeinä tiedostojärjestelmien käsittelemisessä ovat kirjoitus- ja poisto-oikeudet omaavat tiedostot ja kansiot. Tällaisia tiedostoja käsitellessä tulee aina varmistaa käyttäjältä saatujen muuttujien arvojen, esimerkiksi tiedostonimien ja polkujen, oikeellisuus sekä se, että operaatiot kohdistuvat ainoastaan sallittuihin ja sallituissa hakemistoissa sijaitseviin tiedostoihin ja kansioihin. Tämä tapahtuu käytännössä tarkistamalla muuttujien arvoja sekä tarvittaessa muokkaamalla muuttujien sisältöä. Hyvänä perussääntönä tiedostojärjestelmien muokkaamisessa toimii se, että kaikki, mikä ei ole erikseen sallittua, tulisi kieltää. [17, 18]

Tietokantojen tietoturva tulee ottaa huomioon jo suunniteltaessa tietokannan varsinaista rakennetta. Tietokannan luomisen yhteydessä sen omistusoikeudet annetaan yleensä ainoastaan sen luoneelle käyttäjälle. Tästä tulisikin pitää kiinni, eikä omistusoikeuksia tulisi antaa Internet-käyttäjille. Käyttäjien oikeuksien pitäisi riittää ainoastaan pakollisten operaatioiden suorittamiseen. Oikeusryhmät ovat hyvä tapa jakaa käyttäjät ryhmiin oikeuksiensa mukaan. Lisäksi kaikkea tietokannan käyttämiseen liittyvää logiikkaa ei tulisi sisällyttää PHP-tiedostoihin vaan suurin osa niistä tulisi lisätä itse tietokannalle kyselyinä ja operaatioiden laukaisemina toimintoina eli herättiminä (trigger). Olennaisena osana tietoturvan takaamisessa toimivat myös käyttäjän ja palvelimen tai tietokannan välisen yhteyden tai kommunikaation salaamisen mahdollistavat menetelmät, kuten esimerkiksi SSH (Secure Shell) ja SSL (Secure Sockets Layer), sekä tiedon sisällön salaamiseen tarkoitetut algoritmit, kuten esimerkiksi MD5 (Message Digest 5). [12, 17]

Tietokantojen käyttämiseen PHP-koodin avulla kohdistuu edellisten lisäksi myös uhka kyselyiden muokkaamisesta tunkeutujan haluamaan muotoon. Esimerkiksi SQL-kyselyitä ei ole mitenkään salattu ja niitä voidaan muokata vahingollisiksi ainoastaan tietämällä esimerkiksi jonkin tietokannan taulun ja tämän taulun sisältämän kentän nimet. Tällaisia suoraan muokattuja kyselyitä voidaan käyttää esimerkiksi paljastamaan tietokannasta tietoa, jonka ei pitäisi olla julkista, muokkaamaan tietoa tai jopa suorittamaan ohjelmia tietokantapalvelimella. Paras suojautumistapa tällaisilta hyökkäyksiltä on jälleen käyttäjältä saatujen syötteiden huolellinen tarkastaminen ja mahdollisten uhkien ennaltaehkäiseminen. [13]

2.3.3 Yhteydet MySQL-tietokantaan

Ohjelmointikielenä PHP tarjoaa useita rakenteita ja komentoja, joilla voidaan suorittaa erilaisia tehtäviä. Sama pätee myös PHP-kielen ja MySQL-tietokannan välisiin yhteyksiin, sillä PHP-kieli sisältää useita toimintoja, joiden avulla voidaan välittää SQL-komentoja MySQL-tietokannalle. PHP:n välityksellä onnistuu sekä tietojen tallentaminen HTML-lomakkeesta suoraan tietokantaan että tietojen lukeminen tietokannasta suoraan PHP-koodin avulla luodulle HTML-sivulle. Ohjelmointikielen tarjoamien funktioiden avulla tietokannan käyttäminen on yksinkertaista, tosin se edellyttää sekä PHP- että SQL-kielen hallitsemista. [3]

Jotta PHP-kielen MySQL-tietokantojen käsittelyyn tarkoitettuja funktioita voisi käyttää, tulee PHP-kielen yhteydet MySQL-tietokantoihin sallia. Oletuksena asennuksen yhteydessä kyseinen ominaisuus sallitaan, paitsi versiosta 5 lähtien. Ominaisuuden voi ottaa käyttöön myös jälkikäteen kääntämällä PHP:n uudelleen tarvittavilla optioilla, jotka löytyvät lähteestä 15. Samasta lähteestä löytyvät myös MySQL-tietokannan hallintaan liittyvät funktiot ja niiden käyttöohjeet. Kyselyt MySQL-tietokantaan PHP-kielen avulla suoritetaan normaaleina SQL-kyselyinä, jotka sijoitetaan kyseiseen tarkoitukseen tarkoitetun PHP-funktion yhteyteen. Esimerkkinä tällaisesta funktiosta on `mysql_query`, jolle annetaan parametrina etukäteen funktioiden avulla valittuun tietokantaan kohdistuva kysely. Kyselyitä ja MySQL-kielen ominaisuuksia käsitellään tarkemmin kohdassa 3.1. [15]

2.3.4 Yhteydet XML-dokumentteihin

Yhtenä hyvänä PHP-kielen ominaisuutena on, että se sisältää XML-parserin eli jäsentimen. Jäsentimen avulla voidaan lukea XML-tiedostoa ja samalla jäsentää tiedostoa sen sisältämien tagien mukaisesti eri osioihin. Kun tiedosto jaetaan automaattisesti osiin, käsittelyssä tulee ottaa huomioon ainoastaan tiedoston eri osien käsittely. Jäsennin vaatii oman funktionsa sekä aloitustagien että lopetustagien käsittelyyn. Käytännössä tiedostoa lukiessa sen eri osista poimitaan halutut tiedot ja niitä käytetään määritysten perusteella halutuilla tavoilla. [3]

Jotta PHP-kielen XML-jäsentimen funktioita voisi käyttää, tulee PHP-kielen yhteydet XML-tiedostoihin sallia. Oletuksena asennuksen yhteydessä kyseinen ominaisuus on sallittu. Kuten MySQL-tietokantojen hallintaan, myös XML-jäsentimen hallintaan on omat funktionsa. Ensimmäisen jäsennin tulee luoda ja sille tulee asettaa sekä aloitustagin että lopetustagin käsittelyfunktiot. Tämän jälkeen jäsennin on käytännössä käyttövalmis. XML-tiedostojen rakennetta käsitellään tarkemmin kohdassa 3.2 ja lisätietoja jäsentimen käytöstä löytyy lähteestä 16. [16]

3 TIEDON VARASTOINNIN JA HALLINNAN TEKNIIKAT

Tiedon varastointiin ja hallintaan käytetään sovelluksessa kahta tekniikkaa, jotka ovat MySQL ja XML. MySQL toimii tietokantana tärkeässä osassa tiedon varastoinnissa, mutta myös XML-tiedostoissa voidaan tarvittaessa säilyttää tietoa. Yhdessä MySQL ja XML tarjoavat sovellukselle hyvät mahdollisuudet tiedon varastointiin ja hallintaan.

3.1 MySQL

MySQL on WWW-palveluja varten kehitetty avoimen lähdekoodin relaatiotietokanta. Sen lisäksi että MySQL-tietokantaohjelmisto on ilmainen, on se myös monipuolinen, joustava ja suorituskykyinen verrattuna muihin relaatiotietokantoihin. Sen toiminta perustuu SQL-kieleen, joka on ANSI:n (American National Standards Institute) ja ISO:n (International Organization for Standardization) standardi. SQL kieli on tietokannoille tarkoitettu kyselykieli, joka mahdollistaa tietokantakyselyiden lisäksi muun muassa tietokantojen rakenteen määrittelyn ja muuntamisen, tietokannan tietojen päivittämisen ja muuttamisen sekä tietokannan valtuuksien ja turvallisuuden hoitamisen. SQL-kielisiä lauseita voidaan helposti upottaa useisiin ohjelmointikieliin, mikä tekee SQL:n käytöstä joustavaa ja tehokasta. Vaikka SQL onkin standardi, MySQL-tietokantojen käyttämä SQL-kieli eroaa hieman standardoiduista komennoista. Perustoiminnot ovat samat kuin standardissa, mutta MySQL laajentaa toimintovalikoimaa tarjoamalla lisäksi muutamia standardin ulkopuolisia komentoja. [3, 4, 26]

3.1.1 Tietokannan relaatiomalli

MySQL-tietokanta perustuu muiden yleisimpien tietokantojen tavoin relaatiomalliin. Relaatiomalli perustuu pohjimmiltaan joukko-oppiin, matematiikkaan ja predikaattilogiikkaan ja se jakautuu kolmeen osaan; rakenteeseen, käsittelyyn ja eheyssääntöihin. Näiden kolmen osan avulla määritellään käytännössä tietokannan rakenteen ja toiminnan säännöt. [4]

Relaatiotietokannan rakenne koostuu yksinkertaisesti tauluista, joita on yleensä tietokannassa useita. Taulut edustavat käytännössä tietokantaan tallennettavien tietojen asiakokonaisuuksia. Taulun sarakkeille määritellään tietotyyppi ja sarakkeeseen

tallennetaan aina samankaltainen tieto. Taulun rivit vastaavat tietueita eli tietokokonaisuuksia. Esimerkiksi jos taulu sisältää henkilötietoja sisältäisi yksi rivi eli tietue yhden henkilön tiedot ja sarakkeiden tietotyypit voisivat olla esimerkiksi etunimi, sukunimi ja ikä. Jokin taulun sarakkeista tulee aina asettaa perusavaimeksi, jolloin sarakkeen sisältämän arvon tulee olla yksilöllinen kussakin tietueessa. Perusavain voi koostua myös useammasta sarakkeesta. Mikäli useita tauluja halutaan yhdistää toisiinsa, voidaan taululle asettaa perusavaimen lisäksi viiteavaimia. Viiteavainsarakkeen sisältämä arvo viittaa jonkun toisen taulun, lapsitaulun, perusavaimeen. [4]

Relaatiotietokannan käsittely perustuu joukko-opissa määriteltyihin joukko-operaatioihin. Näitä ovat valinta, projektio, yhdiste, leikkaus, erotus ja liitos. Kaikki tietokantoihin suuntautuvat joukko-operaatiot voidaan toteuttaa käyttämällä SQL-kielen SELECT-käskyn erilaisilla muodoilla. [4]

Eheyssäännöt huolehtivat siitä, että tietokannan sisältämät tiedot ovat oikein, ne eivät ole keskenään ristiriidassa ja ne vastaavat reaalimaailmaa. Eheyssääntöjä tarvitaan tilanteissa, joissa esimerkiksi samaa tietoa yritetään tallentaa tietokantaan useampaan kertaan tai tietokannan eri tauluissa on ristiriitaista tietoa. Relaatiotietokannalle on määritelty kaksi tärkeää eheysrajoitetta, jotka ovat avaineheys ja viite-eheys. Avaineheyden mukaan perusavaimelle täytyy antaa aina jokin tyhjästä poikkeava arvo. Toisin sanoen perusavaimen arvo ei voi olla tyhjää tarkoittava NULL. Viite-eheyden mukaan taulusta ei voi poistaa arvoja tai tietueita, joihin on viitattu muissa tauluissa. [4]

3.1.2 Ominaisuudet

MySQL-tietokannan toiminta perustuu niin sanottuun asiakas-palvelin -malliin, jonka ansiosta käyttäjät eivät koskaan pääse käsiksi suoraan tietokantaan. Kaikki prosessointi hoidetaan erityisesti tietokannan muokkaamiseen tarkoitettujen palvelinohjelmistojen kautta. Näitä palvelinohjelmistoja voidaan kehittää useilla eri ohjelmointikielillä, esimerkiksi Java-, PHP- tai Perl-kielillä. [3]

Yksi MySQL-palvelin voi sisältää useita tietokantoja, jotka voivat kukin sisältää useita tauluja. Jokaiselle tietokannalle voidaan määrittää omat käyttäjäasetukset

käyttäjätunnuksineen, salasanoineen ja käyttäjäkohtaisine oikeuksineen. Käyttäjiä voi periaatteessa olla rajaton määrä ja jokaiselle käyttäjälle voidaan tarvittaessa määrittää erilaiset käyttöoikeudet eri tietokantoihin ja niiden sisältämiin tauluihin. MySQL-tietokannat tarjoavat siis erittäin hyvän käyttäjäsuojan. [3, 26]

MySQL-tietokantojen tehokkuutta parantavia ominaisuuksia ovat esimerkiksi tuki useille yhtäaikaisille käyttäjille, monipuoliset, jopa standardia laajemmat, komennot sekä monisäikeinen toteutus. Jälkimmäinen tarjoaa etuja ja nopeampaa prosessointia mikäli MySQL-palvelin toimii usean prosessorin järjestelmissä. Muina etuina voidaan mainita nopea muistinvarausjärjestelmä, koodin optimoinnin ansiosta erittäin nopeasti suoritettavat tietokantaoperaatiot sekä tuki suurillekin tietokannoille. [3, 26]

3.2 XML

XML on yksinkertainen ja joustava merkkauskieli, joka oli alun perin tarkoitettu sähköiseen julkaisutoimintaan. Nykyään se toimii myös tärkeässä roolissa verkon välityksellä tapahtuvassa kommunikoinnissa. XML on johdettu SGML-kielestä (Standard Generalized Markup Language, ISO 8879), jolla voidaan järjestää ja merkata elementtejä dokumenteissa. SGML on XML:n tavoin metakieli, jonka pääasiallisena toimenkuvana on kuvata tietoa tiedosta. Sillä voidaan siis ilmaista tallennetun tiedon sisältö ja siihen liittyvät attribuutit, mutta ei sitä, miten tämä sisältö esitetään. Dokumentin sisältö ja ulkoasu on siis tiukasti erotettu toisistaan. Ulkoasun määrittelyyn voidaan tarpeen vaatiessa käyttää erillisiä tyylikieliä, kuten aiemmin käsiteltyä CSS-kieltä tai yksinomaan XML-kielen ulkoasun määrittelyyn käytettävää XSL (eXtensible Style Language) -kieltä. [1, 4, 27]

XML-dokumentti on normaali tekstitiedosto, joten sitä voidaan käsitellä esimerkiksi tekstieditorissa. HTML-kielen valmiista tageista poiketen XML-kielen tagit voidaan nimetä itse. Tämän ansiosta XML-dokumenttia onkin suhteellisen helposti luettava, mikäli elementit ja niihin liittyvät attribuutit on nimetty järkevästi. Helppolukuisuudesta huolimatta XML-dokumentin lukeminen on helppoa myös tietokoneelle. XML-dokumentti voidaan jakaa rakenteeltaan kahteen osaan, loogiseen ja fyysiseen. XML dokumentin sisältö koostuu erilaisista elementeistä, jotka määritellään tiedoston loogisessa rakenteessa. Pelkän elementtien määrittelyn lisäksi looginen rakenne määrittelee myös elementtien

järjestyksen ja eri elementtien välillä vallitsevan hierarkian. W3C:n XML-määritys määrittelee tarkasti XML-dokumentin kieliopin ja sille asetetut ehdot, jotka sen tulisi täyttää. Fyysisesti dokumentin rakenne koostuu tietokokonaisuuksista, entiteeteistä, jotka ympäröidään kyseisillä merkinnöillä. Jokaisella tietokokonaisuudella on jotain sisältöä sekä yksilöllinen nimi. [1, 4, 27]

XML dokumentti rakentuu kokonaisuudessaan elementeistä, attribuuteista ja entiteeteistä. Dokumentin aloittaa aina juuri-elementti, jonka sisälle kootaan kaikki muut elementit- HTML-kielen tavoin jokainen elementti koostuu aloitustagista, mahdollisista aloitustagin attribuuteista, sisällöstä sekä lopetustagista. Aloitustagi sisältää elementin nimen kulmasulkujen sisällä sekä mahdolliset attribuutit, esimerkiksi `<elementin_nimi attribuutti=″1″>`. Lopetustagit erotetaan aloitustageista `/`-merkin avulla, esimerkiksi `</elementin_nimi>`. Näiden kahden tagin sisälle merkitään elementtien sisältö. Mikäli elementillä ei ole sisältöä, se on tyhjä. Tällöin elementti voidaan merkitä joko alku- ja lopputunnisteella tai tyhjän elementin tunnisteella `<elementin_nimi/>`. [1, 27]

Attribuutteja käytetään kuvaamaan ja tarkentamaan tagien sisältämää tietoa ja sen muotoilua. Attribuutit sijaitsevat tagien sisällä ja ne koostuvat attribuutin nimestä ja attribuutille annetusta arvosta. Arvo annetaan tietylle attribuutille =-merkin avulla ja arvo kirjoitetaan aina heittomerkkien sisään. Koska XML-kielessä elementit voidaan määritellä itse, voidaan attribuutteja käyttää myös omina elementteinään. Tällöin attribuuttien käyttäminen onkin täysin vapaavalintaista. Suurissa dokumenteissa attribuuttien käyttö voi lyhentää dokumenttia, mutta pääsääntöisesti useiden attribuuttien käyttäminen hankaloittaa dokumenttien käsittelyä. [1, 27]

Entiteettien avulla muodostetaan XML dokumentin fyysinen rakenne. Entiteetit tulee erikseen käyttämällä määritellä DTD- (Document Type Definition) tai XML-skeema -kuvauksia. Entiteettien kuvaamisen lisäksi DTD- ja XML-skeema -dokumenteissa voidaan määritellä koko XML dokumentin rakenne. XML-skeema on monipuolisuutensa vuoksi alkanut syrjäyttää DTD-määrityksiä. [1, 4]

4 TOTEUTETTAVAN OHJELMISTON SUUNNITTELU

Työssä toteutettava mielipidekyselyjärjestelmä koostuu useasta erillisestä osasta, joista näkyvimmät ovat käyttöliittymät järjestelmän pääkäyttäjälle ja normaaleille käyttäjille. Pääkäyttäjällä tarkoitetaan käyttäjää, jolla on oikeus laatia ja ylläpitää mielipidekyselyitä. Normaalilla käyttäjällä taas viitataan käyttäjään, joka voi vastata pääkäyttäjän laatimiin kyselyihin. Seuraavissa kappaleissa on suunniteltu kyseisen järjestelmän toteutus laatimalla järjestelmälle yleiset vaatimukset sekä suunnittelemalla tietojen varastointiin käytettävät tekniikat ja pääkäyttäjän ja normaalin käyttäjän tarvitsemat toiminnot.

4.1 Yleiset vaatimukset

Työn tavoitteena on suunnitella ja toteuttaa palautejärjestelmä, joka mahdollistaa palautteen keräämisen halutulta kohderyhmältä internetin välityksellä. Järjestelmään toteutetaan kaksi erillistä käyttöliittymää; pääkäyttäjän käyttöliittymä sekä normaalin käyttäjän käyttöliittymä. Näiden lisäksi järjestelmän taustalla toimii jokin tietojen varastointiin tarvittava järjestelmä, järjestelmän käyttöliittymien ulkoasuun liittyvät tyylitiedot sisältävä CSS-tyylitiedosto sekä XML-muotoinen ohjaustiedosto. Tässä järjestelmässä tietojen varastointi toteutetaan MySQL-tietokannan sijasta tallentamalla järjestelmän tarvitsemat tiedot XML-muotoisiin tiedostoihin. Tätä toteutustapaa käytetään, sillä MySQL-tietokannan käyttäminen olisi aiheuttanut suuren työmäärän myös loppukäyttäjän järjestelmän ylläpitäjälle. Varsinainen järjestelmän toiminnallisuus toteutetaan PHP- ja HTML-kielillä ja järjestelmä laaditaan täysin selainpohjaiseksi. Lopullisen järjestelmän tulee toimia Unix- ja Linux-järjestelmissä, joihin on asennettu PHP-kielen tulkki.

Varsinaisen toteutettavan mielipidekyselyjärjestelmän tulee soveltua hyvin käyttötarkoitukseensa ja sillä tulee pystyä luomaan ja ylläpitämään Internet-selaimen välityksellä toimivia mielipidekyselyitä. Toiminnallisuudeltaan järjestelmän tulee olla helppokäyttöinen eikä se saa vaatia käyttäjiltä minkäänlaista erityisosaamista, kuten ohjelmointitaitoa. Lisäksi kaikkien järjestelmän käytössä vaadittavien toimintojen tulee olla helposti pääkäyttäjän suoritettavissa päävalikon välityksellä. Normaalin käyttäjän osalta mielipidekyselyihin vastaamisen tulee suoraviivaista ja helppoa, jotta järjestelmän

käyttämisen vaikeus ei karsi mahdollisia vastaajia. Lisäksi käyttöliittymän ulkoasun tulee olla normaalien käyttäjien osalta siisti ja selkeä.

Tietoturvan varmistamiseksi ja järjestelmän väärinkäytön estämiseksi sekä pääkäyttäjien että normaalien käyttäjien toiminnot tulee suojata salasanalla. Järjestelmän luonteen ja käyttötarkoituksen vuoksi tämän hetkiseen järjestelmään ei tarvitse sisällyttää kuin yksi pääkäyttäjän käyttäjätunnus ja salasana. Normaalien käyttäjien salasanoja taas tulee pystyä kontrolloimaan, esimerkiksi lisäämään ja poistamaan, pääkäyttäjän toimintojen avulla. Pääkäyttäjän tulee tarvittaessa pystyä muokkaamaan kaikkia järjestelmän käyttämiä tiedostoja turhien virheiden ehkäisemiseksi. Tiedostojen muokkaamisen tulee olla niin helppoa, ettei pääkäyttäjän tarvitse tehdä virheitä tietokantaan osaamattomuuden takia. Tämä tarkoittaa käytännössä sitä, että tiedostojen sisältöä tulee pystyä muokkaamaan pääkäyttäjän käyttöliittymään toteutettujen ohjattujen toimintojen avulla.

4.2 Tietojen varastointi

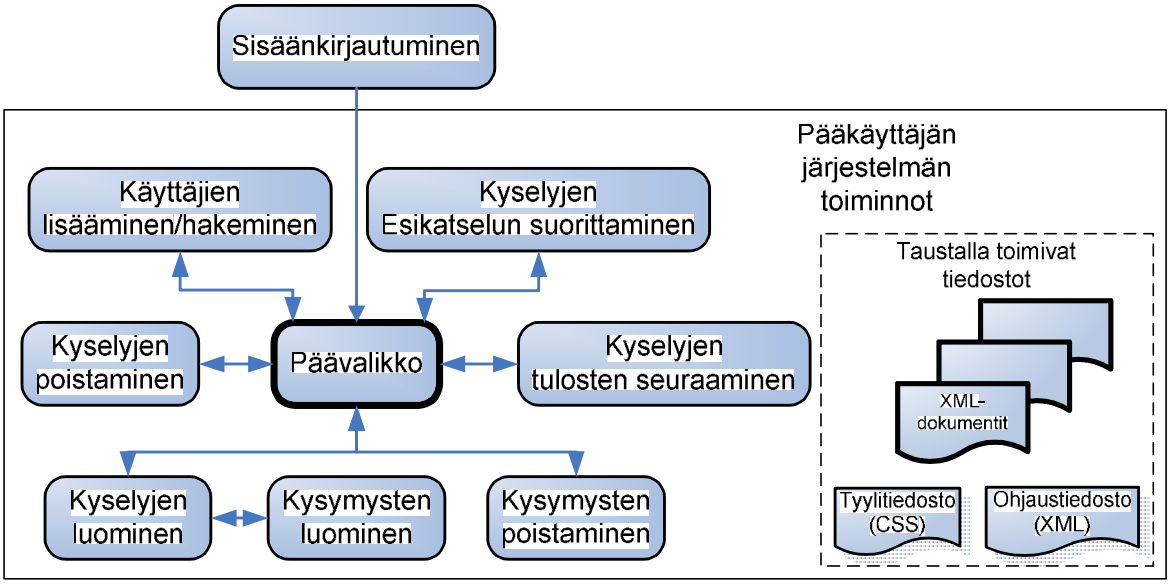
Järjestelmän käyttämät käyttäjien syöttämät ja luomat tiedot tallennetaan järjestelmässä XML-muotoisiin tiedostoihin. Järjestelmään luodaan omat tiedostonsa kyselyissä esiintyville kysymyksille, pääkäyttäjän luomille kyselyille, järjestelmän normaaleille käyttäjille, kysymyksissä esiintyville vastausvaihtoehdoille sekä kyselyiden vastauksille. Kysymystauluun tallennetaan kaikki kyselyissä esiintyneet kysymykset. Lisäksi järjestelmässä käytetään XML-muotoista ohjaustiedostoa, joka sisältää järjestelmän toiminnan kannalta oleellista tietoa.

Kysymystiedostoon tallennetaan kaikki kyselyissä esiintyneet ja esiintyvät kysymykset. Kyselyiden luomisen yhteydessä kyselyyn sijoitettavat kysymykset valitaan kysymystiedoston sisältämistä kysymyksistä. Kyselytiedostoon tallennetaan kaikki pääkäyttäjän järjestelmään luomat kyselyt kysymyksineen ja vastausvaihtoehtoineen. Vastausvaihtoehtotiedoston sisältämät tiedot toimivat kysymyksille asetettavien vastausvaihtoehtojen teksteinä. Kyselyt erotetaan toisistaan yksilöllisen kyselytunnisteen avulla. Järjestelmän käyttäjätiedostoon tallennetaan tiedot kaikista järjestelmän normaaleista käyttäjistä kyselyittäin. Käyttäjätiedostoa käytetään myös varastoimaan käyttäjien yksilöllisiä salasanoja, jotka toimivat käyttäjien tunnistusmenetelmänä

kirjautumisen yhteydessä. Normaalien käyttäjien kyselyihin antamat vastaukset tallennetaan erilliseen vastaustiedostoon. Käyttäjien tavoin vastaukset lajitellaan kyselyittäin siten, että myös eri henkilöiden vastaukset pidetään erillään.

4.3 Pääkäyttäjän toiminnot

Pääkäyttäjän toimintoihin kuuluvat kirjautuminen, mielipidekyselyjen luominen ja poistaminen, luotujen kyselyjen esikatselu, kyselyjen tulosten seuraaminen, käyttäjien lisääminen ja hakeminen sekä kyselyihin liitettävien kysymysten luominen ja poistaminen. Nämä toiminnot suojataan salasanalla, jotta ainoastaan pääkäyttäjä pääsee suorittamaan niitä. Kaikkia toimintoja pääsee suorittamaan pääkäyttäjän päävalikossa sijaitsevien linkkien kautta. Mahdollisista toimintoihin liittyvistä virhetilanteista, esimerkiksi tietojen luomisen tai poistamisen yhteydessä, ilmoitetaan pääkäyttäjälle. Mikäli toiminnoissa tapahtuu virheitä, ei järjestelmän sisältämiä tietoja tule muuttaa. Kuvassa 1 on esitetty pääkäyttäjän järjestelmän toiminnot ja suhteelliset sijainnit järjestelmän sivustolla.



Kuva 1: Pääkäyttäjän järjestelmän toiminnot.

Pääkäyttäjän järjestelmään pääsee erillisen pääkäyttäjän kirjautumissivun kautta. Kirjautumisen onnistuessa pääkäyttäjä ohjataan päävalikkoon, joka sisältää linkit kaikkiin pääkäyttäjän toimintoihin. Poikkeuksena näistä on kysymysten luominen, johon pääsee ainoastaan kyselyjen luomissivun kautta.

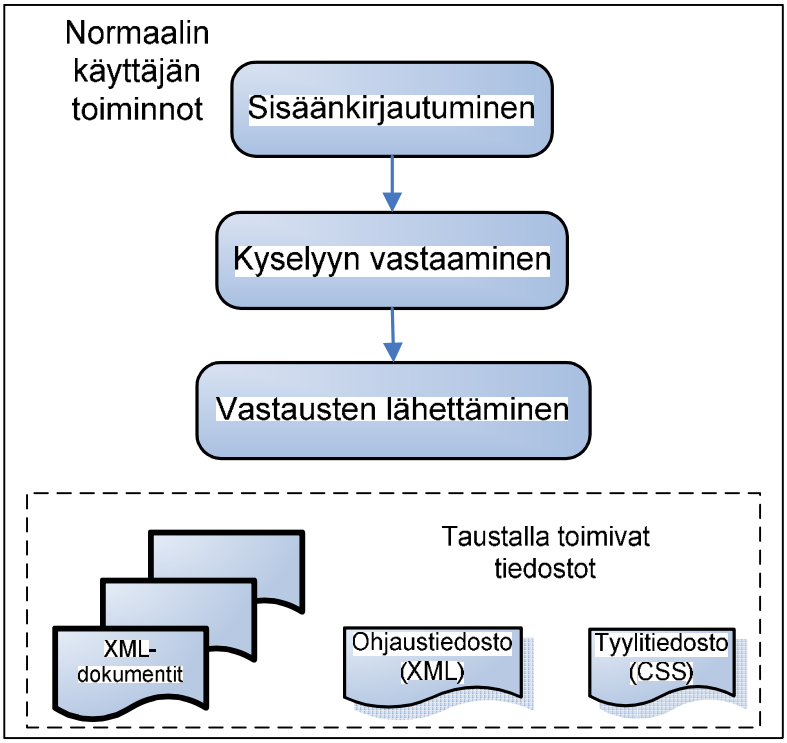
Kyselyn luominen on toiminto, joka opastaa pääkäyttäjää uuden mielipidekyselyn luomisessa. Käyttäjältä kysytään kyselyyn liittyvä tunniste, eli kyselyn otsikko, ja lisäksi kyselyyn liitettävät kysymykset, joita voi lisätä kyselyyn kirjoittamalla tai valitsemalla kysymystiedostossa valmiiksi olevia kysymyksiä. Käyttäjä saa valita myös kysymyksiin liittyvien vastausvaihtoehtojen muodon. Kyselyn luomisen yhteydessä kyselylle luodaan kyselytiedostoon uusi elementti, johon lisätään kaikki kyselyyn liittyvät tiedot. Kyselyn poistamisessa vastaavasti voidaan poistaa jokin luoduista kyselyistä. Tällöin kyselytiedostosta poistetaan kyseiseen kyselyyn liittyvä elementti.

Järjestelmään luotuja kyselyitä voidaan esikatsella suoraan pääkäyttäjän järjestelmän kautta esikatselutoiminnon avulla. Esikatselutoiminto näyttää kyselyt pääkäyttäjälle sellaisena kuin ne näytetään normaaleille käyttäjille. Esikatselutoiminto suoritetaan valitsemalla ensin haluttu järjestelmän sisältämä kysely ja tämän jälkeen suorittamalla toiminto.
Kyselyjen tuloksia ja tulosten kehittymistä voidaan seurata oman toimintonsa avulla. Kyselyjen tulokset tallennetaan vastaustiedostoon ja nämä tulokset yhdistetään tulosten seuraamistoiminnon avulla. Kyselyjen tulosten seurannassa siis pääkäyttäjä valitsee ensin tietyn kyselyn, jonka jälkeen kyseisen kyselyn tulokset haetaan tiedostosta ja näytetään pääkäyttäjälle. Kyselyn tulokset tulostetaan taulukkona, josta on helppo luoda kuvaaja esimerkiksi taulukkolaskentaohjelmalla.

Käyttäjien lisääminen ja olemassa olevien käyttäjien hakeminen tapahtuu erillisellä lomakkeella. Pääkäyttäjän tulee ensin valita kysely listalta ja hakea tämän tietoja tiedostosta. Tämän jälkeen pääkäyttäjä voi halutessaan syöttää valitulle kyselylle halutun vastaajien määrän. Käyttäjiä kyselyille lisätään siten, että tietyt salasanat linkitetään aina tiettyyn kyselyyn. Käyttäjätunnuksena kyselyssä toimii yhteinen pääkäyttäjän valitsema tunnus ja järjestelmä generoi kyselylle yhtä monta salasanaa kuin sille on asetettu vastaajia. Nämä salasanat tulostetaan pääkäyttäjälle, jonka jälkeen ne voidaan tulostaa ja jakaa vastaajille. Vastaajat ovat siis täysin anonyymejä ja käyttäjät linkitetään oikeisiin kyselyihin kyselylle ominaisen käyttäjätunnuksen ja yksilöllisten salasanojen avulla. Käyttäjien poistaminen tapahtuu poistamalla järjestelmästä kysely, jolloin poistetaan samalla myös kyselyyn liittyvä käyttäjätunnus ja salasanat sekä kyselyyn liittyvät vastaukset.

4.4 Normaalikäyttäjien toiminnot

Normaalien käyttäjien ainoat toiminnot ovat järjestelmään kirjautuminen ja käyttäjätunnuksen ja salasanan osoittamaan kyselyyn vastaaminen. Mikäli kirjautuminen epäonnistuu, tulee siitä ilmoittaa käyttäjälle. Kirjautumisen onnistuessa käyttäjä ohjataan suoraan kyselyyn. Kyselyyn vastaamisen ja vastausten lähettämisen tulee olla käyttäjän kannalta mahdollisimman yksinkertaista. Käyttäjät tulee päästää vastaamaan jokaiseen kyselyyn vain kerran ja järjestelmän täytyy pitää huoli kyseisen ehdon toteutumisesta. Kuvassa 2 on esitetty normaalin käyttäjän toiminnot ja niiden suorittamisjärjestys.



Kuva 2: Normaalikäyttäjän järjestelmän rakenne.

5 OHJELMISTON TOTEUTUS

Järjestelmän toteutettiin edellä laaditun suunnitelman mukaisesti pääkäyttäjän ja normaalien käyttäjien käyttöliittymiin ja toimintoihin keskittyen. Lopullinen järjestelmän toteutus suoritettiin PHP-, HTML- ja XML –kielillä sekä käyttöliittymien ulkoasujen osalta CSS-menetelmällä. Käytännössä XML toimii järjestelmän tietoja varastoivien tiedostojen muotoilun perustana, PHP toimii järjestelmän taustalla tiedosto-operaatioiden suorittajana, HTML- ja PHP-kielten avulla luodaan järjestelmän käyttöliittymien ulkoasu ja CSS-menetelmää käytetään ulkoasujen muotoilussa. Koska järjestelmälle ei ole asetettu varsinaisesti muita kuin toimintoihin liittyviä tarkkoja vaatimuksia, voitiin toteutuksessa käyttää vapaasti erilaisia valittuihin toteutuskieliin liittyviä tekniikoita.

5.1 Tietojen varastointi

Tietojen varastointiin järjestelmässä käytettiin suunnitelman mukaisesti XML-muotoista tietoa sisältäviä tiedostoja. Kyseisiä järjestelmän taustalla toimivia tiedostoja ovat käyttäjätiedosto, kyselytiedosto, vastaustiedosto, kysymystiedosto, vastausvaihtoehtotiedosto sekä ohjaustiedosto. Ohjaustiedostoa lukuun ottamatta kaikki järjestelmän tietovarastoina toimivat tiedostot ja niiden sisältöjen muodot on esitetty seuraavissa kappaleissa. Ohjaustiedoston rakennetta ei esitellä, sillä sen sisältö ei suoraan vaikuta normaalin käyttäjän tai pääkäyttäjän toimintaan vaan se sisältää ainoastaan toimintaa ohjaavia tietoja.

5.1.1 Käyttäjätiedosto

Käyttäjätiedostossa säilytetään kyselyihin liitettäviä käyttäjätunnuksia ja niihin liittyviä salasanoja. Jokaista kyselyä varten on määritetty ainoastaan yksi käyttäjätunnus ja jokaisesta käyttäjätunnusta varten on määritetty yksi tai useampi salasana. Salasanan ja käyttäjätunnuksen oikea yhdistelmä sallii normaalin käyttäjän pääsyn käyttäjätunnuksen ja tunnisteen viittaamaan kyselyyn. Salasanat kirjoitetaan käyttäjätiedostoon sha1-menetelmällä salattuina ja niille asetetaan luomisen yhteydessä kaytetty-attribuutin arvoksi false (epätosi). Kun kyseistä salasanaa on käytetty kyselyssä vastaamiseen, muutetaan kaytetty-attribuutin arvoksi true (tosi). Seuraavassa on esitetty käyttäjätiedoston sisällön rakennetta kuvaava XML-skeema –määrittely.

```
<?xml version="1.0"?>

<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
<xs:element name=”kayttajat” maxOccurs=”1”>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=”kayttaja” maxOccurs=”unbounded”>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="salasana">
              <xs:complexType>
                <xs:simpleContent>
                  <xs:extension base="xs:string">
                    <xs:attribute name="kaytetty">
                      <xs:simpleType>
                        <xs:restriction base="xs:string">
                          <xs:enumeration value="true"/>
                          <xs:enumeration value="false"/>
                        </xs:restriction>
                      </xs:simpleType>
                    </xs:attribute>
                  </xs:extension>
                </xs:simpleContent>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        <xs:attribute name=”kayttajatunnus” type=”xs:string” use=”required”>
        <xs:attribute name=”tunniste” type=”xs:string” use=”required”>
        </xs:complexType>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

5.1.2 Kyselytiedosto

Kyselytiedostoon tallennetaan kaikki pääkäyttäjän laatimat kyselyt erillisinä kysely-elementteinä. Jokaiselle kysely-elementille määritellään yksilöivät kayttajatunnus- ja tunniste-attribuutit, joiden avulla kyselyt erotetaan toisistaan. Kysely-elementit sisältävät lisäksi jokaiselle niiden sisältämälle kysymykselle oman kysymys-elementin, joiden sisältönä taas toimivat kysymysnumero-, vastausvaihtoehto- ja sisalto-elementit. Näistä kysymysnumero sisältää kysymyksen järjestysnumeron suhteessa kyseiseen kyselyyn, vastausvaihtoehto sisältää vastausvaihtoehtotiedoston mukaisen vastausvaihtoehdon tunnisteen ja sisalto sisältää varsinaisen kysymyksen. Kaikkia kysymysnumeroita ei välttämättä esiinny kyselyssä, mutta kysymykset järjestetään silti kyselyyn numeroiden mukaan nousevassa järjestyksessä.

Uuden kyselyn luomisen yhteydessä luodaan tiedostoon kyselylle luonnollisesti uusi kysely-elementti ja tämän sisältämät tiedot. Kyseisiä tietoja käytetään kyselyiden

tulostamisessa ja tulosten seuraamisessa. Kyselyiden poistamisen yhteydessä taas poistettavaa kyselyä vastaava kysely-elementti poistetaan kysely-tiedostosta. Seuraavassa on esitetty kyselytiedoston sisällön rakennetta kuvaava XML-skeema –määrittely.

```
<?xml version="1.0"?>

<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
<xs:element name=”kyselyt” maxOccurs=”1”>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=”kysely” maxOccurs=”unbounded”>
        <xs:complexType>
          <xs:sequence>
           <xs:element name=”kysymys” maxOccurs=”unbounded”>
             <xs:complexType>
               <xs:sequence>
                 <xs:element name=”kysymysnumero” type=”xs:integer” minOccurs=”1”
maxOccurs=”1”>
                 <xs:element name=”vastausvaihtoehto” type=”xs:string”
minOccurs=”1” maxOccurs=”1”>
                 <xs:element name=”sisalto” type=”xs:string” minOccurs=”1”
maxOccurs=”1”>
               </xs:sequence>
             </xs:complexType>
         </xs:sequence>
         <xs:attribute name=”kayttajatunnus” type=”xs:string” use=”required”>
         <xs:attribute name=”tunniste” type=”xs:string” use=”required”>
       </xs:complexType>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

5.1.3 Vastaustiedosto

Vastaustiedostoon tallennetaan normaalien käyttäjien lähettämät kyselyvastaukset. Jokaiselle kyselylle luodaan siihen tulleiden vastausten myötä oma kyselynvastaukset-elementtinsä, jonka sisälle tallennetaan kaikki samaan kyselyyn liittyvät vastaukset. Kyselynvastaukset-elementin attribuutteina toimivat kyselyt yksilöivät kayttajatunnus- ja tunniste-attribuutit. Jokaisen erillisen henkilön antamat vastaukset tallennetaan oikean kyselynvastaukset-elementin ja uuden henkilonvastaukset-elementin sisään. Varsinaiset vastaustekstit tallennetaan vielä vastaus-elementtien sisään. Jokaiselle kysymykselle luodaan oma vastaus-elementtinsä, johon tallennetaan varsinaisesti joko valitun vastausvaihtoehdon sisältämä teksti tai avoimen tekstikentän tapauksessa käyttäjän tekstikenttään syöttämä teksti.

Vastaustiedostoa käytetään normaalien käyttäjien vastausten tallentamisen lisäksi myös luonnollisesti kyselyiden tulosten seuraamiseen. Toiminnon avulla pääkäyttäjä näkee vastaajien yksilölliset vastaukset vastaustiedoston välityksellä. Seuraavassa on esitetty vastaustiedoston sisällön rakennetta kuvaava XML-skeema –määrittely.

```
<?xml version="1.0"?>

<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
<xs:element name=”vastaukset” maxOccurs=”1”>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=”kyselynvastaukset” maxOccurs=”unbounded”>
        <xs:complexType>
          <xs:sequence>
            <xs:element name=”henkilonvastaukset” maxOccurs=”unbounded”>
              <xs:complexType>
                <xs:sequence>
                  <xs:element name=”vastaus” maxOccurs=”unbounded”
type=”xs:string”>
                </xs:sequence>
              </xs:complexType>
          </xs:sequence>
        <xs:attribute name=”kayttajatunnus” type=”xs:string” use=”required”>
        <xs:attribute name=”tunniste” type=”xs:string” use=”required”>
        </xs:complexType>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

5.1.4 Kysymystiedosto

Kysymystiedostoon tallennetaan pääkäyttäjän kysymysten luomistoiminnolla luomat kysymykset. Kysymykset tallennetaan erikseen sekä kysymystiedostoon että niihin kyselyihin, joihin ne liittyvät. Kysymystiedoston sisältö ei siis vaikuta varsinaisten kyselyjen sisältöön vaan ainoastaan kyselyn luonnissa käytetyn lomakkeen näyttämiin valmiisiin kysymyksiin. Tämän ansioista kysymyksiä voi myös poistaa kysymystiedostosta, vaikka kysymys olisikin asetettu yhteen tai useampaan kyselyyn. Jokainen kysymystiedoston sisältämä kysymys luodaan erillisen kysymys-elementin sisälle. Seuraavassa on esitetty kysymystiedoston sisällön rakennetta kuvaava XML-skeema –määrittely.

```
<?xml version="1.0"?>

<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
<xs:element name=”kysymykset” maxOccurs=”1”>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=”kysymys” maxOccurs=”unbounded” type=”xs:string”>
    </xs:sequence>
  </xs:complexType>
```

```
</xs:element>
</xs:schema>
```

5.1.5 Vastausvaihtoehtotiedosto

Vastausvaihtoehtotiedosto sisältää kaikki erilaiset vastausvaihtoehdot, joita käyttäjä voi asettaa kyselyyn sijoitettaville kysymyksille. Jokainen vastausvaihtoehto on kuvattu oman vaihtoehto-elementtinsä sisällä ja sisältää pakollisina elementteinä tunniste-, maara- ja tyyppi-elementit sekä vaihtoehdon mukaan mahdollisesti maara-elementin sisällön määräämän määrän teksti-elementtejä. Tunniste-elementti sisältää vastausvaihtoehdon tunnisteen, jonka perusteella pääkäyttäjä valitsee kysymykselle kyseisen vastausvaihtoehdon. Maara-elementti taas sisältää edellä mainittujen teksti-elementtien määrän, joka kertoo samalla myös kyseisen vastausvaihtoehdon normaaleille käyttäjille esitettyjen vastausvaihtoehtojen määrän. Tyyppi taas sisältää vastausvaihtoehdon tyypin, joka voi olla avoin, joka tarkoittaa avointa tekstikenttää, tai radio, joka tarkoittaa radio-tyyppisen näppäimen käyttämistä. Teksti-elementit sisältävät vastausvaihtoehtojen viereen tulostettavat vaihtoehtojen tekstit. Seuraavassa on esitetty vastausvaihtoehtotiedoston sisällön rakennetta kuvaava XML-skeema –määrittely.

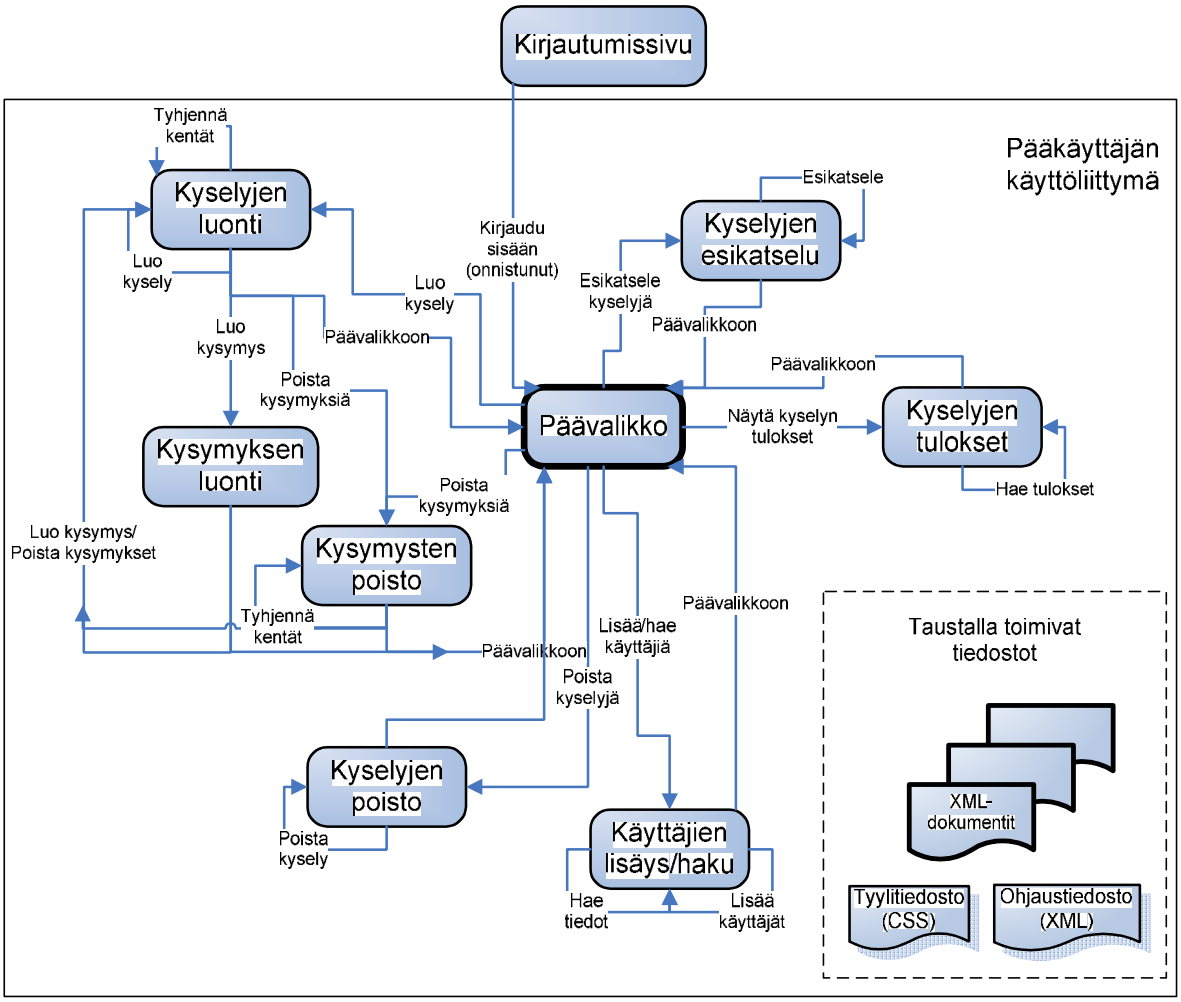
```
<?xml version="1.0"?>

<xs:schema xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
<xs:element name=”vastausvaihtoehdot” maxOccurs=”1”>
  <xs:complexType>
    <xs:sequence>
      <xs:element name=”vaihtoehto” maxOccurs=”unbounded”>
        <xs:complexType>
          <xs:sequence>
            <xs:element name=”tunniste” maxOccurs=”1” minOccurs=”1”
type=”xs:string”>
            <xs:element name=”maara” maxOccurs=”1” minOccurs=”1”
type=”xs:integer”>
            <xs:element name=”tyyppi” maxOccurs=”1” minOccurs=”1”>
             <xs:simpleType>
               <xs:restriction base="xs:string">
                 <xs:enumeration value="true"/>
                 <xs:enumeration value="false"/>
               </xs:restriction>
             </xs:simpleType>
           <xs:element name=”teksti” maxOccurs=”unbounded” type=”xs:string”>
         </xs:sequence>
        </xs:complexType>
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>
```

Vastausvaihtoehtotiedoston muokkaamista ei sisällytetty tämän hetkiseen järjestelmään, vaan tarpeelliset vastausvaihtoehdot luodaan tiedostoon valmiiksi ennen järjestelmän toimittamista. Käytännössä pääkäyttäjä ei siis pysty muokkaamaan ennalta asetettuja vastausvaihtoehtoja järjestelmän nykyisessä versiossa.

5.2 Pääkäyttäjän toiminnot

Pääkäyttäjän toimintoihin kuuluvat laaditun järjestelmäsuunnitelman mukaisesti kirjautuminen, mielipidekyselyjen luominen ja poistaminen, luotujen kyselyjen esikatselu, kyselyjen tulosten seuraaminen, käyttäjien lisääminen ja hakeminen sekä kyselyihin liitettävien kysymysten luominen ja poistaminen. Pääkäyttäjän toiminnot on suojattu salasanalla käyttämällä htaccess-menetelmää. Kuvassa 3 on esitetty pääkäyttäjän järjestelmän toimintojen sivukartta ja eri sivujen väliset suhteet.



Kuva 3: Pääkäyttäjän järjestelmän rakenne.

5.2.1 Kirjautuminen

Pääkäyttäjän kirjautumissivulle pääsee kirjoittamalla selaimen osoitteeksi päävalikon osoitteen. Tällöin järjestelmä kysyy käyttäjältä palvelimelle ennalta määriteltyä käyttäjätunnusta ja salasanaa. Mikäli tiedot täsmäävät, käyttäjä pääsee päävalikkoon. Muussa tapauksessa käyttäjälle tulostetaan palvelimen tuottama virheilmoitus. Pääkäyttäjän ei tarvitse erikseen kirjautua ulos järjestelmästä. Uloskirjautuminen tapahtuu ainoastaan sulkemalla järjestelmän käyttämiseen käytetty selain. Mikäli selainta ei suljeta, pääsee käyttäjä palaamaan takaisin järjestelmään kirjautumatta uudelleen.

5.2.2 Kyselyjen luonti ja poisto

Kyselyn luonti on toiminto, joka sallii pääkäyttäjälle uuden mielipidekyselyn luomisen. Toimintoon pääsee päävalikon linkistä ”Luo kysely”. Kyselyn luontilomakkeella käyttäjältä kysytään pakollisina tietoina kyselyyn liittyvä tunniste, kyselyn käyttäjätunnus ja lisäksi kyselyyn liitettävät kysymykset. Kyselyn tunniste toimii muissa kyselyihin liittyvissä pääkäyttäjän toiminnoissa linkkinä kyseiseen kyselyyn. Käyttäjätunnus taas toimii kyseisen kyselyn käyttäjätunnuksena kaikille kyselyyn vastaaville normaaleille käyttäjille. Kyselyihin tulee myös lisätä vähintään yksi kysymys. Kysymykset, niihin liittyvät vastausvaihtoehdot ja niiden järjestysnumerot kyselyssä voidaan valita luontilomakkeelle tulostuvista listoista. Lisäksi listalle voidaan luoda uusia ja listalta voidaan poistaa vanhoja kysymyksiä lomakkeella sijaitsevien ”Luo kysymys” ja ”Poista kysymyksiä” –linkkien välityksellä.

Uusi kysely voidaan luoda järjestelmään painamalla ”Luo kysely” tai luominen voidaan peruuttaa painamalla ”Päävalikkoon”. Kyselyn luontilomake voidaan myös tyhjentää painamalla ”Tyhjennä kentät”-näppäintä. Kyselyn luomisen yhteydessä lomakkeelle syötetyt tiedot tarkastetaan siltä varalta, että tiedoissa on virheitä. Virhetilanne aiheutuu, mikäli jokin kenttä on jäänyt tyhjäksi, joillekin kysymyksille on asetettu sama järjestysnumero tai syötetyt kyselyille yksilölliset tunniste ja käyttäjätunnus ovat jo käytössä järjestelmässä. Havaituista virheistä ilmoitetaan käyttäjälle, eikä virheellistä kyselyä tallenneta järjestelmään. Mikäli virheitä ei ilmene, luodaan kyselylle kyselytiedostoon oma kysely-elementti, johon lisätään kaikki käyttäjän syöttämät tiedot. Käytännössä lisättäviä tietoja ovat tunniste, käyttäjätunnus sekä kyselyyn liittyvät kysymykset ja näiden tiedot.

Kyselyn poisto –toiminnon avulla voidaan poistaa jokin järjestelmään luoduista kyselyistä. Poistolomakkeelle siirtyminen tapahtuu valitsemalla päävalikosta linkki ”Poista kyselyjä”. Kyselyjen poistolomakkeelle tulostetaan taulukkoon kaikkien järjestelmässä olevien kyselyiden tunnisteet ja näille valintanäppäimet. Poistettava kysely voidaan valita listalta ja tämän jälkeen valittu kysely voidaan poistaa painamalla ”Poista kysely”. Poistaminen voidaan peruuttaa painamalla ”Päävalikkoon”. Mikäli yhtään kyselyä ei ole valittu ja kyselyitä yritetään poistaa, tulostetaan virheilmoitus. Mikäli kysely on valittu ja sitä yritetään poistaa, kysytään käyttäjältä varmistus. Tällöin käyttäjä voi joko hyväksyä poistamisen tai peruuttaa toimenpiteen. Poistamisen yhteydessä tiedostoista poistetaan kyseiseen kyselyyn liittyvät kysely- ja vastaustaulut, mikäli näitä on olemassa. Kyseiset elementit tunnistetaan kyselyn tunnisteen perusteella.

5.2.3 Kysymysten luonti ja poisto

Pääkäyttäjä voi vapaasti luoda uusia kysymyksiä liitettäväksi kyselyihin kysymysten luonti –toiminnon avulla. Kyseiselle lomakkeelle pääkäyttäjä pääsee kyselyiden luontisivun kautta painamalla ”Luo kysymys”. Sivulle on ainoastaan yksi tekstikenttä, johon pääkäyttäjä voi kirjoittaa haluamansa tekstin. Kysymystekstistä poistetaan automaattisesti järjestelmän kannalta vahingolliset merkit, kuten erilaiset XML-elementit. Pääkäyttäjä lisätä kysymyksen tiedostoon painamalla ”Luo kysymys”. Tekstikentän voi tyhjentää painamalla ”Tyhjennä kenttä”. Mikäli käyttäjä ei haluakaan luoda kysymystä, pääsee hän takaisin kyselyn luontisivulle painamalla ”Kyselyn luontiin” tai päävalikkoon painamalla ”Päävalikkoon”. Kysymyksen luonnin yhteydessä kysymystiedostoon luodaan uusi kysymys-elementti, jonka sisällöksi kirjoitetaan käyttäjän syöttämä teksti.

Järjestelmään luotuja kysymyksiä voidaan poistaa valitsemalla päävalikosta tai kyselyn luontisivulta linkki ”Poista kysymyksiä”. Tällöin pääkäyttäjä ohjataan lomakkeelle, jolle tulostetaan kaikki kysymystiedostossa olevat kysymykset. Kysymykset voidaan poistaa valitsemalla halutut kysymykset listalta ja painamalla ”Poista kysymykset”. Lomakkeen kentät voidaan tyhjentää painamalla ”Tyhjennä kentät”. Kysymysten poistaminen voidaan peruuttaa painamalla ”Päävalikkoon” tai ”Kyselyn luontiin”, jolloin siirrytään linkin osoittamalle sivulle. Kysymysten poistamisen yhteydessä kysymystiedostosta poistetaan valittuja kysymyksiä vastaavat kysymys-elementit sisältöineen.

5.2.4 Kyselyjen esikatselu

Kyselyjen esikatselu –toiminnon avulla voidaan esikatsella pääkäyttäjän luomia mielipidekyselyitä. Esikatselutoiminto on pääkäyttäjän tapa selvittää, miltä kysely näyttää normaalin käyttäjän vastatessa siihen. Käytännössä toiminnon ulkoasu siis vastaa normaalin käyttäjän käyttöliittymää ilman järjestelmään kirjautumista ja vastausten varsinaista lähettämistä. Kyselyjen esikatselu –toimintoon pääsee pääkäyttäjän päävalikon kautta painamalla ”Esikatsele kyselyjä”. Tällöin käyttäjä ohjataan lomakkeelle, johon tulostuu esimerkiksi kyselyjen poisto –toiminnon tavoin lista järjestelmän sisältämien kyselyiden tunnisteista sekä näille valintanäppäimet. Haluttua kyselyä voidaan esikatsella valitsemalla se listalta ja painamalla ”Esikatsele”. Mikäli yhtään kyselyä ei ole valittu ja näppäintä painetaan, tulostuu ruudulle virheilmoitus. Muussa tapauksessa pääkäyttäjälle tulostetaan valittu kyselylomake. Esikatselusivulta voidaan palata takaisin päävalikkoon painamalla ”Päävalikkoon”.

Esikatselutoiminnossa haetaan kyselyn tiedot kyselytiedostosta ja tulostetaan ne näytölle järjestelmän taustalla toimivan CSS-tiedoston muotoilujen mukaisesti. Kysymykset ja niihin liittyvät vastausvaihtoehdot tulostetaan allekkain niille määrätyssä järjestyksessä. Huolimatta pääkäyttäjän suorittamista valinnoista kyselyn luomisen yhteydessä, kysymysten numerointi alkaa aina numerosta yksi ja kasvaa yhdellä siirryttäessä seuraavaan kysymykseen. Kyselyyn vastaaminen esikatselutoiminnon välityksellä ei ole mahdollista vaan vastaaminen täytyy aina suorittaa normaalin käyttäjän käyttöliittymän kautta.

5.2.5 Käyttäjien lisäys/haku

Käyttäjien lisääminen tapahtuu erillisellä lomakkeella, johon pääkäyttäjä pääsee painamalla päävalikosta linkkiä ”Lisää käyttäjiä”. Tällöin käyttäjä ohjataan lomakkeelle, joka on esimerkiksi kyselyjen poistamisesta tuttu. Lomakkeella on listattu järjestelmän sisältämien kyselyiden tunnisteet, joista jokaiselle on luotu oma valintanäppäimensä. Käyttäjän tulee ensimmäisenä hakea kyselyn tietoja valitsemalla kysely listalta ja painamalla ”hae tiedot”. Mikäli tietoja haetaan, eikä yhtään kyselyä ole valittu, tulostetaan käyttäjälle virheilmoitus. Muussa tapauksessa näytölle tulostuu valittuun kyselyyn liittyvät tiedot sekä käyttäjien lisäyslomake. Käyttäjien lisäys/haku –sivulta voidaan palata päävalikkoon painamalla ”Päävalikkoon”.

Näytölle tulostettaviin kyselyyn liittyviin tietoihin kuuluvat kyselyn tunniste ja käyttäjätunnus sekä kyselylle aiemmin luotujen käyttäjien määrä. Alle tulostuvaan käyttäjien lisäyslomakkeeseen voidaan halutessa syöttää lisättävien käyttäjien määrä. Tämän jälkeen käyttäjät voidaan lisätä järjestelmään painamalla ”Lisää käyttäjät”. Mikäli lisättävien käyttäjien määrä on negatiivinen, nolla tai kenttä on jätetty tyhjäksi, tulostetaan virheilmoitus. Muussa tapauksessa käyttäjä ohjataan sivulle, jolle tulostetaan kyselyn tiedot sekä lisättyjen käyttäjien salasanat. Automaattisesti luodut salasanat tulee kopioida talteen, sillä lisäyksen yhteydessä ne salataan, eikä niitä enää voi katsella myöhemmin. Salasanojen poimimisen jälkeen sivulta voidaan palata päävalikkoon.

Kyselyn tietojen hakeminen toimii siten, että kyselytiedostosta haetaan oikea kysely-elementti kyselyn tunnisteen perusteella. Kyselytiedostosta poimitaan tunnisteen lisäksi myös kyselyyn liittyvä käyttäjätunnus. Käyttäjien määrä saadaan hakemalla käyttäjätiedostosta oikean tunnisteen omaavaa kayttaja-elementtia ja laskemalla tämän sisältämät salasana-elementit. Käyttäjien lisääminen taas tapahtuu lisäämällä käyttäjätiedostoon oikean tunnisteen sisältämän kayttaja-elementin alle haluttu määrä uusia salasana-elementtejä. Näiden elementtien sisälle sijoitetaan järjestelmän generoimat salasanat, jotka on salattu käyttämällä sha1-menetelmää.

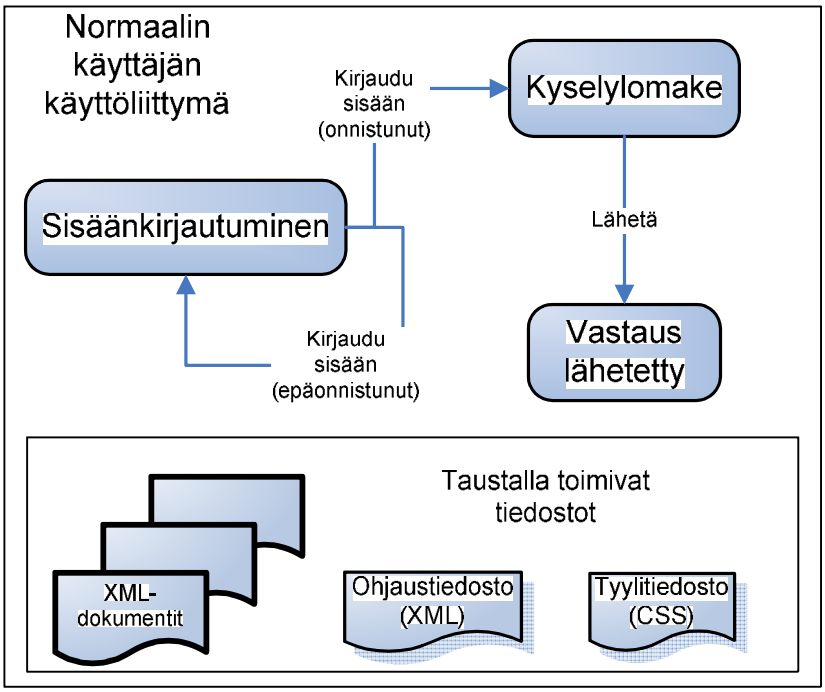
5.2.6 Kyselyjen tulosten seuraaminen

Kyselyjen tulosten seuraaminen hoidetaan järjestelmään luodun ohjatun toiminnon avulla. Toimintoon pääsee valitsemalla päävalikosta ”Näytä kyselyn tulokset”. Tällöin näytölle tulostuu aiemmin esitetyistä toiminnoista tuttu lomake, johon tulostetaan kyselyiden tunnisteet sekä näille valintanäppäimet. Valitsemalla yhden kyselyistä ja painamalla ”Hae tulokset” tulokset tulostetaan näytölle. Mikäli kyselyä ei ole valittu, tulostetaan virheilmoitus. Toiminto voidaan lopettaa painamalla ”Päävalikkoon”.

Tulosten seuraamistoiminnossa tulostetaan haettuun vastanneiden käyttäjien määrä, kyselyyn liittyvät kysymykset, näiden vastausvaihtoehdot sekä vastanneiden käyttäjien vastaukset. Mikäli kysymys on ollut tyypiltään monivalintakysymys - tyyppi-elementin arvo on ”radio” - tulostetaan vastausvaihtoehdoille kysymyksittäin vastausmäärät sekä lukuina että vastanneiden määrän suhteen prosentteina. Mikäli taas kysymyksen vastaus on

syötetty avoimeen tekstikenttään - tyyppi-elementin arvo on ”avoin” - tulostetaan käyttäjien vastaukset lomakkeelle allekkain. Vastanneiden määrä lasketaan laskemalla käyttäjätiedostosta käytettyjen salasanojen lukumäärä – kaytetty-attribuutin arvo on ”true” – ja kysymyksiin annettujen vastausten määrä lasketaan vertaamalla käyttäjien antamien vastausvaihtoehtojen nimiä vastausvaihtotiedoston sisältämien vaihtoehtojen nimiin.

5.3 Normaalikäyttäjien toiminnot



Kuva 4: Normaalikäyttäjän järjestelmän rakenne.

Normaalien käyttäjien ainoat toiminnot esitellyn suunnitelman mukaisesti järjestelmään kirjautuminen ja käyttäjätunnuksen ja salasanan osoittamaan kyselyyn vastaaminen. Mikäli kirjautuminen epäonnistuu, ilmoitetaan siitä käyttäjälle ja kirjautumislomake tulostetaan uudelleen. Kirjautumisen onnistuessa näytölle tulostetaan käyttäjätunnuksen ja salasanan osoittama kyselylomake. Kyselyyn vastatessa käyttäjän tulee antaa vastaus jokaiseen kysymykseen. Mikäli jotain vastausta ei ole annettu, järjestelmä pyytää sitä uudelleen kun kyselyn vastauksia yritetään tallentaa. Vastausten tallentaminen tapahtuu painamalla ”Lähetä”. Vastauksen lähettämisen jälkeen käyttäjä ohjataan sivulle, jossa kerrotaan vastausten lähettämisen onnistumisesta tai epäonnistumisesta.

Käyttäjän kirjautumisen yhteydessä haetaan käyttäjätiedostosta oikea kysely kyselyyn liitetyn käyttäjätunnuksen perusteella. Mikäli käyttäjätunnus löytyy ja kyseinen kayttaja-elementti sisältää myös salattuna käyttäjän syöttämän salasanan, jota ei ole vielä käytetty, päästetään käyttäjä vastaamaan kyselyyn. Muussa tapauksessa käyttäjälle tulostetaan virheilmoitus. Käyttäjän edetessä kyselyyn haetaan kyselytiedostosta oikeaa kyselyä käyttäjän syöttämän käyttäjätunnuksen perusteella. Kysely tulostetaan näytöllä samoin kuin pääkäyttäjän kyselyn esikatselu –toiminnossa, mutta lisäämällä lomakkeelle ”Lähetä”-näppäin. Käyttäjän lähettäessä kyselyn, tarkastetaan edellä mainitulla tavalla käyttäjän vastaukset ja edetään tallennusvaiheeseen. Tällöin tarkastetaan uudelleen käyttäjän syöttämän salasanan kaytetty-attribuutin arvo. Mikäli salasanaa ei ole käytetty, tallennetaan vastaukset vastaustiedostoon tiedoston edellyttämään muotoon ja tulostetaan käyttäjälle tähän viittaava viesti. Muussa tapauksessa käyttäjän vastaukset hylätään ja esitetään käyttäjälle virheilmoitus.

6 TEKNIIKOIDEN SOVELTUVUUS JA TULEVAISUUDEN NÄKYMÄT

Työssä suunnitellulle ja toteutetulle järjestelmälle on olemassa useita erilaisia toteutustapoja. Järjestelmän suunnittelun ja toteutuksen lähestymistavat voivat poiketa toisistaan esimerkiksi erilaisten ohjelmointikielten ja mahdollisesti jopa erilaisten arkkitehtuurien käytössä, tiedon varastointimenetelmien toteutustavoissa sekä toteutettavan järjestelmän sisältämissä toiminnoissa ja niiden toteutustavoissa. Tässä kappaleessa arvioidaan työssä aiemmin esiteltyjen ja järjestelmän toteutuksessa käytettyjen menetelmien soveltuvuutta ja yhteensopivuutta suunnitellun järjestelmän toteutukseen. Samalla tuodaan pintapuolisesti esille muutamia menetelmiä, joita olisi voitu käyttää tässä projektissa tai joita voidaan käyttää mahdollisesti tulevaisuudessa vastaavan järjestelmän toteuttamiseen. Lisäksi käydään läpi mahdollisia ajatuksia toteutetun järjestelmän jatkokehityksestä sekä mahdollisista uusista ominaisuuksista.

6.1 Toteutustekniikoiden arviointi

Järjestelmän toteutuksessa pääosin käytetyt tekniikat olivat toteutuksen osalta HTML ja PHP, järjestelmän ulkoasun muokkaamisen osalta osittain HTML ja pääosin CSS sekä tietojen varastoinnin osalta XML. Kyseiset tekniikat valittiin paitsi aiempien hyvien kokemusten perusteella, myös juuri niiden hyvän soveltuvuuden ja yhteensopivuuden takia. Koko projektin vaatimuksena oli toteuttaa kahdella erillisellä käyttöliittymällä toimiva selainpohjainen mielipidekyselyjärjestelmä, jonka käyttäminen on ennen kaikkea helppoa ja suhteellisen turvallista. Selainpohjaisen järjestelmän kehittämiseen ei käytännössä ole muita pohjaratkaisuja kuin HTML, joten kyseinen kieli soveltui toteutukseen paremmin kuin hyvin.

PHP on lähiaikoina yleistynyt Internet-pohjaisten sovellusten toteutuskielenä ja oli HTML-kielen tavoin luonnollinen valinta järjestelmän toteutuskieleksi. Lisäksi PHP on kehitetty nimenomaan selainpohjaisten järjestelmien toteutusta ja HTML-kieltä silmällä pitäen. Käytännössä PHP tarjosi järjestelmän toteutukselle ohjelmointimahdollisuudet sekä hyvät liittymät HTML-kielisten dokumenttien luomiseen ja HTML tarjosi tiedon selainpohjaisen esitystavan. Nimenomaan HTML- ja XML-kielisten dokumenttien muotoiluun tarkoitettu CSS-menetelmä täydensi yhteensopivan yhdistelmän.

Ainoa heikkous käytetyissä toteutustavoissa ja kielissä oli enemmänkin tiedon kuljettamiseen kuin varastointiin käytetty XML-kieli. XML-kielen pohjalta muotoiltujen tiedostojen toimivuus tiedon varastointiformaattina sekä näiden tiedostojen lukeminen ja tulkitseminen PHP-kielen avulla oli työlästä. Vaikka PHP tarjoaa oman XML-kielen jäsentimen (parser), oli tämän käyttäminen toteutuksessa vähintäänkin työlästä. Tämän vuoksi suurin osa XML- ja PHP-kielten välisistä yhteyksistä toteutettiin itse laadittujen jäsenninfunktioiden avulla. Lähestymistapa oli kohtuullisen työläs, mutta toimiva. Tiedon varastointitapana XML-kielen selkeä heikkous on tiedostojen suuri koko ja suurista elementtimääristä johtuva tavallaan tarpeeton kasvaminen. Käytännössä XML-kielen käyttäminen tiedon varastointitapana tarkoittaa toteutetun järjestelmän kannalta sitä, että esimerkiksi suurien kysely- tai käyttäjämäärien palveleminen saattaa hidastaa järjestelmän toimivuutta. Tosin tällöin puhutaan jo erittäin suurista tietomääristä, jotka on käytännössä helppo välttää rajoitetuissa käyttöympäristössä.

Muita käyttökelpoisia menetelmiä toteutukseen olisi ollut ilman muuta jo aiemmin esitelty MySQL-tietokanta tiedon varastoinnissa. Kyseinen tiedontallennustapa olisi mahdollistanut tiedon lisäämisen ja hakemisen tietokannasta ilman nykyiseen järjestelmään toteutettua tiedon parsimista. Lisäksi tieto olisi voitu varastoida huomattavasti pienempään tilaan ja suojata helposti tietokannan omien tietoturvaominaisuuksien avulla. PHP-kielen avuksi toteutuksessa olisi voitu ottaa myös toinen selaimessa tulkattavana toimiva ohjelmointikieli, JavaScript. Kyseisellä kielellä olisi voitu toteuttaa esimerkiksi lomakkeille syötettyjen tietojen oikeellisuus. Varsinaista PHP-kieltä parempaa korvaajaa järjestelmän taustalla toimivan toiminnallisuuden toteutukseen on vaikea keksiä.

Yhtenä vaihtoehtona koko järjestelmän toteutustapojen korvaajaksi olisivat kuitenkin nykyään suurten järjestelmien toteutustapoja hallitseva J2EE (Java 2 Enterprise Edition), tähän helposti liitettävä sivujen ulkoasujen toteutukseen tarkoitettu JSP (Java Server Pages) sekä kyseiset menetelmät toisiinsa yhdistävä Struts. Lisäksi J2EE- ja JSP-menetelmien taustalla toimiva tietokanta voidaan yhdistää toteutukseen käyttämällä samaa Struts-runkoa. Kyseinen järjestelmäarkkitehtuuri sopii kuitenkin paremmin suuremman luokan järjestelmiin ja tässä työssä suunnitellun ja toteutetun suhteellisen pienen järjestelmän toteutus mainittuja menetelmiä käyttämällä vaatisi ainakin huomattavasti suurempia

valmisteluja kuin tämän hetkinen järjestelmän toteutus. Myös menetelmiä tukevien kehityssovellusten hankkiminen ja asentaminen vaatisi aikaa ja todennäköisesti myös pääomaa.

6.2 Järjestelmän kehitysmahdollisuudet

Toteutetulla mielipidekyselyjärjestelmällä on laajat kehittämismahdollisuudet. Järjestelmää voidaan kehittää esimerkiksi vaihtamalla järjestelmän pohjalla toimivia toteutusmenetelmiä tai luomalla järjestelmään erilaisia uusia ominaisuuksia. Käytännössä uusia ominaisuuksia voi luoda joko liittyen tämänhetkiseen toiminnallisuuteen tai laajentamalla järjestelmää mielipidekyselyiden lisäksi esimerkiksi palautteen keräämiseen tai vaikkapa työhakemusten tai tuotetilausten vastaanottoon. Kuitenkin mikäli järjestelmän käyttötarkoitus ja toimintaperiaate pidetään samana, rajoittuu kehittäminen lähinnä menetelmien päivittämiseen ja uusien ominaisuuksien kehittämiseen. Näiden osalta järjestelmän kehittäminen kannattaa aloittaa järjestelmän tämänhetkisten puutteiden korjaamiseen. Selkeitä puutteita ovat järjestelmän tietovarastoina toimivat XML-tiedostot sekä muutamien järjestelmän toimivuuden kannalta suhteellisen tärkeiden ominaisuuksien puuttuminen.

Tietovarastoina toimivat XML-tiedostot tulisi korvata PHP-ohjelmointikielen kanssa yhteensopivalla tietokannalla, esimerkiksi MySQL-tietokannalla. Tietokannan käyttäminen selkeyttää järjestelmän rakennetta, parantaa suorituskykyä ja toimintavarmuutta sekä mahdollistaa tietojen varastoinnin pienempään tilaan. Käytännössä myös tietojen haku ja tallentaminen nopeutuisi uuden tietojenvarastointimenetelmän myötä. Siirtyminen XML-tiedostoista tietokantaan olisi myös kohtuullisen helppoa, sillä ohjelma on toteutettu tätä muutosta silmällä pitäen. Myös PHP-kielen hyvät tietokantayhteydet tarjoavat mahdollisuudet järjestelmän tietovaraston helpompaan muuttamiseen.

Järjestelmän kehittämisen yhteydessä järjestelmään voisi toteuttaa muutamia siitä puuttuvia ominaisuuksia. Näistä näkyvimpänä puutteena on kyselyiden vastausvaihtoehtojen muokkaamisominaisuuden puuttuminen. Tällä hetkellä muutokset tulee tehdä suoraan vastausvaihtoehtoja kontrolloivaan XML-tiedostoon, joka vaikeuttaa järjestelmän pääkäyttäjän toimintaa merkittävästi. Käytännössä vastausvaihtoehtojen muokkaamisen mahdollistamiseen tähtäävä uudistus tarkoittaisi uuden ominaisuuden

kehittämistä pääkäyttäjän käyttöliittymään. Uudistus voitaisiin toteuttaa suhteellisen pienellä vaivalla käyttämällä olemassa olevia ominaisuuksia varten kehitettyjä haku- ja parsimisfunktioita. Muita mahdollisia uusia ominaisuuksia olisivat esimerkiksi kyselyiden ulkoasun muuttaminen käyttämällä vaihtoehtoisia pääkäyttäjän kyselykohtaisesti valitsemia CSS-tiedostoja sekä kyselyihin asetettujen kysymysten pakollisuuden määrittäminen. Myös molemmat näistä ominaisuuksista olisivat helposti toteutettavissa tämänhetkisen järjestelmän menetelmien avulla.

7 JOHTOPÄÄTÖKSET JA JATKOKEHITYSKOHTEET

Tämän työn tarkoituksena oli suunnitella ja toteuttaa Internet-pohjainen mielipidekyselyjärjestelmä. Menetelmiksi järjestelmän toteutukseen valittiin jo etukäteen HMTL- ja PHP-kielet sekä CSS-menetelmä. HTML-kielen avulla toteutettiin järjestelmään sekä pääkäyttäjän että normaalin käyttäjän käyttöliittymät, joiden muotoiluun käytettiin lisäksi CSS-menetelmän tarjoamia mahdollisuuksia. PHP-kieli taas toimi järjestelmän taustalla käyttäjille laadittujen operaatioiden toteutuskielenä. Käytettävien menetelmien valinnat suoritettiin perustuen aiempiin kokemuksiin selainpohjaisten järjestelmien kehittämisestä sekä menetelmien yhteensopivuudesta. Tiedon varastointitavaksi valittiin olosuhteiden pakottamana XML-muotoisista tiedostoista ja MySQL-tietokannasta XML, vaikka MySQL-tietokannan oletettiin soveltuvan paremmin kyseiseen käyttötarkoitukseen. Mikäli järjestelmän loppukäyttäjällä olisi ollut mahdollisuus MySQL-tietokannan käyttöön, olisi toteutustapa luultavasti vaihdettu tähän. Puutteistaan huolimatta XML osoittautui kuitenkin lopullisen järjestelmän kannalta toimivaksi ratkaisuksi.

Järjestelmän suunnittelu ja toteutus onnistui kohtuullisen hyvin, mutta projektin aikana huomattiin joitakin puutteita. Suunnitteluun vaikutti huomattavasti tarkkojen järjestelmävaatimusten puute sekä ehkä osittain jopa liiallinen toteutukseen liittyvä vapaus. Puutteellisten järjestelmävaatimusten vuoksi järjestelmään ei ehditty toteuttaa aivan kaikkia oleellisia ominaisuuksia. Tähän ei auttanut myöskään se, että loppukäyttäjän kanssa keskusteltiin tiiviisti projektin edetessä. Toteutuksessa onnistuttiin kuitenkin kohtuullisen hyvin ottaen huomioon loppukäyttäjän järjestelmän aiheuttamat rajoitteet ja suunnitteluosuuden osittainen vajavaisuus. Tärkeimmät ominaisuudet, kuten kyselyiden luominen, käyttäjien lisääminen ja kyselyiden tulosten seuraaminen saatiin toteutettua hyvin. Parantamisen varaa jäi lähinnä järjestelmän suunnitteluvaiheeseen, mutta puutteista johtuvat järjestelmän vajavaisuudet voidaan suhteellisen helposti korjata mahdollisessa jatkokehitysprojektissa.

Tärkeimmät jatkokehityksen kohteet järjestelmässä ovat toteutukseen käytettyjen menetelmien muokkaaminen sekä järjestelmän kehittäminen luomalla siihen uusia ominaisuuksia. Menetelmien muokkaamisella tarkoitetaan lähinnä haluttujen toteutusmenetelmien vaihtamista toisiin paremmin toimiviin menetelmiin. Tässä tapauksessa järjestelmän toteutuksessa voitaisiin siirtyä esimerkiksi käyttämään XML-

muotoisten tiedostojen sijaan MySQL-tietokantaa tietojen varastointiin tai siirtää koko järjestelmä toimimaan J2EE-arkkitehtuurin ja siihen liittyvien menetelmien mukaisesti. Jälkimmäinen vaihtoehto ei kuitenkaan ole kannattava kyseessä olevan kokoluokan järjestelmässä. Uusien ominaisuuksien kehittäminen kannattaa aloittaa täydentämällä tämän hetkistä järjestelmää puuttuvien ominaisuuksien osalta. Suurimmat puutteet ovat tällä hetkellä vastausvaihtoehtojen muokkausominaisuuden puuttuminen, kysymyksiin liittyvien vastauksien pakollisuuden määrittäminen sekä kyselyiden rajoittaminen käyttämään ainoastaan yhtä CSS-tyylitiedostoa. Kaikki nämä ominaisuudet ovat lisäksi helposti toteutettavissa järjestelmän yksinkertaisen toteutuksen ansiosta.

Järjestelmään valitut toteutusmenetelmän toimivat XML-kielen puutteita lukuun ottamatta saumattomasti yhteen ja siltä osin sekä suunnittelu että toteutus onnistuivat erinomaisesti. Kuten jo mainittu, kehitysmahdollisuuksia järjestelmään jäi lähinnä liittyen tiedon varastointimenetelmien muuntamiseen ja uusien menetelmien kehittämiseen. Järjestelmä kuitenkin toimii nykyisessä käyttötarkoituksessaan erittäin hyvin myös tämänhetkisten ominaisuuksien puitteissa, eikä järjestelmän jatkokehitys ole välttämätöntä. Joidenkin uusien ominaisuuksien toteuttaminen voi kuitenkin olennaisesti helpottaa järjestelmän käyttämistä ja tuoda sen käyttäjille lisäarvoa.

LÄHTEET

1. Bray Tim, Paoli Jean, Sperberg-McQueen C.M., Maler Eve, Yergeau François, Cowan John (editors). 2004. Extensible Markup Language (XML) 1.1. W3C Recommendation 4.2.2004, edited in place 15.4.2004. [Viitattu 25.9.2006]. Saatavissa: http://www.w3.org/TR/xml11/

2. Cascading Style Sheets Home Page. [verkkodokumentti]. 2006. [viitattu 2.9.2006]. W3C. Saatavissa: http://www.w3.org/Style/CSS/#specs

3. Heinisuo R. PHP ja MySQL. 2. painos. 2003. Jyväskylä: Gummerus Kirjapaino Oy, Talentum Media Oy, 237 s. Sivut 16-19, 36-38, 63-65 ja 174-177. ISBN 951-762-833-1

4. Hovi, A. SQL-opas. 1. painos. 2004. Saarijärvi: Saarijärven Offset Oy, Docendo Finland Oy, 208 s. Sivut 4, 5-10, 14-20 ja 237-241. ISBN 941-846-228-3

5. Häggman C. Web-Design. 1. painos. 2001. Jyväskylä: Tummavuoren kirjapaino, Docendo Finland Oy, 166 s. Sivut 35-40 ja 112-134. ISBN 951-846-094-9

6. Introduction to Cascading Style Sheets, level 2. [verkkodokumentti]. 1998. [viitattu 2.9.2006]. W3C. Saatavissa: http://www.w3.org/TR/CSS2/intro.html

7. Introduction to PHP: What can PHP do? [verkkodokumentti]. 2005. [viitattu 5.9.2006]. Saatavissa: http://fi.php.net/manual/fi/print/intro-whatcando.php

8. Introduction to PHP: What is PHP? [verkkodokumentti]. 2005. [viitattu 5.9.2006]. Saatavissa: http://fi.php.net/manual/fi/print/introduction.php

9. Korpela J.K. & Linjama T. Web-suunnittelu. 1. painos. 2003. Porvoo: WS Bookwell, Docendo Finland Oy, 457 s. Sivut 72-109 ja 304-315. ISBN 951-846-172-4

10. Lemay L. Opeta itsellesi WWW-julkaiseminen – HTML 4. 1998. Jyväskylä: Gummerus Kirjapaino Oy, Suomen ATK-kustannus Oy, 618 s. Sivut 48-58 ja 517-539. ISBN 951-762-622-4

11. PHP Characteristics: Persistent Database Connections. [verkkodokumentti]. 2005. [viitattu 11.9.2006]. Saatavissa: http://fi.php.net/manual/fi/features.persistent-connections.php

12. PHP Database Security: Encrypted Storage Model. [verkkodokumentti]. 2005. [viitattu 18.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.database.storage.php

13. PHP Database Security: SQL Injection. [verkkodokumentti]. 2005. [viitattu 18.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.database.sql-injection.php

14. PHP Functions: Introduction to Session Handling Functions. [verkkodokumentti]. 2005. [viitattu 11.9.2006]. Saatavissa: http://fi.php.net/manual/fi/ref.session.php

15. PHP Functions: MySQL Functions. [verkkodokumentti]. 2005. [viitattu 19.9.2006]. Saatavissa: http://www.php.net/manual/fi/ref.mysql.php

16. PHP Functions: XML Parser Functions. [verkkodokumentti]. 2005. [viitattu 19.9.2006]. Saatavissa: http://fi.php.net/manual/fi/ref.xml.php

17. PHP Security: Database Security. [verkkodokumentti]. 2005. [viitattu 18.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.database.php

18. PHP Security: File System Security. [verkkodokumentti]. 2005. [viitattu 14.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.filesystem.php

19. PHP Security: General considerations. [verkkodokumentti]. 2005. [viitattu 11.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.general.php

20. PHP Security: Installed as a CGI binary. [verkkodokumentti]. 2005. [viitattu 11.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.cgi-bin.php

21. PHP Security: Installed as an Apache module. [verkkodokumentti]. 2005. [viitattu 14.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.apache.php

22. PHP Security: Introduction. [verkkodokumentti]. 2005. [viitattu 11.9.2006]. Saatavissa: http://fi.php.net/manual/fi/security.intro.php

23. PHP:n ominaisuudet: Evästeet. [verkkodokumentti]. 2005. [viitattu 10.9.2006]. Saatavissa: http://fi.php.net/manual/fi/features.cookies.php

24. PHP:n ominaisuudet: PHP ja HTTP-autentikaatio. [verkkodokumentti]. 2005. [viitattu 10.9.2006]. Saatavissa: http://fi.php.net/manual/fi/features.http-auth.php

25. Raggett D., Le Hors A. & Jacobs I. HTML 4.01 Specification. [verkkodokumentti]. 1999. [viitattu 1.9.2006]. W3C. Saatavissa: http://www.w3.org/TR/REC-html40/

26. The Main Features of MySQL. [verkkodokumentti]. 2006. [viitattu 23.9.2006]. Saatavissa: http://dev.mysql.com/doc/refman/4.1/en/features.html

27. Tuikka Tommi, Kanala Sari. XML Ohjelmoinnin perusteet. 2001. Helsinki: Oy Edita Ab, 132 s. Sivut 3-17. ISBN 951-826-407-4

28. Wium Lie H. & Bos B. Cascading Style Sheets, level 1. [verkkodokumentti]. 1999. [viitattu 2.9.2006]. W3C. Saatavissa: http://www.w3.org/TR/CSS1