2018

Käyttölupajärjestelmän tekniset ratkaisut ja sovellusarkkitehtuuri

26.4.2018

Versiohistoria

Versio:	Pvm:	Laatijat:	Selitys:
Versio 0.1	9.7.2014	Kati Laakso	Dokumentin luonti
Versio 0.2	12.9.2014	Kati Laakso	Käytettäviä tekniikoita tarkennettu
Versio 0.3	04.12.2014	Timo Ojala	Arkkitehtuurikuvausta tarkennettu
Versio 0.31	28.1.2016	Henri Tenhunen	Pieniä tarkennuksia eri kohtiin
Versio 0.32	6.9.2016	Irma-Leena Notkola	Korjattu version 0.31 pvm
Versio 0.4	3.1.2017	Sami Nousiainen	Dokumentin rakenne uusittu.
Versio 0.5	4.1.2017	Sami Nousiainen, Henri Tenhunen, Suvi Lumivirta	Dokumentin sisältöä päivitetty ja ajantasaistettu.
Versio 0.51	5.1.2017	Henri Tenhunen	Päivitetty osio 4.3.3.
Versio 0.52	9.1.2017	Sami Nousiainen	Päivitetty kuva 2 (selainsovelluksen rakenne).
Versio 0.6	15.12.2017	Suvi Lumivirta, Henri Tenhunen	Lisätty uusi osio (konfiguroitavuus) sekä päivitetty dokumentin sisältöä
Versio 0.61	28.3.2018	Henri Tenhunen	Päivitetty osio 5.1
Versio 0.62	28.3.2018	Irma-Leena Notkola	Päivitetty Johdanto ja muutettu tiedostodokumentin otsikko (uusi otsikko: Lupajärjestelmä_Arkkitehtuuri_v…; vanha otsikko: FMAS_Arkkitehtuuri_v…)
Versio 0.63	28.3.2018	Henri Tenhunen	Päivitetty osio 3.4.1
Versio 0.7	10.4.2018	Henri Tenhunen	Päivitetty osiot 3.2, 4.1.4, 4.1.7, 4.2.1, 5.1, 7.1 ja 7.2
Versio 0.8	25.4.2018	Henri Tenhunen	Päivitetty osiot 2.1, 3.4.1 ja 4.1.4
Versio 0.9	26.4.2018	Henri Tenhunen	Päivitetty osio 3.1 ja lisätty prosessikaaviot

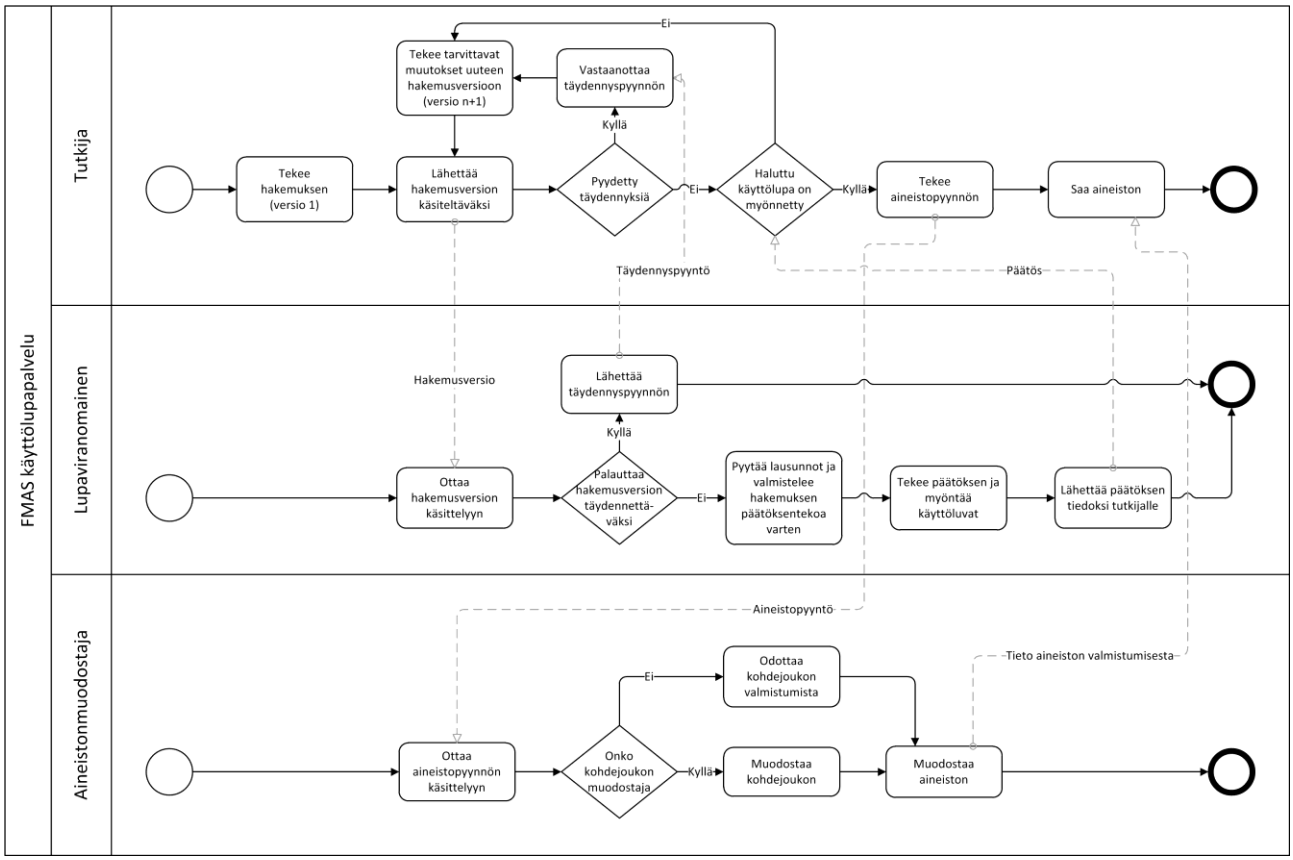
Sisällys

1 Johdanto

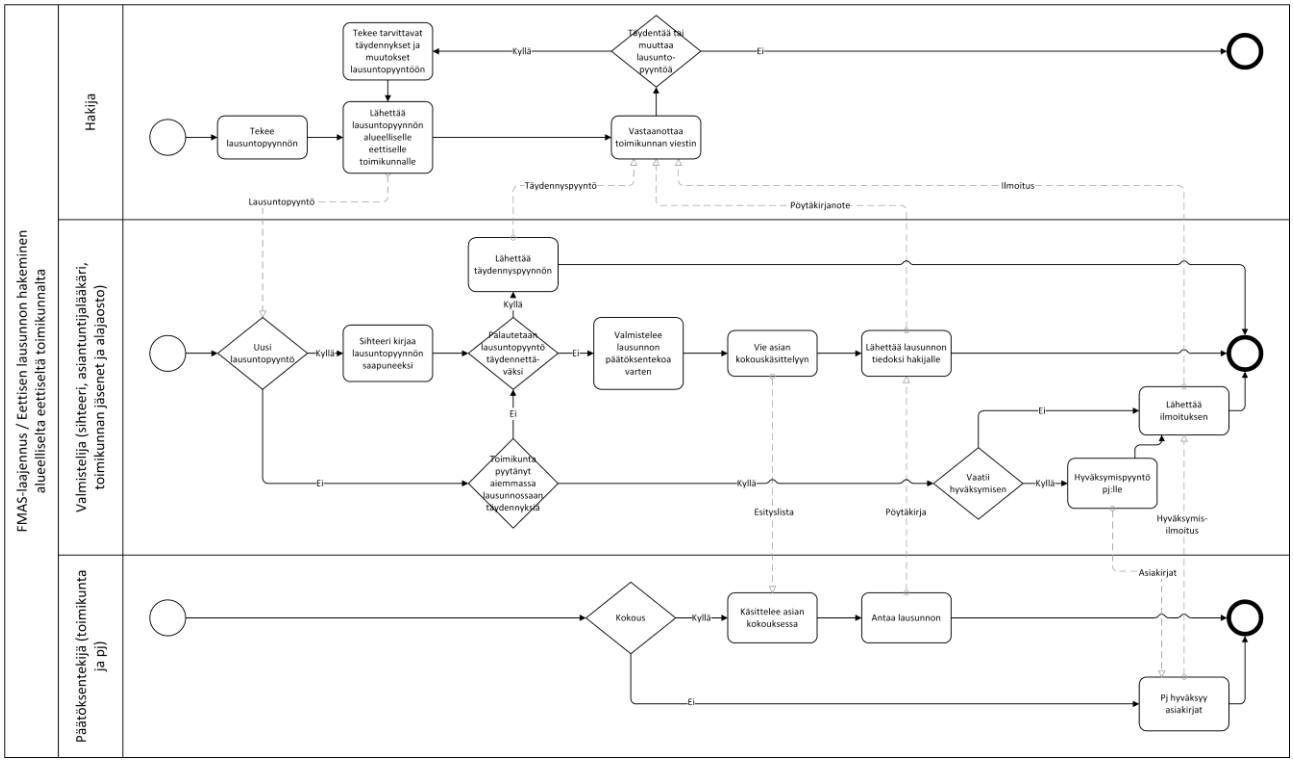
Tässä dokumentissa kuvataan sähköisen lupapalvelun teknisiä ratkaisuja ja sovellusarkkitehtuuria. Lupapalvelun kehittäminen aloitettiin vuonna 2014 hankkeessa Kansallinen rekisteri- ja mikroaineistojen tutkijapalvelu (Finnish Microdata Access Services, FMAS) ja sitä on jatkettu vuoden 2016 loppupuolelta lähtien Sitran rahoittamassa Isaacus-esituotantohankkeessa Kansallinen lupapalvelu ja informaatio- ja tukipalvelu.

2 Prosessi

Lupapalvelun perustoiminnallisuus on suoraviivainen: tutkimusryhmään kuuluva hakija kirjautuu järjestelmään, luo hakemuksen ja lähettää sen yhdelle tai useammalle viranomaiselle. Käyttölupien myöntämisen jälkeen hakija voi tehdä aineistotilauksen ja lupapalvelu ylläpitää tietoa keillä on voimassa oleva käyttölupa mihinkin aineistoon. Lupaprosessia on pääpiirteissään kuvattu seuraavissa kuvissa.



Kuva 1, FMAS-lupapalvelun toiminnallinen kaaviokuva; vain varsinainen lupakäsittelyprosessi kuvattu, esim. arkistointi puuttuu



Kuva 2, Eettisen arvioinnin sähköisen palvelun toiminnallinen kaaviokuva; vain varsinainen hakemuksen käsittelyprosessi kuvattu, esim. arkistointi puuttuu

3 Toteutusympäristö

3.1 Toteutustapa ja käytetyt tekniikat

- Linux-käyttöjärjestelmä
- Apache HTTP-palvelin (sisältää kaiken lupajärjestelmän toiminnallisuuden)
- MySQL Community Edition –tietokannanhallintajärjestelmä
- PHP-ohjelmointikieli
- Määrittelytason tietomalli kuvataan MS Visio 14-versiolla
- Suunnittelutason tietomalli/tietokanta kuvataan ER-kaaviona Visual Paradigm avulla
- Käyttäjien käyttäjätunnukset tallennetaan lupajärjestelmän tietokantaan
- Salasana tallennetaan tietokantaan md5 (cryptographic hash) -funktio tiivisteenä
- Käyttäjätunnus on järjestelmän käyttäjän sähköpostiosoite
- Hakijoiden ja viranomaisten käyttäjätunnus/salasana-pari tallennetaan eri tauluihin
- Käyttäjätunnukset tallennetaan tekstimuodossa tietokantaan
- Tietokannan merkistöksi asetetaan UTF-8
- Taulut luodaan transaktio-turvallisiksi käyttämällä InnoDB-engineä MyISAM-moottorin sijaan (create table(...)engine=InnoDB default charset=UTF8;)
- Tietokantaoperaatiot hoidetaan ACID-doktriinia noudattamalla (atomicity, consistency, isolation and durability)
- Samanaikaisuus tietokannassa ns. *Optimistic concurrency control* (OCC)–periaatteiden mukaisesti

- Järjestelmän käyttöliittymä toteutetaan suomeksi ja englanniksi
- Käyttöliittymän tyylit (esim. fonttikoot) kuvataan tyylitiedoston (*.css) avulla
- SOAP tietoliikenneprotokollan toteutuksessa käytetään PHP WSDL Creatoria
- Liikenne verkon yli salataan SSL (Secure Sockets Layer) –tietoturvaprotokollan avulla
- SÄHKE2-määräyksen mukainen toteutus (tietokannat ja rajapinnat)
- Järjestelmien välinen kommunikointi toteutetaan SOAP–protokollaa käyttäen (tai vaihtoehtoisesti: REST)
- Testitapaukset suunnitellaan ja kirjataan JIRA-sovellukseen
- Lupajärjestelmän kehitys- ja ylläpidossa käytetään SVN versionhallinta- ohjelmistoa

3.2 Käytetyt toteutuskielet ja komponentit

Lupapalvelu on toteutettu käyttäen HTML-, CSS-, JavaScript-, PHP- ja SQL-kieliä. Näiden kielten perustoiminnallisuuksien lisäksi on käytetty muutamaa ulkoista kirjastoa/komponenttia: JQuery-JavaScript-kirjastoa[1], Pikaday-kalenterikomponenttia[2], TCPDF-pdf-kirjastoa[3] ja WSDL creator -kirjastoa[4]. Toteutuksessa on pyritty siihen, että yhdessä tiedostossa olisi sekoitettu mahdollisimman harvaa ohjelmointikieltä keskenään. Ohjelmistokerroksittain (kts. kappale 4.2 ja Kuva 4) tarkasteltuna toteutuskieliä on käytetty seuraavasti:

- Käyttöliittymäkerros: HTML, CSS, JavaScript, PHP
- Logiikkakerros: PHP, HTML, CSS
- Tietokantakerros: PHP, SQL

Käyttöliittymäkerroksessakin käytetyt kielet on rajattu erillisiin tiedostoihin mm. seuraavasti:

- ui/views/*_view.php-tiedostoissa sekaisin HTML-koodia ja PHP-koodia, joka tulostaa arvoja/dataa HTML-koodin joukkoon
- JavaScript-koodi sijaitsee ensisijaisesti static/js/-hakemiston tiedostoissa
- CSS-koodi sijaitsee static/css/-hakemiston tiedostoissa

Tietokantakerroksen SQL-koodi sijaitsee *DAO.php-nimisissä tiedostoissa.

Toteutuksessa tulee vastaan seuraavanlaisia eri ohjelmointikielten välisiä rajapintoja:

- HTML – PHP: PHP-koodista voidaan tulostaa HTML-koodia ja HTML-koodin sisällä voi olla linkkejä PHP-koodiin ja itse PHP-koodia (<?php ?> -tagien sisällä).
- HTML – JavaScript: JavaScript-koodissa voidaan käsitellä HTML-koodissa muodostuneita elementtejä/komponentteja (esim. document.getElementById:n avulla).

[1] https://jquery.com/
[2] https://github.com/dbushell/Pikaday
[3] https://tcpdf.org/
[4] https://github.com/piotrooo/wsdl-creator

- JavaScript – PHP: Muuttujien arvoja voidaan siirtää PHP-koodista JavaScript-koodiin (esim. käyttöliittymäkerroksen ui/template/header.php-tiedostossa tyyliin var x=<?php echo $x; ?>;, missä x on JavaScript-muuttuja ja $x on PHP-muuttuja, jonka arvo siirretään JavaScript-muuttujaan).

3.3 Nimeämiskäytännöt

PHP-koodin kirjoituksessa on pyritty noudattamaan seuraavia nimeämiskäytäntöjä:

- DTO- ja DAO-luokkien nimet (nämä muodostavat käytännössä suurimman osan lähdekoodin luokista) on kirjoitettu isolla alkukirjaimella ja luokkien nimien lopussa on DTO tai DAO
- DTO-luokkien attribuuttien nimet on kirjoitettu isolla alkukirjaimella. Metodeita DTO-luokissa ei ole ollenkaan.
- DAO-luokkien metodien nimet on kirjoitettu pienellä alkukirjaimella
- Funktioiden nimet on kirjoitettu pienellä alkukirjaimella
- Muuttujien nimet on kirjoitettu pienellä alkukirjaimella
- Vakioiden (ml. käyttöliittymässä näkyvät tekstit sisältävät vakiot) nimet on kirjoitettu isoilla kirjaimilla kokonaan

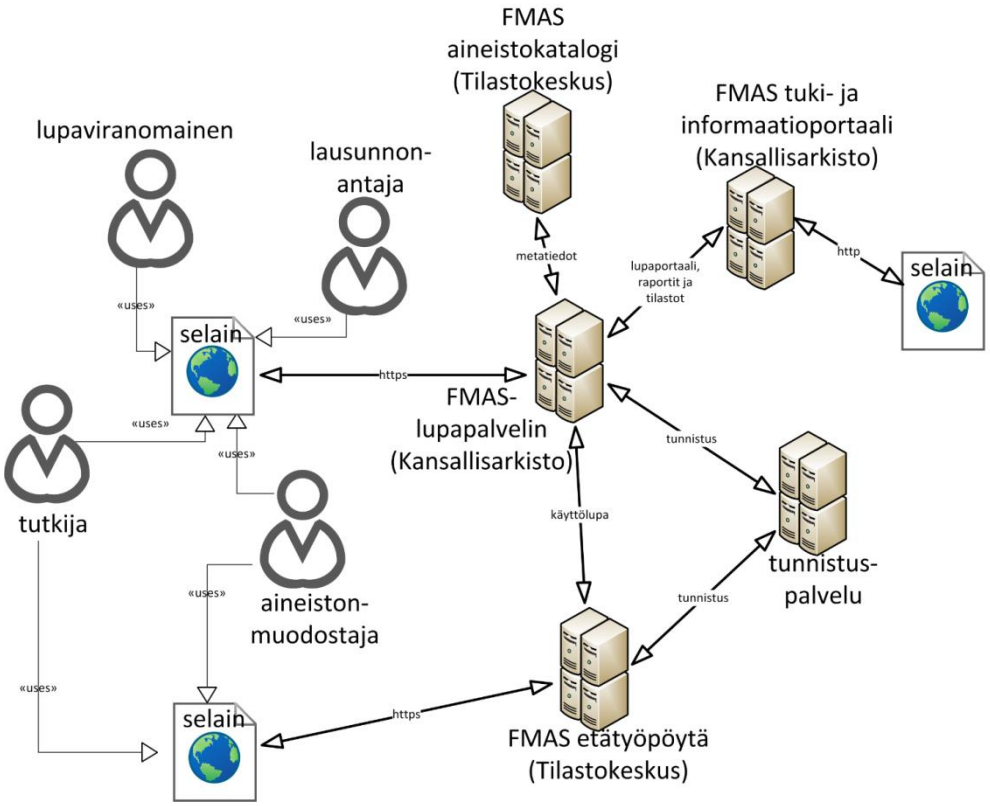
Lisäksi DAO-luokkien metodien nimien alkuosalla on pyritty kertomaan, millaisen tietokantaoperaation kyseinen metodi suorittaa:

- hae_*: metodi hakee tietokannasta tietoa (SQL:n SELECT-lause)
- poista_*: metodi poistaa tietokannasta tietoa (SQL:n DELETE-lause)
- paivita_*: metodi päivittää tietokannassa olevaa tietoa (SQL:n UPDATE-lause)
- alusta_*, luo_*: metodi alustaa/luo tietokantaan tietoa (SQL:n INSERT-lause)

4 Arkkitehtuuri

4.1 Arkkitehtuurin yleiset periaatteet

Lupajärjestelmässä on viisi käyttäjäryhmää: hakija, viranomainen, lausunnonantaja, aineistonmuodostaja ja pääkäyttäjä. Näiden käyttäjäryhmien roolit ja käyttöoikeudet on kuvattu järjestelmän vaatimusmäärittelydokumentaatiossa. Kaikki käyttäjäryhmät tekevät vaatimusmäärittelyssä kuvattuja toimintoja selaimen avulla. Nämä toiminnot lähtevät kutsuina Apache HTTP-palvelimelle. Sovelluspalvelin sisältää kaiken lupajärjestelmän toiminnallisuuden ja käyttää tietokannanhallintajärjestelmää tietojen hakuun ja tallentamiseen.



Kuva 3. FMAS-lupapalvelun ylätason tekninen toteutuskuva.

Lupajärjestelmä toteutetaan LAMP–teknologioiden avulla. Akronyymi tarkoittaa seuraavia toteutustekniikoita:

- Linux-käyttöjärjestelmä
- Apache HTTP-palvelin
- MySQL –tietokannanhallintajärjestelmä
- PHP-ohjelmointikieli

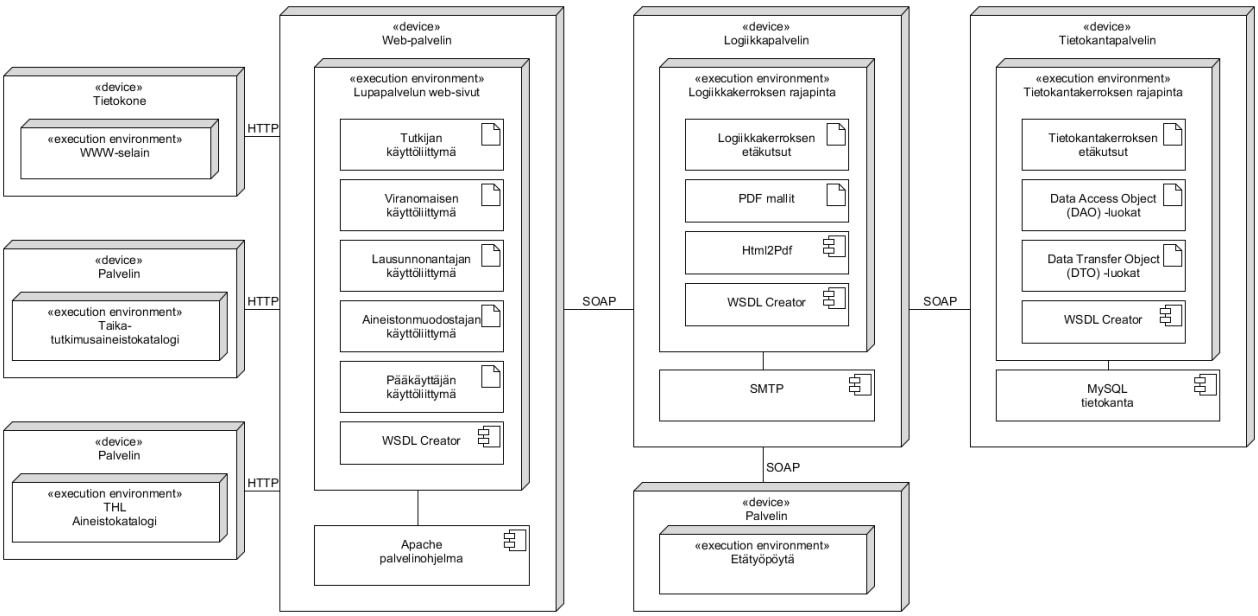
Kehitystyössä käytetään Eclipse for PHP Developers –kehitysympäristöä. Liikenne verkon yli salataan SSL (Secure Sockets Layer) –tietoturvaprotokollan avulla.

4.2 Ohjelmiston hajautus ja kerrosrakenne

Lupapalvelun toteutuksessa noudatetaan kolmikerrosarkkitehtuuria (Kuva 4), jossa ohjelmisto (lupapalvelu) on jaettu kolmeen eri kerrokseen, jotka voidaan myös hajauttaa konkreettisesti eri palvelimille. Ohjelmiston muodostavat kerrokset ovat käyttöliittymäkerros (presentation layer), logiikkakerros ((business) logic layer) ja tietokantakerros (data access layer). Kerrosten välinen kommunikaatio (tiedonsiirto) hoidetaan XML-pohjaista SOAP-protokollaa (Simple Object Access Protocol) käyttäen. Sen lisäksi, että ohjelmisto voidaan tällöin hajauttaa useammalle palvelimelle, voidaan periaatteessa mikä tahansa kerros vaihtaa jollain muulla ohjelmointikielellä toteutettuun, vastaavan toiminnallisuuden tarjoavaan kerrokseen. Hajautus useammalle palvelimelle taas mahdollistaa pääsyoikeuksien (protokolla- ja porttikohtaisten) säätämisen ohjelmistokerroksittain ja lisää täten myös tietoturvaa.

Kolmikerrosarkkitehtuurissa esim. tietokantakerros on ainut, joka tietää itse tietokannasta mitään. Muut kerrokset näkevät vain itse datan tietämättä, miten se on fyysisesti talletettu. Vastaavasti käyttöliittymäkerros on ainut, joka tuottaa näkymiä käyttöliittymään ja käsittelee käyttäjältä tulevat vastaukset (esim. klikkaukset).

Kolmikerrosarkkitehtuuri ei ole sama kuin Model-View-Controller[5] (MVC) –suunnittelumalli. MVC-suunnittelumallissa malli (Model) huolehtii sekä tiedon talletuksesta että sovelluslogiikasta, näkymä (View) taas huolehtii käyttäjälle näkyvän käyttöliittymän tuottamisesta ja päivittämisestä ja käsittelijä (Controller) vastaanottaa käyttäjältä käyttöliittymän kautta tulevan syötteen ja välittää sen mallille tai näkymälle. Täten siis MVC-suunnittelumallin malli (Model) vastaa kolmikerrosarkkitehtuurin logiikka- ja tietokantakerroksia ja näkymä (View) ja käsittelijä (Controller) sisältyvät kolmikerrosarkkitehtuurin käyttöliittymäkerrokseen.



Kuva 4. Lupajärjestelmän arkkitehtuuri

Sovelluksen rakenne koostuu esitys, sovellus- ja tiedonsaantikerroksista (Kuva 4).

1. Tietokantakerros

   Tietokannan lisäys, päivitys ja hakuoperaatiot suoritetaan tiedonsaantikerroksen ohjelmointirajapinnan kautta. Rajapintaan yhdistetään käyttämällä SOAP tietoliikenneprotokollaa. SOAP palvelin sekä tiedonsaantikerroksen ohjelmointirajapinta sijaitsevat tiedostossa fmas_db_api.php. Rajapinnan apufunktiot ovat tiedostossa helper_functions.php.

[5] Tony Marston, ”What is the 3-Tier Architecture?”, 14.10.2012. http://www.tonymarston.co.uk/php-mysql/3-tier-architecture.html

2. Logiikkakerros

   Logiikkakerroksen tehtävänä on koordinoida sovellusta, käsitellä komentoja, siirtää, luokitella sekä prosessoida dataa käyttöliittymän ja tiedonsaantikerroksen välillä. Tiedosto fmas_business_logic.php toimii SOAP – asiakkaana sekä palvelimena. Rajapinnan apufunktiot ovat tiedostossa helper_functions.php.

3. Käyttöliittymäkerros

   Tutkijan, viranomaisen, lausunnonantajan, aineistonmuodostajan sekä pääkäyttäjän käyttöliittymiä yhdistävät tiedosto nimeltä fmas_ui.php. Kyseinen tiedosto on käyttöliittymälogiikan keskus, johon on koottu PHP-apufunktiot sekä liitetty SOAP-liitäntä sovelluskerrokseen, kieliasetukset sekä käännökset. Käyttöliittymillä on myös yhteiset header- ja footer tiedostot. Header-tiedostossa määritetään web-sivun yläosion HTML-rakenne, ladattavat JavaScript- ja CSS- tiedostot sekä käyttöliittymäkohtaiset valikot.

4.3 Järjestelmän sisäiset rajapinnat

Keskeiset järjestelmän sisäiset rajapinnat ovat ohjelmiston eri kerrosten (käyttöliittymä, logiikka, tietokanta) väliset rajapinnat. Kommunikaatio kerrosten välillä tapahtuu SOAP-protokollaa käyttäen. PHP-ohjelmointikielessä SOAP-protokollaa varten hyödynnetään WSDL creator -kirjastoa[6]. Ko. kirjasto huolehtii automaattisesti rajapintafunktioiden WSDL-kuvausten tuottamisesta, kunhan rajapintafunktioiden yläpuolelle on PHP-koodissa laitettu tarpeelliset wsdl creator –kirjaston tunnistamat direktiivit, joissa kerrotaan mm. syötteen ja vastauksen tietotyypit. Esimerkiksi logiikkakerroksen tiedostossa fmas_business_logic.php on luokalla fmas_business_logic metodi rekisteroi_kayttaja, jonka yläpuolella on PHP:n kommenteissa seuraavat direktiivit:

```
/**
 * @WebMethod
 * @desc Kuvaus rajapinnan metodista
 * @param string[] $syoteparametrit
 * @return string[] $dto
 */
```

Lupajärjestelmän rajapinnat toteutetaan web-palveluina, joilla on WSDL -rajapintakuvaus. Järjestelmien välinen kommunikointi toteutetaan SOAP–protokollaa käyttäen (tai vaihtoehtoisesti: REST).

4.4 Järjestelmän ulkoiset rajapinnat

Lupapalvelusta siirretään tietoa seuraaviin järjestelmiin / lupapalvelu vastaanottaa tietoa seuraavista järjestelmistä:

[6] https://github.com/piotrooo/wsdl-creator

- Informaatio- ja tukipalvelu (tulossa)
- Metatietokatalogi (TAIKA)
- THL Aineistokatalogi
- Suomi.fi -tunnistautuminen ja asiointivaltuudet (tulossa)
- Etätyöpöytä (tulossa)

4.4.1 Rajapinnat aineistokatalogeihin

Lupapalvelun tietokannassa säilytetään tietoja Tilastokeskuksen Taika-aineistokatalogin ja Terveyden ja hyvinvoinnin laitoksen aineistokatalogin (tilasto)aineistoista sekä niiden muuttujista. Säilytettäviä tietoja ovat (tilasto)aineiston nimi, viiteajankohta ja kuvaus sekä muuttujan mittayksikkö, luokitus, tunnus ja aineistoon/muuttujaan liittyvä tunniste. Tiedot (tilasto)aineistoista ja muuttujista lisätään tai päivitetään lupapalvelun tietokantaan skriptillä, jonka palvelin suorittaa esimerkiksi muutaman kerran vuorokaudessa Unix-pohjaisella cron ajastuspalvelulla. Tiedonharavoinnin suorittava skripti on UI-kerroksen tiedostossa metadata_rajapinta.php. Skriptin voi ajaa vain komentoliittymällä.

Rajapintayhteys Taika-aineistokatalogiin on PHP:n curlilla (Client URL Library) toteutettu GET-pyyntö osoitteisiin https://taika.stat.fi/api/$lang /datasets ja https://taika.stat.fi/api/$lang/variables/$X, missä

$X = aineiston tunniste ja $lang = kielikoodi.

Vastaavasti etäkutsu THL:n aineistokatalogiin lähetetään GET-pyynnöllä osoitteisiin https://aineistoeditori.fi/api/v1/public/studies ja https://aineistoeditori.fi/api/v1/public/studies/$Y/datasets/$X, missä

$X = aineiston ID ja $Y = study ID.

Vastaus (aineistot, muuttujat ja niiden metatiedot) saadaan XML-muodossa, josta poimitut tiedot päivitetään lupapalvelun tietokantaan.

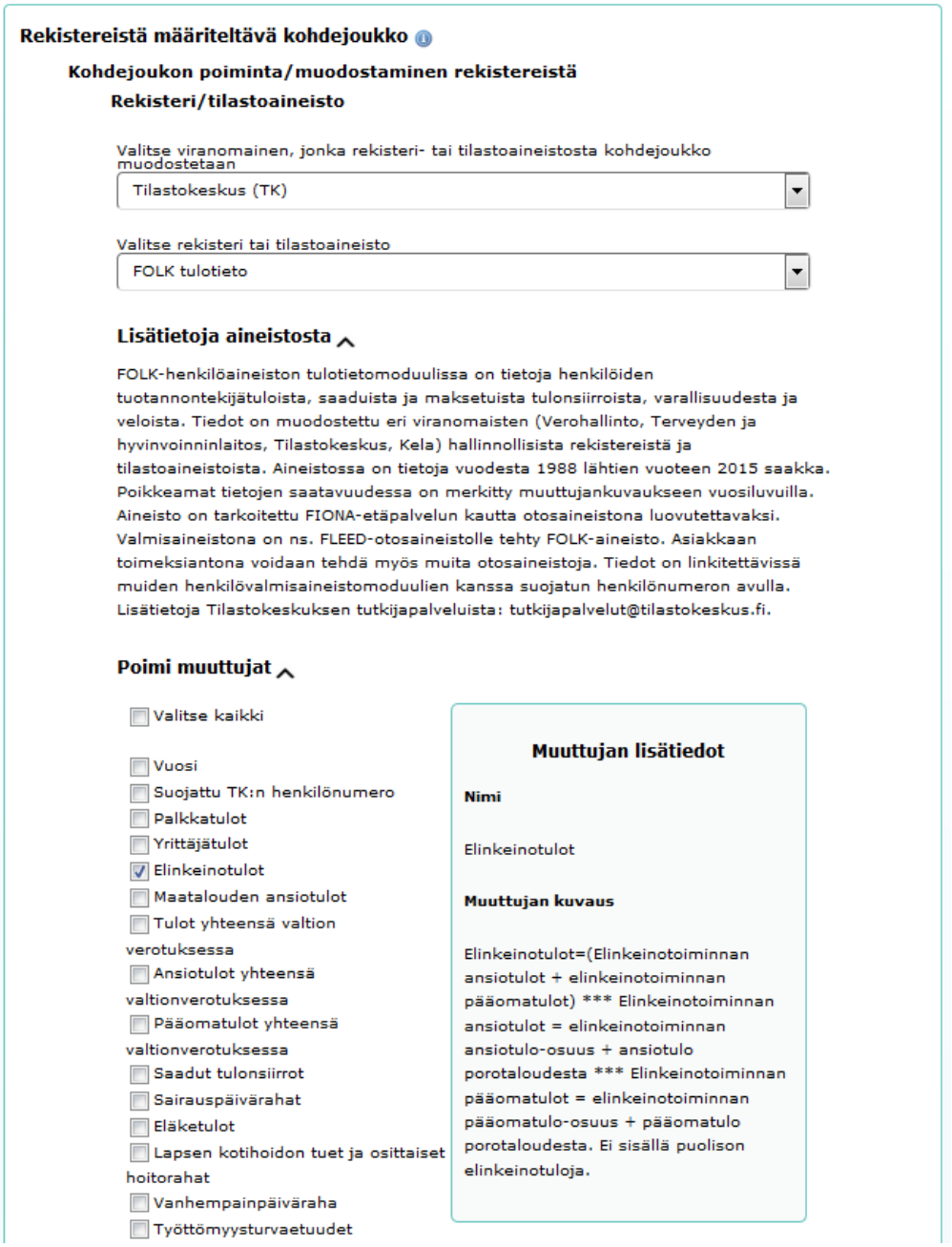
```
Haettu aineistosta UTH-aineisto 2014 (terveys ja hyvinvointi, THL) 191 kpl muuttujia.
Haettu aineistosta FOLK tutkinto  22 kpl muuttujia.
Haettu aineistosta FOLK  perhe  15 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), otos, työnhakujaksot (YA244) 6 kpl muuttujia.
Haettu aineistosta UTH-aineisto 2014 (ATH-verrokin a-osio) 17 kpl muuttujia.
Haettu aineistosta FOLK perustieto 57 kpl muuttujia.
Haettu aineistosta Yritysrekisterin tilastotiedostot: toimipaikat 2012 (YA222) 46 kpl muuttujia.
Haettu aineistosta Yritystietovarasto: toimipaikat 2015 (YA222) 29 kpl muuttujia.
Haettu aineistosta FOLK työssäkäynti  45 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), kokonaisaineisto, työsuhdejaksot 22 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), kokonaisaineisto, henkilöt 165 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), otos, työsuhdejaksot (YA244) 22 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), otos, sijoitusjaksot (YA244) 7 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), otos, henkilöt (YA244) 165 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED),  työnhakujaksot (kokonaisaineisto) 6 kpl muuttujia.
Haettu aineistosta Yhdistetty työntekijä-työnantaja-aineisto (FLEED), sijoitusjaksot (kokonaisaineisto) 7 kpl muuttujia.
Haettu aineistosta Työ- ja elinkeinomineisteriön työvälitysaineisto : Työnhaku 108 kpl muuttujia.
Haettu aineistosta Työ ja elinkeinoministeriön työnvälitystilaston aineisto: Työpaikka 29 kpl muuttujia.
Haettu aineistosta Työ ja elinkeinoministeriön työnvälitystilaston aineisto: Työnhakija 26 kpl muuttujia.
Haettu aineistosta Työ ja elinkeinoministeriön työnvälitystilaston aineisto: Työkunto 12 kpl muuttujia.
Haettu aineistosta Yritystietovarasto: toimipaikat 2016 (YA222) 29 kpl muuttujia.

Metatietojen päivitys

Luvan kohteita lisätty: 24
Luvan kohteita päivitetty: 72
Muuttujia lisätty: 773
Muuttujia päivitetty: 4432
```

Kuva 5, Tiedonharavointi aineistokatalogeista

Haetut tiedot näkyvät käyttäjälle muuttujalistauksena siten, että kustakin muuttujasta saa näkyviin muuttujan nimen lisäksi muuttujan metatiedot. Käyttäjä voi poimia lupahakemukseen listasta haluamansa muuttujat.



Kuva 6. Taika-tutkimusaineistokatalogin muuttujien ja muuttujien kuvausten näkyminen lupapalvelussa.

5 Sovelluskerrokset

5.1 Tietokantakerros

5.1.1 Keskeisin toiminnallisuus

Tietokantakerros huolehtii tiedon tallettamisesta tietokantaan ja tiedon hakemisesta tietokannasta logiikkakerroksesta tulevien pyyntöjen mukaisesti. Tietokantaan liittyviä tietoja ei ole lähdekoodissa muualla kuin itse tietokantakerroksen lähdekoodissa ja kaikki SQL-kieliset lauseet on eristetty tietokantakerroksen DAO-luokan metodeihin.

5.1.2 Tiedostot

Tietokantakerrokseen kuuluu seuraavia hakemistoja ja tiedostoja:

- dto:
  - *DTO.php:
    - Data Transfer Object (DTO) –luokat PHP-tiedostoissa. Näitä luokkia käytetään kaikissa sovelluskerroksissa tiedonsiirtoon.
- vendor:
  - WSDL Creator -kirjasto
- _config.php
- helper_functions.php
- fmas_db_api.php
- *DAO.php:
  - Data Access Object (DAO) –luokat PHP-tiedostoissa.

5.1.3 Tietokannan nimeämiskäytännöt

Lupajärjestelmän tietokantaelementtien nimeämisessä (esimerkiksi taulujen nimet) käytetään suomen kieltä.

Taulukko 1. Tietokannan elementtien nimeäminen.

Tietokannan elementti	**Kuvaus**
indeksi	IX_Taulunimi_Kenttänimi.
kenttä	Kuvattavan asian nimi, esimerkiksi ”Sukunimi”
pääavain	Pääavainrajoittimet nimetään PK_Taulunimi. Esimerkiksi: PK_Hakemus
taulu	Kuvattavan asian nimi yksikössä, esimerkiksi ”Hakemus”. Jos taulun nimessä on tarpeen yhdistää eri sanoja, erotetaan taulut alaviivalla. Esimerkiksi: Hakemus_Tila.
viiteavain	Viiteavainrajoittimet nimetään FK_lapsitaulu_vanhempitaulu. Esimerkiksi: FK_hakemustila_tila.

Muita tietokannan suunnittelussa huomioon otettavia asioita:

- pääavainkentät nimetään aina ”ID”
- lähes jokaisesta taulusta löytyvät seuraavat kentät: Lisaaja, Lisayspvm, Muokkaaja, Muokkauspvm. Näihin kenttiin tallennetaan tietoa tiedon alkuperäisestä lisääjästä (insert käyttäjätunnuksella x), lisäyksen ajanhetkestä sekä vastaavasti viimeisimmän päivätyksen tekijästä (update käyttäjätunnuksella x) sekä päivitysajankohta.
- true/false-tyyppiset kentät ovat bit-tyyppisiä siten, että true=1 ja false=0.
- Pienet, luettelotyypiset listat kuvataan tietokannan Koodistot-taulussa. Jokaisesta koodista on taulussa sekä koodi (esim. ”HYV”) että selite (esim. ”hyväksytty”).

Koodistoarvoista tallennetaan niitä käyttäviin tauluihin koodi ilman viite-eheyksiä Koodistot-tauluun.
- Taulujen, kenttien, rajoittimien ja indeksien nimien maksimipituus on 64 merkkiä.

5.1.4 Tietomalli ja tietokanta

Lupajärjestelmän määrittelytason tietomalli kuvataan MS Visio 14-versiolla. Suunnittelutason tietomalli (tietokanta) kuvataan ER-kaaviona Visual Paradigm –työvälineen avulla.

Lupajärjestelmä rakennetaan käyttämään MySQL Community Edition -tietokannanhallintajärjestelmää.

Tietokannan suunnittelussa tulee ottaa huomioon vaatimus SÄHKE2-määräyksen toteuttamisesta. Tietokannan sisältöön (ei siis esim. kenttänimiin) tulee tuottaa määräyksen mukaiset kentät.

Tietokannan ER-kaavio (reverse engineer –menetelmällä tietokannasta saatu) löytyy erillisestä tiedostosta.

Osa tietokannan tauluista on toistaiseksi käyttämättä tai täysin hyödyntämättä, mm. Paatetty_aineisto-taulu (jossa FK_Paatos).

Tietokannan sisältöä tyhjennettäessä kannattaa jättää tiettyjen taulujen sisällöt tyhjentämättä:

- Asiakirjahallinta_liite: sisältää liitetyyppien metatiedot
- Asiakirjahallinta_saanto: sisältää tiedot liitetyyppien riippuvuussäännöistä
- Jarjestelman_hakijan_roolit: sisältää tiedot tutkimusryhmän rooleista
- Kayttaja-taulu: sisältää lupapalvelun käyttäjien tiedot
- Koodistot-taulu: sisältää viranomaisten koodeja vastaavat nimet
- Lomake: sisältää tiedot lupajärjestelmän lomakkeista
- Lomake_hakemus: sisältää hakemuslomakkeiden lisätiedot
- Lomake_paatos: sisältää päätöslomakkeiden lisätiedot
- Lomakkeen_sivut: sisältää tiedot lomakkeen sivuista
- Luvan_kohde-taulu: sisältää viranomaisten rekisterien nimet
- Muuttuja –taulu: sisältää tiedot aineistokatalogien muuttujista
- Osio-taulu: sisältää tiedot lomakkeiden elementeistä
- Osio_saanto –taulu: sisältää tiedot lomakkeiden elementtien riippuvuussäännöistä
- Osio_lause –taulu: sisältää tiedot osioiden riippuvuussääntöjen ehtolauseista
- Paakayttajan_rooli: sisältää tiedot lupajärjestelmän pääkäyttäjistä
- Viranomaisen_rooli-taulu: sisältää tiedot lupapalvelun viranomaisten käyttäjärooleista

Liitetiedostot on talletettu myös tietokantaan eikä tiedostoiksi web-palvelimelle. Tämä on tehty tietoturva- ja tietosuojasyistä. Liitetiedostot ovat blob-objekteina MySQL-tietokannassa.

Muutamien tietokantataulujen merkitys on selitetty alla:

- **Hakemusversio**: sisältää mm. tutkimuksen nimen, hakemuksen tyypin ja version tiedot.
- **Hakemus:** sisältää viranomaisen koodin.
- **Haettu_aineisto**: sisältää aineistonmäärittelytiedot mm. tiedon tunnistetaanko kohdejoukko / tapaukset / verrokit / viitehenkilöt aikaisemmasta. Haettu_aineisto-taulu liittyy Hakemusversio-tauluun.
- **Viestit:** sisältää lupapalvelun käyttäjien palvelussa lähettämät viestit ja viestien linkityksen keskusteluketjuiksi (viesti – vastaus viestiin). Viestit-taulu liittyy Hakemus-tauluun ja Kayttaja-tauluun (kahdesti: sekä viestin lähettäjän että vastaanottajan osalta).
- **Lausuntopyynto**: sisältää mm. lausuntopyyntötekstin ja lausunnon määräpäivän. Lausuntopyynto-taulu liittyy Tutkimus-tauluun sekä Kayttaja-tauluun (kahdesti: sekä lausunnon pyytäjän että lausunnon antajan osalta).
- **Lausunto**: sisältää mm. vastauksen (teksti) lausuntopyyntöön. Lausunto-taulu liittyy Lausuntopyynto-tauluun.

5.1.5 Tietokantarajapinta

PHP-koodista tehdään SQL-kyselyt MySQL-tietokantaan käyttäen PHP:n mukana tulevaa PDO-rajapintaa (PHP Data Objects). SQL-kysely määritellään prepare-funktiolla, jolle annetaan parametrina mm. SQL-kyselyn sisältävä merkkijono, jossa ei tarvitse vielä olla sijoitettuna kyselyssä käytettäviä muuttujien arvoja, vaan niiden tilalla voi olla :var –tyyliset merkkijonot; ko. merkkijonot korvautuvat sitten muuttujien arvoilla SQL-lauseessa, kun se ajetaan execute-funktiolla.

Tietokantarajapinnassa on käytetty Data Access Object (DAO) –suunnittelumallia. Kaikki tietokannan taulujen käsittelyyn liittyvä SQL-koodi löytyy DAO-luokista (niiden metodeista). Yleensä yksi DAO vastaa yhtä tietokannan taulua ja hoitaa ko. tauluun liittyvän tiedon talletuksen, päivityksen, hakemisen ja poistamisen. DAO-luokan vastaavat funktiot alkavat merkkijonoilla: luo_, paivita_, hae_ ja poista_. Esim. Hakija-taulua vastaava DAO on HakijaDAO. Arvojen välittämiseksi DAO-luokille tai DAO-luokista käytetään Data Transfer Object (DTO) –suunnittelumallia DTO-luokat sisältävät vain attribuutteja, joihin tietokantaan talletettavat / tietokannasta haetut arvot sijoitetaan. Tietokannasta tietoa hakevat DAO-luokkien metodit palauttavat (PHP:n return-lauseella) tyypillisesti DTO-luokan objektin. DTO-luokan objektit sisältävät vain tietoa eivätkä lainkaan metodeita ja lisäksi ne eivät tiedä enää mitään fyysisestä tietokantaratkaisusta. DTO-luokan kaikki attribuutit ovat julkisia (public) eli niitä voi käsitellä (hakea arvon tai muuttaa arvoa) suoraan DTO-luokan olion ulkopuolelta. Tämä on tehty yksinkertaisuuden ja tehokkuuden vuoksi eikä erillisiä getter- ja setter-metodeita ole haluttu lähteä määrittelemään jokaiselle attribuutille (jotka sitten määriteltäisiin yksityisiksi eli private).

Tietokantayhteyden avaaminen ja sulkeminen hoidetaan fmas_db_api.php-tiedoston fmas_db_api-luokan metodeissa _connectToDb ja _disconnectFromDb. Myös tietokantatransaktiot aloitetaan (PDO:n beginTransaction-funktio) ja lopetetaan (PDO:n

commit-funktio) fmas_db_api-luokan metodeissa. Varsinaisia SQL-lauseita fmas_db_api-luokka ei kuitenkaan sisällä.

5.1.6 Salasanat ja käyttäjätunnukset tietokannassa

Käyttäjien käyttäjätunnukset ja salasanojen tiivisteet tallennetaan lupajärjestelmän tietokantaan. Käyttäjätunnus on järjestelmän käyttäjän sähköpostiosoite.

Käyttäjätunnukset tallennetaan tekstimuodossa tietokantaan. Käyttäjätunnukseen liittyvä salasana tallennetaan tietokantaan salasanasta laskettuna tiivisteenä. Tiivistefunktiona käytetään md5 (cryptographic hash) -funktiota. Ennen tiivisteen laskemista salasanaan liitetään ”suola”, joka pidentää salasanan ja vaikeuttaa siten rainbow-taulujen käytön tietoturvahyökkäystilanteessa.

5.1.7 Muut tietokantakonfiguraatiot

Seuraavat konfiguraatiot tehdään tietokantaan:

- Taulut luodaan transaktio-turvallisiksi käyttämällä InnoDB-engineä MyISAM-moottorin sijaan (create table(…)engine=InnoDB default charset=UTF8;)

5.1.8 Muita tietokannan käyttöön liittyviä huomioita

Tietokantaoperaatiot hoidetaan ACID-doktriinia noudattamalla (atomicity, consistency, isolation and durability).

5.2 Logiikkakerros

5.2.1 Keskeisin toiminnallisuus

Logiikkakerros huolehtii käyttöliittymäkerroksen ja tietokantakerroksen välisten pyyntöjen ja vastausten välittämisestä. Logiikkakerros ei tiedä mitään käyttöliittymäelementeistä eikä tietokannasta. Logiikkakerroksen toiminnallisuuksiin kuuluvat mm:

- Hakemusten, lausuntojen ja päätösten generointi PDF-muotoon
- Sähköposti-ilmoitukset
- Lajittelualgoritmit
- Käyttöoikeuksiin liittyvät tarkistukset
- Hakemuksen puuttuvien tietojen tarkistus
- Datan käsittely (esimerkiksi puu-tietorakenteen muuttaminen tauluksi)
- Sääntömoottori: lomakkeen tilatietojen päivittäminen riippuvuussääntöjen perusteella
- Koontitoiminto, joka etsii eroavaisuudet hakemuksen versioiden välillä (kesken)

5.2.2 Tiedostot

Logiikkakerrokseen kuuluu mm. seuraavia hakemistoja ja tiedostoja:

- templates:
  - *_html2pdf.html:
    - PDF pohjien rakenteet
  - html2pdf.css:

        - PDF pohjien tyylimäärittelyt
- vendor:
    - wsdl creator –kirjasto
    - tcpdf -kirjasto
- _config.php
- helper_functions.php
- fmas_business_logic.php

5.3 Käyttöliittymäkerros

5.3.1 Keskeisin toiminnallisuus

Käyttöliittymäkerros huolehtii lupapalvelun käyttäjälle selaimessa näkyvien näkymien tuottamisesta sekä käyttäjän käyttöliittymässä tekemien toimintojen (mm. klikkaamiset) käsittelystä. Tarvittaessa käyttöliittymäkerros lähettää käyttäjän (esim. lomakkeisiin) syöttämää tietoa logiikkakerroksen kautta tietokantakerrokselle tai hakee käyttäjälle näytettävää tietoa logiikkakerroksen kautta tietokantakerroksesta.

5.3.2 Tiedostot

Käyttöliittymäkerrokseen kuuluu mm. seuraavia hakemistoja ja tiedostoja:

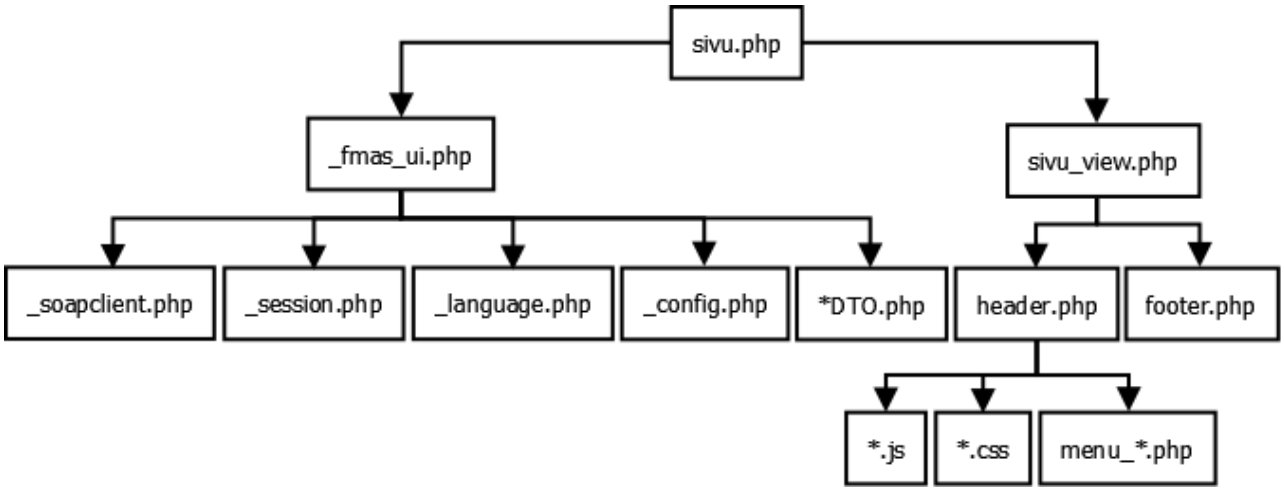
- static:
    - css:
        - jquery-ui.css
        - pikaday.css
        - tyylit.css: Tyylimäärittelyt lupapalvelun käyttöliittymäelementeille
    - images:
        - Käyttöliittymän ikonit (mm. checkmark)
        - Käyttöliittymässä näkyvät Kansallisarkiston logot
    - js:
        - JQuery-JavaScript-kirjasto
        - kalenterikomponentti
        - fmas_front_end.js
        - yleiset.js
- ui:
    - language:
        - lang_*.php, lang_*_arrays.php: Käyttöliittymän tekstit tietyllä kielellä
    - template:
        - footer.php
        - header.php
        - menu_*.php
        - vasen_menu.php
    - views:

        - *_view.php: Tietyn näkymän HTML-sivun luova PHP-koodi. Itse käyttöliittymässä näytettävän tiedon haku tapahtuu pääkansiossa olevassa tiedostossa, jonka nimi on muuten sama, mutta siitä puuttuu _view.
- vendor:
    - wsdl creator -kirjasto
- _fmas_ui.php: käyttöliittymän apufunktiot
- _soap_config.php
- *.php (muut)

5.3.3 Toteutusratkaisut

Käyttöliittymän rakenne

Web-sivut generoidaan PHP- ohjelmointikielellä käyttöliittymäkerroksen palvelimella. Käyttäjän avaama sivu koostuu useasta eri osasta (Kuva 7).



Kuva 7. Käyttöliittymän rakenne

sivu.php

Käyttäjän avaama web-sivu. Alussa sivulle liitetään apufunktiot/asetukset tiedostosta _fmas_ui.php ja tarkistetaan mm. $_POST / $_GET parametrit. Luodaan SOAP-yhteys, jos logiikkakerroksesta haetaan tai logiikkakerrokseen viedään dataa. Lopuksi liitetään tiedosto sivu_view.php, joka sisältää varsinaisen HTML- sisällön.

_fmas_ui.php

Tiedosto, josta haetaan käyttöliittymäkerroksen asetukset, apufunktiot ja alustetaan tai päätetään käyttäjän istunto.

sivu_view.php

Tiedosto määrittää sivun HTML-struktuurin ja sisällön. Sivu_view.php koostuu header templatesta (header.php), sivun sisällöstä sekä footer templatesta (footer.php).

header.php

Header template, eli sivun ”yläosa”, joka vastaa pääosin HTML-rakenteen <head> elementin sisällöstä, sekä <body> tagin sisällä olevan ”yla_sisalto” –div elementin luokan sisällöstä (johon kuuluu mm. kieli- ja roolivalikko sekä logo). Head- tagien sisällä muutetaan tarvittaessa PHP:n muuttujia JavaScriptin muuttujiksi sekä liitetään tyylitiedostot (mm. tyylit.css) ja JavaScript tiedostot (esim. fmas_front_end.js). Headeriin sisällytettävä valikko määräytyy käyttäjän roolin perusteella. Valikot ovat tiedostoissa: menu_aineistonmuodostaja.php, menu_hakija.php, menu_lausunnonantaja.php, menu_paakayttaja.php, menu_viranomainen.php.

footer.php

Footer template on sivun loppuosassa sijaitseva linkkejä sisältävä elementti.

Käyttöliittymän toiminnallisuudet (DOM)

Sivujen dynaamiset toiminnot toteutetaan JavaScript komentosarjakielellä sekä AJAX-tekniikoilla. Näihin toiminnallisuuksiin lukeutuvat mm:

- Lomakkeiden elementtien välisten riippuvuuksien määrittely (piilottaminen / näyttäminen)
- TAIKA-metatietojen haku
- Lomakkeiden tietojen automaattinen tallennus
- Elementtien lisäys / poisto
- Ilmoitukset (varoitus, jos sivulta poistutaan tallentamatta muutettuja tietoja)

Toiminnallisuuksien toteutuksessa otetaan esimerkkiä Unobtrusive JavaScript-suunnittelumallista, jonka pääpiirteenä on eristää toiminnallisuus web-sivun struktuurista/sisällöstä. Toisin sanoen, JavaScript koodit ovat erillisissä *.js –tiedostoissa ja web-sivujen sisältöön ei sotketa JavaScriptin funktionaalisuutta.

Esimerkiksi jos event handler sijoitettaisiin HTML-kielen sekaan:

<input type="text" name="date" onchange="validateDate()" />

tulisi kehityksestä vaikeammin ylläpidettävää. HTML:n tehtävä on kuvata dokumentin struktuuria eikä sen ohjelmallisia toimintoja.

Alla oleva esimerkki havainnollistaa kuinka JavaScript eristetään HTML-kielestä:

HTML:

```
<input type="text" name="date" id="date" />
```

JavaScript:

```
document.getElementById('date').onchange = validateDate;
```

Käyttöliittymän tyylit

Käyttöliittymän tyylit (esim. fonttikoot) kuvataan tyylitiedoston (*.css) avulla. Tyylitiedoston avulla tehdään myös joitain toiminnallisuuksia esim. kun käyttäjä vie hiirikursorin ”Uusi hakemus” –napin päälle, vaihdetaan napin taustaväri ja hiirikursori nuolesta kädeksi, jotta käyttäjä tietää, että ko. nappia voi nyt klikata. Tämä hoidetaan CSS-tiedoston input.nappi:hover ja input[type=”button”] –määrittelyillä. Ihan vastaavalla tavalla tab-osion linkkien (esim. ”Etusivu”, ”Saapuneet viestit”) tekstin väri vaihdetaan, kun hiirikursori on linkin päällä. Se tapahtuu mm. ul.tabnav li a:hover –määrittelyn avulla.

Tyylitiedostoissa määritellään joitain keskeisiä, toistuvasti käyttöliittymässä käytettyjä elementtejä. Yksi niistä on esim. div.laatikko, jota käytetään mm. tutkimuksen perustiedoissa erottelemaan eri osiot (”1. Tutkimuksen perustiedot”, ”2. Julkaisusuunnitelma”, jne.) ja ko. laatikolle on määritelty CSS-tiedostossa pyöristetyt kulmat ja sininen reunaviiva.

6 Toiminnallisuudet

6.1 Istunnon (session) hallinta

Käyttäjän istuntoon liittyviä tietoja säilytetään PHP:n session-muuttujissa. Istunnon käynnistämisen ja vanhenemisen määrittelyt löytyvät käyttöliittymäkerroksen tiedostosta _session.php. Kirjautuneen käyttäjän istunto päättyy, mikäli käyttäjä on ollut epäaktiivisena viimeiset 30 minuuttia.

Sisäänkirjautumisen yhteydessä käyttäjälle luodaan satunnaismerkeistä koostuva token, jota säilytetään session-muuttujana (käyttöliittymä) sekä tietokannan Suojaus-taulussa. Käyttäjän istuntoon sidottu token on autentikointiin vaadittava parametri tietokantakerroksen funktioissa.

Tietoturvasyistä session-muuttujiin kirjoitetaan ja poistetaan niiden lukitus aina ennen etäkutsua logiikkapalvelimelle.

6.2 Samanaikaisuuden hallinta sovelluksessa

Samanaikaisuudella tarkoitetaan tässä tilannetta, jossa monta käyttäjää yrittää päivittää samoja tietoja samanaikaisesti. Edellä mainitun kaltainen tilanne hoidetaan siten, että tietojen muokkaaminen mahdollistetaan vain yhdelle käyttäjälle kerrallaan. Esimerkiksi jos käyttäjä A avaa hakemuksen, tästä seuraa että hakemus lukitaan muokkaamisen ajaksi, jolloin käyttäjä B voi tarkastella samaa hakemusta vain luku- tilassa. Lukitus poistuu, kun käyttäjä A lopettaa hakemuksen tarkastelun/muokkauksen. Jos käyttäjä A jättää selaimen auki ja unohtaa siirtyä pois hakemuksesta tai kirjautua ulos, niin lukitus poistuu automaattisesti lukon vanhetessa (30 min).

Tiivistettynä lukitseminen toteutuu seuraavanlaisesti (esimerkkinä hakemuksen lukitus):

- Hakemusversio- tietokantataulun Muokkaaja- kenttään lisätään tai päivitetään tieto siitä, kuka taulun on lukinnut (FK_Kayttaja).
- Tietokantataulun Muokkauspvm –kenttään lisätään tai päivitetään lukitsemisen aika/päivämäärä.
- Tietokantakerroksen tiedoston _config.php globaali muuttuja HAKEMUSVERSIO_LOCK_TIME määrittää ajan, kuinka pitkään lukitus kestää.
- Hakemusta voi muokata jos Muokkaaja- kenttä on tyhjä, tai jos Muokkaaja on sama kuin kirjautunut käyttäjä, tai jos lukitsemisaika on umpeutunut. Muussa tapauksessa hakemus näytetään vain luku- tilassa.
- Kun lukitsija siirtyy pois hakemuksesta tai kirjautuu ulos, Hakemusversio- taulun Muokkaaja ja Muokkauspvm kentät tyhjennetään. Tällöin muiden ei tarvitse jäädä odottamaan lukon päättymistä (aika, joka määritetään edellä mainittuun globaaliin muuttujaan).

6.3 Lokitus

Lupapalvelussa tallennetaan lokitietoa tietokantaan Kayttoloki-tauluun. Seuraavat tapahtumat tallentuvat tietokantaan aikaleimojen kera:

- sisäänkirjautuminen ja uloskirjautuminen (sekä viittaus käyttäjään: FK_Kayttaja)
- hakemuksen poisto (sekä viittaus käyttäjään ja hakemusversioon: FK_Kayttaja, FK_Hakemusversio)
- liitteen avaaminen (sekä viittaus käyttäjään, hakemusversioon ja liitteeseen: FK_Kayttaja, FK_Hakemusversio, FK_Liitteet)

Lisäksi useimmat tietokantataulut sisältävät tiedot lisäyksen, muokkauksen ja poistamisen ajankohdasta sekä viittaukset lisääjään, muokkaajaan ja poistajaan. Esimerkiksi Osio_sisalto-taulusta selviää lomakkeiden tallennushistoria.

6.4 Raportit

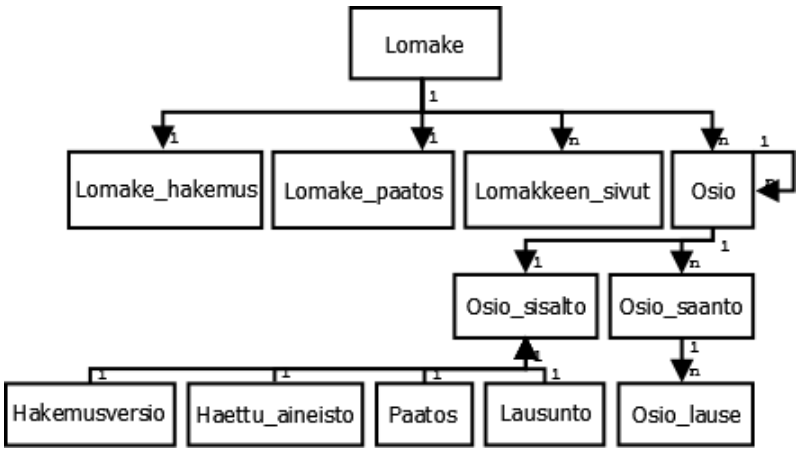
Järjestelmään ei suunnitella erillistä raportit-osioita, mutta sen sijaan jatkokehityksessä tehdään kaikille kiinnostuneille avoin käyttöliittymä tietojen hakuun järjestelmän tietokannasta. Käytännössä tietoja voidaan tulostaa vain tiedoista, jotka eivät riko yksityisyyden suojaa, vaan sen sijaan voidaan tulostaa esimerkiksi tieto hakemuksista tieteenalakohtaisesti. Tarkat hakukriteerit kuvataan järjestelmän Käyttötapaukset-dokumentissa.

7 Konfiguroitavuus

7.1 Lomakkeet

Lupapalvelussa lomakkeiden sisältöjä ja niiden välisiä riippuvuuksia on mahdollista konfiguroida (lisäillä/muokata/poistaa), jotta niitä ei tarvitsisi kirjoittaa suoraan PHP- ja JavaScript ohjelmakoodiin. Näin ollen erilaisia lomakkeita voi suunnitella ja toteuttaa lupapalveluun pääkäyttäjän lomake-editorilla (tai pelkästään tietokantakyselyillä). Lomakkeiden dynaamisella muokattavuudella pyritään parantamaan ohjelmiston ylläpidettävyyttä ja laajennettavuutta.

Lupajärjestelmässä voi konfiguroida hakemus, päätös ja lausuntolomakkeita. Lomake määritellään tietokantaan (Kuva 8, Lomakkeen relaatiomalli).



Kuva 8, Lomakkeen relaatiomalli

Lomakkeelle asetetaan nimi, asiakirjan metatiedot ja lomakkeen tyyppi (Lomake-taulu). Hakemus ja päätöslomakkeeseen liittyvät lisämäärittelyt löytyvät tauluista Lomake_hakemus ja Lomake_paatos. Hakemuslomake koostuu sivuista/segmenteistä, joiden määrittelyt (tunniste, nimi ja sivun järjestys) ovat Lomakkeen_sivut-taulussa. Lomake rakennetaan erityyppisistä elementeistä. Yksi Osio-taulun rivi vastaa yhtä lomakkeen elementtiä. Elementin tyyppejä ovat esimerkiksi:

- laatikko: selaimelle tulostuva laatikko, jonka sisälle voi laittaa muita elementtejä
- kysymys: otsikkotyyppinen teksti
- textarea: tekstimuotoinen vastauskenttä
- radio: valintapainike (voi valita yhden vaihtoehdon)
- checkbox: valintaruutu (voi valita useita vaihtoehtoja)
- date_start, date_end: päivämäärä (jakso)
- taulukko

Osiosta luodaan hierarkkinen relaatio toisen osion välille viittaamalla FK_Osio_parent attribuuttiin. Muita attribuutteja ovat mm:

- ID: elementin yksilöivä tunniste
- FK_Lomake
- Osio_nimi
- Sivun_tunniste
- Viranomaiskohtainen_tunniste
- Osio_tyyppi
- Osio_luokka
- Otsikko_fi: Kentän otsikko suomeksi
- Otsikko_en: Kentän otsikko englanniksi
- Pakollinen_tieto
- Infoteksti_fi
- Infoteksti_en
- Sarakkeiden_lkm (jos osion tyyppi on taulukko)
- Maksimi_merkit: (tekstimuotoiselle vastauskentälle)
- Jarjestys

Elementit muodostavat tietorakennetyypiltään puurakenteen, jota käydään läpi käyttöliittymässä rekursiivisesti hierarkiatasojen ja järjestys-attribuuttien mukaisessa järjestyksessä.

Lomakkeelle tallennettuja tietoja säilytetään Osio_sisalto-taulussa, jonka rakenne on muotoa:

- ID: yksilöivä tunniste
- FK_Hakemusversio: viite hakemukseen (jos lomake on hakemus)
- FK_Haettu_aineisto: viite haettavaan aineistoon (jos lomake on hakemus)

- FK_Paatos: viite päätökseen (jos lomake on päätös)
- FK_Lausunto: viite lausuntoon (jos lomake on lausunto)
- FK_Osio: viite elementtiin, johon tallennus kohdistuu
- Sisalto_text: tallennettu tieto tekstimuodossa
- Sisalto_date: tallennettu päivämäärä
- Sisalto_boolean: tallennettu boolean-arvo

Taulujen Osio_saanto ja Osio_lause avulla voi määritellä monimutkaisiakin riippuvuuksia lomakkeelle. Esimerkiksi riippuvuussäännöillä kysymysten/elementtien näyttämistä käyttäjälle voidaan rajata aiempien valintojen perusteella.

Sääntö muodostuu tietokantaan määritetystä propositiologiikan lausekkeesta, joka on konjunktiivisessa normaalimuodossa:

$$S(o_0) \leftarrow \left(P_1(o_1) \vee \ldots \vee P_n(o_n)\right) \wedge \ldots \wedge \left(P_j(o_j) \vee \ldots \vee P_m(o_m)\right)$$

o-symboli on viitattavan Osion ID-avain. S on syötteeseen assosioitu sääntö (esim. piilota tai näytä elementti) ja P on syötteeseen määritetty predikaatti/ehto (esim. onko valinta/kenttä valittu/täytetty). Osion o_0 tilaksi määräytyy $S(o_0)$, mikäli sääntöön liitetty ehtolause on tosi. Osion tila päivitetään logiikkakerroksessa lomakkeen latauksen sekä automaattisen asynkronisen tallennuksen yhteydessä.

Osioon viittaavat säännöt asetetaan Osio_saanto-tauluun. Alkeisdisjunktiot yhdistettynä konjunktioilla muodostavat ehtolauseen, joka määritetään Osio_lause-tauluun.

7.2 Liitetyypit

Hakemuslomakkeelle mahdollisesti vaadittavia liitetyyppejä on mahdollista lisätä/poistaa/konfiguroida. Liitetyyppien määrittelyt löytyvät Asiakirjahallinta_liite-taulusta. Liitetyypille asetetaan nimi, sallitut tiedostotyypit, lisätiedot, asiakirjan metatiedot sekä viittaus lomakkeeseen. Liitetyyppi voi olla lomakkeella ei-pakollinen, pakollinen tai ehdollisesti pakollinen. Pakollisissa tapauksissa Asiakirjahallinta_saanto-tauluun määritetään viite liitetyypin ID-arvoon sekä pakollisuuden tyyppi. Jos liitetyyppi on ehdollisesti pakollinen, niin ehtolause muodostetaan Osio_lause-taulun määrittelyillä samalla periaatteella kuin Osion säännön muodostamisessa (Osio_lause-taulusta viitataan Asiakirjahallinta_saanto-tauluun). Näillä menetelmillä voidaan esimerkiksi luoda ehto: *Vaadi lomakkeella liite "Opinnäytteen ohjaajan lausunto" jos käyttäjä vastaa "Kyllä" kysymykseen "Onko tietojen käyttötarkoitus opinnäytetyö?"*

8 Yleiset asiat

8.1 Tietoturva ja tietosuoja

Lupapalvelun toteutuksessa on pyritty huomioimaan tietoturva-asiat. Mm. seuraavat mahdolliset tietoturvaongelmat on tunnistettu ja niihin on varauduttu:

JavaScript-koodin syöttö lomakkeiden kenttiin. HTML-sivun lomakkeiden kenttiin voi syöttää esimerkiksi merkkijonon ”><script>alert(”Huu!”);</script>, joka saattaa johtaa siihen, että ko. teksti ajetaan JavaScript-koodina selaimessa. Ko. skripti johtaa ainoastaan ponnahdusikkunan avautumiseen, mutta se voisi tehdä muutakin vähemmän harmitonta. Ohjelmistokoodissa on huolehdittava siitä, että yo. tyylinen merkkijono ei tulostu sellaisenaan sivun HTML-koodin sekaan, vaan erikoismerkit (esim. >) korvataan HTML:n vastaavilla entiteeteillä[7]. Tämä onnistuu PHP:ssä htmlspecialchars-funktiolla.

Palvelun PHP-tiedostojen ajaminen suoraan selaimen osoiteriviltä. Web-pohjaisissa sovelluksissa siirrytään usein sivulta/näkymästä toiseen ajamalla uuden sivun/näkymän tuottama PHP-tiedosto palvelimella. Näitä PHP-tiedostoja käyttäjä voi kuitenkin ajaa myös suoraan kirjoittamalla tiedoston osoitteen ja nimen suoraan selaimen osoiteriville. Jos ko. PHP-tiedostolle voi antaa osoitteen (URL) yhteydessä myös parametreja, voi käyttäjä koittaa syöttää virhetilanteen aiheuttavia parametreja PHP-tiedostolle suoraan selaimen osoiteriviltä. Tätä varten PHP-tiedoston pitää tarkistaa syötteenään saamansa URL-parametrit.

Hakemistolistaus. Palvelun yleistä tietoturvan tasoa voi parantaa estämällä loppukäyttäjiä näkemästä palvelun hakemistojen tiedostolistan suoraan selaimen osoiteriville hakemiston nimen kirjoittamalla. Toki käyttäjät voivat saada muutenkin selville hakemistojen sisältämien tiedostojen nimet (mm. HTML-sivun lähdekoodia tarkastelemalla). Eston voi kuitenkin tehdä esim. web-palvelimen (Apache) konfiguraatiotiedostoja muokkaamalla.

Liitetiedostot. Käyttäjien on tarkoitus ladata liitetiedostoja (esim. pdf-tyyppisiä) lupapalveluun. Nämä liitetiedostot eivät saa näkyä toisille käyttäjille. Täten liitetiedostot voivat aiheuttaa tietosuorariskin. Ne voivat myös aiheuttaa tietoturvariskin, jos käyttäjä pystyy lataamaan palveluun ohjelmistokoodia sisältävän liitetiedoston ja ajamaan sen. Näiden syiden vuoksi liitetiedostot talletetaan suoraan MySQL-tietokantaan blob-tyyppisinä olioina sen sijaan, että ne talletettaisiin suoraan web-palvelimelle /var/www/html-hakemiston alle. Liitetiedostoa avattaessa palvelu tarkastaa kyseisen käyttäjän oikeudet saada nähdä ko. liitetiedoston sisältö.

Palvelunestohyökkäykset. Lupapalveluun kohdistuvien pyyntöjen määrää rajoitetaan tiedostossa _security.php. Tietyllä aikavälillä samasta ip-osoitteesta tai sessiosta ei saa tulla enempää kuin rajattu määrä pyyntöjä.

PHP:n session-muuttujien käyttöön liittyy seuraavanlainen DOS-haavoittuvuus:

- HTTP-pyyntö tulee käyttöliittymäpalvelimelle, jolloin istunto muodostetaan ja lukitaan
- Käyttöliittymäpalvelin lähettää SOAP-pyynnön logiikkakerrokseen. Vastausta odotellessa session-tiedosto on lukittuna

[7] http://www.w3schools.com/html/html_entities.asp

- Lukituksen aikana DOS-hyökkääjä voi lähettää samalla session-cookiella pyyntöjä palvelimelle jonossa niin kauan, kunnes apachen 'max_connections' ja 'max_workers' – määrä ylittyy – jolloin palvelin kaatuu.

Edellä mainitusta haavoittuvuudesta johtuen session-muuttujiin kirjoitetaan tietoa ja poistetaan lukitus ennen SOAP-pyyntöä, jotta session-tiedoston lukitus kestäisi mahdollisimman vähän aikaa.

8.2 Käyttöliittymän tekstit ja kieliversiointi

Eri käyttäjäryhmät valitsevat asiointikielen käyttäjän perustiedoissa. Järjestelmän käyttöliittymä on toteutettu suomeksi ja englanniksi.

Kieliversiointi tehdään siten, että kaikki käyttöliittymässä esiintyvät tekstit otetaan kielikohtaisissa PHP-tiedostoissa määritellyistä vakioista tai taulukoista. Tarkkaan ottaen kielitiedostot sijaitsevat käyttöliittymäkerroksen ui/language-kansiossa:

- lang_fi.php, lang_se.php: Näissä tiedostoissa on määritelty isoilla kirjaimilla PHP-vakioita, joita sitten käytetään käyttöliittymässä tekstien tulostamiseen.
- lang_fi_arrays.php, lang_en_arrays.php
    - Näissä tiedostoissa on määritelty assosiatiivisia taulukoita (muuttujia) pienillä kirjaimilla ja $-merkillä, joita sitten käytetään käyttöliittymässä tekstien tulostamiseen.
    - PHP 5.6+ tukee const-avainsanalla määriteltyjä vakiotaulukoita ja PHP 7 vakiotaulukoiden määrittelyä myös define-avainsanalla.

Kielitiedostot sisältävät myös mm. seuraavanlaisia määrittelyjä:

- liitteen tyyppi ja liitteen nimi
- hakijan roolin tyyppi ja roolin nimi
- viranomaisen roolin tyyppi ja roolin nimi
- viranomaisen koodi ja nimi
- hakemuksen tila ja tilan nimi

Tietokannan Koodistot-taulua käytetään nykyään vain siihen, kun haetaan viranomaisten koodeille selite.