Ohjelmistotuotantoprojekti

Muutos- ja korjauspyyntöjen priorisointityökalu

Ryhmä Muppett

SUUNNITTELUDOKUMENTTI

Helsinki 31.7.2008

HELSINGIN YLIOPISTO
Tietojenkäsittelytieteen laitos

Ohjelmistotuotantoprojekti, kesä 2008

Projekti:

Muutos- ja korjauspyyntöjen priorisointityökalu

Asiakas:

Oodi-konsortio/ Sampo Lehtinen

Ryhmä:

Arto Chydenius
Laura Haverinen
Merja Lindén
Topi Musto
Laura Ojala
Toni Sormunen

Ohjaaja:

Marko Lehtimäki

Kurssin vastuuhenkilö:

Kimmo Simola

Dokumentin versiohistoria

Versio	**Päiväys**	**Muutokset**	**Muuttaja**
0.0	09.06.2008	Pohja	LH
0.1	22.7.2008	Sisällön tuonti	LH
0.2	26.7.2008	Korjauksia	LO
1.0	28.7.2008	Sisältöä ja korjauksia	AC,LH,TM
1.1	31.7.2008	Viimeistely	Muppett
1.2	7.8.2008	Katselmoinnin jälkeiset korjaukset	Muppett

Sisältö

1 Johdanto

Tässä suunnitteludokumentissa kuvataan Muppett-ohjelmistoa, jonka Helsingin yliopiston tietojenkäsittelytieteen laitoksen ohjelmistotuotantoprojektikurssin kesän 2008 Muppett-ryhmä toteuttaa Oodi-konsortiolle. Suunnitteludokumentin tavoitteena on kuvata tuotettava järjestelmä sellaisella tasolla, että sen toteutus on suoraviivaista. Tämä suunnitteludokumentti on tuotettu vaatimusmäärittelydokumentin [RyM08] pohjalta.

1.1 Dokumentin sisällön kuvaus

Luvussa 2 esitetään suunnitteludokumenttiin liittyvä sanasto. Järjestelmän yleiskuvaus on luvussa 3 ja luku 4 arkkitehtuuri tarkentaa suunnittelun periaatetta. Luku 5 käsittelee toteutettavia rajapintoja ja luvut 6 ja 7 tarkentavat arkkitehtuurin luokka-tasolle. Tietokanta ja käyttöliittymä esitellään luvuissa 8 ja 9.

2 Sanasto

Abstrakti tehdas -suunnittelumalli (Abstract Factory)

Abstrakti tehdas kapseloi yhteen saman tapaisia oliotehtaita. Yksittäinen oliotehdas luo konkreettisia olioita. Abstraktille tehtaalle luodaan konkreettinen toteutusmalli, jota sitten toteutetaan erillisten rajapintojen välityksellä [Abs08, GHJ94].

Arkkitehtuuri

Tässä dokumentissa arkkitehtuuri-sanalla viitataan ohjelmistoarkkitehtuuriin. Arkkitehtuurista on esitetty useita erilaisia määritelmiä. Tässä dokumentissa arkkitehtuurin tehtävänä on ottaa kantaa keskeisiin ohjelmiston ratkaisuihin, jotka koskevat paitsi yleistä jakamista osiin myös osien välistä kommunikointitapaa [KoM05].

Komento-suunnittelumalli (Command)

Komento-suunnittelumallissa yhteen olioon kapseloidaan tietty komento ja sen parametrit. Näin saadaan sopiva tallennuspaikka muuttuville parametreille ja voidaan helposti yhdistää tietyntyyppisiä toimintoja yhdeksi luokaksi. [Com08, GHJ94].

CSS

Akronyymi sanoista Cascading Style Sheets. CSS-tyylimääritykset ovat ohjeita siitä, miten dokumentin elementit esitetään [Lai07].

DAO

Akronyymi sanasta Data Access Object (DAO) [SDN08]. DAO:t ovat luokkia, jotka kapseloivat järjestelmän tietokantaoperaatiot ja näin erottavat tietokantakohtaisen koodin muusta koodista. Ne tarjoavat rajapinnan tietokannan käsittelyyn, jolloin järjestelmän tietokannan voi vaihtaa toteuttamalla uudestaan pelkät DAO-luokat. DAO-oliot hakevat tietokannasta tarvittavat tiedot ja muodostavat niiden avulla DTO-olioita (kts. DTO). Tässä dokumentissa DAO-suunnittelumalliin perustuvia luokkia kutsutaan DAO-luokiksi ja niiden ilmentymiä DAO-olioiksi.

DTO

Akronyymi sanoista Data Transfer Object [SDN08]. DTO:t ovat olioita, jotka sisältävät dataa, mutta eivät sisällä metodeja. DTO-oliot ovat tiedon siirtoa varten: ne luodaan DAO:n tietokannasta saamien tietojen perusteella ja välitetään muille järjestelmän osille kuten mitkä tahansa oliot. DTO:ita voidaan välittää myös tietokannalle päin, jolloin niistä saadaan suoraan tietokantaan lisättävät rivit.

Front Controller -suunnittelumalli

Front Controller -suunnittelumallia käytetään usein verkkosovelluksissa. Front Controller -luokan toteutus keskittää järjestelmän pyyntöjen hallinnoinnin: kaikki järjestelmän pyynnöt ohjataan Front Controllerille, joka ratkaisee, mitä tehdään seuraavaksi [Fro08].

JavaScript

Selaimessa toimiva tapahtumaperustainen, tulkattava skriptikieli, joka on suunniteltu interaktiivisuuden lisäämiseksi HTML-merkkauskielellä laadittuihin sivuihin [Lai07].

JSP

Akronyymi sanoista Java Server Pages [JSP08]. JSP tarjoaa tavan luoda dynaamisia web-sivuja Java-ohjelmointikielellä.

MVC-arkkitehtuuri/MVC-malli

Akronyymi MVC viittaa englanninkielisiin sanoihin Model-View-Controller. MVC-mallin

(model-view-controller, malli-näkymä-ohjain) perusajatus on erottaa käyttöliittymä varsinaisesta sovelluslogiikasta ja -datasta. Tällä pyritään siihen, että käyttöliittymää olisi helppo muuttaa ja että järjestelmä olisi helppo siirtää toiselle graafiselle alustalle. Lisäksi halutaan, että käyttöliittymä heijastaa aina sovellusdatan tilaa ja näyttää sen tarvittaessa erilaisissa näkymissä aina konsistenssissa, oikeassa muodossa [KoM05].

PostgreSQL

PostgreSQL on avoimeen lähdekoodiin perustuva relaatiotietokanta. PostgreSQL tarjoaa lähes kaikki SQL-standardissa määritetyt muuttujatyypit ja sillä on useita alkuperäisiä ohjelmointirajapintoja esimerkiksi C++:lle ja Javalle [Pos08].

Rajapinta

Monet arkkitehtuurit ja suunnittelumallit perustuvat rajapintojen käyttöön. Ne ovat olennainen osa arkkitehtuuria, koska ne määräävät tavat, joilla ohjelmiston arkkitehtuurin eri osat kommunikoivat keskenään. Rajapintoja käytetään myös arkkitehtuuria alemman tason järjestelmän osien väliseen kommunikointiin [KoM05].

Servletti (Servlet)

Web-palvelimen toiminnallisuutta laajentava Java-ohjelma. Verrattavissa palvelimelta ajettavaan Java-sovelmaan [JST08].

SQL

Akronyymi sanoista Structured Query Language, joka on tietokantakyselyissä käytetty kieli. SQL mahdollistaa muun muassa tietojen kysymisen, lisäämisen, päivittämisen ja poistamisen tietokannasta. SQL ei ota kantaa verkkoprotokollaan tai standardiin, jolla tietoa kantaan syötetään, ja tästä johtuen SQL on lähes kaikkien relaatiotietokantojen tukema kieli [SQL08].

Suunnittelumalli

Suunnittelumallit ovat tunnettujen ja käytännössä hyväksi havaittujen ratkaisujen kuvauksia yleisiin ohjelmiston suunnittelua koskeviin ongelmiin tietyissä tilanteissa [GHJ94, KoM05].

Tomcat

Apachen tarjoama palvelinohjelma, joka vastaa Servlettien ja JSP- tiedostojen suoritusaikaisesta hallinnasta ja Javan kääntämisestä [Tom08].

XHTML

Akronyymi, joka on muodostettu sanoista EXtensible HyperText Markup Language. Se on XML-pohjainen merkkauskieli, jolla pyritään korvaamaan HTML. [Lai07].

3 Järjestelmän yleiskuvaus

Muppett-järjestelmä on muutos- ja korjauspyyntöjen priorisointijärjestelmä. Järjestelmä tarjoaa eri yliopistoille mahdollisuuden ilmaista mielipiteensä esitettävistä muutoksista Oodi-konsernin yliopistoille tarjoamiin järjestelmiin. Ylläpitäjälle järjestelmä tarjoaa kätevän tavan kerätä eri yliopistojen mielipiteet muutosten tärkeydestä. Muutosten priorisointiin käytetään erilaisia äänestyksiä.

3.1 Järjestelmän toteutus

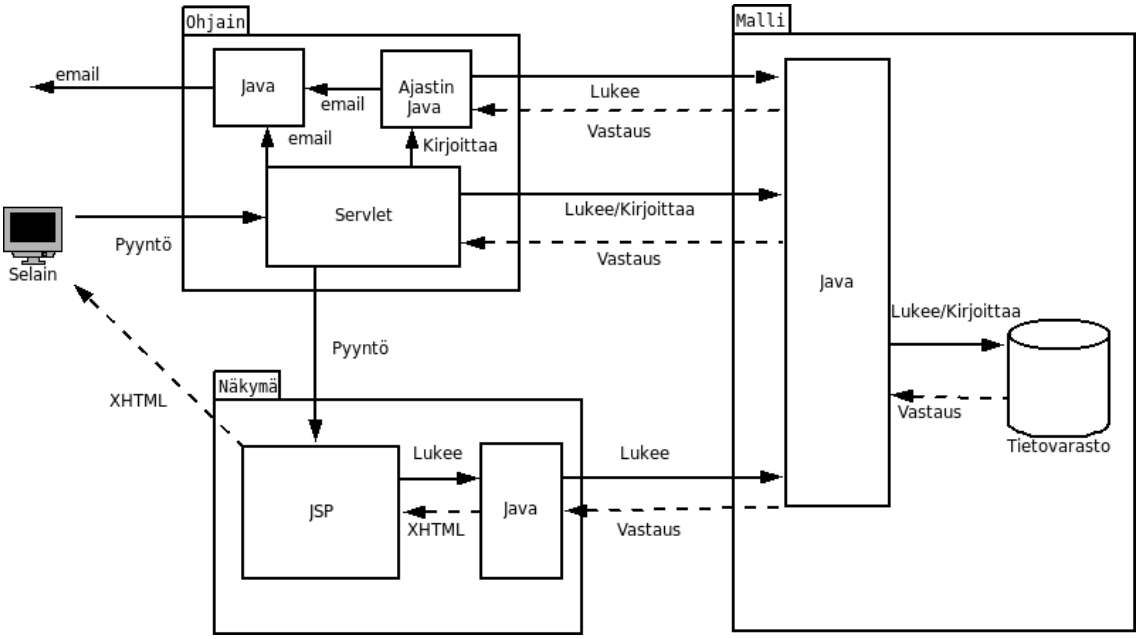
Muppett-järjestelmää käytetään WWW-selaimella. Järjestelmän toimintalogiikka toteutetaan Java-kielen versiolla 1.6. Järjestelmässä käytetään PostgreSQL-ohjelmaa tietovaraston toteuttamiseen. Järjestelmän käyttöliittymä toteutetaan XHTML 1.0 -merkkauskielellä. Sivujen ulkoasu määritellään CSS 2.1 -tyylimäärittelyillä ja toiminnallisuutta lisätään JavaScript-kielellä. Käyttöliittymä liitetään toimintalogiikkaan JavaServlet- ja JSP-tekniikoiden avulla. Tomcat 5.5.7 -palvelinohjelmisto välittää järjestelmän käyttöliittymän käyttäjän selaimelle.

3.2 Järjestelmän käyttöympäristö

Muppett-järjestelmä toteutetaan ja testataan tietojenkäsittelytieteen laitoksen db-palvelimella. Halutessaan asiakas voi siirtää järjestelmän itselleen paremmin soveltuvaan ympäristöön.

4 Arkkitehtuuri

Muppett-järjestelmä pohjautuu malli-näkymä-ohjain (MVC, Model-View-controller) -arkkitehtuuriin. Kuva 1 havainnollistaa eri osia ja niiden välisiä yhteyksiä.



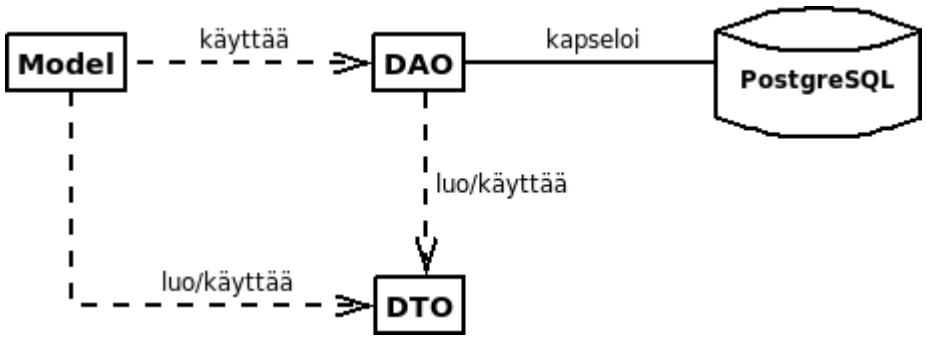
Kuva 1. Järjestelmän jakautuminen malliin, näkymään ja ohjaimeen.

Muppett-järjestelmän malli koostuu tietokannasta ja Java-luokista, jotka tarjoavat rajapinnan järjestelmän palveluihin. Luokat sisältävät sovellusalueen toimintalogiikan ja esittävät sovellusalueen tietosisällön. Näkymä koostuu JSP-tiedostoista, jotka sisältävät staattista XHTML:ää sekä Java-koodia, joka kutsuu mallia ja luo sivun dynaamisen XHTML:n. Ohjain koostuu yhdestä servletistä ja sen käyttämistä tavallisista Java-luokista. Näihin luokkiin sisältyy mm. luokat sähköpostin lähettämiseen ja taustasäikeitä käyttävä ajastinluokka. Servletin kautta WWW-selain kommunikoi Muppett-järjestelmän kanssa. Ohjain käyttää mallia esimerkiksi tallentamaan lomakkeelta lähetetyt tiedot tietokantaan, jonka jälkeen se ohjaa käyttäjän oikeaan näkymään.

4.1 Malli

Malli toteutetaan DAO-suunnittelumallin (Data Access Object) mukaisesti. DAO-luokat kapseloivat tietovarastoa koskevat operaatiot ja tarjoavat rajapinnan tietovaraston käsittelyyn. DAO-suunnittelumallia mukaillen tietovarastoa koskeva koodi erotetaan mallin muusta koodista. Muppett-järjestelmässä tietovarastona käytetään PostgreSQL -tietokantaa. Tietokannan taulut esitetään DTO-olioina (Data Transfer Object), jotka ovat julkisia kenttiä sisältäviä luokkia ilman aksessoreita tai toimintalogiikkaa. DAO-luokkien tarjoamat rajapinnat välittävät tietoa käyttäen DTO-olioita. Kuva 2 havainnol-

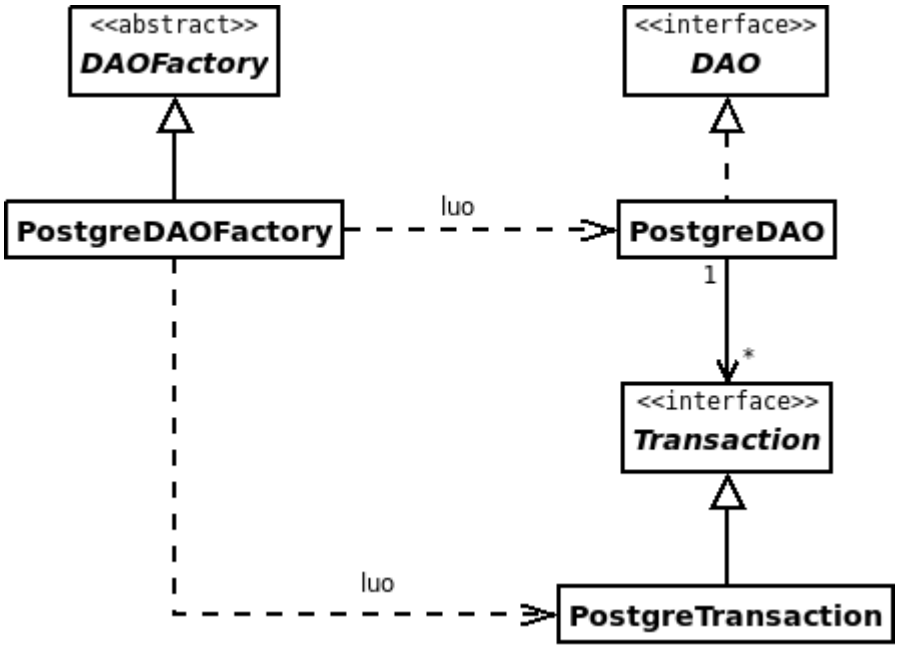
listaa mallin suhdetta DAO:jen ja DTO:iden kautta tietokantaan



Kuva 2. Malli, DAO, DTO ja tietovarasto

DAO-oliot luodaan abstrakti tehdas -suunnittelumallin (Abstract Factory) mukaisesti. DAO:t ovat määritelty rajapintaluokkina, joten niiden taustalla oleva tietokantatoteutus voidaan vaihtaa. Abstrakti tehdasluokka, DAOFactory, määrittelee, mitä rajapintoja käyttäviä DAO-olioita järjestelmä luo. Muppett-järjestelmässä PostgreDAOFactory toteuttaa abstraktin DAOFactory-luokan tarjoten rajapinnan PostgreSQL-tietokantaan (kuva 3).

Tietokantaoperaatiot suoritetaan transaktion sisällä. Näin pyritään varmistamaan, että ongelmien sattuessa tietokanta voidaan palauttaa eheään tilaan. Jokaisen transaktion jälkeen tietokannan on oltava eheässä tilassa. Aikaisemmin suoritetut onnistuneet transaktiot eivät saa hävitä myöhempien ongelmien seurauksena. Kuva 3 havainnollistaa transaktioiden suhdetta DAO-olioihin.

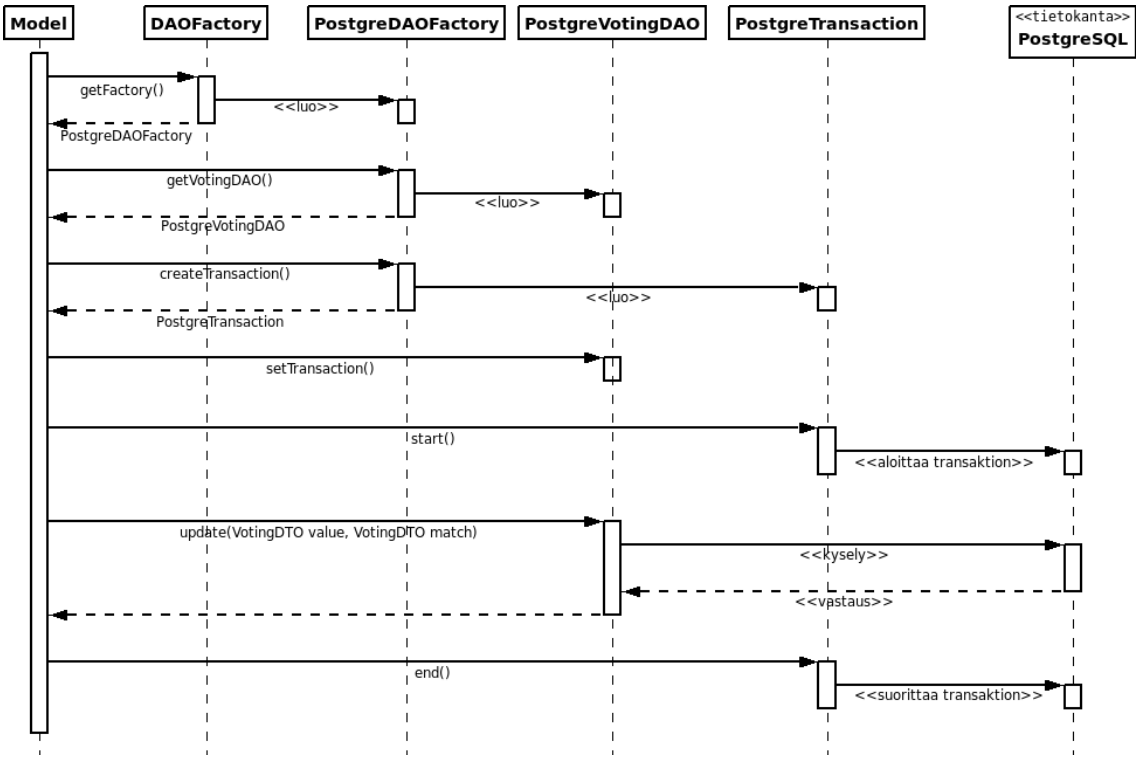


Kuva 3. PostgreDAOFactory-luokka toteuttaa abstraktin DAOFactory-luokan vaatimat metodit siten, että järjestelmällä on käytössään pääsy PostgreSQL-tietokantaan. PostgreDAOFactory-luokka luo PostgreDAO-olioita ja PostgreTransaction-olioita, joiden avulla suoritetaan tietokantaoperaatiot.

4.1.1 Esimerkki: Tietokanta, DAO ja DTO

Esimerkiksi otetaan tietokannan university-taulu. Taulussa on tiedot yliopiston tunnisteesta, nimestä, lyhenteestä ja painoarvosta. UniversityDAO-oliossa on metodit näiden tietojen listaamiseen ja viemiseen tietokantaan. UniversityDAO muodostaa tietokannalta saamistaan tiedoista UniversityDTO-olioita, joiden kentät vastaavat suoraan tietokannasta saatuja sarakkeiden arvoja. UniversityDTO-olioita käyttämällä voidaan siis välittää tietokannan tietoja järjestelmän muiden osien käyttöön esimerkiksi näkymälle, joka listaa ne käyttäjälle. UniversityDAO-luokat vastaanottavat myös UniversityDTO-olioita, joita ne välittävät edelleen tietokantaan.

Kuvassa 4 on sekvenssikaavio tietokantaoperaatioista. Malli luo abstraktin oliotehtaan, joka luo Muppett-järjestelmässä käytettävän PostgreDAOFactory-oliotehtaan. PostgreDAOFactory toteuttaa PostgreSQL-tietokannalle soveltuvat operaatiot. Malli pyytää PostgreDAOFactory:ltä VotingDAO-oliota. PostgreDAOFactory luo ja palauttaa PostgreVotingDAO-olion. Malli pyytää PostgreDAOFactorya luomaan uuden transaktion ja saa PostgreTransaction-olion. Malli alustaa transaktion ja käynnistää sen. Tämän jälkeen mallilla on yhteys tietokantaan ja malli voi suorittaa tietokantaoperaatioita. Lopuksi transaktio lopetetaan ja yhteys suljetaan.



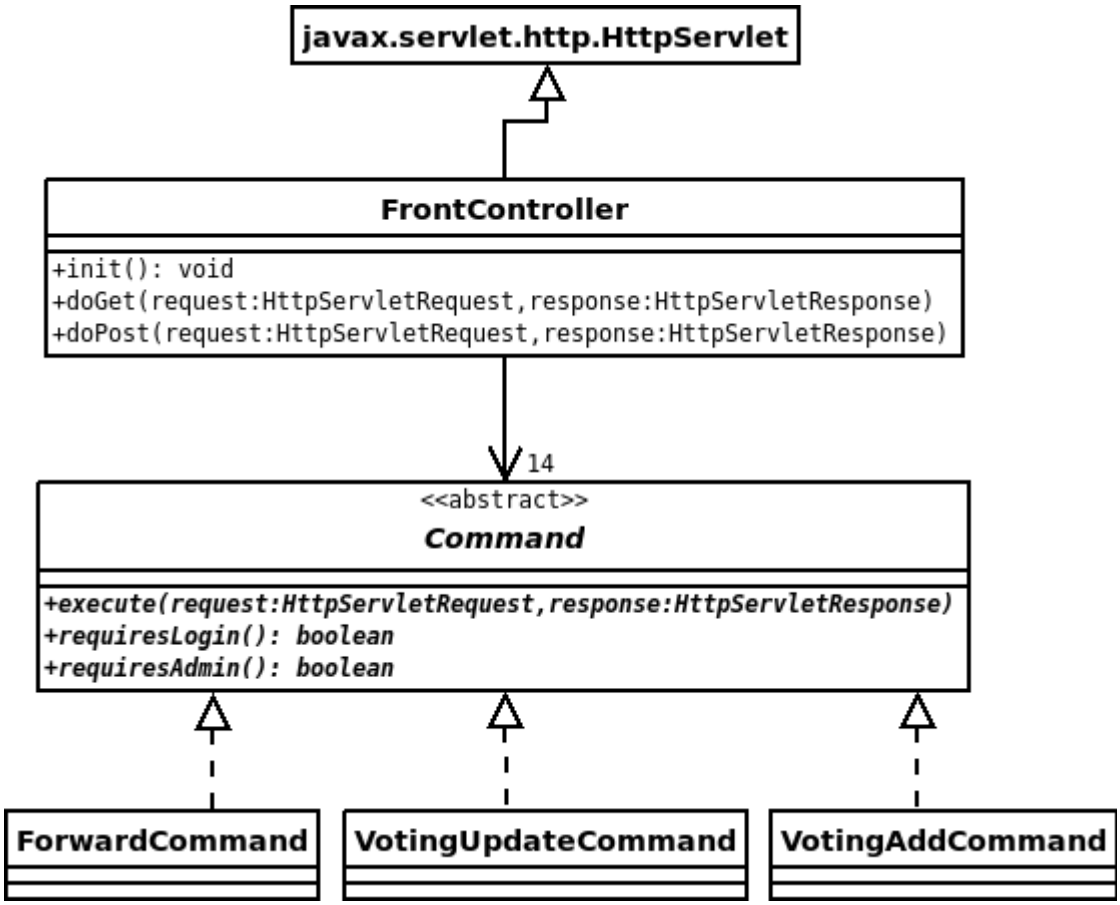
Kuva 4. Sekvenssikaavio tietokantaoperaatiosta, jossa malli päivittää tietokannan tietoja.

4.2 Näkymä

Näkymä on järjestelmän osa, jonka käyttäjä näkee selaimen välityksellä. Näkymä esittää mallin tilaa erilaisten JSP-sivujen avulla. Näkymä saa ohjeet mallin esittämisestä ohjaimelta.

4.3 Ohjain

Ohjain noudattaa Front Controller -suunnittelumallia, jonka vuoksi järjestelmässä on ainoastaan yksi servletti. Kaikki HTTP-palvelupyynnöt käyttäjän selaimelta Muppett-järjestelmälle kulkevat FrontController-luokan kautta. Tämä mahdollistaa esimerkiksi näkymä- tai palvelukohtaisten käyttöoikeuksien tarkastamisen keskitetysti. Yksittäiset palvelut on jaettu omiin luokkiinsa komento-suunnittelumallin (Command) mukaisesti. Kuva 5 havainnollistaa FrontController-luokan ja komentojen suhdetta. Vastaanottaessaan käyttäjän palvelupyynnön, FrontController-luokka valitsee sen URL:n perusteella suoritettavan komennon. URL:ien muoto esitetään kappaleessa 5.3.



Kuva 5. Front Controller toteutetaan servlettinä. Front Controller -luokalle ohjataan kaikki pyynnöt, esimerkiksi tallentaa äänestystilanteen muuttuneet tiedot. Tällöin FrontController -luokka ohjaa pyynnön VotingUpdateCommand-luokalle, kutsumalla luomansa VotingUpdateCommand-olion execute-metodia. Kuvassa ei ole esitetty järjestelmän kaikkia komentoja.

4.4 Poikkeukset

Kun järjestelmä ei itse pysty suorittamaan pyydettyä palvelua loppuun virheen vuoksi, josta se ei voi itse toipua, esimerkiksi tietokantayhteyden katketessa, se heittää poikkeuksen, jonka Tomcat-palvelin ottaa kiinni. Tällöin käyttäjä ohjautuu Tomcatin asetustiedostoissa määritellylle virhesivulle. Järjestelmän heittämä oma poikkeusolio kapseloidaan RuntimeException-poikkeukseksi tai ServletException-poikkeukseksi, jotta se voidaan välittää ulos HttpServletin do-metodeista. Järjestelmän omat poikkeusluokat DAOException, ModelException ja ControllerException peritään kaikki suoraan java.lang.Exceptionista.

5 Rajapinnat

Tässä luvussa esitellään lyhyesti mallin, näkymän ja ohjaimen tarjoamat palvelut eli rajapinnat.

5.1 Malli

Mallin tarjoama rajapinta toteutetaan luokilla UserModel, IssueModel, ModuleModel, UniversityModel ja VotingModel. Nämä luokat käyttävät ID-, DTO- ja VotingResult-luokkia, jotka ovat myös osa mallin rajapintaa.

5.2 Näkymä

Näkymä esittää käyttäjälle mallin tietoja, jotka ovat staattisia ja dynaamisia sivuja sekä niiden yhdistelmiä. Erilaiset näkymät on listattu alla.

Staattiset sivut
- Käyttöohje
- Uloskirjautuminen (”olet kirjautunut ulos”-sivu)

Dynaamiset sivut
- Sisäänkirjautuminen
- Käyttäjälistaus
- Käyttäjätietojen lisääminen/muokkaaminen
- Muutospyyntöjen lisääminen/tuominen/muokkaaminen
- Omien tietojen muokkaaminen
- Äänestystuloksen katsominen (äänestystyypin mukaisesti)
- Äänestyksen luominen/muokkaaminen
- Äänestyslistaus
- Äänestäminen (äänestystyypin mukaisesti)
- Etusivu

5.3 Ohjain

Ohjain prosessoi ja käsittelee tapahtumat. Lisäksi ohjain muuttaa tarvittaessa mallin tilaa. Ohjain huolehtii myös selväkielisten salasanojen tiivistämisestä ja suolaamisesta. Ohjaimen tarjoamat palvelut on listattu alla.

Palvelut
- Käyttäjän ohjaaminen URL:n perusteella
- Käyttäjän syötteiden lukeminen (lomakkeiden käsittely)
    - käyttäjätietojen lisääminen/muuttaminen/poistaminen
    - muutospyyntöjen lisääminen/tuominen/muuttaminen
    - sisäänkirjautuminen
    - uloskirjautuminen
    - äänestyksen luominen/muokkaaminen
    - äänestäminen
- Luettavat syötteet (POST ja GET)
- Luettavien syötteiden tarkastaminen. Esim. sähköpostiosoite, käyttäjän nimitiedot, salasanan pituus jne.
- Mallin lukeminen
- Mallin muuttaminen
- Näkymän valinta
- Sähköpostin lähettäminen
- Äänestystietojen vieminen järjestelmästä CSV:nä
- Sähköpostiviestien lähettämisen ajastaminen
- Salasanan käsittely

Käyttäjän syöttämien tietojen tarkistus ja eri sivuille ohjaaminen tapahtuu erilaisten komentojen ohjaamana. Alla on esitelty ohjaimen komennot. Ensimmäinen osa taulukosta esittää ForwardCommandin vastaanottamat URL:t ja niitä vastaavat JSP-sivut, joihin ForwardCommand ohjaa pyynnön. Taulukon jälkimmäisessä osassa on listattu ohjaimen muut komennot ja URL:t joilla niitä kutsutaan.

URL	**ForwardCommandin kohde**
forgotten_password.do	forgotten_password.jsp
frontpage.do	frontpage.jsp
issue_list.do	issue_list.jsp
ordering_voting.do	ordering_voting.jsp
classifying_voting.do	classifying_voting.jsp
upload_issues.do	upload_issues.jsp
user_list.do	user_list.jsp

voting.do	voting.jsp
voting_report.do	voting_report.jsp
edit_user.do	edit_user.jsp
edit_issue.do	edit_issue.jsp
edit_voting.do	edit_voting.jsp

URL	**Komennot**
login.do	LoginCommand
logout.do	LogoutCommand
user_update.do	UserUpdateCommand
user_add.do	UserAddCommand
user_delete.do	UserDeleteCommand
voting_update.do	VotingUpdateCommand
voting_add.do	VotingAddCommand
vote_classifying.do	VoteClassifyingCommand
vote_ordering.do	VoteOrderingCommand
voting_export.do	VotingExportCommand
issue_add.do	IssueAddCommand
issue_update.do	IssueUpdateCommand
issue_import.do	IssueImportCommand
reset_password.do	ResetPasswordCommand

5.4 Ohjaimen ja näkymän yhteistoiminta

5.4.1 Sessionhallinta

Käyttäjän kirjautuessa järjestelmään ohjain luo käyttäjälle session palvelimelle. Sessioon tallennetaan käyttäjän tiedot UserDTO-oliossa. Ohjain ja näkymä käyttävät session tietoja yhdistääkseen käyttäjän käyttäjätunnukseen, esimerkiksi käyttäjän oikeustaso saadaan session tiedoista. Käyttäjän sessio säilyy palvelimella kunnes käyttäjä kirjautuu ulos tai sessio vanhenee.

5.4.2 Lomakkeiden käsittely

JSP-sivut välittävät käyttäjän syöttämän lomakedatan ohjaimelle. Ohjain tarkistaa tietojen oikeellisuuden. Jos lomakkeen lähetys epäonnistuu, koska osa tiedoista ei ole oikeellisia, ohjain lähettää oikeelliset tiedot takaisin JSP-sivulle ja palauttaa tiedon virheestä. Jos lähetys onnistuu, ohjain palauttaa tiedon onnistuneesta tallennuksesta. Luvussa 9.3 dokumentoidaan sivujen POST-parametrien nimet.

6 Malliluokat

Tässä luvussa listataan malliluokkien tarjoamat metodit ja niiden tarkoitukset. Luokille on annettu prioriteetti luokan nimen jälkeen. Mikäli luokalle ei ole mahdollista antaa kokonaisprioriteettia tai jokin metodi poikkeaa luokan prioriteetista, on metodin perässä sen oma prioriteetti.

6.1 UserModel

UserModel-luokka tarjoaa näkymälle ja ohjaimelle rajapinnan mallin sellaiseen käsittelyyn, johon liittyy ensisijaisesti järjestelmän käyttäjien tiedot.

boolean addUser(UserAccountDTO user) throws ModelException

PRIORITEETTI 2

- Lisää järjestelmään uuden käyttäjän. Lisättävän käyttäjän tiedot annetaan DTO:na, jonka id-kenttä on tyhjä. Lisäysfunktio muokkaa sille annetun DTO:n id-kentän vastaamaan juuri lisätyn käyttäjän id:tä. Mikäli järjestelmässä on jo käyttäjä, jolla on sama sähköpostiosoite kuin lisättävällä käyttäjällä, lisääminen epäonnistuu ja metodi palauttaa false-arvon. Onnistuneessa lisäyksessä metodi palauttaa true-arvon.

boolean modifyUser(UserAccountDTO user) throws ModelException

PRIORITEETTI 3

- Muokkaa järjestelmässä olevaa käyttäjää. Muokattava käyttäjä valitaan annetun DTO:n id-kentän avulla, ja muut kentät kertovat käyttäjätietojen uudet arvot. Metodi palauttaa true-arvon, mikäli muuttaminen onnistui, ja false-arvon tilanteessa, jossa sähköpostiosoitetta yritetään muuttaa samaksi kuin jollain toisella käyttäjällä.

void deleteUser(UserAccountID user) throws ModelException

PRIORITEETTI 4

- Poistaa järjestelmästä käyttäjän. Poistettava käyttäjä valitaan annetun id:n avulla.

List<UserAccountDTO> listUsers() throws ModelException

PRIORITEETTI 4

- Palauttaa järjestelmän kaikki käyttäjätiedot listassa. Listan järjestys on määrittelemätön.

List<UserAccountDTO> listUsers(UniversityID university) throws ModelException
PRIORITEETTI 2

- Palauttaa listassa järjestelmän ne käyttäjät, jotka kuuluvat tiettyyn yliopistoon. Yliopisto valitaan annetun id:n perusteella. Listan järjestys on määrittelemätön.

UserAccountDTO listUser(UserAccountID user) throws ModelException

PRIORITEETTI 4

- Palauttaa järjestelmän yhden käyttäjän tiedot. Käyttäjä valitaan annetun id:n perusteella.

UserAccountDTO listUser(String email) throws ModelException

PRIORITEETTI 1

- Palauttaa järjestelmän yhden käyttäjän tiedot. Käyttäjä valitaan annetun sähköpostiosoitteen perusteella.

6.2 IssueModel

IssueModel-luokka tarjoaa näkymälle ja ohjaimelle rajapinnan mallin sellaiseen käsittelyyn, johon liittyy ensisijaisesti järjestelmän muutospyyntöjen tiedot.

boolean addIssue(IssueDTO issue) throws ModelException

PRIORITEETTI 1

- Lisää järjestelmään uuden muutospyynnön. Lisättävä muutospyyntö annetaan DTO:na, jonka id-kentän arvo jätetään huomioimatta. Lisäysfunktio muokkaa sille annetun DTO:n id-kentän vastaamaan juuri lisätyn muutospyynnön id:tä. Mikäli järjestelmässä on jo muutospyyntö, jolla on sama ulkoinen tunniste kuin lisättävällä muutospyynnöllä, lisääminen epäonnistuu ja metodi palauttaa false-arvon. Onnistuneessa lisäyksessä metodi palauttaa true-arvon.

boolean modifyIssue(IssueDTO issue) throws ModelException

PRIORITEETTI 4

- Muokkaa järjestelmässä olevaa muutospyyntöä. Muokattava muutospyyntö valitaan annetun DTO:n id-kentän avulla. Muut kentät kertovat muutospyynnön tietojen uudet arvot. Metodi palauttaa true-arvon mikäli muuttaminen onnistui, ja false-arvon mikäli muutospyyntöä yritetään muuttaa sellaiseksi, että sillä on sama ulkoinen tunniste kuin jollain toisella järjestelmässä olevalla muutospyynnöllä. Tällä toiminnolla voidaan myös merkitä muutospyyntö poistetuksi asettamalla sen deleted-kentän arvoksi true.

void markIssuesDeleted(Set<IssueID> issues) throws ModelException

PRIORITEETTI 4

- Merkitsee kerralla useita muutospyyntöjä poistetuksi. Merkittävät muutospyynnöt annetaan joukkona muutospyyntö-id:itä. Kaikki merkitsemiset tehdään kerralla,eikä virheen tapahtuessa yhtään muutospyyntöä merkitä.

List<IssueDTO> listIssues() throws ModelException

PRIORITEETTI 4

- Palauttaa järjestelmän kaikkien muutospyyntöjen tiedot listassa. Listan järjestys on määrittelemätön.

List<IssueDTO> listIssues(ModuleID module) throws ModelException

PRIORITEETTI 1

- Palauttaa järjestelmän tiettyyn moduuliin kuuluvien muutospyyntöjen tiedot listassa. Moduuli valitaan annetun id:n perusteella. Listan järjestys on määrittelemätön.

List<IssueDTO> listIssues(VotingID voting) throws ModelException

PRIORITEETTI 1

- Palauttaa järjestelmän tiettyyn äänestykseen liitettyjen muutospyyntöjen tiedot listassa. Äänestys valitaan annetun id:n perusteella. Listan järjestys on määrittelemätön.

List<IssueDTO> listIssues(List<ExternalID> issues) throws ModelException

PRIORITEETTI 1

- Palauttaa järjestelmän ne muutospyynnöt, joiden ulkoinen tunniste löytyy annetusta tunnistelistasta. Listan järjestys vastaa annetun tunnistelistan järjestystä.

6.3 ModuleModel

PRIORITEETTI 1

ModuleModel-luokka tarjoaa näkymälle ja ohjaimelle rajapinnan mallin sellaiseen käsittelyyn, johon liittyy ensisijaisesti järjestelmän moduulien tiedot.

List<ModuleDTO> listModules() throws ModelException

Palauttaa järjestelmän kaikkien moduulien tiedot listassa. Listan järjestys on määrittelemätön.

6.4 UniversityModel

PRIORITEETTI 1

UniversityModel-luokka tarjoaa näkymälle ja ohjaimelle rajapinnan mallin sellaiseen käsittelyyn, johon liittyy ensisijaisesti järjestelmän yliopistojen tiedot.

List<UniversityDTO> listUniversities(ModuleID module) throws ModelException

- Palauttaa listassa niiden yliopistojen tiedot, jotka on liitetty tiettyyn moduuliin. Moduuli valitaan annetun id:n perusteella. Listan järjestys on määrittelemätön.

List<UniversityDTO> listUniversities(VotingID voting) throws ModelException

- Palauttaa listassa niiden yliopistojen tiedot, jotka saavat äänestää tietyssä äänestyksessä. Äänestys valitaan annetun id:n perusteella. Listan järjestys on määrittelemätön.

boolean allowedVoter(UniversityID university, VotingID voting) throws ModelException

- Kertoo annettun yliopiston oikeuden äänestää annetussa äänestyksessä. Palauttaa true-arvon mikäli saa, ja false-arvon muussa tapauksessa.

6.5 VotingModel

VotingModel-luokka tarjoaa näkymälle ja ohjaimelle rajapinnan mallin sellaiseen käsittelyyn, johon liittyy ensisijaisesti järjestelmän äänestyksen tiedot.

void addVoting(VotingDTO voting, Set<IssueID> issues, Set<UniversityID> universities) throws ModelException

PRIORITEETTI 1

- Lisää järjestelmään uuden äänestyksen. Lisättävän äänestyksen tiedot annetaan VotingDTO:na, jonka id-kenttä on tyhjä. Metodi muokkaa sille annetun DTO:n id-kentän vastaamaan juuri lisätyn äänestyksen id:tä. Äänestyksessä äänestettävät muutospyynnöt annetaan joukkona muutospyyntöjen id:itä ja äänestyksessä sallitut äänestäjät joukkona yliopistojen id:itä.

void modifyVoting(VotingDTO voting, Set<IssueID> issues, Set<UniversityID> universities) throws ModelException

PRIORITEETTI 3

- Muokkaa järjestelmässä olevaa äänestystä. Muokattava äänestys valitaan annetun VotingDTO:n id-kentän perusteella. Muut kentät kertovat äänestyksen tietojen uudet arvot. Äänestyksessä äänestettävät muutospyynnöt annetaan joukkona muutospyyntöjen id:itä ja äänestyksessä sallitut äänestäjät joukkona yliopistojen id:itä.

void addVote(Set<VoteDTO> votes) throws ModelException

PRIORITEETTI 1

- Lisää yliopiston antaman äänen järjestelmään. Ääni muodostuu joukosta VoteDTO:ita, joilla on kaikilla sama yliopisto ja sama äänestys, mutta eri muutospyyntö.

void modifyVote(Set<VoteDTO> votes) throws ModelException

PRIORITEETTI 1

- Muokkaa yliopiston antamaa ääntä järjestelmässä. Ääni muodostuu joukosta VoteDTO:ita, joilla on kaikilla sama yliopisto ja sama äänestys, mutta eri muutospyyntö.

List<VotingDTO> listVotings() throws ModelException

PRIORITEETTI 1

- Palauttaa järjestelmän kaikkien äänestysten tiedot listassa. Listan järjestys on määrittelemätön.

VotingDTO listVoting(VotingID voting) throws ModelException

PRIORITEETTI 1

- Palauttaa tietyn äänestyksen tiedot. Äänestys valitaan annetun id:n perusteella.

List<VoteDTO> listVotes(VotingID voting, UniversityID university) throws ModelException

PRIORITEETTI 1

- Palauttaa järjestämättömän listan yliopiston antamista äänistä tietyssä äänestyksessä.

VotingResult calcResult(VotingID voting) throws ModelException

PRIORITEETTI 1

- Palauttaa annetun äänestyksen tämänhetkisestä äänestystilanteesta muodostetun äänestystulosolion, joka kertoo äänestyksen tuloksen.

6.6 VotingResult (interface)

PRIORITEETTI 1

VotingResult-rajapintaluokka kuvaa yhden äänestyksen tilannetta tietyllä hetkellä. Luokalta voi pyytää tietoa äänestyksen osallistujista, äänestyksessä annetuista äänistä ja äänestyksen lopputuloksesta. Järjestelmän kullekin äänestystyypille on oma tämän rajapintaluokan toteuttava luokka. Malli välittää VotingResult-olioita näkymälle, jotta näkymä voi tulostaa äänestyksen lopputuloksen. Jokaisella VotingResult-rajapinnan toteuttavalla luokalla on äänestystyyppikohtaisia sarakkeita, jotka antavat lisätietoa äänestyksen lopputuloksesta tai annetuista äänistä.

Alla on annettu esimerkki äänestystuloksen laskennasta vaihe vaiheelta:

1. Näkymä kutsuu VotingModel-luokan calcResult-metodia
2. calcResult-metodi hakee tietokannasta yksittäisen äänestyksen tiedot (voting) tarkistaakseen äänestyksen tyypin.
3. calcResult-metodi hakee tietokannasta muun äänestyksen tietosisällön (vote, allowed_voter, issue).
4. calcResult-metodi valitsee äänestyksen tyypin perusteella jonkin VotingResult-rajapinnan toteuttavan äänestystyyppiluokan.
5. calcResult-metodi luo valitun äänestystyyppiluokan olion, joka toteuttaa VotingResult-rajapintaluokan tietokannasta haetun tietosisällön perusteella.
6. calcResult-metodi palauttaa luodun äänestystyyppiluokan olion näkymälle.
7. VotingResult-olio laskee äänestyksen tuloksen.
8. Näkymä kysyy oliolta tarvitsemansa tiedot.

VotingDTO voting();

- Palauttaa tämän VotingResult-olion kuvaaman äänestyksen tiedot.

List<IssueDTO> results();

- Palauttaa tämän VotingResult-olion kuvaaman äänestyksen muutospyyntöjen tiedot järjestetyssä listassa muutospyyntöjen saamien äänien mukaan siten, että eniten ääniä saaneet muutospyynnöt ovat listan ensimmäisinä.

List<UniversityDTO> voters();

- Palauttaa tämän VotingResult-olion kuvaaman äänestyksen äänestäjien tiedot järjestämättömässä listassa.

Map< IssueID, Map<UniversityID, VoteValue> > votes();

- Palauttaa tämän VotingResult-olion kuvaamassa äänestyksessä annetut äänet. Äänet on järjestetty sisäkkäisiin assosiaatiotauluihin niin, että yksittäiselle muutospyynnölle voidaan helposti hakea, minkä äänen kukin äänestäjä on kyseiselle

muutospyynnölle antanut.

List<String> votingSpecificColumnNames()

- Palauttaa listan yhteenvetotietojen nimistä, jotka tämä äänestystyyppi tarjoaa.

Map< IssueID, List<String> > votingSpecificColumns()

- Palauttaa assosiaatiotaulullisen listoja tämän äänestyksen yhteenvetotiedoista. Kullekin muutospyynnölle tarjotaan lista yhteenvetotiedoista, joka on järjestetty vastaavaan järjestykseen votingSpecificColumnNames()-metodin palauttaman listan järjestyksen kanssa.

6.6.1 ClassifyingVoting (implements VotingResult)

PRIORITEETTI 1

ClassifyingVoting-luokka laskee luokitteluäänestyksen lopputuloksen. Äänestyskohtainen sarake on keskiarvo.

ClassifyingVoting(VotingDTO voting, List<VoteDTO> votes, List<AllowedVoterDTO> allowedVoters, List<IssueDTO> issues)

- Muodostaa uuden olion yhden äänestyksen tietojen perusteella. Äänestyksestä annetaan yleiset tiedot, äänestyksessä annetut äänet, ne yliopistot, jotka saavat osallistua äänestykseen, ja ne muutospyynnöt, joita äänestyksessä saa äänestää.

List<String> votingSpecificColumnNames()

- Palauttaa listan { "Keskiarvo" }

6.6.2 OrderingVoting (implements VotingResult)

PRIORITEETTI 1

OrderingVoting-luokka laskee tärkeysjärjestysäänestyksen lopputuloksen. Äänestyskohtainen sarake on virhepisteiden summa.

OrderingVoting(VotingDTO voting, List<VoteDTO> votes, List<AllowedVoterDTO> allowedVoters, List<IssueDTO> issues)

- Muodostaa uuden olion yhden äänestyksen tietojen perusteella. Äänestyksestä annetaan yleiset tiedot, äänestyksessä annetut äänet, ne yliopistot, jotka saavat osallistua äänestykseen, ja ne muutospyynnöt, joita äänestyksessä saa äänestää.

List<String> votingSpecificColumnNames()

- Palauttaa listan { "Yhteensä" }

6.7 DAOFactory (abstract)

PRIORITEETTI 1

Luokan tehtävänä on luoda DAO-olioita mallin muiden luokkien käytettäväksi ja se noudattaa abstrakti tehdas -suunnittelumallia.

static void init()

- Alustaa abstraktin tehtaan. Tätä on kutsuttava kerran ennen muiden metodien käyttöä.

static DAOFactory getFactory()

- Palauttaa konkreettisen DAOFactory-olion mallin luokkien käytettäväksi.

abstract Transaction createTransaction()

- Luo uuden transaktio-olion annettavaksi tehtaalta pyydetyille DAO-oliolle.

abstract void releaseTransaction(Transaction transaction)

- Vapauttaa transaktio-olion käyttämät resurssit kun oliota ei enää käytetä.

abstract UserAccountDAO createUserAccountDAO()

abstract IssueDAO createIssueDAO()

abstract ModuleDAO createModuleDAO()

abstract UniversityDAO createUniversityDAO()

abstract AllowedVoterDAO createAllowedVoterDAO()

abstract VotingDAO createVotingDAO()

abstract VoteDAO createVoteDAO()

- Luovat uusia DAO-olioita mallin luokille käytettäväksi.

6.8 Transaction (interface)

PRIORITEETTI 1

Rajapinnan mahdollistaa tietovaraston käsittelyn niin, että useat operaatiot suoritetaan joko yhtenä kokonaisuutena tai virhetilanteessa ei ollenkaan. Transaktio-oliota käytetään niin, että ensin abstrakti tehdas luo sen, jonka jälkeen se annetaan yhdelle tai useammalle DAO:lle. Seuraavaksi transaktio aloitetaan, jonka jälkeen DAO:ja käytetään ja lopulta transaktio lopetetaan ja palautetaan tehtaalle tuhottavaksi. Mikäli transaktiota ei voida suorittaa kerrallaan kokonaisena, voi DAO:jen käsittely heittää poikkeuksen. Tällöin transaktion voi peruuttaa alkutilaansa ja suoritusta voi yrittää samalla transaktiolla uudestaan. DAO voi myös käyttää transaktiota aloittamatta sitä koskaan. Tällöin DAO:n tekemät muutokset menevät heti yksi kerrallaan tietovarastoon asti.

begin()

- Aloittaa transaktion. Aloittamisen jälkeen DAO, joka käsittelee tietovarastoa tämän transaktion läpi, ei tee mitään näkyvää muutosta tietovarastoon ennen transaktion lopettamista.

rollback()

- Peruuttaa transaktion. Transaktion läpi tehdyt muutokset tietovarastolle eivät tapahdukaan. Transaktio pitää tällöin aloittaa uudelleen.

end()

- Lopettaa transaktion. Toteuttaa transaktion läpi tehdyt muutokset tietovarastolle.

6.9 DAO-luokat

DAO-rajapintaluokat tarjoavat palvelut mallin tietovaraston lukemiseen ja kirjoittamiseen. Mallin luokat luovat DAO-rajapinnan toteuttavia oliota pyytämällä abstraktia tehdasta luomaan sellaisen. Jokaista DTO-luokkaa vastaa oma DAO-luokkansa, joka mahdollistaa kyseisen DTO-luokan tietojen hakemisen tietovarastosta. DAO-luokkien metodit ovat kaikille luokille samat, joten niistä kuvataan tarkemmin vain yksi esimerkki, UserAccountDAO.

6.9.1 UserAccountDAO

PRIORITEETTI 1

Luokka mahdollistaa tietovaraston UserAccountDTO:ita koskevan käsittelyn.

void setTransaction(Transaction transaction)

- Asettaa DAO:lle transaktion. Ennen DAO:n käyttöä sille täytyy asettaa abstraktilta tehtaalta pyydetty transaktio. DAO käyttää transaktiota muodostaakseen yhteyden tietovarastoon. Saman transaktion voi antaa usealle DAO:lle samaan aikaan, jolloin usean DAO:n toiminta saadaan yhdistettyä yhdeksi tietovaraston toimenpiteeksi. Kun DAO:n käyttö on lopetettu, tulee transaktio palauttaa abstraktille tehtaalle lopetettavaksi.

void insert(UserAccountDTO value) throws DAOException

- Lisää uuden käyttäjätiedon tietovarastoon. Annetun DTO:n id-kenttä jätetään huomioimatta. Metodi tallentaa id-kenttään tietovarastosta saadun id:n. Muut kentät tallennetaan sellaisenaan.

void delete(UserAccountID user) throws DAOException

- Poistaa käyttäjän tietovarastosta. Poistettava käyttäjä valitaan annetun id:n perusteella.

List<UserAccountDTO> find(UserAccountDTO match) throws DAOException

- Hakee käyttäjätietoja tietovarastosta. Annettu match-DTO kertoo, mitä käyttäjätietoja haetaan. Palauttaa järjestämättömän listan niistä käyttäjistä, joiden tiedot

vastaavat kaikkia match-olion epätyhjiä kenttiä.

void update(UserAccountDTO value, UserAccountDTO match) throws DAOException

- Päivittää käyttäjän tietoja tietovarastossa. Annetun value-DTO:n id-kenttä jätetään huomioimatta. Value-DTO:n muut kentät kertovat käyttäjän mahdollisesti päivitettävät tiedot. Annetun match-DTO:n vertailu tietovarastoon kertoo, kenen käyttäjätietoja päivitetään.

6.9.2 IssueDAO

PRIORITEETTI 1

void insert(IssueDTO value) throws DAOException

void update(IssueDTO value, IssueDTO match) throws DAOException

void delete(IssueDTO match) throws DAOException

List<IssueDTO> find(IssueDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.3 ModuleDAO

PRIORITEETTI 1

void insert(ModuleDTO value) throws DAOException

void update(ModuleDTO value, ModuleDTO match) throws DAOException

void delete(ModuleDTO match) throws DAOException

List<ModuleDTO> find(ModuleDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.4 UniversityDAO

PRIORITEETTI 1

void insert(UniversityDTO value) throws DAOException

void update(UniversityDTO value, UniversityDTO match) throws DAOException

void delete(UniversityDTO match) throws DAOException

List<UniversityDTO> find(UniversityDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.5 AllowedVoterDAO

PRIORITEETTI 1

void insert(AllowedVoterDTO value) throws DAOException

void update(AllowedVoterDTO value, AllowedVoterDTO match) throws DAOException

void delete(AllowedVoterDTO match) throws DAOException

List<AllowedVoterDTO> find(AllowedVoterDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.6 VotingDAO

PRIORITEETTI 1

void insert(VotingDTO voting) throws DAOException

void update(VotingDTO value, VotingDTO match) throws DAOException

void delete(VotingDTO match) throws DAOException

List<VotingDTO> find(VotingDTO value, VotingDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.7 VoteDAO

PRIORITEETTI 1

void insert(VoteDTO vote) throws DAOException

void update(VoteDTO vote) throws DAOException

void delete(VoteDTO match) throws DAOException

List<VoteDTO> find(VoteDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.8 VotingIssueDAO

PRIORITEETTI 1

void insert(VotingIssueDTO votingIssue) throws DAOException

void update(VotingIssueDTO votingIssue) throws DAOException

void delete(VotingIssueDTO match) throws DAOException

List<VotingIssueDTO> find(VotingIssueDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.9.9 UniversityModuleDAO

PRIORITEETTI 1

void insert(UniversityModuleDTO universityModule) throws DAOException

void update(UniversityModuleDTO universityModule) throws DAOException

void delete(UniversityModuleDTO match) throws DAOException

List<UniversityModuleDTO> find(UniversityModuleDTO match) throws DAOException

void setTransaction(Transaction transaction)

6.10 ID- luokat

Tietokannan taulujen pääavainten tyypit on kapseloitu erillisiin luokkiin, jotta järjestelmän tietokanta on helpommin vaihdettavissa toiseen. Muppett-järjestelmän ID:t ovat int-tyyppisiä, ja kaikki ID-luokat sisältävät samat metodit, joten ainoastaan UserAccountID on kuvattu tarkemmin.

6.10.1 UserAccountID

PRIORITEETTI 1

Luokka UserAccountID vastaa user_account-taulun pääavainta

Konstruktorit:

UserAccountID(int value)

- Asettaa uuden olion ID:n arvon parametrina annetuksi arvoksi.

  UserAccountID(String value)

- Muuntaa parametrina annetun String-tyyppisen olion int-tyyppiseksi ja asettaa sen uuden olion arvoksi.

Metodit:

int getValue()

- Palauttaa ID:n arvon.

void setValue(int value)

- Asettaa ID:n arvon.

String toString()

- Muuntaa ID:n arvon String-tyyppiseksi ja palauttaa sen.

6.10.2 UniversityID

PRIORITEETTI 1

Luokka UniversityID vastaa university-taulun pääavainta.

6.10.3 IssueID

PRIORITEETTI 1

Luokka IssueID vastaa issue-taulun pääavainta.

6.10.4 ModuleID

PRIORITEETTI 1

Luokka ModuleID vastaa module-taulun pääavainta.

6.10.5 VotingID

PRIORITEETTI 1

Luokka VotingID vastaa voting-taulun pääavainta.

6.11 DTO-luokat

6.11.1 UserAccountDTO

PRIORITEETTI 1

Kentät:

UserAccountID id

String firstName

String lastName

String hashedPassword

String passwordSalt

String email

UniversityID university

Boolean admin

Konstruktorit:

UserAccountDTO()

UserAcountDTO(UserAccountID id, String firstName, String lastName, String hashedPassword, String passwordSalt, String email, UniversityID university, Boolean admin)

6.11.2 IssueDTO

PRIORITEETTI 1

Kentät:

IssueID id

String url

Integer workload_estimate

String workload_guess

String externalID

String title

String version

ModuleID module

String type

Boolean deleted

konstruktorit:

IssueDTO()

IssueDTO(IssueID id, String url, Integer workload_estimate, Integer workload_guess, String externalID, String title, String version, ModuleID module, String type, Boolean deleted)

6.11.3 AllowedVoterDTO

PRIORITEETTI 1

kentät:

VotingID voting

UniversityID university

konstruktorit:

AllowedVoterDTO()

AllowedVoterDTO(VotingID voting, UniversityID university)

6.11.4 VoteDTO

PRIORITEETTI 1

kentät:

UniversityID university

VotingID voting

IssueID issue

String voteValue

konstruktorit:

VoteDTO()

VoteDTO(UniversityID university, VotingID voting, IssueID issue, String voteValue)

6.11.5 UniversityDTO

PRIORITEETTI 1

kentät:

UniversityID id

String name

String abbreviation

Float weight

konstruktorit:

UniversityDTO()

UniversityDTO(UniversityID id, String name, String abbreviation, Float weight)

6.11.6 VotingDTO

PRIORITEETTI 1

kentät:

VotingID id

String name

Date startTime

Date endTime

Integer type

Boolean closed

Boolean deleted

Boolean weighted

Boolean public_votes

konstruktorit:

VotingDTO()

VotingDTO(VotingId id, String name, Date startTime, Date endTime, Integer type, Boolean closed, Boolean deleted, Boolean weighted, Boolean public_votes)

6.11.7 VotingIssueDTO

PRIORITEETTI 1

kentät:

VotingID voting

IssueID issue

konstruktorit:

VotingIssueDTO()

VotingIssueDTO(VotingID voting, IssueID issue)

6.11.8 ModuleDTO

PRIORITEETTI 1

kentät:

ModuleID id

String name

konstruktorit:

ModuleDTO()

ModuleDTO(ModuleID id, String name)

6.11.9 UniversityModuleDTO

PRIORITEETTI 1

kentät:

UniversityID university

ModuleID module

konstruktorit:

UniversityModuleDTO()

UniversityModuleDTO(UniversityID university, ModuleID module)

7 Ohjainluokat

Tässä luvussa listataan ohjainluokkien tarjoamat metodit ja niiden tarkoitukset. Luokille on annettu prioriteetti luokan nimen jälkeen.

7.1 FrontController

PRIORITEETTI 1

Kaikki käyttäjän selaimelta tulevat pyynnöt järjestelmän palveluihin kulkevat FrontController-rollerin kautta. FrontControllerin yksittäiset toiminnallisuudet on jaettu komento-suunnittelumallin mukaisesti erillisiin luokkiin, eli komentoihin. FrontController tunnistaa palvelut pyynnön URL:n .do -päätteestä ja valitsee .do:ta edeltävästä URL:n osasta oikean komennon. URL:ien ja komentojen välinen suhde esitetään kappaleessa 5.3.

void init() throws ControllerException

- Luo command-luokasta perityt komennot ja alustaa muun järjestelmän.

void doPost(HttpServletRequest request, HttpServletResponse response) throws ControllerException

- Käsittelee käyttäjän selaimen lähettämät POST-pyynnöt. Valitsee komennon pyynnön URL:n perusteella, tarkistaa komennon vaatiman oikeustason ja suorittaa komennon execute-metodin, mikäli kirjautuneella käyttäjällä on riittävä oikeustaso.

void doGet(HttpServletRequest request, HttpServletResponse response) throws ControllerException

- Käsittelee käyttäjän selaimen lähettämät GET-pyynnöt. Valitsee komennon pyynnön URL:n perusteella, tarkistaa komennon vaatiman oikeustason ja suorittaa komennon execute-metodin, mikäli kirjautuneella käyttäjällä on riittävä oikeustaso.

7.2 Command (abstract)

PRIORITEETTI 1

Abstrakti Command-luokka toimii yliluokkana ohjaimen eri toimintoja toteuttaville aliluokille. Kukin aliluokka toteuttaa abstraktin execute-metodin, jota FrontController kutsuu, mutta vain mikäli sen hetkisellä käyttäjällä on riittävät oikeustasot kyseisen komennon suorittamiseen.

void dispatch(HttpServletRequest request, HttpServletResponse response, String location) throws ServletException, IOException

- Ohjaa käyttäjän pyynnön johonkin osoitteeseen, eli yleensä jollekin JSP-sivulle käsiteltäväksi.

abstract boolean requiresLogin()

- Kertoo vaatiko tämän komennon käyttö sisäänkirjautuneen käyttäjän. Palauttaa true-arvon mikäli vaatii ja false-arvon mikäli ei vaadi.

abstract boolean requiresAdmin()

- Kertoo vaatiko tämän komennon käyttö sisäänkirjautuneen käyttäjän, joka on ylläpitäjä. Palauttaa true-arvon mikäli vaatii ja false-arvon mikäli ei vaadi.

abstract void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

- Sisältää tämän komennon toiminnallisuuden. Suorituksen loputtua komento ohjaa

käyttäjän pyynnön jollekin JSP-sivulle käsiteltäväksi, eli valitsee mikä näkymä näytetään.

7.2.1 ForwardCommand (extends Command)

PRIORITEETTI 1

Luokkaa käytetään ohjaamaan käyttäjän selaimen GET- tai POST-pyyntö muille komennoille tai jollekin muulle palvelimella olevalle resurssille. Kohteena oleva resurssi määritellään luodessa ilmentymää luokasta. Instanssia luodessa määritellään myös vaatiiko kohteena oleva resurssi kirjautumisen tai ylläpitäjän oikeudet.

ForwardCommand(String forwardLocation, boolean requiresLogin, boolean requiresAdmin) throws ControllerException

- Luo uuden ilmentymän ForwardCommand-luokasta.

void execute(HttpServletRequest request, HttpServletResponse) throws ControllerException

- Sisältää ForwardCommand -luokan toiminnallisuuden.

boolean requiresLogin()

- Palauttaa konstruktorissa määritellyn arvon.

boolean requiresAdmin()

- Palauttaa konstruktorissa määritellyn arvon.

7.2.2 LoginCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on näyttää käyttäjälle sisäänkirjautumissivu, kirjata käyttäjä sisään järjestelmään ja ohjata pyyntö pääsivulle mikäli sähköpostiosoite ja salasana olivat oikein. Jos komentoa kutsutaan GET:llä, näytetään aina pelkkä kirjautumissivu

(login.jsp). Jos komentoa kutsutaan POST:lla, se tarkastaa lomakkeelta lähetetyn tunnuksen ja salasanan. Mikäli nämä ovat oikein, komento tallentaa sessioon käyttäjän tiedot UserDTO:na ja ohjaa käyttäjän järjestelmän etusivulle (frontpage.jsp). Komento ei vaadi kirjautumista eikä ylläpitäjän oikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.3 LogoutCommand (extends Command)

PRIORITEETTI 2

Luokan tehtävänä on tyhjentää käyttäjän tiedot sessiosta ja ohjata käyttäjä "olet kirjautunut ulos" -sivulle (logout.jsp). Komento ei vaadi kirjautumista eikä ylläpitäjäoikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.4 UserUpdateCommand (extends Command)

PRIORITEETTI 3

Luokan tehtävänä on tarkistaa käyttäjätietojen muokkaamissivulta (edit_user.jsp) lähetetty POST-data, päivittää mallia (UserModel) ja ohjata pyyntö takaisin muokkaamissivulle (edit_user.jsp). Komentoa käytetään kun käyttäjä haluaa muokata omia tietojaan. Lisäksi ylläpitäjällä on oikeus muokata kenen tahansa käyttäjän tietoja. Komento vaatii kirjautumisen muttei ylläpitäjän oikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.5 UserAddCommand (extends Command)

PRIORITEETTI 2

Luokan tehtävänä on tarkistaa käyttäjätietojen muokkaamissivulta (edit_user.jsp) lähetetty POST-data, päivittää mallia (UserModel) ja ohjata pyyntö takaisin muokkaamissivulle (edit_user.jsp). Komento vaatii sekä kirjautumisen että ylläpitäjän oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.6 UserDeleteCommand (extends Command)

PRIORITEETTI 4

Luokan tehtävänä on päivittää malliin (UserModel) käyttäjä poistetuksi ja ohjata pyyntö käyttäjälistaussivulle (user_list.jsp). Komento vaatii sekä kirjautumisen että ylläpito-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.7 VotingAddCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on tarkistaa äänestyksen lisäämissivulta (edit_voting.jsp) lähetetty POST-data, päivittää mallia (VotingModel) ja ohjata pyyntö takaisin äänestyksen muokkaamissivulle (edit_voting.jsp). Komento vaatii sekä kirjautumisen että ylläpito-

oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.8 VoteClassifyingCommand (extends Command):

PRIORITEETTI 1

Luokan tehtävänä on tarkistaa luokitteluäänestyssivulta (classification_voting.jsp) lähetetty POST-data, päivittää mallia (VotingModel) ja ohjata käyttäjä takaisin luokitteluäänestyssivulle (classification_voting.jsp). Komento vaatii kirjautumista muttei ylläpitäjä-oikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.9 VoteOrderingCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on tarkistaa järjestysäänestyssivulta (ordering_voting.jsp) lähetetty POST-data, päivittää mallia (VotingModel) ja ohjata käyttäjä takaisin järjestysäänestyssivulle (ordering_voting.jsp). Komento vaatii kirjautumista muttei ylläpitäjä-oikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.10 VotingExportCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on luoda äänestystilanteesta raportti ja lähettää se käyttäjän selaimelle CSV-tiedostona. Äänestys valitaan käyttäjän pyynnön GET- tai POST-datasta luetun äänestys-id:n perusteella. Komento hakee äänestyksen tiedot mallilla ja tulostaa ne käyttäjän selaimelle sopivasti muotoiltuna. Komento vaatii sekä kirjautumisen että ylläpitäjä-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.11 VotingUpdateCommand (extends Command)

PRIORITEETTI 3

Luokan tehtävänä on tarkistaa äänestyksenmuokkaussivulta (edit_voting.jsp) lähetetty POST-data, päivittää mallia (VotingModel) ja ohjata käyttäjä takaisin äänestyksenmuokkaussivulle (edit_voting.jsp). Komento vaatii sekä kirjautumisen että ylläpitäjä-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.12 IssueAddCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on lukea ylläpitäjän täyttämän muutospyynnön lisäys-sivun (edit_issue.jsp) lomakkeen tiedot käyttäjän pyynnön POST-datasta ja tarkistaa niiden oikeellisuus. Mikäli tiedot ovat kelpoja, komento pyytää mallia lisäämään muutospyynnön tietokantaan. Lopuksi käyttäjän pyynnön ohjataan uudelleen lisäys-sivulle, jolloin

pyynnön mukaan liitetään tieto siitä onnistuiko lisääminen tai miksi se ei onnistunut. Komento vaatii sekä kirjautumisen että ylläpitäjä-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.13 IssueUpdateCommand (extends Command)

PRIORITEETTI 4

Luokan tehtävänä on tarkistaa muutospyynnönmuokkaussivulta (edit_issue.jsp) tullut POST-data, päivittää mallia (IssueModel) ja ohjata pyyntö takaisin muutospyynnön-muokkaussivulle (edit_issue.jsp). Komento vaatii sekä kirjautumisen että ylläpitäjä-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.14 IssueImportCommand (extends Command)

PRIORITEETTI 1

Luokan tehtävänä on lukea muutospyyntöjen lisäyssivulta (upload_issues.jsp) lähetetty CSV-tiedosto, tarkistaa sen oikeellisuus, parsia siinä määritellyt muutospyynnöt ja lisätä ne tietokantaan mallin (IssueModel) avulla ja ohjata pyyntö takaisin muutospyyntöjen lisäyssivulle (upload_issues.jsp). Komento vaatii sekä kirjautumisen että ylläpitäjä-oikeudet.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.2.15 ResetPasswordCommand (extends Command)

PRIORITEETTI 2

Luokan tehtävänä on asettaa uusi satunnainen salasana käyttäjälle, joka annetaan forgotten_password.jsp -sivun lomakkeessa. Salasanan asettamisesta lähetetään tieto käyttäjän sähköpostiin. Komento ei vaadi kirjautumista eikä ylläpitäjän oikeuksia.

void execute(HttpServletRequest request, HttpServletResponse response) throws ControllerException

boolean requiresLogin()

boolean requiresAdmin()

7.3 Hasher

PRIORITEETTI 1

Luokan tehtävä on tarjota palvelu salasanojen tiivistämiseksi, jottei niitä tallenneta tietokantaan selväkielisenä. Tiivistämiseen käytetään SHA-512 algoritmia. Luokka tarjoaa myös metodit salasanan ja suolan luomiseen.

static String hashPassword(String password, String passwordSalt) throws ModelException

- Tiivistää annetun selväkielisen salasanan ja annetun suolauksen yhdistelmän.

static String createPassword()

- Palauttaa satunnaisen merkkijonon, jota käytetään salasanana.

static String createPasswordSalt()

- Palauttaa satunnaisen merkkijonon, jota käytetään salasanan suolana.

7.4 Validate

PRIORITEETTI 1

Validate-luokka tarjoaa rajapinnan näkymän kautta tulevan tiedon tarkistamiseen. Validate-luokka koostuu tiedon tarkistamisen suorittavista staattisista metodeista ja metodien palauttamista enumeraatioista. Enumeraatiot ilmaisevat tiedon oikeellisuuden. Näkymä välittää ainoastaan String-tyyppisiä parametreja, ja näkymältä tulevat String-tyyppiset parametrit tarkistetaan sellaisenaan. Jos parametri kuvaa muun tyyppistä, esimerkiksi päivämäärä-tietoa, sisältöä, muutetaan se oikeaan muotoonsa. Uuteen muotoon muutettu parametri asetetaan metodikutsun toiseen olioparametriin. Esimerkiksi votingStartTime-metodi saa parametrinaan sekä String-muotoisen tiedon alkupäivästä että Date-olion, jotta metodi voi muuntaa Stringin Date-olioksi ja tallentaa sen viiteparametrin osoittamaan olioon.

Validaten määrittelemät enumeraatiot:

UniversityID { OK, EMPTY, MALFORMED }

UserAccountID { OK, EMPTY, MALFORMED }

IssueID { OK, EMPTY, MALFORMED }

ModuleID { OK, EMPTY, MALFORMED }

VotingID { OK, EMPTY, MALFORMED }

UserAccountFirstName { OK, EMPTY, TOO_LONG }

UserAccountLastName { OK, EMPTY, TOO_LONG }

UserAccountPassword { OK, EMPTY, TOO_LONG, TOO_SHORT }

UserAccountPasswordSalt { OK, EMPTY, TOO_LONG, TOO_SHORT }

UserAccountEmail { OK, EMPTY, MALFORMED }

IssueURL { OK, EMPTY, TOO_LONG }

IssueWorkloadEstimate { OK, EMPTY, MALFORMED, NEGATIVE }

IssueWorkloadGuess { OK, EMPTY, TOO_LONG }

IssueExternalID { OK, EMPTY, TOO_LONG }

IssueTitle { OK, EMPTY, TOO_LONG }

IssueVersion { OK, EMPTY, TOO_LONG }

IssueType { OK, EMPTY, TOO_LONG }

VotingName { OK, EMPTY, TOO_LONG }

VotingStartTime { OK, EMPTY, MALFORMED, IN_PAST }

VotingEndTime { OK, EMPTY, MALFORMED, IN_PAST }

VotingType { OK, EMPTY, NEGATIVE, NO_SUCH_TYPE }

VoteValue { OK, EMPTY, MALFORMED, TOO_LONG }

validaten metodit:

Validate.UniversityID universityID(String universityID, UniversityID validatedUniversityID)

Validate.UserAccountID userAccountID(String userAccountID, UserAccountID validatedUserAccountID)

Validate.IssueID issueID(String issueID, IssueID validatedIssueID)

Validate.ModuleID moduleID(String moduleID, ModuleID validatedModuleID)

Validate.VotingID votingID(String votingID, VotingID validatedVotingID)

Validate.UserAccountFirstName userAccountFirstName(String firstName)

Validate.UserAccountLastName userAccountLastName(String lastName)

Validate.UserAccountPassword userAccountPassword(String password)

Validate.UserAccountPasswordSalt userAccountPasswordSalt(String passwordSalt)

Validate.UserAccountEmail userAccountEmail(String AccountEmail)

Validate.IssueURL issueURL(String url)

Validate.IssueWorkloadEstimate issueWorkloadEstimate(String workloadEstimate, Integer validatedWorkloadEstimate)

Validate.IssueWorkloadGuess issueWorkloadGuess(String workloadGuess)

Validate.IssueExternalID issueExternalID(String externalID)

Validate.IssueTitle issueTitle(String title)

Validate.IssueVersion issueVersion(String version)

Validate.IssueType issueType(String issueType)

Validate.VotingName votingName(String name)

Validate.VotingStartTime votingStartTime(String startTime, Date validatedStart-Time)

Validate.VotingEndTime votingEndTime(String endTime, Date validatedEndTi-me)

Validate.VotingType votingType(String votingType, Integer validatedVotingTy-pe)

Validate.VoteValue voteValue(String voteValue)

7.5 SendMail

PRIORITEETTI 1

SendMail-luokka vastaa järjestelmän sähköpostien lähettämisestä. Se tarjoaa ohjaimen luokille rajapinnan, jolla lähettää viestejä.

void sendMail(List<String> to, String subject, String message)

- Lähettää annetulla otsikolla varustetun sähköpostin annettuihin osoitteisiin.

7.6 MailTimer

PRIORITEETTI 2

MailTimer-luokka vastaa järjestelmän ajastettujen sähköpostien lähettämisen ajastamisesta. Ajastinluokka pitää yllä rekisteriä järjestelmässä olevista äänestyksistä ja lähettää ajastettuja muistutusviestejä kaikille käyttäjätunnuksille, jotka kuuluvat niihin yliopistoihin, jotka saavat äänestää rekisteröidyssä äänestyksessä, mutta eivät ole vielä äänestäneet. Muistutusviestejä lähetetään äänestyksen alkaessa, äänestyksen puolessa välissä ja kun äänestysaikaa on enää neljäsosa jäljellä. Äänestystä luotaessa tai sen tietojen muuttuessa pitää äänestys rekisteröidä ajastajalle, jotta ajastaja käsittelee kyseisen äänestyksen. Mikäli jonakin muistutusajankohtana äänestys on suljettu, poistettu tai päättymis- tai alkamispäivä on muuttunut, ei ajastaja lähetä muistutusviestejä vaan äänestys poistetaan ajastajan rekisteristä. Mikäli ajastaja ei pääse muistutusajankohtana käsiksi malliin, ei kyseistä muistutusviestiä lähetetä. MailTimerin käyttämät säikeet herätetään aina kun ajastajalle rekisteröityjen äänestysten alkamis- tai päättymisajankohtia muutetaan.

void registerVotingNotification(VotingID voting) throws ControllerException

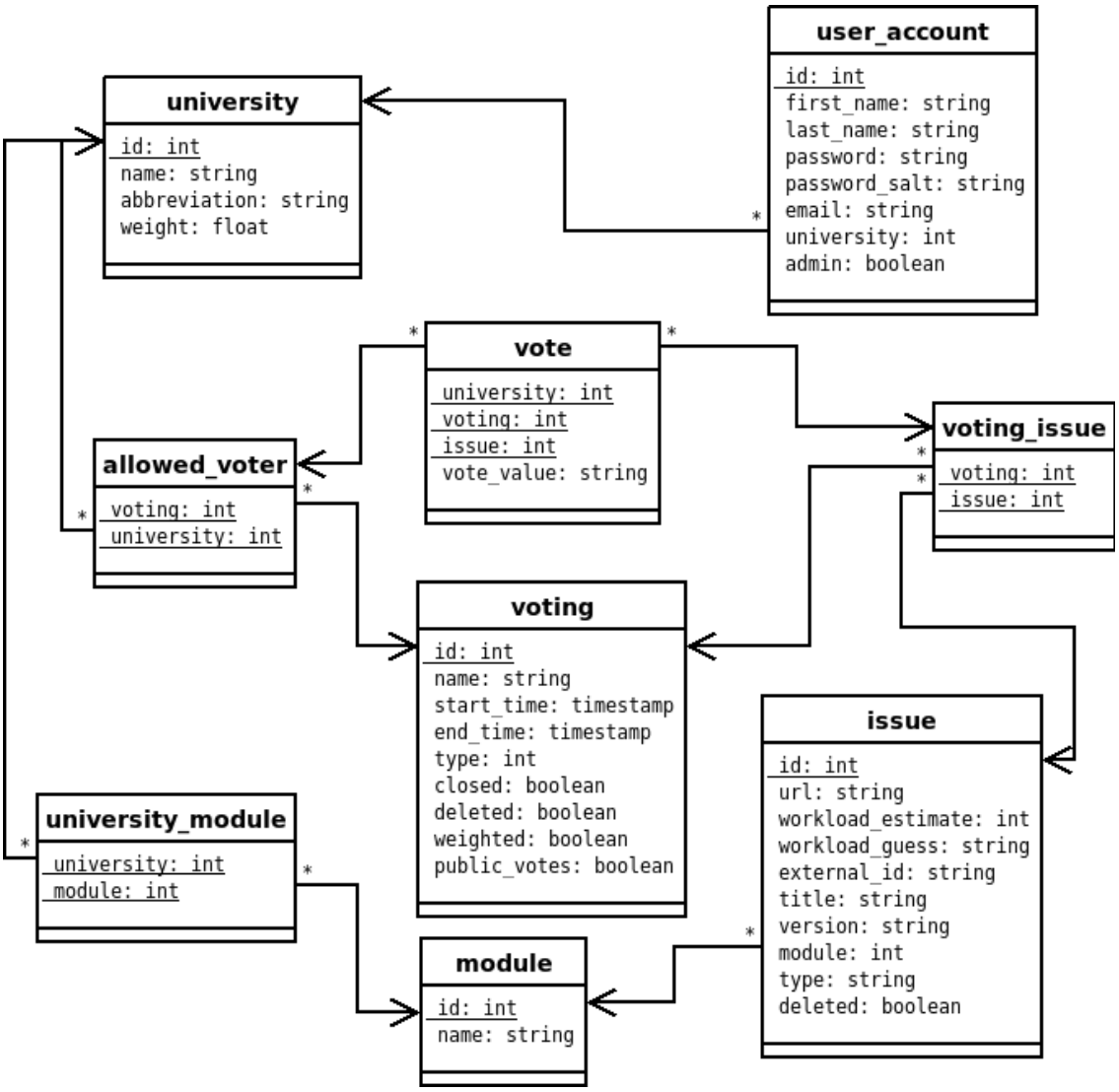
- Rekisteröi uuden tai muuttuneen äänestyksen ajastimelle. Ajastin hakee äänestyksen tiedot mallilta. Erityisesti lähetettävien sähköpostien osoitteet haetaan vasta muistutusajankohtana.

void unregisterVotingNotification(VotinID voting)

- Poistaa rekisteröidyn äänestyksen ajastimen rekisteristä. Tämän jälkeen kyseisestä äänestyksestä ei enää lähetetä muistutusviestejä, ellei äänestystä rekisteröidä uudelleen.

8 Tietokanta

Tässä luvussa kerrotaan tarkemmin järjestelmän tietovarastona toimivasta tietokannasta. Jokainen taulu ja sen merkitys eritellään omassa alaluvussaan. Kuva 6 esittää tietokannan tietokantakaaviona. Tietokannan luontilauseet on esitetty liitteessä 1.



Kuva 6. Tietokantakaavio

8.1 user_account

Taulu user_account kokoaa käyttäjien tiedot yhteen. Käyttäjästä tallennetaan tähän tauluun järjestelmän sisäinen tunniste, etunimi, sukunimi, salasana salattuna, salauksen suolaus, sähköpostiosoite, käyttäjän edustaman yliopiston tunniste numeerisena arvona ja tieto siitä onko käyttäjällä ylläpito-oikeuksia. Taulun pääavaimena toimii käyttäjätunniste ja yliopisto- sarakkeen arvo on viite university-tauluun.

8.2 university

Taulu university kokoaa yliopistotiedot yhteen. Yliopistolla on järjestelmän sisäinen tunniste, nimi, lyhenne ja painoarvo. University- taulun pääavaimena toimii tunniste.

8.3 allowed_voter

Taulu allowed_voter on liitostaulu, joka yhdistää yliopiston niihin äänestyksiin, jossa kyseinen yliopisto saa äänestää. Taulun pääavaimeen tarvitaan liitoksen molemmat osat.

8.4 vote

Taulu vote kertoo kunkin yliopiston äänestyksessä antamat äänet. Äänestä tallennetaan tieto äänen antaneesta yliopistosta, äänestyksestä, johon ääni liittyy, tieto muutospyynnöstä, jota kyseinen äänestyksen arvo koskee, ja tieto varsinaisesta äänen arvosta. Pääavain muodostuu yliopiston tunnisteesta, äänestyksen tunnisteesta ja muutospyynnön tunnisteesta.

8.5 voting_issue

Taulu voting_issue on liitostaulu, joka liittää äänestykseen ne muutospyynnöt, joita voi äänestää kyseisessä äänestyksessä. Taulun pääavaimeen tarvitaan liitoksen molemmat osat.

8.6 voting

Taulu voting kertoo äänestyksen tiedot. Äänestyksestä tallennetaan järjestelmän sisäinen tunniste, nimi, alku- ja loppuaika ja tyyppi. Äänestyksen tyyppi ilmaistaan numerona, ja ohjain tulkitsee numeron ja välittää sitä vastaavan äänestysnäkymän. Lisäksi äänestykseen liitetään tieto onko kyseinen äänestys suljettu, poistettu, lasketaanko sen tulos painotettuna ja onko sen äänet julkisia. Pääavaimena toimii äänestyksen tunniste.

8.7 university_module

Taulu university_module on liitostaulu, joka yhdistää yliopiston johonkin moduuliin, jotta uusien äänestysten luonti helpottuu. Taulun pääavaimeen tarvitaan liitoksen molemmat osat.

8.8 module

Tauluun module tallennetaan moduulin järjestelmän sisäinen tunniste ja sen nimi. Pääavaimena toimii moduulin tunniste.

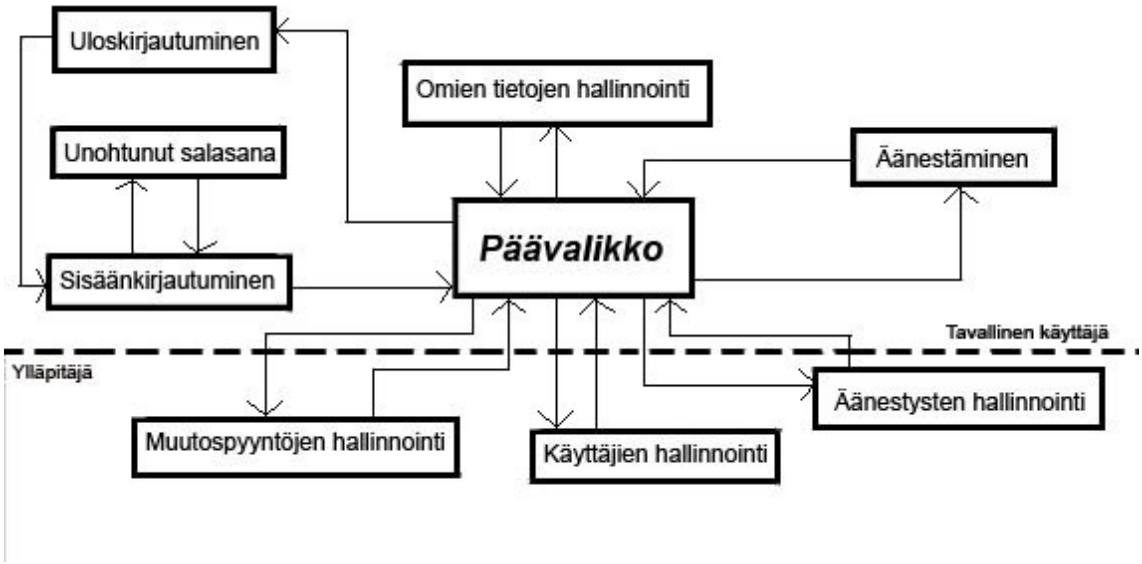
8.9 issue

Taulu issue kokoaa yksittäisten muutospyyntöjen tiedot. Muutospyynnöstä tallennetaan järjestelmän sisäinen tunniste, url, työmääräarvio tai -arvaus, ulkoinen tunniste, otsikko, versio, moduuli, johon se liittyy, tyyppi ja tieto siitä, onko kyseinen muutospyyntö merkitty poistetuksi. Pääavaimena toimii muutospyynnön tunniste.

9 Käyttöliittymä

9.1 Yleistä

Muppett-järjestelmän käyttöliittymä pyritään toteuttamaan siten, että järjestelmä on mahdollisimman miellyttävä ja intuitiivinen käyttää. kuva 7 havainnollistaa JSP-sivujen välisiä yhteyksiä.



Kuva 7. Järjestelmän sivujen väliset yhteydet

9.2 Sivujen sisältö ja käyttötarkoitus

Tässä luvussa selitetään kunkin sivun sisältö ja merkitys yleisellä tasolla. Kunkin JSP-sivun kohdalla on mainittu sivun toteutusprioriteetti.

9.2.1 frontpage.jsp (käyttäjä)

PRIORITEETTI 3

Sivu frontpage avautuu kun käyttäjä kirjautuu järjestelmään. Sivulla toivotetaan käyttäjä tervetulleeksi ja kerrotaan yleistä tietoa järjestelmästä. Kaikki valikko-toiminnot ovat käytettävissä jo heti frontpage-sivulta alkaen.

9.2.2 edit_user.jsp (ylläpitäjä)

PRIORITEETTI 2

Sivu edit_user tarjoaa mahdollisuuden lisätä uuden käyttäjän järjestelmään. Käyttäjän lisääminen näkyy ylläpitäjälle käyttäjähallintavalikon alla. Samalla lomakkeella ylläpitäjä myös muokata käyttäjien aiempia tietoja.

9.2.3 edit_issue.jsp (ylläpitäjä)

PRIORITEETTI 1

Sivu edit_issue tuottaa yksittäisen muutospyynnön lisäyspalvelun. Yksittäisen muutospyynnön lisääminen on muutospyyntöjen hallinta -valikon alla. Samalla lomakkeella voidaan myös muokata yksittäistä muutospyyntöä.

9.2.4 edit_voting.jsp (ylläpitäjä)

PRIORITEETTI 1

Sivua edit_voting käytetään uuden äänestyksen luomiseen. Äänestyksen luomiseen pääsee äänestykset-valikon alta. Samalla lomakkeella voidaan myös muokata jo olemassa olevaa äänestystä.

9.2.5 forgotten_password.jsp (käyttäjä)

PRIORITEETTI 2

Sivun forgotten_password kautta on mahdollista resetoida unohtunut salasana. Uusi salasana lähetetään käyttäjätunnuksen mukaiseen sähköpostiosoitteeseen. Salasanan resetointiin on linkki kirjautumissivulta.

9.2.6 issue_list.jsp (ylläpitäjä)

PRIORITEETTI 4

Sivu issue_list listaa järjestelmässä olevat muutospyynnöt. Listaus hakee tiedot tieto-

kannasta. Listauksesta pääsee muokkaamaan muutospyyntöjä.

9.2.7 login.jsp (käyttäjä)

PRIORITEETTI 1

Sivun login kautta kirjaudutaan järjestelmään käyttäjätunnuksella ja salasanalla. Jos salasana on unohtunut, pääsee sen uusimaan unohtunut salasana -sivun kautta.

9.2.8 logout.jsp (käyttäjä)

PRIORITEETTI 2

Sivu logout ilmoittaa onnistuneesta uloskirjautumisesta. Logout sulkee järjestelmän avaaman session, jotta muut käyttäjät eivät pääse käyttämään luvatta järjestelmää.

9.2.9 navigation.jsp (käyttäjä)

PRIORITEETTI 1

Sivu navigation on järjestelmän päävalikko ja se liitetään kaikkiin sivuihin. Se tarjoaa päävalikon, jonka kautta pääsee navigoimaan kaikkiin järjestelmän osiin ja myös kirjautumaan ulos järjestelmästä. Navigaation sisältö riippuu käyttäjän oikeuksista: ylläpitäjälle valikossa näkyy useampia vaihtoehtoja.

9.2.10 classification_voting.jsp (käyttäjä)

PRIORITEETTI 1

Sivu classification_voting tarjoaa äänestäjälle luokitteluäänestyksen, jossa luokitellaan muutospyynnöt asteikolla todella tärkeän ja ei ollenkaan tärkeän välillä. Valitun äänestyksen tiedot haetaan mallilta.

9.2.11 ordering_voting.jsp (käyttäjä)

PRIORITEETTI 1

Sivu ordering_voting tarjoaa äänestäjälle mahdollisuuden asettaa haluamansa muutospyynnöt tärkeysjärjestykseen. Äänestyksen tiedot haetaan mallilta.

9.2.12 upload_issues.jsp (ylläpitäjä)

PRIORITEETTI 1

Sivu upload_issues tarjoaa ylläpitäjälle mahdollisuuden tuoda CSV-muodossa olevia

uusia muutospyyntöjä järjestelmään. Käyttäjä valitsee tiedoston koneellaan olevista tiedostoista.

9.2.13 user_list.jsp (ylläpitäjä)

PRIORITEETTI 4

Sivu user_list listaa kaikki järjestelmässä olevat käyttäjät. Listaus tarjoaa pääsyn käyttäjien poistoon ja muokkaukseen.

9.2.14 voting_report.jsp (käyttäjä)

PRIORITEETTI 2

Sivu voting_report tuottaa sulkeutuneista äänestyksistä raportin, josta näkee annetut äänet ja äänestyksen lopputuloksen. Palvelu on kaikkien järjestelmässä olevien käyttäjien käytössä, jos ylläpitäjä on asettanut raportin julkiseksi.

9.3 Tietojen syöttäminen

Tässä luvussa on lueteltu kunkin järjestelmässä olevan sivuston kentät, joiden avulla käyttäjä välittää tiedot järjestelmälle. Lisäksi lomakkeissa on piilokenttiä, joiden avulla välitetään esimerkiksi ääniä äänestyssivulta käsiteltäväksi.

9.3.1 edit_user.jsp

- form: create_user
- input-kentät:
- lastname: tekstikenttä, johon syötetään käyttäjän sukunimi
- firstname: tekstikenttä, johon syötetään käyttäjän etunimi
- username: tekstikenttä, johon syötetään käyttäjän käyttäjätunnus eli sähköpostiosoite
- university: select-valinta, jossa valitaan olemassaolevista yliopistoista se, jota käyttäjä edustaa. Tiedot haetaan mallilta.
- rights: radiobutton valinta, jolla määrätään käyttäjän käyttöoikeudet, vaihtoehdot ovat user ja admin
- mail_info: checkbox-valinta, joka merkitään jos halutaan lähettää tiedot käyttäjä-tunnuksen luomisesta käyttäjälle

- user_create: button-tyyppinen lomakkeen lähetysnappi

9.3.2 edit_issue.jsp

-form: create_issue

-input-kentät:

- issue_id: tekstikenttä, johon syötetään muutospyynnön tunniste

- issue_header: tekstikenttä, johon syötetään muutospyynnön otsikko

- work_time: radiobutton-valinta, jolla kerrotaan onko muutokselle esitetty työaika tarkka vai arvio, vaihtoehdot ovat exact ja guess

- working_time: tekstikenttä, johon kerrotaan työaika, kun siitä on tarkkaa tietoa

- working_time_guess: tekstikenttä, johon kerrotaan työaika, kun se on vain arvio

- type: radiobutton-valinta, joka kertoo onko suoritettava työ muutos vaiko korjaus, vaihtoehdot ovat change ja fix

- url: tekstikenttä, johon syötetään muutospyynnön tarkemman kuvauksen web-osoite

- version: tekstikenttä, johon voidaan ilmaista muutospyynnön versio

- module: select-valinta, jolla kerrotaan mihin moduuliin uusi muutospyyntö halutaan liittää. Tiedot haetaan mallilta.

- issue_save: button-tyyppinen lomakkeen lähetysnappi

9.3.3 edit_voting.jsp

- form: create_voting

- input-kentät:

- voting_name: tekstikenttä, johon voidaan syöttää äänestykselle kuvaava nimi

- voting_module: select-valinta, jolla voidaan valita mihin moduuliin äänestys liittyy. Tiedot haetaan mallilta.

- voting_type: select-valinta, jolla osoitetaan minkälainen äänestys on kyseessä, tällä hetkellä vaihtoehdot ovat numeroitu seuraavasti:

1.Luokitteluäänestys

2.Järjestysäänestys

- start_time: tekstikenttä, johon syötetään äänestyksen alkuaika

- end_time: tekstikenttä, johon syötetään äänestyksen loppuaika

- voters: checkbox-valinta, jolla osoitetaan, mitkä yliopistot saavat osallistua äänestykseen. Tiedot haetaan mallilta.

- issues: checkbox-valinta, jolla merkitään mitkä muutospyynnöt otetaan mukaan äänestykseen, vaihtoehdot haetaan kannasta moduuliin liittyen, jokainen saa oman numeerisen valuen

- copy_votes: checkbox-valinta, jolla osoitetaan, että aiemmin pohjaksi kopioidun äänestyksen äänetkin otetaan mukaan uuteen äänestykseen

- voting_save: button-tyyppinen lomakkeen lähetysnappi

9.3.4 forgotten_password.jsp

-form: forgotten_password

-input-kentät:

- user: tekstikenttä, johon käyttäjä syöttää käyttäjätunnuksensa

- password_forgotten:button-tyyppinen lomakkeen lähetysnappi,

9.3.5 issue_list.jsp

- form:issue_list_form

- input-kentät:

- deleted_issues: poistettava muutospyyntö

- issue_delete: button-tyyppinen lomakkeen lähetysnappi

9.3.6 login.jsp

- form: login

- input-kentät:

- user: tekstikenttä, johon käyttäjä syöttää käyttäjätunnuksensa

- password: password-tyyppinen tekstikenttä, johon käyttäjä syöttää salasanansa

- login_button: button-tyyppinen lomakkeen lähetysnappi

9.3.7 classification_voting.jsp

- input-kentät:

    - Haetaan kannasta, jokaista äänestyksessä mukana olevaa muutospyyntöä kohden on viisi radiobutton-tyyppistä kenttää, joilla valitaan arvo 1-5

    - Radiobuttonit saavat nimekseen issueNxM, jossa N on kunkin muutospyynnön numero ja M valittava arvo 1-5.

- classVoteOk: button-tyyppinen lomakkeen lähetysnappi

9.3.8 ordering_voting.jsp

- input-kentät:

    - dragOk: button-tyyppinen form-lähetysnappi

9.3.9 upload_issues.jsp

- form: upload

-input-kentät:

    - file-tyyppinen kenttä, johon annetaan tuotavan tiedoston polku etsimällä se koneelta

    - select-tyyppinen kenttä, josta valitaan mihin moduuliin muutospyynnöt halutaan liittää. Tiedot haetaan mallilta.

- file_upload: button-tyyppinen form-lähetysnappi

9.3.10 user_list.jsp

- form: delete_users

- input-kentät:

    - delete_users: jokaista kannasta haettua käyttäjää kohden on checkbox-tyyppinen kenttä, johon merkitsemällä käyttäjä voidaan poistaa, checkboxeilla juokseva numerointi arvona

- users_delete: button-tyyppinen form-lähetysnappi

9.3.11 voting_report.jsp

- input-kentät:

- file_upload_from_report: button-tyyppinen form-lähetysnappi
- muita buttoneita, jotka tuovat toimintoja:
    - voting_edit: muokkaus
    - csv_load: CSV:n lataaminen

Lähteet

Abs08 http://en.wikipedia.org/wiki/Abstract_factory [25.7.2008]

Com08 http://en.wikipedia.org/wiki/Command_pattern [25.7.2008]

Fro08 http://en.wikipedia.org/wiki/Front_controller [25.7.2008]

GHJ94 Gamma, H., Helm, R., Johnson, R., Vlissides, J.,
Design Patterns, Elements of Reusable Object-Oriented Software.
Addison-Westley, 1994

JSP08 JavaServer Pages http://java.sun.com/products/jsp/ [29.7.2008]

JST08 Java Servlet Technology http://java.sun.com/products/servlet/ [29.7.2008]

KoM05 Koskimies, K., Mikkonen, T., Ohjelmistoarkkitehtuurit.
Talentum, Helsinki, 2005

Lai07 Laine, H., Digitaalisen median tekniikat -kurssin 2007 luentokalvot
Helsingin yliopisto matemaattis-luonnontieteellinen tiedekunta tietojenkäsittelytieteen laitos

Pos08 http://www.postgresql.org/about/ [24.7.2008]

RyM08 Ryhmä Muppett: Chydenius, A., Haverinen, L., Lindén M., Musto, T., Ojala, L., Sormunen, T.,

Vaatimusmäärittelydokumentti: Muutos- ja korjauspyyntöjen priorisointityökalu.
Ohjelmistotuotantaprojekti -kurssin osasuorituksena laadittu julkaisematon dokumentti, Tietojenkäsittelytieteen laitos, Helsinki, 2008

SDN08 Sun Developer Network (SDN), Core J2EE Patterns - Data Access Object.

http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html

[11.7.2008]

SQL08 http://fi.wikipedia.org/wiki/SQL [24.7.2008]

Tom08 http://tomcat.apache.org/ [24.7.2008]

Liitteet

Liite 1. Tietokannan luontilauseet, tietokanta.txt