Kasi

Suunnitteludokumentti

Helsinki 26.03.07

Ohjelmistotuotantoprojekti

HELSINGIN YLIOPISTO

Tietojenkäsittelytieteen laitos

Kurssi

581260 Ohjelmistotuotantoprojekti (9+1op/6ov)

Projektiryhmä

Lauri Holmas

Veli-Pekka Kestilä

Joni Lahtinen

Tuukka Palomäki

Markus Penttilä

Antti-Pekka Sarin

Ilkka Tikkala

Asiakas

Harri Laine

Petri Kutvonen

Vastuuhenkilö

Kimmo Simola

Ohjaaja

Ilari Moilanen

Kotisivu

www.cs.helsinki.fi/group/kasi

Versiohistoria

Versio	Päiväys	Tehdyt muutokset
0.1	16.2.2007	VPK: Aloitettu dokumentin teko.
0.2	18.2.2007	MP: Lisätty alustavat luokkakuvaukset.
0.3	18.2.2007	VPK: asiakasliittymän tekstien parannus.
0.4	19.2.2007	VPK: Lisätty tietokantakuvaus.
0.5	21.2.2007	JKL: Editoitu GUI:n luokkakuvauksia.
0.52	21.2.2007	VPK: Lisätty raportointiliittymän kappale.
0.55	22.2.2007	VPK: Lisätty kuvat dokumentin liite-osaan.
0.56	23.2.2007	TP: Parannettu raportointipuolen kuvausta.
0.59	24.2.2007	MP: Lisätty pakettien nimet, tyylittelyä.
0.60	24.2.2007	MP: Pieniä muutoksia tuoteolioihin
0.61	24.2.2007	AS: Sanasto päivitetty
0.62	24.2.2007	LH:tk-kuvaus ja yhteyskomponentti päivitetty
0.63	25.2.2007	MP: Product muuttui + sekalaista viilausta
0.64	25.2.2007	VPK: Tieto-olioihin muutoksia.
0.65	26.2.2007	MP: Muutin hiukan Main-komponenttia.
0.66	26.2.2007	MP: Sanasto taulukkoon, TK-kaavio, yms.
0.67	26.2.2007	VPK: Korjatut komponentti ja tieto-oliokuvat Korjattu FingerprintAuthentication.
0.68	27.2.2007	MP: Yleistä korjailua, päivitystä
0.69	27.2.2007	TP: Muutettu raporttiliittymää
0.70	1.3.2007	MP: Päivitetty Main ja Settings tarkastuksen mukaisesti
0.71	2.3.2007	MP: Korjattu muita osia tarkastuksen mukaisesti (ei kpl 6, kpl 4 osaksi).
0.82	4.3.2007	VPK: Lisätty GUI-komponentin arkkitehtuuri.
0.83	6.3.2007	MP: Lisätty AuthenticationSettings, päivitetty sen mukaisesti, korjailtu virheitä
0.84	6.3.2007	LH: TK-kaavio ja DBConnection-luokkakaavio korjattu.
0.85	6.3.2007	A-P:Authentication komponentin kuvaus ja kaavio päivitetty.
0.90	7.3.2007	MP: Korvattu GUI:n luokkakuvaukset uusilla
0.91	12.3.2007	MP: Pieniä muutoksia toteutuksen perusteella
0.92	13.3.2007	A-P:Authentication komponentin kuvaukset ja luokkakaavio päivitetty. Sanastoa lisätty.
0.93	13.3.2007	LH: virhe tk-kaaviossa korjattu
0.94	26.3.2007	IT: oikoluku, ulkoasua muokattu

Sisällys

1 Johdanto....1
2 Sanasto....2
3 Arkkitehtuurisuunnitelma....4
4 Tietokanta....6
4.1 Userdata....6
4.2 Groupdata....8
4.3 Finger....8
4.4 Fingerprint....8
4.5 Userbalance....9
4.6 Rawproduct....9
4.7 Productsize....9
4.8 Finalproduct....10
4.9 Productmix....10
4.10 Productin....11
4.11 Productout....11
4.12 Log....11
5 Asiakasliittymän arkkitehtuuri....13
5.1 Komponentit....13
5.1.1 GUI-komponentin alikomponentit....16
5.2 Tieto-oliot....17
5.2.1 UserBalance....18
5.2.2 UserInfo....19
5.2.3 Product....21
5.2.4 ExportProduct....22
5.2.5 ImportProduct....23
5.2.6 ProductGroup....23
5.2.7 CancelProduct....24
5.2.8 GUISettings....25
5.2.9 Language....26
5.2.10 DBSettings....27
5.2.11 FingerprintSettings....28
5.2.12 FPData....29
5.2.13 AuthenticationSettings....30
5.3 Rajapinnat....30
5.3.1 UserInterface....31
5.3.2 UserInterfaceEvent....32
5.3.3 Database....33
5.3.4 Configuration....35
5.3.5 UserAuthentication....36
5.3.6 Fingerprint....36
5.3.7 FingerprintEvent....37
5.3.8 GUI-komponentin alikomponenttien rajapinnat....37
5.3.8.1 ViewEventInterface....38
5.3.8.2 ViewInterface....41
5.3.8.3 RegisterViewInterface....42
5.3.8.4 ShopViewInterface....42
5.3.8.5 ImportViewIterface....42
5.3.8.6 AlertViewInterface....43
5.3.8.7 StartViewInterface....43

5.3.8.8 ConfigViewInterface....44
6 Raportointiliittymän arkkitehtuuri....45
6.1 PHP:n erityisominaisuuksia....45
6.2 Luokkien ja tiedostojen toiminta....45
6.2.1 Istunnonhallinta....45
6.2.2 Kielenvalinta ja kielen vaihtaminen....46
6.3 Raportointiliittymän hakemistorakenne....46
6.3.1 Alihakemistot....46
6.3.2 Tiedostopuun juuressa olevat tiedostot....47
6.4 Raportointiliittymän luokat....51
6.4.1 Language....52
6.4.2 Connection....54
6.4.3 Logger....55
6.4.4 Product....56
6.4.5 User....58
6.4.6 Report....59
7 Asiakasliittymän luokkakuvaukset....60
7.1 MainProgram-komponentti....60
7.1.1 CoffeeTouch....60
7.1.2 MainGuiHandler....62
7.1.3 MainFingerprintHandler....64
7.2 Authentication-komponentti....65
7.2.1 Authentication....66
7.2.2 ConnectionData....67
7.3 Settings-komponentti....68
7.3.1 Settings....68
7.3.2 XML-asetustiedostot....70
7.4 Database-komponentti....70
7.4.1 DBCommunication....71
7.4.2 DBConLostException....73
7.4.3 ProductNotFoundException....73
7.5 FingerprintAuthentication-komponentti....73
7.5.1 FingerprintAuthentication....73
7.5.2 FingerprintCache....74
7.5.3 FingerprintCacheData....75
7.5.4 FingerprintData....76
7.5.5 ReaderListener....77
7.6 GUI-komponentti....77
7.6.1 GuiMain-alikomponentti....77
7.6.1.1 GuiMain....77
7.6.2 StartView-alikomponentti....78
7.6.2.1 StartPanel....79
7.6.2.2 CPanel....79
7.6.2.3 CsPanel....80
7.6.2.4 WarningPanel....80
7.6.2.5 AlertEvent (rajapinta)....81
7.6.3 RegisterView-alikomponentti....81
7.6.3.1 RegisterPanel....82
7.6.3.2 CPanel....82
7.6.3.3 Lower....83
7.6.3.4 Upper....84
7.6.3.5 CnPanel....84

7.6.3.6 RegisterEvent (rajapinta)....85
7.6.4 ConfigView-alikomponentti....85
7.6.4.1 ConfigPanel....85
7.6.4.2 CPanel....86
7.6.4.3 CnPanel....86
7.6.4.4 CcPanel....87
7.6.4.5 CccPanel....87
7.6.4.6 CcwPanel....88
7.6.4.7 CcnPanel....88
7.6.4.8 RightSide....88
7.6.4.9 LeftSide....89
7.6.4.10 FingerEvent (rajapinta)....89
7.6.5 ShopView-alikomponentti....90
7.6.5.1 ShopPanel....90
7.6.5.2 CPanel....90
7.6.5.3 SPanel....91
7.6.5.4 CancelButton....91
7.6.5.5 BuyButton....92
7.6.5.6 BuyEvent (rajapinta)....92
7.6.5.7 CancelEvent (rajapinta)....92
7.6.6 ImportView-alikomponentti....93
7.6.6.1 ImportPanel....93
7.6.6.2 SPanel....93
7.6.6.3 CPanel....94
7.6.6.4 RightSide....94
7.6.6.5 ImportEvent (rajapinta)....95
7.6.7 AlertView-alikomponentti....95
7.6.7.1 CPanel....95
7.6.7.2 AlertPanel....96
7.6.7.3 AlertEvent (rajapinta)....96
7.6.8 General-alikomponentti....96
7.6.8.1 NPanel....97
7.6.8.2 LanguageButton....98
7.6.8.3 ImportButton....98
7.6.8.4 ConfigButton....99
7.6.8.5 LogoutButton....99
7.6.8.6 CancelProductButton....99
7.6.8.7 ViewEvent (rajapinta)....100
7.6.8.8 CancelEvent (rajapinta)....100
7.6.8.9 ImportableList....100
7.6.8.10 GroupList....101
7.6.8.11 ProductList....101
7.6.8.12 ProductChangeEvent (rajapinta)....101
7.6.8.13 ProductList....102
7.6.8.14 ShopableList....102
7.6.8.15 ProductChangeEvent (rajapinta)....102
7.6.8.16 FingerConfigPanel....103
7.6.8.17 NPanel....103
7.6.8.18 CPanel....103
7.6.8.19 FingerButton....104

Liitteet

Liite A: Luokkakaaviot..99

1 Johdanto

Projektin tarkoituksena on tehdä työyhteisön kahvikassa. Valmis tuote tulee korvaamaan järjestelmän, jossa seinälle kiinnitettyyn paperiin on vedetty viivoja juodun kahvimäärän mukaan. Lisäksi projektin tehtävänä on tutustua sormenjälkitunnistuksen mahdollisuuksiin ja toteuttaa valmis komponentti sormenjälkitunnistusten tekemiseen.

Tässä dokumentissa käydään läpi projektin ohjelmistoarkkitehtuuri tarvittavalla tarkkuudella toteutuksen mahdollistamiseksi. Ensin arkkitehtuuri kuvataan yleisellä tasolla ja sen jälkeen esitellään ohjelmiston tietokannan rakenne.

Tietokannan lisäksi ohjelmistossa on kaksi osajärjestelmää. Näistä monimutkaisempi on asiakaskäyttöliittymä. Sen tehtävänä on mahdollistaa tuotteiden ostaminen (tai oikeammin käytön kirjaaminen, koska rahaa ei liiku) ja tuotteiden tuomisesta saatavan hyvityksen kirjaaminen helppokäyttöisesti. Tämän osajärjestelmän kautta myös lisätään ohjelmistoon uusia käyttäjiä. Asiakaskäyttöliittymä käyttää jo rekisteröityjen käyttäjien tunnistamiseen sormenjälkitunnistusta.

Ohjelmiston toinen osajärjestelmä on raportointikäyttöliittymä, joka puolestaan antaa verkkoselaimella käytettävän, HTML-pohjaisen näkymän ohjelman tietovarastoon. Sen kautta käyttäjät pystyvät seuraamaan omaa kahvinkulutustaan ja koko järjestelmän toimintaa. Lisäksi ylläpitokäyttäjät voivat tämän käyttöliittymän kautta muuttaa tarjolla olevaa tuotevalikoimaa ja poistaa käyttäjiä.

Molempien osajärjestelmien tarkempi arkkitehtuuri on kuvattu omissa luvuissaan. Näissä kuvataan niiden komponentit ja komponenttien väliset rajapinnat. Lisäksi kuvataan rajapintojen käyttämät tieto-oliot. Olio- ja komponenttikuvauksien tekemiseen on käytetty UML-kieltä, ja toteutukseen ja luokkasuunnitteluun tarvittavat tiedot on kirjoitettu auki.

Asiakaskäyttöliittymän komponenteista on lisäksi tarkat luokkakaaviot, joista selviää tarkemmin komponenttien sisäinen toteutus. Raportointikäyttöliittymän toisenlaisesta toteutustavasta johtuen siitä ei ole erillisiä luokkakaavioita, vaan kaikki toteutuksen kannalta tarpeellinen tieto on esitetty komponenttikuvauksen yhteydessä.

2 Sanasto

Käsite	*Selitys*
aktiivikäyttö	tuote on näkyvissä käyttöliittymässä, kun se on merkattu tietokannassa olevan aktiivikäytössä
asetustiedosto	tietyn muotoinen XML-tiedosto, josta asiakasliittymä hakee asetuksensa
asiakasliittymä / asiakaskäyttöliittymä	osajärjestelmä, jonka kautta käyttäjät pystyvät ostamaan ja tuomaan tuotteita kahvihuoneessa
csv-vienti	mahdollisuus tuottaa järjestelmän raporteista tiedosto, jossa kentät on eroteltu toisistaan erotinmerkillä
Event Interface	tapahtumarajapinta
Factory-olio	Olio, joka on tarkoitettu uusien olioiden luomiseen
Griaule GrFinger	ulkopuolinen Java-kirjasto sormenjälkitunnistimen käyttöä varten
JDBC-rajapinta	Java Database Connectivity, tarjoaa metodit tietokantakyselyihin
käyttäjäsaldo	käyttäjän tuoteryhmäkohtainen saldo, joka kertoo tuontien ja vientien välisen tasapainon
ostettava tuote	järjestelmästä ostettavissa olevat tuotteet, esim. kahvi maidolla tai espresso
raportointiliittymä / raportointikäyttöliittymä	osajärjestelmä, joka on tarkoitettu käyttäjien ja tuotteiden hallinnointiin, sekä raporttien tuottamiseen
tuotava tuote	tuote, jolla käyttäjä voi hyvittää ostoksiaan, esim. paketillinen kahvia tai tölkillinen maitoa
TKTL	Helsingin yliopiston tietojenkäsittelytieteen laitos
tuotehälytys	tuotteelle asetettava hälytys, joka ilmaisee tuotteen olevan loppumaisillaan
tuoteryhmä	usean vientituotteen muodostama kokonaisuus, jonka sisältämien tuotteiden arvot eroavat muista tuoteryhmistä
tuontiarvo	järjestelmään tuotavasta tuotteesta omalle tilille saatava hyvitys perusyksikössä (kahvikupeissa)
tuontitapahtuma	käyttäjän järjestelmään tekemä hyvitys, joka kasvattaa hänen saldoaan tuoteryhmässä
vakiotuote	käyttäjän itselleen asettama oletustuote

Käsite	*Selitys*
vientitapahtuma	käyttäjän järjestelmällä tekemä ostos, joka vähentää hänen saldoaan tuoteryhmässä

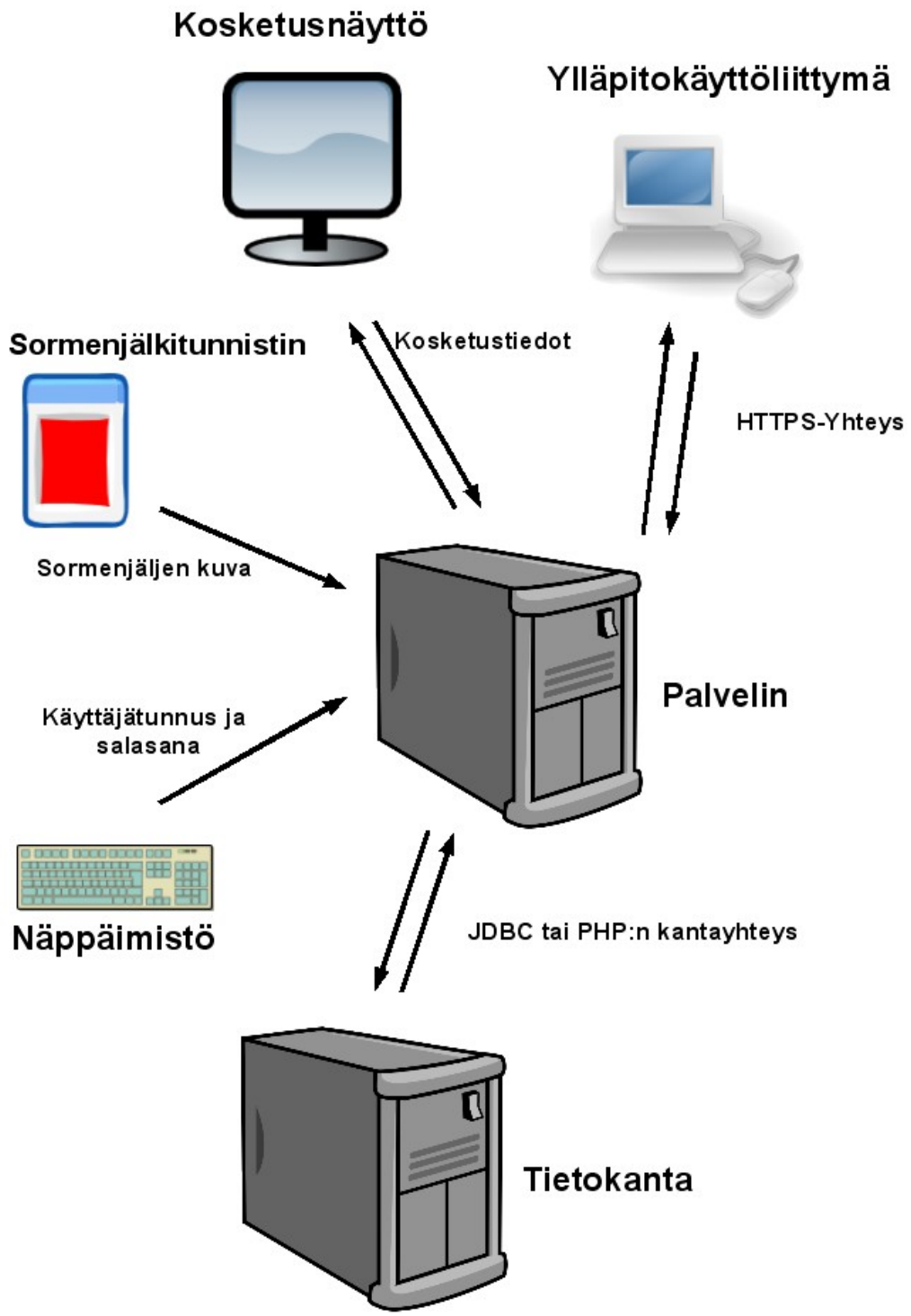
3 Arkkitehtuurisuunnitelma

Ohjelmistossa on kolme osajärjestelmää. Nämä osajärjestelmät ovat tietokanta, asiakasliittymä ja raportointiliittymä. Tietokanta toteutetaan siten, että se ei ole sidoksissa mihinkään määrättyyn tietokantajärjestelmään. Toteutusvaiheessa tämä pyritään varmistamaan siten, että tietokanta toimii PostgreSQL ja Oracle -kantamoottoreiden kanssa.

Asiakasliittymän tehtävänä on toimia kahvihuoneen näkymänä järjestelmään. Sen kautta käyttäjät pystyvät ostamaan tuotteita ja tuomaan tuotteita järjestelmään. Tämän järjestelmän osan toteutukseen käytetään Java-ohjelmointikieltä. Asiakaskäyttöliittymä keskustelee tietokannan lisäksi myös kosketusnäytön (joka korvaa hiiren osoitinlaitteena), sormenjälkilukijan ja näppäimistön kanssa. Kosketusnäyttöä käytetään pääasiallisena syöttö ja osoitinlaitteena. Sormenjälkilukijan avulla käyttäjä tunnistetaan järjestelmässä, ja näppäimistön avulla käyttäjä syöttää käyttäjätunnuksensa ja salasanansa rekisteröimisvaiheessa, jotta käyttäjä saadaan yhdistettyä Tietojenkäsittelytieteen laitoksen käyttäjätietoihin myöhempiä tunnistustarpeita varten.

Raportointikäyttöliittymä on toteutettu PHP-ohjelmointikielellä ja toimii Apache http-palvelimen päällä. Sen avulla käyttäjät pääsevät perumaan vääriä ostoksia ja katselemaan raportteja osto- ja tuontitapahtumistaan. Lisäksi raportointikäyttöliittymän kautta järjestelmän ylläpitäjät pystyvät poistamaan järjestelmästä käyttäjiä, ja sekä lisäämään että poistamaan tuoteryhmiä ja tuotteita.

Kuvassa 1. esitellään ohjelmiston toimintaympäristö ja sen eri osat. Lisäksi kerrotaan minkälaisia syötteitä tai protokollia ohjelmiston osien ja toimintaympäristön välillä on. Ohjelmiston tietovarasto sijaitsee tietokantakoneella ja ohjelmakomponentit sijaitsevat palvelimella. Palvelimeen on myös yhdistetty kosketusnäyttö, sormenjälkitunnistin ja näppäimistö. Näiden avulla asiakas ohjaa ohjelmiston toimintaa kahvihuoneessa. Ylläpitokäyttöliittymään liitytään normaalin www-selaimen avulla käyttäen HTTPS-protokollaa.



Kuva 1: Yleiskuva ohjelmiston arkkitehtuurista.

4 Tietokanta

Tässä luvussa kuvataan ohjelmiston tietokanta. Siinä on esitelty jokainen taulu ja taulujen kentät. Tietokanta on suunniteltu mahdollisimman kantariippumattomaksi, mutta tietokannan yksinkertaistamiseksi käytetään joitakin tietokantojen ominaisuuksia, jotka täytyy toteuttaa kantariippuvaisesti.

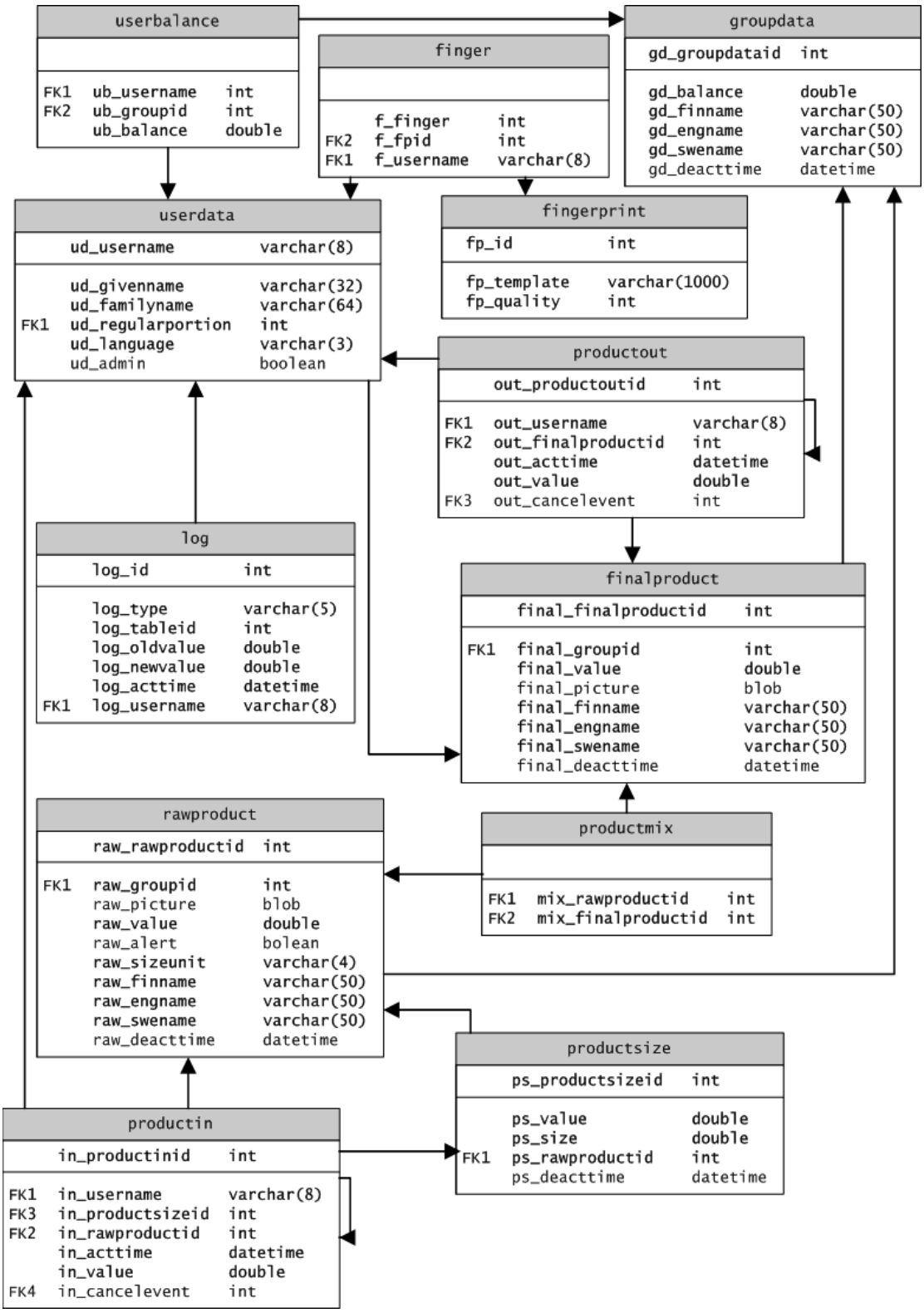
Tästä kantariippuvuudesta johtuen on laadittu erilliset kannanluontilauseet kohdetietokannoiksi tarkoitetuille PostgreSQL:lle ja Oraclelle. Normaalit haut ja lisäykset pyritään toteuttamaan siten, että niihin ei tarvitse tehdä muutoksia, jos tietokantaa vaihdettaan. Jos lauseita kuitenkin joudutaan muuttamaan, ne on tallennettu ohjelmiston asetustiedostoihin, joissa niiden editoiminen on helppoa, ja ne ovat heti käytettävissä ohjelman uudelleenkäynnistyksen jälkeen.

Kuvassa 2 esitellään myös tietokantakaavio, josta on helpompi saada käsitys taulujen riippuvaisuussuhteista.

4.1 Userdata

Taulu sisältää ohjelman käyttäjätiedot. Näihin tietoihin kuuluu esimerkiksi käyttäjätunnus ja käyttäjän etu- ja sukunimi. Lisäksi käyttäjästä talletetaan hänen oletuskielensä ja vakiotuotteensa. Myös tieto siitä, onko käyttäjä järjestelmän ylläpitäjä vai ei, on tallennettu tähän tauluun.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
ud_username	Varchar(8)	X	TKTL:n käyttäjätunnus, pääavain
ud_givenname	Varchar(32)	X	Käyttäjän etunimi
ud_familyname	Varchar(64)	X	Käyttäjän sukunimi
ud_regularportion	Int	X	Käyttäjän valitsema vakiotuote. Viite finalproduct-tauluun
ud_language	Varchar(3)	X	Käyttäjän valitsema vakiokieli.
ud_admin	Boolean		Tieto käyttäjän ylläpitäjäoikeuksista



Kuva 2: Tietokantakaavio

4.2 Groupdata

Taulu sisältää tiedot ohjelman tuoteryhmistä. Jokaisella tuoteryhmällä on ID-numero, jonka arvoa tietokanta kasvattaa automaattisesti, kun tauluun lisätään rivejä. Muita taulun tietoja ovat tuoteryhmän yleinen saldo, joka kertoo erotuksen siitä, paljonko tuoteryhmään on tuotu tuotteita, ja paljonko niitä on käytetty vakioannoksina. Lisäksi tuoteryhmästä on sen nimi suomeksi, ruotsiksi ja englanniksi. Käytöstäpoistopäiväyksellinen tuoteryhmä ei näy käyttäjälle asiakaskäyttöliittymässä. Jos käytöstäpoistopäiväys on tyhjä, on tuoteryhmä aktiivikäytössä.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
gd_groupdataid	Int	X	Pääavain, kasvatetaan automaattisesti
gd_balance	Double	X	Tuoteryhmän yhteinen saldo
gd_finname	Varchar(50)	X	Tuoteryhmän nimi suomeksi
gd_engname	Varchar(50)	X	Tuoteryhmän nimi englanniksi
gd_swename	Varchar(50)	X	Tuoteryhmän nimi ruotsiksi
gd_deacttime	Datetime		Käytöstäpoistopäiväys

4.3 Finger

Taulu sisältää tiedon siitä, kenelle mikäkin sormenjälki kuuluu ja mihin sormeen se on liitetty.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
f_finger	Int	X	0-4, eli mikä sormi on kyseessä
f_fpid	Int	X	Viite fingerprinttiin.
f_username	Varchar(8)		Käyttäjä, viiteavain userdata-tauluun

4.4 Fingerprint

Taulu sisältää sormenjäljen tiedot. Taulussa on lisäyksien myötä automaattisesti kasvava pääavain, joka yksilöi jokaisen tallennetun sormenjäljen. Sormenjäljistä tallennetaan tunnistustieto ja tieto sen laadusta.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
fp_id	Int	X	Pääavain, kasvatetaan automaattisesti
fp_template	Varchar(1000)	X	Sormenjäljen tunnistustieto
fp_quality	Int	X	Tunnistustiedon laatu

4.5 Userbalance

Taulussa on tiedot käyttäjien saldoista eri tuoteryhmissä. Taulua päivitetään, kun käyttäjä tuo tuotteita järjestelmään tai vie niitä järjestelmästä.. Taulussa on kaksi viiteavainta, toinen käyttäjään ja toinen tuoteryhmään. Lisäksi on kyseisen ryhmän saldotieto.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
ub_username	Varchar(8)	X	Käyttäjä, viite userdata-tauluun
ub_groupid	Int	X	Tuoteryhmä, viite groupdata-tauluun
ub_balance	Double	X	Käyttäjän saldo kyseisessä tuoteryhmässä

4.6 Rawproduct

Taulu sisältää tiedot tuotavista tuotteista. Pääavain yksilöi tuotteet tietokannassa ja lisäksi jokainen tuote on yhdistetty johonkin tuoteryhmään. Taulussa on tieto siitä, onko tuotteella loppumishälytys ja mitä laatua tuotekoot ovat (l, g, kg, jne.). Tuotteen arvon laskemiseksi taulussa on tuotteen perustuontiarvo. Tuotavien tuotekokojen tuontiarvot lasketaan kertoimen ja perustuontiarvon mukaan. Käytöstäpoistopäiväyksellinen tuote ei näy käyttäjille asiakaskäyttöliittymässä. Jos päivämäärä on tyhjä, tuote on aktiivikäytössä.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
raw_rawproductid	Int	X	Pääavain, kasvatetaan automaattisesti
raw_groupid	Int	X	Tuoteryhmä, viite groupdata-tauluun
raw_picture	blob		Sisältää tuotteen kuvan.
raw_value	Double	X	Tuotteen tuontiarvo
raw_alert	Boolean		Tieto onko tuotteessa loppumishälytys
raw_sizeunit	Varchar(4)	X	Tuotteen koon määre (esim. kg, l, g)
raw_finname	Varchar(50)	X	Tuotteen nimi suomeksi
raw_engname	Varchar(50)	X	Tuotteen nimi englanniksi
raw_swename	Varchar(50)	X	Tuotteen nimi ruotsiksi
raw_deacttime	Datetime		Käytöstäpoistopäiväys

4.7 Productsize

Taulu sisältää tiedot yhdestä tuotavaan tuotteeseen liittyvästä pakkauskoosta. Taulussa on pääavainkenttä, jonka luku kasvaa automaattisesti rivilisäyksen yhteydessä. Lisäksi taulussa on tieto kertoimesta, jolla tietyn koon tuontiarvo lasketaan tuotteen perustuontiarvosta, ja paketin tuotekoko. Käytöstäpoistopäiväyksellinen tuotekoko ei näy käyttäjille asiakaskäyttöliittymässä. Jos päivämäärä on tyhjä, on tuotekoko aktiivikäytössä.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
ps_productsizeid	Int	X	Pääavain, kasvatetaan automaattisesti
ps_value	Double	X	Tuotteen arvo kertoimena. Varsinainen arvo lasketaan kertomalla sen tuotteen arvo, johon tuotekoko liittyy, tällä kertoimella.
ps_size	Double	X	Paketin tuontikoko
ps_rawproductid	Int	X	Tuote, viite rawproduct-tauluun
ps_deacttime	Datetime		Käytöstäpoistopäiväys

4.8 Finalproduct

Taulu sisältää tiedot ostettavista tuotteista. Jokaisella tuotteella on sen yksiselitteisesti tunnistava pääavain, jonka kanta luo tuotteen lisäämisen yhteydessä. Lisäksi tuotteessa on tieto siitä, mihin tuoteryhmään se kuuluu, ja muut tuotteen tiedot, kuten sen ostoarvo, kuva ja nimi suomeksi, ruotsiksi ja englanniksi. Käytöstäpoistopäiväyksellinen tuote ei näy käyttäjille asiakaskäyttöliittymässä. Jos päivämäärä on tyhjä, on tuote aktiivikäytössä.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
final_finalproductid	Int	X	Pääavain, kasvatetaan automaattisesti
final_groupid	Int	X	Tuoteryhmä, viite groupdata-tauluun
final_value	Double	X	Tuotteen hinta (normaalisti **1**)
final_picture	Blob		Kuva tuotteesta
final_finname	Varchar(50)	X	Tuotteen nimi suomeksi
final_engname	Varchar(50)	X	Tuotteen nimi englanniksi
final_swename	Varchar(50)	X	Tuotteen nimi ruotsiksi
final_deacttime	Datetime		Käytöstäpoistopäiväys

4.9 Productmix

Taulussa kuvataan mitä raakatuotteita, eli tuotavia tuotteita (esim. kahvi, maito), johonkin valmiiseen tuotteeseen kuuluu. Taulussa on siis viittaukset kummankin tyyppisten tuotteiden pääavaimiin.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
mix_rawproductid	Int	X	Raakatuote, viite rawproduct-tauluun
mix_finalproductid	Int	X	Valmistuote, viite finalproduct-tauluun

4.10 Productin

Taulu sisältää tapahtumakirjaukset käyttäjien tuonneista. Sen perusteella tuotavien tuotteiden listaa voidaan muokata sopivaan järjestykseen käyttäjälle. Lisäksi tämän perusteella voidaan tehdä raportteja käyttäjien tuonneista ja tuontitottumuksista.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
in_productinid	Int	X	Pääavain, kasvatetaan automaattisesti
in_username	Varchar(8)	X	Käyttäjä, viite userdata-tauluun
in_productsizeid	Int	X	Tuotekoko, viite productsize-tauluun
in_rawproductid	Int	X	Tuote, viite rawproduct-tauluun
in_acttime	Datetime	X	Tapahtuma-aika
in_value	Double	X	Tapahtuman arvo, saldoihin tehty muutos

4.11 Productout

Taulu sisältää tapahtumakirjaukset käyttäjien ostoista. Tästä taulusta lasketaan kuinka monta ostoa käyttäjä on tehnyt eri tuotteista, ja tämän tiedon perusteella järjestetään ostettavat tuotteet sellaiseen järjestykseen, että käyttäjän useimmin ostama tuote näkyy käyttöliittymässä listan ensimmäisenä ja harvimmin ostama viimeisenä. Lisäksi taulu mahdollistaa tuotteiden ostojen perumiset ja erilaisten raporttien teon käyttäjien käyttötottumuksista.

Kentän nimi	***Tyyppi***	***Not null***	***Kuvaus***
out_productoutid	Int	X	Pääavain, kasvatetaan automaattisesti
out_username	Varchar(8)	X	Käyttäjä, viite userdata-tauluun
out_finalproductid	Int	X	Tuote, viite finalproduct-tauluun
out_acttime	Datetime	X	Tapahtuma-aika
out_value	Double	X	Tapahtuman arvo, saldoihin tehty muutos
out_cancelevent	Int		Mahdollinen peruttava tapahtuma, viite productout-tauluun. Jos tyhjä niin normaalitapahtuma, jos olemassa niin peruutustapahtuma jolla korvataan viitattu tapahtuma.

4.12 Log

Taulu sisältää tapahtumakirjauksia arvomuutoksista tuonti- ja vientituotteissa, pakkauskoissa sekä tuoteryhmissä. Taulussa lukee log_type-kentässä tekstimuodossa mihin tauluun viitataan, sekä log_tableid-kentässä kyseisen taulun tunnus. Tauluun talletetaan myös tieto kuka teki muutoksen ja milloin.

Kentän nimi	*Tyyppi*	*Not null*	*Kuvaus*
log_id	Int	X	Pääavain, kasvatetaan automaattisesti
log_type	Varchar(5)	X	Tekstimuotoinen tieto mihin tauluun viitataan: raw, final, size tai group
log_tableid	Int	X	Viitattavan taulun tunnus
log_oldvalue	Double	X	Muutetun vanha arvo
log_newvalue	Double	X	Muutetun uusi arvo
log_acttime	Datetime	X	Tapahtuma-aika
log_username	Varchar(8)	X	Käyttäjä, viite userdata-tauluun

5 Asiakasliittymän arkkitehtuuri

5.1 Komponentit

Asiakasliittymässä on kuusi komponenttia, jotka on esitelty kuvassa 3. Näistä kuudesta MainProgram-komponentti hoitaa kaiken tiedonkulun muiden asiakasliittymän osien kesken. Loput viisi puolestaan hoitavat ohjelman toimintojen ja ulkoisten rajapintojen toteuttamisen. Nämä viisi muuta komponenttia ovat GUI, Authentication, Settings, Database ja FingerprintAuthentication.

MainProgram-komponentti sovittaa muut erilliset komponentit toisiinsa ja helpottaa niiden toteuttamista toisistaan erillisinä osina. Muiden apuna olevat Database ja Settings -komponenteista käyttävät kuitenkin esimerkiksi muiden komponenttien määrittelemiä asetusluokkia asetusten välittämiseen, joten täysin erillisiä ne eivät ole. Database-komponentin rajapinnoissa on suoraan jokaiselle kantahaulle tehty oma metodi, joka palauttaa kyselyn antaman tiedon oikeassa muodossa.

GUI-komponentti on ohjelman suurin ja monimutkaisin osa. Sen tehtävänä on toteuttaa käyttäjälle näkyvä ohjelman käyttöliittymä ja muuntaa käyttäjän ohjelmalle antamat käskyt sellaiseen muotoon, että ohjelman muiden osien on niitä helppo käsitellä. Lisäksi GUI-komponentti pitää kirjaa ohjelman sen hetkisestä tilasta ja reagoi sen mukaan myös MainProgram-komponentin kautta tuleviin sormenjälkien tunnistustapahtumiin.

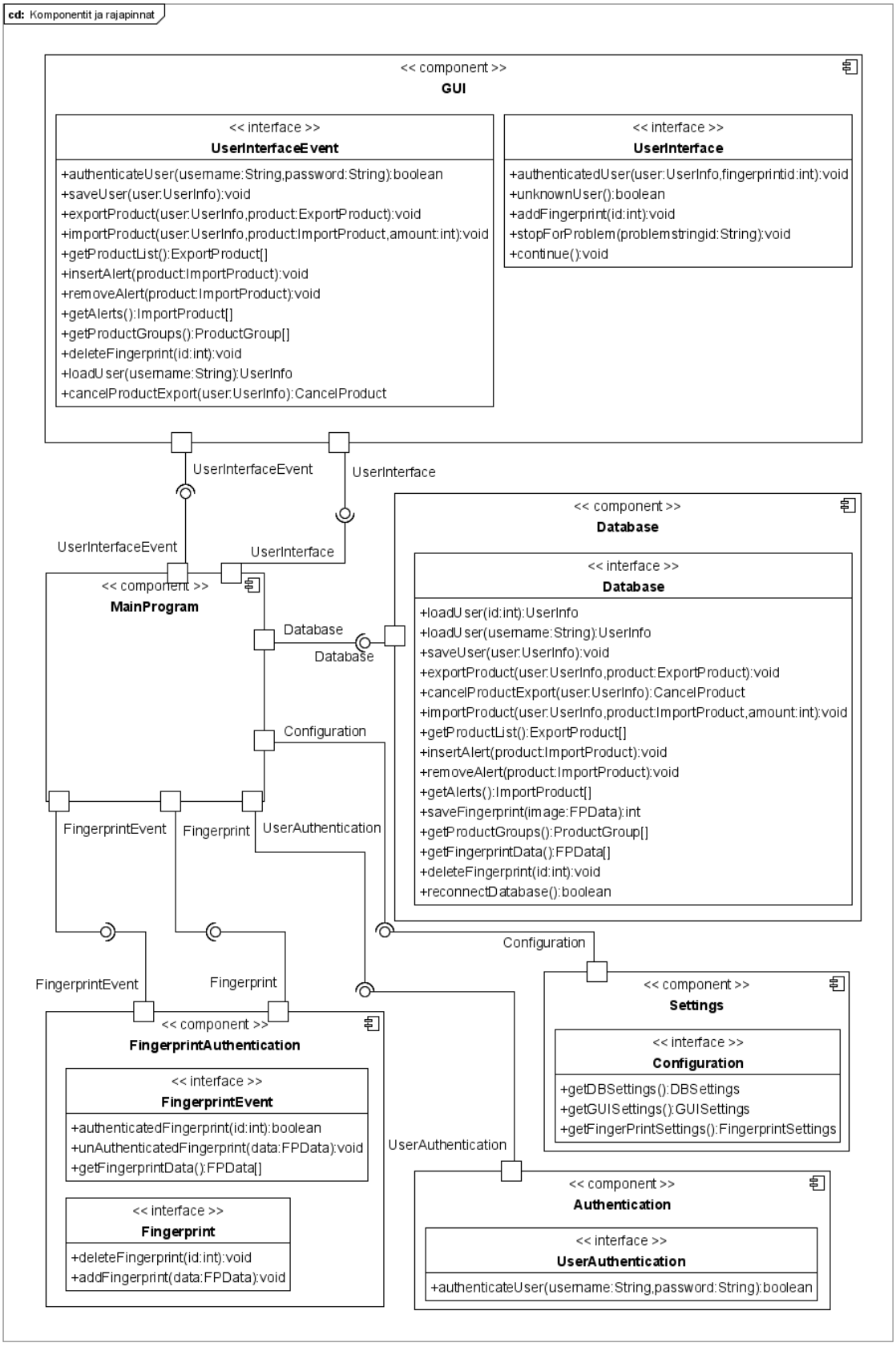
Authentication-komponentin tehtävä on liittyä käyttöjärjestelmään tai muuhun vastaavaan käyttäjätietojen lähteeseen, jonka kautta käyttäjä pystytään tunnistamaan (TKTL:n) käyttäjätunnuksen ja salasanan avulla.

Settings-komponentin tehtävänä on hakea ohjelman tarvitsemat asetukset tiedostojärjestelmässä sijaitsevasta asetustiedostosta. Asetustiedoston sijainti voidaan määrittää komentoriviltä, tai jos sitä ei ole määritelty, komponentti käyttää ohjelmakoodissa lukevaa vakiopaikkaa tiedoston sijainnille. Lisäksi komponentti käyttää GUI, Database ja FingerprintAuthentication -komponenttien määrittämiä asetusluokkia.

Database-komponentin tehtävänä on toimia ohjelman ja ohjelmiston tietokannan välisenä rajapintana. Sen rajapinnassa on jokaiselle kantahaulle tehty oma metodi, joka palauttaa kyselyn antaman tiedon oikeassa muodossa. Komponentti ottaa yhteyttä tietokantaan JDBC-rajapinnan kautta. Tietokannasta haettava tieto asetetaan myöhemmin tässä dokumentissa esiteltäviin tieto-olioihin.

FingerprintAuthentication-komponentti kuuntelee ohjelman käyttämää sormenjälkilukijaa. Tätä tarkoitusta varten se käyttää hyödykseen Griaulen GrFingerJava-kirjastoa ja -ajuria. Komponentti saa sormenjälkitunnistimelta ilmoituksia

uudesta luetusta sormenjäljestä. Tämän jälkeen se vertaa luettua sormenjälkeä sillä varastossa oleviin sormenjälkitunnistustietoihin, ja jos tunnistus onnistuu, lähettää MainProgram-komponentille sormenjäljen ID-numeron. Muussa tapauksessa lähetetään sormenjäljen tunnistustieto sen mahdollista tallentamista varten.



Kuva 3: Komponentit ja rajapinnat

5.1.1 GUI-komponentin alikomponentit

Ohjelman GUI-komponentti on jaettu kuvan 4. mukaisesti 8-alikomponenttiin. Niistä GuiMain-komponentti on GUI-komponentin sydän. General-komponentissa on muiden komponenttien yhdessä käyttämiä luokkia. Muut komponentit on puolestaan jaettu sen mukaan, että minkä ohjelman näytön ne toteuttavat. Tämä mahdollistaa uusien näyttöjen lisäämisen tarvittaessa ohjelmaan mahdollisimman pienin muutoksin.

GUI-Komponentin kaikki "älykäs"-toiminnallisuus on toteutettu GuiMain-alikomponentissa. Tämän alikomponentin tehtävänä on olla GUI-komponentin muiden osien rajapintana itse pääohjelmaan. Se toteuttaa ViewEventInterface rajapinnan ja pitää sisällään käyttöliittymän tilatiedot.

StartView-alikomponentti toteuttaa käyttäjälle ensimmäisenä näkyvän osan ohjelmasta. Tämän komponentin toteuttamassa näkymässä käyttäjä voi aloittaa rekisteröitymisen ohjelman käyttäjäksi, aloittaa tuotteen loppumisesta kertovan varoituksen lisäämisen, poistaa tuotteen loppumisen varoitus tai sormenjäljen antamalla kirjautua sisään ohjelmaan. Toteuttaa StartViewInterface-rajapinnan.

RegisterView-alikomponentti mahdollistaa uuden käyttäjän rekisteröitymisen. Siinä käyttäjä kirjoittaa TKTL:n käyttäjätunnuksen ja salasanansa. Jos nämä hyväksytään, niin sen jälkeen käyttäjälle annetaan mahdollisuus antaa itselleen etunimi ja sukunimi järjestelmään. Näiden tietojen antamisen jälkeen käyttäjä siirretään normaaliin ConfigView-alikomponentin näkymään. Toteuttaa RegisterViewInterface-rajapinnan

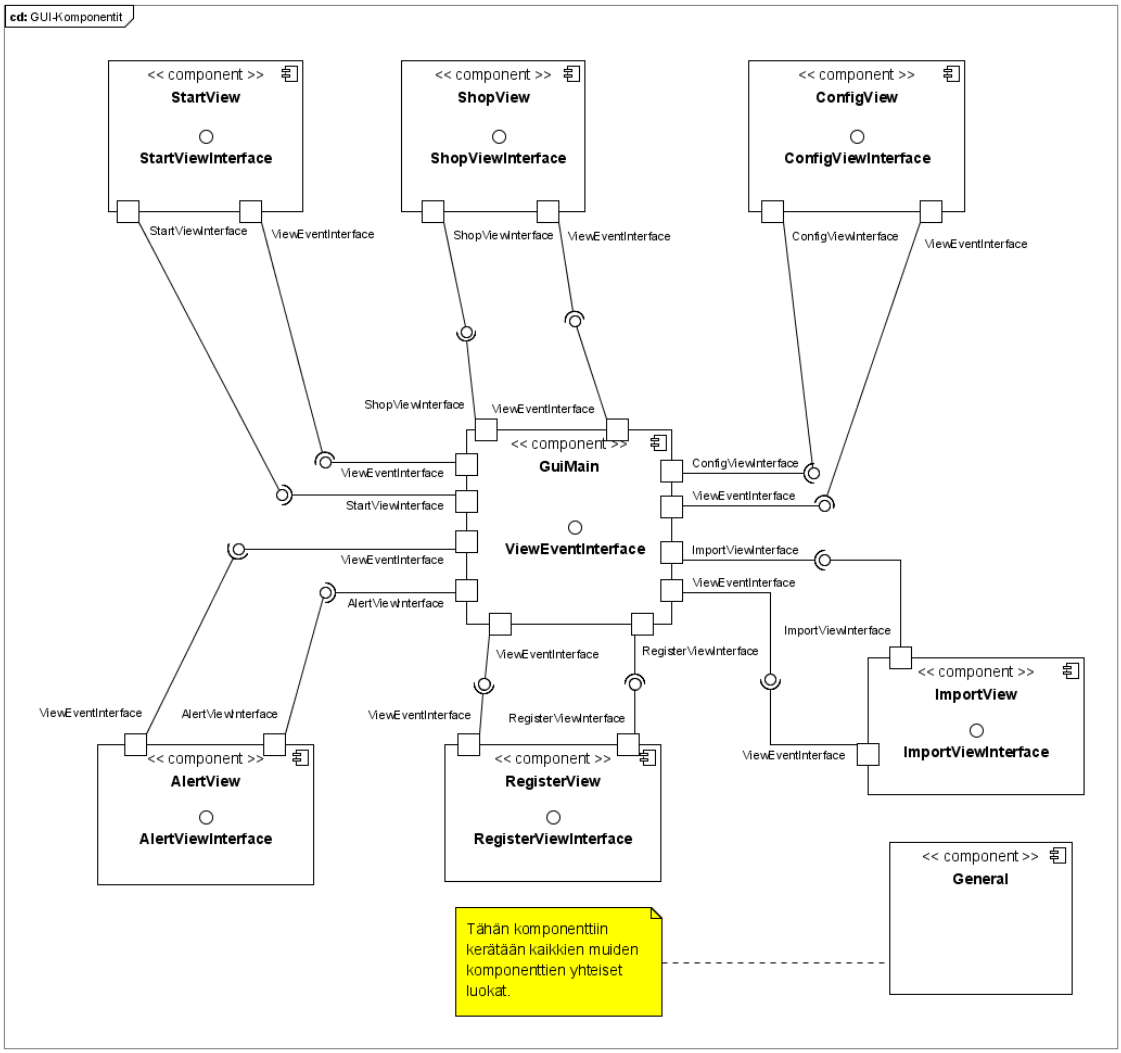
ConfigView-alikomponentin tuottama näkymä mahdollistaa käyttäjän käyttäjätietojen muuttamisen ohjelmassa. Sen avulla käyttäjä voi vaihtaa tuotetta, joka häneltä normaalisti veloitetaan ostoliittymässä. Ja lisätä itselleen uusia tai korvata vanhoja sormenjälkiä, jotka mahdollistavat käyttäjän sisäänkirjautumisen sormenjälkilukijan avulla. ConfigView-alikomponentin näkymästä käyttäjä voi siirtyä tuotteiden tuonti tai ostonäkymään. Alikomponentti toteuttaa ConfigViewInterface-rajapinnan.

ShopView-alikomponentti tuottaa näkymän, jonka kautta käyttäjä voi merkitä ostaneensa tuotteita järjestelmästä. Näkymä tulee näkyville, kun käyttäjä joko syöttää ohjelman alkuruudussa sormenjälkensä tai siirtyy siihen jostain muusta näkymästä nappia painamalla. Jos käyttäjä on syöttänyt sormenjälkensä alkuruudussa ja ei tee mitään muuta tämän jälkeen tai kirjautuu ulos järjestelmästä. Veloitetaan häneltä asetuksissa määritelty vakiotuote. Tämän komponentin näkymästä käyttäjä voi siirtyä tuotteiden tuonti tai omat tiedot näkymiin. Alikomponentti toteuttaa ShopViewInterface-rajapinnan.

ImportView-alikomponentin tuottaman näkymän kautta käyttäjät pystyvät merkitsemään tiedon siitä, että ovat tuoneet tuotteita. Nämä merkinnät lisäävät sitten käyttäjän saldoa järjestelmässä. Näkymästä pääsee osto ja omat tiedot näkymiin. Alikomponentti toteuttaa ImportViewInterface-rajapinnan.

AlertView-alikomponentti tuottaa näkymän, jossa käyttäjä pystyy lisäämään varoituksen jokin tuotavan tuotteen loppumisesta. Käyttäjän lisättyä varoituksen ohjelma siirtyy takaisin aloitusnäkymään. Alikomponentti toteuttaa AlertViewInterface-rajapinnan.

General-alikomponenttiin on lisätty muiden komponenttien yhteisesti tarvitsemia luokkia. Näitä ovat esimerkiksi NPanel ja LanguageButton, joita monet alikomponenteista käyttävät. Komponentilla ei ole omaa toiminnallisuutta ja se toimiikin lähinnä luokkavarastona.



Kuva 4: GUI-Komponentin alikomponentit

5.2 Tieto-oliot

Asiakasliittymän eri osien välistä tiedonkulkua varten ohjelmassa on tieto-olioita. Näiden tehtävänä on tiedon välittäminen komponentista toiseen sellaisessa muodossa, että tietoon liittyvät riippuvuussuhteet pysyvät helposti hallittavina. Ohjelmasta löytyvät seuraavissa luvuissa esiteltävät tieto-oliot. Tieto-olioiden luokat on laitettu niille läheisen järjestelmän komponentin kanssa samaan pakettiin, jotka on mainittu kunkin luokan kohdalla.

Tieto-oliot on esitelty UML-luokkakaaviona dokumentin liitteessä (liite A, kuva 1).

5.2.1 UserBalance

Paketti: gui

Luokka on UserInfo-luokan apuluokka, joka pitää sisällään käyttäjän saldon järjestelmässä olevassa tuoteryhmässä.

private String[] group;

Sisältää tuoteryhmän nimen eri kielillä.

private int balance;

Käyttäjän saldo kyseisessä tuoteryhmässä.

public static final int FINNISH = 0;

Suomenkielisen nimen paikka nimitaulukossa.

public static final int ENGLISH = 1;

Englanninkielisen nimen paikka nimitaulukossa.

public static final int SWEDISH = 2;

Ruotsinkielisen nimen paikka nimitaulukossa.

Metodit

public UserBalance(String[] group, int balance);

Konstruktori ottaa parametreina vastaan olion tiedot. String-taulu on kolmen yksikön kokoinen ja kielivaihtoehdot tulee tallentaa nimitaulukkomuuttujien määräämille paikoille.

public int getBalance(void);

Metodi palauttaa saldon balance-muuttujasta.

public void setBalance(int balance);

Metodi asettaa muuttujaan uuden saldon.

public String getGroupName(int language);

Palauttaa tuoteryhmän nimen halutulla kielellä. Parametrina annetaan yksi luokan julkisista kielen määrittävistä muuttujista.

5.2.2 UserInfo

Paketti: gui

Luokan tehtävänä on välittää käyttäjään liittyvä tieto Database-komponentilta GUI-komponentille. Luokka sisältää kaiken tarpeellisen tiedon käyttäjästä, hänen tilinsä tilanteesta ja tuotteista, joita käyttäjä voi ostaa. Luokkaa käytetään myös apuna, kun uusi käyttäjä lisätään tai käyttäjän tietoihin halutaan tehdä muutoksia. Tällöin luokan avulla siirretään tieto siitä, että mitä asetuksia käyttäjästä on muutettu Database-komponentille, joka tallettaa tiedot kantaan.

Muuttujat

private String username;

Käyttäjän käyttäjätunnus. (Tämä on sama kuin TKTL:n järjestelmissä).

private String givenname;

Olion kuvaaman käyttäjän etunimi. Tämän avulla välitetään rekisteröitymisvaiheessa käyttäjän ilmoittama etunimi tietokantaan.

private String familyname;

Olion kuvaaman käyttäjän sukunimi. Tämän avulla välitetään rekisteröitymisvaiheessa käyttäjän ilmoittama sukunimi tietokantaan.

private UserBalance[] balance;

Taulu UserBalance-olioista, jotka sisältävät tiedot käyttäjän saldosta eri tuoteryhmissä. Tämän tiedon avulla GUI-komponentti pystyy esittämään käyttäjän saldotiedot asiakkaalle.

private int[] fingerprints;

Taulu, johon on tallennettu käyttäjän sormenjälkien ID-numerot siten, että elementti nolla on peukalo ja elementti neljä on pikkurilli ja muut sormet tulevat näiden väliin. Taulun koko on viisi alkiota, ja jos jossain alkiossa ei ole sormenjälkeä, sen arvoksi on asetettu **"-1"**.

private ExportProduct regularportion;

Käyttäjän suosikkituotteen kertova ExportProduct-olio. Sen avulla käyttöliittymä mahdollistaa sen, että asiakas voi ostaa ennalta määrätyn tuotteen antamalla ainoastaan sormenjälkensä ja tekemättä mitään muuta.

private ExportProduct[] userproductlist;

Kaikki käyttäjälle vietäväksi tarjottavat tuotteet, taulukossa henkilökohtaisen suosituimmuusjärjestyksen mukaan ExportProduct-olioina.

private String defaultlanguage;

Muuttujassa on tallennettuna tekstinä käyttäjän oletuskieli. Mahdollisia arvoja ovat fin, swe ja eng.

private boolean cancelAllowed;

Tieto siitä, onko käyttäjällä jokin vientitapahtuma, joka on mahdollista peruuttaa sääntöjen puitteissa. Jos näin on muuttujan arvona on **"true"** tai jos se ei ole mahdollista, niin arvona on silloin **"false"**.

Metodit

public UserInfo(String username, String givenname, String familyname, UserBalance[] balance, int[] fingerprints, ExportProduct regularportion, ExportProduct[] userproductlist, String language, boolean cancelAllowed);

Konstruktori saa parametreina kaikki olion sisältämät tiedot, kun olio luodaan.

public String getUsername(void);

Palauttaa käyttäjätunnuksen.

public void setGivenName(String givenname);

Asettaa etunimen.

public String getGivenName(void);

Palauttaa etunimen.

public void setFamilyName(String familyname);

Asettaa sukunimen.

public String getFamilyName(void);

Palauttaa sukunimen.

public UserBalance[] getBalances(void);

Palauttaa käyttäjän saldotiedot.

public void setBalances(UserBalance[] balances);

Asettaa käyttäjän saldotiedot.

public void setFingerprint(int finger, int id);

Asettaa tietyn sormen sormenjäljeksi jonkin sormenjälki-id:n.

public int getFingerprint(int finger);

Palauttaa tietyn sormen sormenjälki-id:n.

public void setRegularPortion(ExportProduct product);

Asettaa vakiotuotteen.

public ExportProduct getRegularportion(void);

Palauttaa vakiotuotteen.

public ExportProduct[] getUserproductlist(void);

Palauttaa kaikki ostettavat tuotteet.

public void setDefaultLanguage(String defaultlang);

Asettaa käyttäjän oletuskielen.

public String getDefaultLanguage(void);

Palauttaa käyttäjän oletuskielen.

public boolean isCancelAllowed(void);

Palauttaa cancelAllowed-muuttujan arvon.

5.2.3 Product

Paketti: main

Luokka on ExportProduct ja ImportProduct -luokkien yliluokka. Se sisältää tuotteen perustiedot, jotka ovat molemmille tuotteille yhtenevät. Näitä ovat esimerkiksi tuotteen id-numero, nimi eri kielillä ja kuva.

Muuttujat

private int id;

Tuotteen id-numero, joka on sen pääavain tietokannassa.

private String[] name;

Kolmen yksikön kokoinen taulu, jossa on tuotteen nimi eri kielillä kielimuuttujien määräämissä paikoissa.

private String[] group;

Kolmen yksikön taulu, jossa on tuotteen tuoteryhmän nimi eri kielillä kielimuuttujien määräämillä paikoilla.

private ImageIcon productpic;

Tuotteen kuva/symboli, jota käytetään käyttöliittymässä.

public static final int FINNISH = 0;

Suomenkielisen nimen paikka nimitaulukossa.

public static final int ENGLISH = 1;

Englanninkielisen nimen paikka nimitaulukossa.

public static final int SWEDISH = 2;

Ruotsinkielisen nimen paikka nimitaulukossa.

Metodit

public Product(int id, String[] name, String[] group, ImageIcon productpic);

Konstruktori saa parametrinaan muuttujien arvot ja luo uuden olion tästä luokasta.

public int getId(void);

Palauttaa id-numeron muuttujasta.

public String getName(int language);

Palauttaa nimen valitulla kielellä.

public String getGroupName(int language);

Palauttaa tuoteryhmän nimen valitulla kielellä.

public ImageIcon getProductPic(void);

Palauttaa tuotteen kuvan.

5.2.4 ExportProduct

Paketti: main

Tämä luokka pitää sisällään tiedon yksittäisistä tuotteista, joita käyttäjät voivat järjestelmästä ostaa. Se perii suurimman osan toiminnallisuudestaan Product-luokalta ja lisää näihin ominaisuuksiin mahdollisuuden tallettaa kuinka paljon tuotteen käyttäminen vähentää käyttäjän saldoa.

Muuttujat

private double useamount;

Tuotteen "hinta", eli paljonko se vähentää käyttäjän saldoa.

Metodit

public ExportProduct(int id, String[] name, String[] group, double useamount, ImageIcon productpic);

Konstruktori tallettaa parametrina saadut tiedot vastaaviin muuttujiin.

public int getUseAmount(void);

Palauttaa tuotteen hinnan muuttujasta.

5.2.5 ImportProduct

Paketti: main

Luokka perii Product-luokan. Sen avulla siirretään tietoa siitä minkälaisia tuotteita käyttäjät voivat järjestelmään tuoda. Tuodut tuotteet ovat jokainen luokan ilmentymiä. Luokassa on tieto siitä, mitä kokoja tuotteita voi tuoda (amounts), ja minkälainen arvo (values) näillä tuoduilla koilla on. Lisäksi luokka pitää kirjaa mitä määrettä tuotava tuote on, eli onko tuotteen määrä esimerkiksi litroissa (l) vai grammoissa (g).

Muuttujat

private double[] amounts;

Koot, joissa tuotteita voi tuoda.

private double[] values;

Kokoja vastaavat tuontiarvot.

private String unit;

Koon yksikkö, esimerkiksi gramma (g).

Metodit

public ImportProduct(int id, String[] name, String[] group, double[] amounts, double[] values, String unit, ImageIcon productpic);

Konstruktori saa parametreinaan tuotteen tiedot, ja tallettaa ne vastaaviin muuttujiin.

public double[] getIntakeAmounts(void);

Palauttaa taulukon tuotteelle sallituista koista.

public String getUnit(void);

Palauttaa koon yksikön.

public double getValue(double amount);

Palauttaa tiettyä kokoa vastaavan tuotteen arvon.

5.2.6 ProductGroup

Paketti: main

Luokka kerää yhteen kaikki tiettyyn ryhmään kuuluvat tuotteet, joita järjestelmään voi tuoda.

Muuttujat

public String[] groupname;

Tuoteryhmän nimi eri kielillä.

public static final int FINNISH = 0;

Suomenkielisen nimen paikka nimitaulukossa.

public static final int ENGLISH = 1;

Englanninkielisen nimen paikka nimitaulukossa.

public static final int SWEDISH = 2;

Ruotsinkielisen nimen paikka nimitaulukossa.

private ImportProduct[] products;

Tuoteryhmään kuuluvat tuotavat tuotteet taulukossa.

Metodit

public ProductGroup(String[] groupname, ImportProduct[] products);

Konstruktori ottaa vastaan tuoteryhmän nimet ja siihen kuuluvat tuotteet, ja tallettaa ne muuttujiin.

public String getName(int language);

Palauttaa tuoteryhmän nimen valitulla kielellä.

public ImportProduct[] getProducts(void);

Palauttaa ryhmään kuuluvat tuontituotteet taulukkona.

5.2.7 CancelProduct

Paketti: main

Luokka sisältää tiedon viimeisimmästä ostosta, joka on peruttu. Tähän tarkoitukseen luokan luomaan ilmentymään tallennetaan kyseisen oston tapahtumapäivä ja ostetun tuotteen tiedot.

Muuttujat

private String timedate;

Oston tapahtumapäivä, muodossa esimerkiksi ”1.3.2007 23:00”.private ExportProduct product;

Tuote-olio, joka sisältää tiedot tuotteesta, jonka oston käyttäjä voi peruuttaa.

Metodit

public CancelProduct(ExportProduct product, String time);

Konstruktori saa parametreissa olion tiedot, jotka se tallettaa vastaaviin muuttujiin.

public ExportProduct getExportProduct(void);

Palauttaa peruutettavan tuotteen tiedot sisältävän olion.

public String getTimeDateString(void);

Palauttaa tiedon peruutettavan oston tapahtumahetkestä.

5.2.8 GUISettings

Paketti: gui

Nimensä mukaisesti luokka pitää sisällään käyttöliittymän asetukset. Sen avulla käyttöliittymä saa Settings-komponentilta tiedon muuttujista, joita järjestelmän hallinnoja voi muuttaa. Lisäksi luokkaan on Language-luokan avulla tallennettu käyttöliittymän käyttämät kielet.

Muuttujat

private int idletimeout;

Aika sekunneissa, jonka kuluttua käyttöliittymä kirjaa käyttäjän ulos, jos tämä ei ole tehnyt mitään.

private int paytimeout;

Aika sekunneissa, jonka kuluttua valittu (tai suosikki) tuote veloitetaan käyttäjän tililtä, ellei hän peru veloitusta.

private int errortimeout;

Aika (sekunneissa), jonka virheilmoitus (lähinnä tunnistamattomasta sormenjäljestä) on ruudulla näkyvissä.

private Language[] languages;

Taulukkoon tallennetaan tallennetaan käyttöliittymän tekstit Language-oliona eri kielillä, kielivakiolukujen määräämiin paikkoihin.

public static final int FINNISH = 0;

Suomea vastaava vakio.

public static final int ENGLISH = 1;

Englantia vastaava vakio.

public static final int SWEDISH = 2;

Ruotsia vastaava vakio.

Metodit

public GUISettings(Language[] languages , ImageIcon picture, int idletimeout, int paytimeout, int errortimeout);

Konstruktori saa parametrinaan olioon talletettavat tiedot.

public String getString(String stringid, int language);

Hakee valitusta kielestä stringid-parametrin mukaisen käyttöliittymän tekstin kutsumalla oikean Language-olion getText(String id)-metodia.

public ImageIcon getLangPicture(int language);

Palauttaa valittua kieltä vastaavan kuvan (lipun) kutsumalla oikean Language-olion getLangPicture()-metodia.

public int getIdleTimeout(void);

Palauttaa vastaavan muuttujan arvon.

public int getPayTimeout(void);

Palauttaa vastaavan muuttujan arvon.

public int getErrorTimeout(void);

Palauttaa vastaavan muuttujan arvon.

5.2.9 Language

Paketti: gui

Luokka on GUISettings-luokan apuluokka, jonka ilmentymiin järjestelmä tallentaa kaikkien käyttöliittymän tarvitsemien tekstien eri kieliversiot.

Muuttujat

private Hashtable uiStringTable;

Hajautustaulukko, joka sisältää kieleen kuuluvat merkkijonot. Avaimena toimii merkkijonoa kuvaava nimi, esim. ”warningButton”.

private ImageIcon picture;

Kielen kuva (lippu).

Metodit

public Language(Hashtable uiStringTable, ImageIcon picture);

Konstruktori saa parametrinaan hajautustaulun, joka sisältää tietyn kielen käyttöliittymässä näkyvät tekstit , ja tallettaa sen uiStringTable-muuttujaan. Lisäksi parametrina saatu kuva talletetaan omaan muuttujaansa.

public String getText(String id);

Hakee hajautustaulusta merkkijonoa käyttäen saatua parametria avaimena, ja palauttaa avainta vastaavan merkkijonon. Mikäli sellaista merkkijonoa ei löydy, palautetaan **null**.

public ImageIcon getLangPicture(void);

Palauttaa muuttujaan tallennetun kielen kuvan.

5.2.10 DBSettings

Paketti: database

Luokka pitää sisällään Database-komponentin asetukset. Nämä asetukset pitävät sisällään tiedon siitä miten tietokantaan saa yhteyden ja mikä on suurin sallittu aika edellisen vientitapahtuman perumiselle. Luokkaan varastoidaan myös kantaan tehtävien kyselyiden SQL-lauseet.

Muuttujat

private String databaseurl;

Tietokannan osoite, sisältäen myös kannan nimen.

private String dbuser;

Tietokannan käyttäjätunnus.

private String dbpassword;

Tietokannan salasana.

private Hashtable dbqueries;

Hajautustaulu, joka sisältää tarvittavat SQL-lauseet. Lausetta vastaavana avaimena on jokin kuvaava nimi, kuten ”getExportProducts”. Lauseiden muuttujat on korvattu kysymysmerkillä.

private int cancelTimeout;

Sisältää tiedon ajasta (minuutteina), jota vanhempia ostotapahtumia käyttäjä ei voi peruuttaa.

Metodit

public DBSettings(String databaseurl, String dbuser, String dbpassword, Hashtable dbqueries, int timeout);

Konstruktori tallettaa parametreina saadut tiedot vastaaviin muuttujiin.

public String getDBUrl(void);

Palauttaa tietokannan osoitteen.

public String getPassword(void);

Palauttaa tietokannan salasanan.

public String getUser(void);

Palauttaa tietokannan käyttäjätunnuksen.

public String getQuery(String queryname);

Hakee hajautustaulusta ja palauttaa tietokantahaun, joka vastaa annettua nimeä tai **null**, mikäli kyselyä ei löydy.

public int getCancelTimeout(void);

Palauttaa sallitun peruutusajan.

5.2.11 FingerprintSettings

Paketti: fingerprintauthentication

Tässä luokassa on tiedot sormenjälkilukijan asetuksista. Näitä asetuksia ovat lukijan sormenjäljille varaaman välimuistin koko ja polut GrFingerJavan tarvitsemiin lisenssitiedostoon ja natiivikirjastoihin.

Muuttujat

private String licensepath;

Tämä muuttuja pitää itsessään tiedon siitä, että mistä hakemistosta GrFingerJavan tarvitsema lisenssitiedosto löytyy. Tämä lisenssitiedosto tulee antaa kirjastolle, kun se luodaan.

private String nativelibrarypath;

Tämä muuttuja pitää sisällään tiedon siitä, että missä sijaitsevat GrFingerJava-kirjaston tarvitsemat natiivikirjastot.

Metodit

public FingerprintSettings(String licensepath, String nativelibpath);

Konstruktori, joka luo FingerprintSettings-olion. Se saa syötteenä sormenjälkipuskurin koon ja lisenssitiedoston hakemistopolun, sekä hakemistopolun natiivikirjastoihin.

public String getLicensePath(void);

Metodi, jolla saadaan FingerprintSettings-oliosta lisenssitiedoston polku merkkijonona.

public String getNativeLibraryPath(void);

Metodi, joka palauttaa FingerprintSettings-oliosta natiivikirjastojen polun.

5.2.12 FPData

Paketti: fingerprintauthentication

Tämän tieto-olion avulla Database ja FingerprintAuthentication -komponentit siirtävät sormenjälkien tunnistustietoja toistensa välillä. Olioon on tallennettuna tunnistustieto tavutaulukkona ja kokonaislukuna tallennetun tiedon laatu, sekä lisäksi sormenjäljen tunnusnumero.

Muuttujat:

private byte[] data;

Muuttujassa on tallennettuna sormenjäljen tunnistustieto tavutaulukkona.

private int quality;

Muuttuja kertoo sormenjäljen tunnistetiedon tarkkuuden. Tätä tarvitaan, kun tiedosta tehdään uusi tunnistusolio uudelleenluontiin.

private int id;

Kuvan sormenjäljen id-numero kannassa. Jos kuvaa ei ole vielä talletettu kantaan on tämän arvo **-1.**

Metodit

public FPData(byte[] data, int quality, int id);

Luo uuden FPData-olion, jonka avulla sormenjäljen tunnistustietoja voidaan siirtää.

public byte getData(void);

Palauttaa sormenjäljen tunnistustiedon tavutaulukkona.

public int getQuality(void);

Palauttaa kokonaislukuna tunnistustiedon tarkkuuden.

public int getId(void);

Palauttaa kokonaislukuna kuvan ID-numeron. Jos kuvalla ei ole vielä ID-numeroa, niin palauttaa **-1.**

5.2.13 AuthenticationSettings

Paketti: authentication

Tämän luokan avulla välitetään autentikointikomponentille tieto URL-osoitteesta, josta autentikointi tehdään.

Muuttujat

private String authenticationUrl;

URL-osoite, johon autentikointikomponentin pitää ottaa yhteyttä.

private String keyStorePath;

Polku Java-virtuaalikoneen SSL-sertifikaattien säilytystiedostoon.

private String keyStorePassword;

Java-virtuaalikoneen sertifikaattivaraston salasana.

Metodit

public AuthenticationSettings(String authenticationUrl, String keyStorePath, String keyStorePasswd);

Konstruktori tallettaa parametrina saadut asetustiedot vastaaviin luokan muuttujiin.

public String getAuthenticationUrl(void);

Palauttaa URL-osoitteen muuttujasta.

5.3 Rajapinnat

Ohjelman komponenttien välissä on kuusi komponenttien välisen liikennöinnin hoitavaa rajapintaa. Näiden lisäksi komponenttien pääluokilla on konstruktorimetodit, jotka hoitavat asetus-tieto-olioiden ja tapahtumarajapintojen (eventinterface) rekisteröimisen luokille. Tapahtumarajapintojen avulla komponentit pääsevät ilmoittamaan muulle järjestelmälle käyttäjän suorittaneen jonkin toiminnon. Niiden lisäksi komponenteilla on normaaleja rajapintoja, joiden kautta komponentin tarjoamia palveluita voi käyttää.

Rajapinnat on esitelty aiemmin kuvassa 3, ja seuraavissa luvuissa ne kuvataan tarkemmin.

5.3.1 UserInterface

Rajapinnan tehtävänä on antaa mahdollisuus kutsua käyttöliittymäkomponentin tarjoamia palveluita. Rajapinta antaa mahdollisuuden kutsua käyttöliittymää, kun käyttäjä on pistänyt sormensa sormenjälkilukijaan. Käyttöliittymää voidaan kutsua myös, jos ohjelman suorituksessa tapahtuu jokin virhe, ja käyttöliittymä halutaan jäädyttää väliaikaisesti näyttämään virheilmoitusta.

Metodit

public void authenticatedUser(UserInfo user, int fingerprintid);

Metodia kutsutaan, kun järjestelmään on saapunut tunnistettu sormenjälki. GUI-komponentin tilasta seuraa mitä tunnistetulle sormenjäljelle tehdään. Tavallisessa tapauksessa sitä käytetään käyttäjän tunnistamiseen. Jos kuitenkin käyttäjä on tekemässä sormenjälkien lisäystä, niin silloin sormenjälki liitetään osaksi jo käyttöliittymällä tiedossa olevaa UserInfo-komponenttia. Käyttöliittymän tulee tarkistaa silloin, kun se liittää sormenjälkeä osaksi UserInfo-oliota, että sormenjälki ei kuulu eri käyttäjälle.

public boolean unknownUser(void);

Metodia kutsutaan silloin, kun järjestelmään tulee tunnistamaton käyttäjä. Jos käyttöliittymä ei ole sormenjälkien syöttämisen tilassa, niin silloin se vain näyttää ilmoituksen, että se on havainnut tunnistamattoman sormenjäljen, ja metodi palauttaa "false"-arvon. Jos metodi on sormenjälkien syöttämiseen tarkoitetussa tilassa, niin silloin metodi palauttaa "true"-arvon. Tällöin rajapintaa käyttävä komponentin tulisi kutsua rajapintaa addFingerprint-metodilla.

public void addFingerprint(int id);

Metodi välittää käyttöliittymälle uuden sormenjäljen id-numeron, jolla sormenjälki on tallennettu kantaan. Tämä sormenjälki liitetään sitten osaksi käyttäjän UserInfo-oliota, joka tallennetaan kantaan, kun käyttäjä on tehnyt muutoksensa tietoihinsa.

public void stopForProblem(String problemstringid);

Jos ohjelman käsittelyssä tapahtuu hetkellinen virhe, esimerkiksi yhteys tietokantaan häviää väliaikaisesti, menee käyttöliittymä tätä metodia kutsumalla suojattuun tilaan, jossa käyttäjät eivät voi tehdä muutoksia. Käyttöliittymä näyttää problemstringid-parametrin määrittelemän virheilmoituksen, joka löytyy kielitiedostoista.

public void continue(void);

Järjestelmän ongelman poistuttua kutsutaan tätä metodia. Käyttöliittymä poistaa virheilmoituksen ja menee takaisin normaalitilaan odottamaan uutta sisäänkirjautumista.

5.3.2 UserInterfaceEvent

Rajapinnan tehtävänä on välittää käyttäjän tekemien valintojen vaikutukset käyttöliittymästä muulle ohjelmalle. Tämän kaltaisia tapahtumia ovat esimerkiksi ostosten teko tai tuotteiden tuominen, sekä käyttäjän omien asetusten muuttaminen.

Metodit

public boolean authenticateUser(String username, String password);

Käyttöliittymä kutsuu tätä metodia uuden käyttäjän rekisteröinnin yhteydessä, kun käyttäjä antaa käyttäjätunnuksensa ja salasanansa. Metodi palauttaa "**true**"-arvon, jos käyttäjän salasana ja käyttäjätunnus täsmäävät toisiinsa. Jos ne eivät täsmää, metodi palauttaa "**false**"-arvon.

public void saveUser(UserInfo user);

Käyttöliittymä kutsuu tätä metodia, kun se haluaa tallettaa UserInfo-oliossa olevat tiedot tietokantaan. Parametrina on UserInfo-olio. Metodi ei palauta mitään.

public void exportProduct(UserInfo user, ExportProduct product);

Käyttöliittymä kutsuu tätä metodia, kun käyttäjä ostaa jonkin tuotteet järjestelmästä. Parametrina se saa UserInfo-olion, joka kertoo käyttäjän, joka tuotetta on ostamassa ja tuotteen, jonka käyttäjä osti. Metodilla ei ole paluuarvoa.

public void importProduct(UserInfo user, ImportProduct product, int amount);

Käyttöliittymä kutsuu tätä metodia, kun käyttäjä tuo jonkin tuotteen ja rekisteröi tuonnin järjestelmään. Parametreina metodilla on UserInfo-olio, joka kertoo kenestä käyttäjästä on kyse, ja ImportProduct-olio kertomassa tuote ja tuotavan tuotteen määrä. Metodi ei palauta mitään.

public ExportProduct[] getProductList(void);

Tämän metodin avulla käyttöliittymä hakee listan järjestelmässä olevista ostettavista tuotteen. Metodilla ei ole parametreja ja se palauttaa ExportProduct-olioiden listan.

public void insertAlert(ImportProduct product);

Lisää kantaan tiedon, että jokin tuote on loppumassa. Saa parametrina ImportProduct-olion, joka kertoo mistä tuotteesta on kyse. Käyttöliittymä kutsuu tätä metodia, kun jokin käyttäjä pistää hälytyksen tuotteelle päälle.

public void removeAlert(ImportProduct product);

Poistaa kannasta tiedon, että jokin tuote on loppumassa. Saa parametrina ImportProduct-olion, joka kertoo mistä tuotteesta on kysymys. Käyttöliittymä kutsuu

tätä metodia, kun joku käyttäjä poistaa hälytyksen tai käyttäjä tuo hälytyksen alaista tuotetta järjestelmään.

public ImportProduct[] getAlerts(void);

Tämä metodi hakee käyttöliittymälle listan voimassa olevista hälytyksistä. Metodi ei saa mitään parametreja ja palauttaa listan ImportProduct-olioita, joille hälytys on pistetty päälle. Jos yhdelläkään tuotteella ei ole hälytystä päällä, palauttaa tyhjän listan.

public ProductGroup[] getProductGroups(void);

Käyttöliittymä hakee tämän metodin avulla listan tuoteryhmistä ja niihin kuuluvista tuotavista tuotteista. Metodi ei saa mitään parametreja ja palauttaa listan ProductGroup-olioita. Käyttöliittymä kutsuu tätä metodia tavallisimmin, kun se rakentaa tuotteiden tuontinäkymää ja sinne listaa tuotavista tuotteista.

public void deleteFingerprint(int id);

Poistaa id:n määrittelemän sormenjäljen järjestelmästä. Käyttöliittymä kutsuu tätä metodia, kun käyttäjä on merkinnyt id:llä varustetun sormenjäljen korvattavaksi uudella ja sormenjäljen tilalle tulee uusi, joko tunnistamaton sormenjälki tai jokin muu tunnistettu sormenjälki, kuin kohdassa jo aikaisemmin varastoitu. Käyttöliittymän tehtävänä on ennen metodin kutsumista varmistaa, että kyseinen sormenjälki ei ole merkitty myös saman käyttäjän muihin sormiin kuuluvaksi.

public UserInfo loadUser(String username);

Metodin tehtävänä on hakea käyttäjän tiedot. Metodi saa parametrina käyttäjätunnuksen ja palauttaa UserInfo-olion. Tätä metodia käytetään silloin, kun jo rekisteröitynyt käyttäjä haluaa vaihtaa sormenjälkiään, mutta ei joko halua tai pysty kirjautumaan sisään omilla sormenjäljillään ja kirjautuu sisään käyttäjätunnuksellaan.

public CancelProduct cancelProductExport(UserInfo user);

Käyttöliittymä kutsuu tätä metodia, kun käyttäjä peruu viimeisimmän ostotapahtuman. Parametrina metodi saa käyttäjän tiedot UserInfo-oliona ja metodin tulee palauttaa CancelProduct-olio, joka sisältää tiedon siitä minkä tuotteen osto peruttiin ja milloin ostaminen oli tapahtunut. Jos UserInfo-oliossa oleva cancelAllowed-muuttuja on false, käyttäjän ei ole mahdollista tehdä peruutusta. Jos metodia tällaisessa tapauksessa kutsutaan, se palauttaa **null**.

5.3.3 Database

Rajapinta tarjoaa tarvittavat metodit tiedon hakemiseen tietokannasta ja sen

tallettamiseen sinne. Monet järjestelmän käyttämistä tieto-olioista luodaan rajapinnan toteuttavassa komponentissa tietokannan tiedoista.

Metodit

public UserInfo loadUser(int id);

Metodi palauttaa UserInfo-olion sille annettua id-numeroa vastaan. Id-numero on sama kuin jokin käyttäjään liitettyjen sormenjälkien numeroista.

public UserInfo loadUser(String username);

Metodi palauttaa käyttäjän UserInfo-olion käyttäjänimen perusteella.

public void saveUser(UserInfo user);

Metodi tallettaa sille annetun UserInfo-olion tiedot kantaan. Metodilla ei ole paluuarvoja.

public void exportProduct(UserInfo user, ExportProduct product);

Metodi saa parametreina UserInfo ja ExportProduct -oliot. Näiden perusteella Database-komponentti tallettaa UserInfo-olion määrittämälle käyttäjälle oston ExportProduct-olion määrittämästä tuotteesta. Metodilla ei ole paluuarvoja.

public CancelProduct cancelProductExport(UserInfo user);

Metodin avulla perutaan viimeisin tuotteen osto. Saa parametrina käyttäjän, jolta osto perutaan ja palauttaa perutun oston tiedot CancelProduct-oliossa.

public void importProduct(UserInfo user, ImportProduct product, int amount);

Metodin avulla rekisteröidään käyttäjän tuotteen tuominen järjestelmään. Metodi saa parametreina käyttäjätiedot UserInfo-oliossa, Tuotteen tiedot ImportProduct-oliossa ja tuotavan määrän. Näiden tietojen perusteella tuotavan määrän mukaan käyttäjän tilin saldoa kasvatetaan ryhmässä, johon tuote kuuluu. Metodi ei palauta mitään.

public ExportProduct[] getProductList(void);

Metodin avulla ohjelma saa listan ostettavista tuotteista. Tätä listaa käytetään, kun käyttäjä ensimmäistä kertaa valitsee vakiotuotteen, jonka haluaa ostaa. Metodi ei saa mitään parametreja ja palauttaa ExportProduct-olioiden listan.

public void insertAlert(ImportProduct product);

Metodi tallettaa tietokantaan määrätylle tuotteelle hälytyksen, että tuote on loppumassa. Tuote saada parametrina, joka on ExportProduct-olio ja metodi ei palauta mitään.

public void removeAlert(ImportProduct product);

Metodi poistaa hälytyksen tuotteelta, jolle on määrätty aikaisemmin hälytys. Jos tuotteella ei ole hälytystä metodi ei tee mitään. Saa parametrina tuotteen ImportProduct-oliona ja ei palauta mitään.

public ImportProduct[] getAlerts(void);

Metodi hakee kannasta listan tuotteista, joille on asetettu hälytys. Se ei saa mitään parametreja ja palauttaa listan ImportProduct-olioita.

public int saveFingerprint(FPData data);

Metodi tallettaa sormenjäljen tunnistetiedon kantaan ja antaa vastauksena id:n sormenjäljelle, joka juuri talletettiin. Metodin parametrina on FPData-olio ja se palauttaa kokonaisluvun.

public ProductGroup[] getProductGroups(void);

Metodin avulla saadaan lista tuoteryhmistä, jotka sisältävät listan ryhmään kuuluvista tuotavista tuotteista. Metodi ei ota mitään parametreja ja palauttaa listan ProductGroup-olioita.

public FPData[] getFingerprintData(void);

Metodin tehtävänä on hakea kannasta kaikki sormenjälkien tunnistetiedot. Metodi palauttaa listan sormenjälkiä FPData-olioina.

public void deleteFingerprint(int id):

Poistaa id-numeron määrittämän sormenjäljen kannasta. Metodi saa parametrina id-numeron kokonaislukuna ja ei palauta mitään.

public boolean reconnectDatabase(void);

Metodia kutsutaan, jos tietokantayhteys on jostain syystä katkennut. Metodi yrittää avata yhteyden tietokantaan, ja jos yhteyttä ei saada, palauttaa "**false**"-arvon. Jos yhteys alkaa metodin suorittamisen aikana toimimaan, palautetaan "**true**"-arvo.

5.3.4 Configuration

Rajapinnan tehtävänä on antaa näkymä komponenttiin, jonka tehtävä on lukea asetukset ohjelman asetustiedostosta. Komponentti muuttaa asetukset asetusolioiksi, jotka MainProgram-komponentti noutaa ja välittää ohjelman muille osille.

Metodit

public DBSettings getDBSettings(void);

Metodi palauttaa DBSettings-olion, joka sisältää tietokannan asetukset ja hakulauseet. Tämä olio annetaan alustusparametrina Database-komponentille.

public GUISettings getGUISettings(void);

Metodi palauttaa GUISettings-olion, joka sisältää käyttöliittymän asetukset ja eri kieliversiot käyttäjille näytettävistä teksteistä. Tämä olio annetaan alustusparametrina GUI-komponentille.

public FingerprintSettings getFingerprintSettings(void);

Metodi palauttaa FingerprintSettings-olion, jonka avulla ohjelma antaa FingerprintAuthentication-komponentille sen tarvitsemat asetukset komponentin alustuksen yhteydessä.

public AuthenticationSettings getAuthenticationSettings(void);

Metodi palauttaa AuthenticationSettings-olion, jonka avulla Authentication-komponentti saa tarvitsemansa asetukset.

5.3.5 UserAuthentication

Rajapinnan tehtävänä on antaa normaaliin käyttäjätunnistukseen yhtenäinen rajapinta. Rajapinnan avulla käyttäjät voivat tunnistautua TKTL:n normaalilla käyttäjätunnus ja salasana -parillaan.

Metodit

public boolean authenticateUser(String username, String password);

Metodi saa parametreinaan käyttäjänimen ja salasanan String-olioina. Metodi palauttaa "**true**"-arvon, jos käyttäjän salasana ja käyttäjätunnus ovat oikein. Muussa tapauksessa metodi palauttaa "**false**"-arvon.

5.3.6 Fingerprint

Rajapinnan avulla ohjelma voi antaa sormenjälkitunnistuskomponentille käskyjä.

Metodit

public void deleteFingerprint(int id);

Metodin avulla voidaan FingerprintAuthentication-komponentin välimuistista poistaa järjestelmästä poistettuja sormenjälkiä. Metodi saa parametrina poistettavan sormenjäljen id-numeron ja ei palauta mitään.

public void addFingerprint(FPData data);

Metodin avulla voidaan uusi sormenjäljen tunnistustieto tallettaa FingerprintAuthentication-komponentin välimuistiin. Näin komponentin ei tarvitse käydä turhaan hakemassa kaikkia tunnistetietoja tietokannasta ajantasaisuuden varmistamiseksi.

5.3.7 FingerprintEvent

Rajapinnan tehtävänä on kuunnella FingerprintAuthentication-komponentilta tulevia tapahtumia ja välittää näitä eteenpäin muiden ohjelman osien käyttöön. Rajapintaan tulee pääasiassa tapahtumia, kun käyttäjä laittaa sormen sormenjälkilukijaan ja ohjelmaa käynnistettäessä.

Metodit

public boolean authenticatedFingerprint(int id);

Komponentti kutsuu tätä metodia, kun se saa sormenjäljen, jonka se onnistuneesti tunnistaa id-numeron mukaiseksi sormenjäljeksi. Mikäli sormenjälkeä vastaavaa käyttäjää ei löydy enää tietokannasta (eli tunnistetieto pitää poistaa myös välimuistista), metodi palauttaa ”**false**”. Muuten palautetaan ”**true**”.

public void unAuthenticatedFingerprint(FPData data);

Komponentti kutsuu tätä metodia, kun se saa sormenjäljen, jota ei tunnistettu. Metodi saa parametrina uuden sormenjäljen, jolloin ohjelma voi päättää mitä se haluaa tehdä sormenjäljellä.

public FPData[] getFingerprintData(void);

Komponentti lataa tämän metodin avulla itselleen sormenjälkien tunnistusdatan tietokannasta. Vastauksena palautetaan lista sormenjälkiä FPData-olioina.

5.3.8 GUI-komponentin alikomponenttien rajapinnat

GUI-komponentilla on useampia sisäisten osien välillä olevia rajapintoja. Näiden

tehtävänä on mahdollistaa eri komponenttien välinen tiedonsiirto ja samalla selkeyttää, sekä jakaa GUI-komponentin rakennetta. Seuraavaksi kuvaamme rajapintojen rakenteen ja kerromme tarkemmin mitä ne ja niiden metodikutsut tekevät.

5.3.8.1 ViewEventInterface

Rajapinta tarjoaa tarvittavat metodit GUI-Komponentin muille alikomponenteille, jotta ne voivat kutsua GuiMain-komponenttia ja sen avulla muita sen aliosia tai muuta ohjelmaa. Rajapinnasta on esitelty kaikki sen tarjoamat metodit. Kaikki GUI-komponentin osat käyttävät tätä rajapintaa.

paketti:

gui.interfaces

metodit:

public void changeStartView(void);

Komponentit kutsuvat tätä metodia, kun käyttäjä painaa uloskirjautumisnappia.

GuiMain luo uuden ilmentymän startView.StartPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeShopView(void);

Komponentit kutsuvat tätä metodia, kun käyttäjä siirtyy käyttöliittymän ostonäkymään.

GuiMain luo uuden ilmentymän shopView.ShopPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeImportView(void);

Komponentit kutsuvat tätä metodia, kun käyttäjä siirtyy käyttöliittymän tuontinäkymään.

GuiMain luo uuden ilmentymän importView.ImportPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeConfigView(void);

Komponentit kutsuvat tätä metodia, kun käyttäjä siirtyy käyttöliittymän asetusnäkymään.

GuiMain luo uuden ilmentymän configView.ConfigPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeAlertView(void);

StartView-komponentti kutsuu tätä metodia, kun käyttäjä siirtyy käyttöliittymän hälytysten lisäysnäkymään. GuiMain luo uuden ilmentymän alertView.AlertPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeRegisterView(void);

Komponentit kutsuvat tätä metodia, jos käyttäjä siirtyy käyttöliittymän uuden käyttäjän rekisteröimisnäkymään.

GuiMain luo uuden ilmentymän registerView.RegisterPanel luokasta ja tekee siitä uuden contentPanensa.

public void changeLanguage(void);

Komponentit kutsuvat tätä metodia, jos käyttäjä vaihtaa käyttöliittymän kieltä.

GuiMain vaihtaa kielen ja luo uudelleen sen hetkisen contentPanen ja päivittää itsensä.

public String getGuiText(String id);

Komponentti kutsuu tätä metodia saadakseen jonkin käyttöliittymässä esiintyvän tekstin. Koska GuiMain tietää, että mikä kieli on milläkin hetkellä käytössä se osaa palauttaa tekstin oikealla kielellä.

public void importProduct(ImportProduct product);

Komponentti kutsuu tätä metodia, kun käyttäjä on valinnut tuotavan tuotteen.

GuiMain kertoo muulle järjestelmälle, että käyttäjä on tuonut tuotteen *product* ja hyväksynyt sen tuonnin.

public void setFavoriteProduct(ExportProduct product);

Komponentti kutsuu tätä metodia, kun käyttäjä vaihtaa tuotetta, jonka hän haluaa vakiona veloitettavan pelkästään sormenjäljen antamalla.

GuiMain kertoo asiasta muulle järjestelmälle.

public void addAlert(void);

Komponentti kutsuu tätä metodia, kun jollekin tuotavalle tuotteelle halutaan asettaan hälytys sen loppumisen merkiksi. Tätä metodia kutsutaan kun käyttäjä on jo valinnut tuotteen ja painaa hälytyksen varmistuspainiketta.

GuiMain kertoo asiasta muulle järjestelmälle.

public String getSelectedAlertProduct(void);

Metodi palauttaa käyttäjällä valittuna olevan tuotteen nimen, kun käyttäjä on asettamassa tuotehälytystä jollekin tuotteelle.

public void alertProduct(ImportProduct product);

Tällä metodilla tuotehälytyksen lisäysnäkymä ilmoittaa GuiMainille, että käyttäjä on valinnut jonkin tuotteen tuotelistasta.

public void authenticateUser(String username, String password);

Komponentti kutsuu tätä metodia, kun se on saanut käyttäjältä käyttäjätunnuksen ja salasanan, joiden avulla käyttäjä haluaa rekisteröityä järjestelmään.

GuiMain kysyy muulta järjestelmältä onko käyttäjä jo rekisteröinyt. Mikäli ei ole, mutta on laitoksen käyttäjä GuiMain luo uudelleen registerView.RegisterPanel luokan ja kertoo, että on näytettävä lisäkentät. Mikäli käyttäjä oli jo rekisteröitynyt luodaan configView.ConfigPanel luokka ja laitetaan se GuiMainin contentPaneksi. Mikäli yritetään salasanaparilla, joka ei ole laitoksen, käyttäjälle kerrotaan vain, että tunnus ei kelpaa.

public void setUserName(String firstname, String familyname);

Komponentti kutsuu tätä metodia, kun käyttäjä haluaa asettaa itselleen etunimen ja sukunimen järjestelmään.

GuiMain muuttaa UserInfo oliota.

public void buyProduct(ExportProduct product);

Komponentti kutsuu tätä metodia, kun käyttäjä ostaa tuotteen järjestelmästä.

GuiMain kertoo tapahtuneesta muulle järjestelmälle.

public void productSelected(ExportProduct product);

Komponentti kutsuu tätä metodia, kun käyttäjä valitsee, että minkä tuotteen hän aikoo ostaa järjestelmästä.

GuiMain kertoo muulle järjestelmällä, että käyttäjä ostaa tuotteen *product*.

public void removeAlert(ImportProduct product);

Komponentti kutsuu tätä metodia, kun käyttäjä kuittaa jonkin hälytyksen käsitellyksi. (Eli on tuonut tai aikoo tuoda hälytettävää tuotetta.)

GuiMain kertoo järjestelmälle, että hälytys poistetaan, jonka jälkeen luo uudelleen alertView.AlertPanelin ja laittaa sen contentPanekseen.

public int getCurrentLanguage(void);

Komponentti kutsuu tätä metodia, saadakseen tietää mikä kieli on tällä hetkellä käytössä. Metodia tarvitaan, koska tuotteiden nimet ovat tuotteissa ja näitä nimiä ei kysytä tämän takia rajapinnan kautta.

public ImageIcon getLanguageImage(void);

Komponentti kutsuu tätä metodia saadakseen tämänhetkisen kielen kuvan.

public void addingFingerprint(int finger);

Komponentti kutsuu tätä metodia kertoakseen, että käyttäjä on lisäämässä parametrina

saatavaan sormeen sormenjäljen. GuiMain menee tilaan, jossa se tietää, että seuraava sormenjälki on jälki joka talletetaan vaatimusten täyttyessä käyttäjälle.

public void cancelFingerprintAdd(void);

Komponentti kutsuu tätä metodia, kun käyttäjä peruu sormenjäljen lisäämisen johonkin sormistaan.

GuiMain poistuu tilasta, jossa se odottaa käyttäjälle lisättävän uuden sormenjäljen.

public void cancelLastAction(void);

Komponentti kutsuu tätä metodia, jos käyttäjä haluaa perua viimeisimmän ostotapahtuman.

GuiMain kertoo muulle järjestelmälle, että käyttäjältä on peruttava sen viimeinen tapahtuma.

public int getBalance(String[] productGroup);

Metodi saa parametrinaan jonkin tuoteryhmän nimen (nimitaulukon), ja palauttaa järjestelmään kirjautuneena olevan käyttäjän saldon tässä tuoteryhmässä.

5.3.8.2 ViewInterface

Luokka on kaikkien muiden komponenttikohtaisten rajapintaluokkine yliluokka. Se määrittelee kaikille yhteiset metodit, joita GuiMain tarvittaessa kutsuu. Yksikään komponentti ei suoraan toteuta tätä rajapintaa. Määriteltävät metodit ovat komponenttien tarjoamien näyttöjen näyttämiseen ja piilottamiseen, sekä kielenvaihtoon liittyviä.

paketti:

gui.interfaces

metodit:

public void changeLanguage(void);

Tätä metodia kutsutaan, kun käyttöliittymän halutaan vaihtavan siinä olevat tekstit uusiksi. Metodin kutsumisen jälkeen komponentti kyselee uudet tekstit ViewEventInterface-rajapinnan getGuiText-metodin avulla.

public void showView(JFrame guimain);

Tätä metodia kutsutaan, kun halutaan tämän komponentin pistävän itsensä näkyväksi käyttöliittymässä.

public void hideView(void);

Tätä metodia kutsutaan, kun halutaan piilottaa tämä komponentti käyttöliittymässä.

5.3.8.3 RegisterViewInterface

Tämä on RegisterView-komponentin rajapinta, jonka kautta sitä hallitaan. Se perii ViewInterface-rajapinnan metodit ja lisäksi määrittelee komponenttikohtaisia metodeja.

paketti:

gui.interfaces

metodit:

public void showAskName(boolean authOK)

Metodia kutsutaan, kun käyttäjän antamat salasana ja käyttäjätunnus on tarkistettu ja on havaittu, että kyseessä on uusi käyttäjä. Metodi saa RegisterView-komponentin näyttämään etu- ja sukunimen kyselyyn tarkoitetut syöte kentät käyttäjälle.

5.3.8.4 ShopViewInterface

Tämä on ShopView-komponentin rajapinta, jonka avulla sille annetaan tiedot käyttäjän ostettavissa olevista tuotteista ja tieto myös käyttäjän saldoista eri tuoteryhmissä. Se myös perii ViewInterface-rajapinnan metodit.

paketti:

gui.interfaces

metodit:

public void addExportProducts(ExportProduct[] products);

Metodi syöttää tuotteet, joita käyttäjä voi ostaa järjestelmästä komponentin näyttämään tuotelistaan. Tuotteet on järjestetty sen mukaan mitä tuotteita käyttäjä on eniten ostanut.

public void setBalance(UserBalance[] balances);

Metodin avulla komponentille syötetään tieto käyttäjän saldoista eri tuoteryhmissä. Jokaista järjestelmässä olevaa tuoteryhmää vastaa yksi UserBalance-olio.

5.3.8.5 ImportViewIterface

Tämä on ImportView-komponentin rajapinta, joka myös perii ViewInterface-rajapinnan toiminnallisuuden. Se lisäksi määrittää omat metodinsa tuotavien tuotteiden ja käyttäjän saldotietojen syöttämiseen komponentille.

paketti:

gui.interfaces

metodit:

public void addImportProductGroups(ProductGroup[] groups);

Metodin avulla ImportView-komponentille syötetään lista tuoteryhmistä, joita järjestelmässä on ja niiden sisällä ryhmään kuuluvista tuotavista tuotteista.

public void setBalance(UserBalance[] balances);

Metodin avulla komponentille syötetään tieto käyttäjän saldoista eri tuoteryhmissä. Jokaista järjestelmässä olevaa tuoteryhmää vastaa yksi UserBalance-olio.

5.3.8.6 AlertViewInterface

Toteuttaa AlertView-komponentin ohjausrajapinnan. Rajapinta perii itse määrittelemiensä metodien lisäksi myös ViewInterface-rajapinnan metodit.

paketti:

gui.interfaces

metodit:

public void addImportProductGroups(ProductGroup[] groups);

Metodin avulla AlertView-komponentille syötetään lista tuoteryhmistä, joita järjestelmässä on ja niiden sisällä ryhmään kuuluvista tuotavista tuotteista. Näiden avulla komponentti sitten tekee listan tuotteista, joille voidaan asettaa hälytys siitä, että tuote on loppumassa.

5.3.8.7 StartViewInterface

Rajapinta StartView-komponentin hallintaan. Rajapinta myös perii ViewInterface-rajapinnan metodit ja lisäksi toteuttaa komponentille ominaisia metodeita, joiden avulla sen tilaa voidaan muuttaa. Nämä metodit auttavat lisäämään hälytyksiä ja poistamaan niitä komponentista.

paketti:

gui.interfaces

metodit:

public void removeAlert(ImportProduct product);

Metodin avulla ilmoitetaan StartView-komponentille, että se voi lopettaa näyttämästä parametrina olevalle tuotteelle loppumisvaroitusta.

public void addAlert(ImportProduct product);

Metodin avulla ilmoitetaan StartView-komponentille, että parametrina ilmoitetulle tuotteelle pitää näyttää varoitus, että tuote on loppumassa.

5.3.8.8 ConfigViewInterface

Rajapinnan avulla voidaan hallita ConfigView-komponentin tilaa. Sen kautta pystyy lisäämään komponentille tiedon järjestelmästä ostettavista tuotteista, lisätä sormenjälkiä ja antaa komponentille tiedon käyttäjän sen hetkisestä saldosta järjestelmässä.

paketti:

gui.interfaces

metodit:

public void addExportProducts(ExportProduct[] products);

Lisää listan järjestelmästä ostettavista tuotteista komponenttiin. Jos käyttäjä on uusi lista on aakkosjärjestyksessä. Muussa tapauksessa lista on käyttäjän suosituimmuusjärjestyksessä.

public void addFingerprints(boolean[] fingerprints);

Antaa komponentille tiedon, että missä käyttäjän sormista on talletettuna sormenjälki. Jos sormessa ei ole sormenjälkeä, niin sen kohdalla taulussa on **"false"**-arvo , muuten **"true"**-arvo.

public void setBalance(UserBalance[] balances);

Metodin avulla komponentille syötetään tieto käyttäjän saldoista eri tuoteryhmissä. Jokaista järjestelmässä olevaa tuoteryhmää vastaa yksi UserBalance-olio.

public void fingerprintAdded(void);

Kertoo komponentille, että käyttäjälle on onnistuneesti tallennettu sen valitsemaan sormeen sormenjälki, jotta komponentti voi näyttää tiedon käyttäjälle.

6 Raportointiliittymän arkkitehtuuri

Raportointiliittymän avulla käyttäjät pystyvät tulostamaan raportteja järjestelmän käytöstä. Näistä raporteista käy ilmi esimerkiksi tuotteiden tuonnit ja viennit, käyttäjien saldot järjestelmässä ja tuotteiden hintoihin tehdyt muutokset.

Raportointiliittymää käytetään myös järjestelmän yleiseen hallintaan. Sen avulla pääkäyttäjät voivat poistaa käyttäjiä järjestelmästä ja tehdä tuotteisiin liittyviä muutoksia, joita ovat esimerkiksi tuoteryhmien ja tuotteiden lisäys ja poisto. Lisäksi on mahdollista muuttaa tuotteiden ominaisuuksia, kuten tuotteen ostohintaa ja tuotteen tuonnista annettavaa hyvitystä.

6.1 PHP:n erityisominaisuuksia

Luokkien metodit ovat kaikki public-metodeita, koska PHP:n versio 4.x ei tue private-metodeita. Samoin luokissa olevat muuttujat ovat public-muuttujia. Muuttujien nimissä käytetään ns. unkarilaista notaatiota, jolla kerrotaan muuttujan tyyppi. Esimerkiksi $a_ on array- tyyppinen muuttuja, $s_ on string-tyyppinen muuttuja. Muuttujat array vastaavat Javan list muuttujia.

Boolean-muuttujia ei käytetä, koska PHP ei tunnista **true** ja **false** -arvoja oikein. Tämä on huomattu useissa käytännön toteutuksissa. Sen takia on parempi palauttaa kokonaisluku (int) 0 tai kokonaisluku (int) 1, joiden arvoja on helpompi tunnistaa PHP kielellä. 0 tarkoittaa raportointiliittymässä boolean muuttujaa **false** ja 1 tarkoittaa **true**.

6.2 Luokkien ja tiedostojen toiminta

Linkeissä kulkeviin muuttujiin viitataan $_GET[”muuttujan_nimi”] nimellä ja lomakkeiden välittämiin muuttujien arvoihin, joiden metodina on post, viitataan $_POST[”lomakkeen_kentän_nimi”]. Luokkien sisälle nämä arvot viedään asettamalla luokan sisäisen muuttujan $a_get arvoksi $_GET ja $a_post arvoksi $_POST. Luokkien ulkopuolella muuttujiin viitataan $_GET[”muuttujan_nimi”] arvolla ja luokkien sisällä $this->a_get[”muuttujan_nimi”] arvolla. Post muuttujaan viitataan vastaavasti.

6.2.1 Istunnonhallinta

Istunnonhallinta toteutetaan PHP:n $_SESSION muuttujien avulla. Kun käyttäjä ensi kertaa tulee raportointiliittymän aloitussivulle, hänelle luodaan istunto, joka on voimassa raportointiliittymässä. $_SESSION["user"] -muuttujassa on käyttäjän tunnus jolla käyttäjä tunnistetaan. Mikäli käyttäjää ei tunnisteta, hänet ohjataan ulos

raportointiliittymästä ja hänet yritetään tunnistaa uudelleen TKTL:n käyttäjätunnusta vastaan. Istunnon voimassaoloaika riippuu PHP:n asennuksesta ja sen pituus on php.ini-tiedostossa määritellyn istunnon maksimipituus.

Istuntoa hallitaan tiedostossa session.php, joka liitetään jokaisen raportointiliittymään juuressa olevan sivun alussa. Tiedostossa session.php on istunnon tarkistus, kielenvalinnan hallinta ja tietokantayhteyden avaus.

6.2.2 Kielenvalinta ja kielen vaihtaminen

Käyttäjän kirjautuessa ladataan tietokannasta käyttäjän kieli. Se tallennetaan muuttujaan $_SESSION["lang"]. Raportointiliittymän käyttämät sanat ovat tiedostoissa include/fin.php, include/swe.php, include/eng.php. Jokaisessa tiedostossa on luokka Language, jota käytetään raportointiliittymässä. Istunnonhallinnassa liitetään käyttäjän kielen mukainen tiedosto. Kieltä voi vaihtaa klikkaamalla raportointiliittymän oikeassa kulmassa lipun kuvaa, joka vaihtaa istunnossa valittua kieltä karusellin omaisesti. Suomi -> Ruotsi -> Englanti -> Suomi -> ja niin edelleen.

6.3 Raportointiliittymän hakemistorakenne

Hakemistorakenne on pidetty mahdollisimman yksinkertaisena. Itse ohjelman pääsivut sijaitsevat hakemistopuun juuressa ja eri hakemistoihin on sijoitettu tietyn tyyppistä tietoa. Tämä helpottaa ohjelman rakenteen pitämistä mahdollisimman siistinä ja helposti hallittavana.

6.3.1 Alihakemistot

beans

Hakemistossa on PHP:n luokat, joilla hallinnoidaan raportointiliittymän tietoja. Hakemistossa ovat productBean, userBean, sekä reportBean.

connection

Hakemistossa on tietokantayhteyden luomiseen sekä tietokantakyselyiden suorittamiseen tarvittavat tiedostot. Connection.php, Connection_oracle.php ja Connection_postgres.php

css

Hakemistossa raportointiliittymän tyyli- ja Javascript-tiedostot.

pics

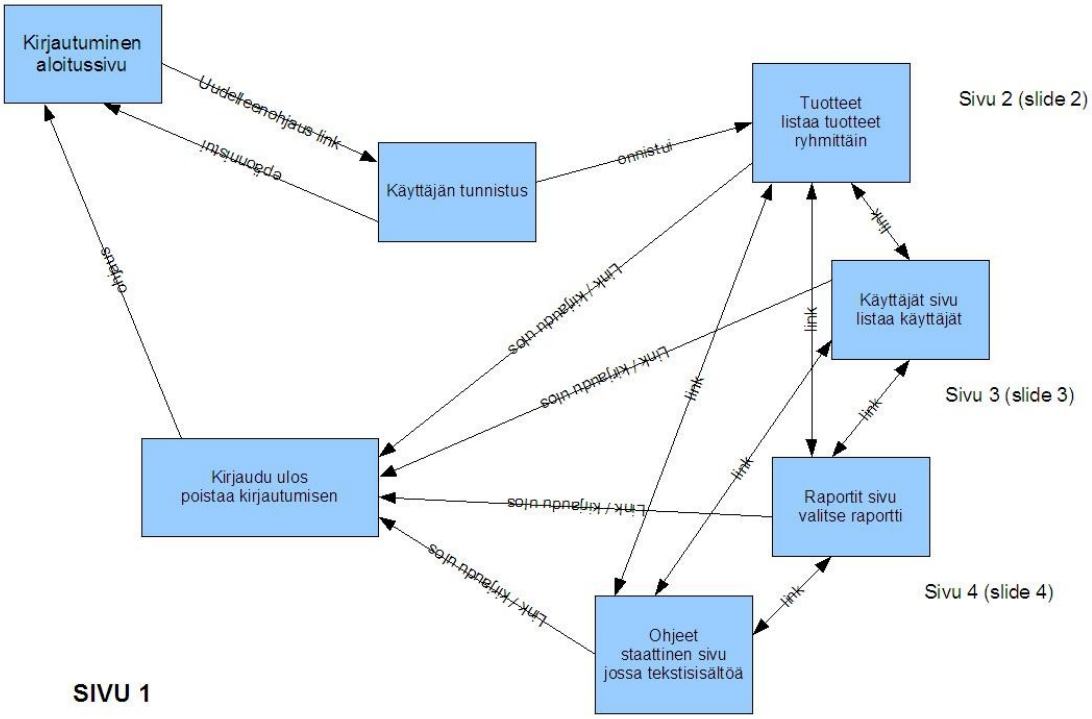
Hakemistossa on raportointiliittymän käyttämät kuvat.

include

Hakemistossa on istunnon hallintaan tarvittava tiedosto session.php sekä kielenhallintaan tarvittavat tiedostot fin.php, swe.php ja eng.php.

6.3.2 Tiedostopuun juuressa olevat tiedostot

Seuraavaksi on kuvattu ne tiedostot, jotka ovat tiedostopuun juuressa. Nämä tiedostot toteuttavat käyttöliittymän pääsivut ja sisältävät ohjelman pääasiallisen toteutuksen. Sivujen väliset siirtymät on kuvattu kuvissa 5, 6, 7 ja 8.



Kuva 5: Aloitusnäkymä

index.php

Tiedosto, joka ohjaa kirjautumiseen. Tiedosto ohjaa Apachen kirjautumisikkunaan, jossa annetaan tietojenkäsittelytieteen laitoksen tunnus ja salasana. Tunnistuksen jälkeen siirrytään itse sovellukseen ja sivulle product.php.

product.php

Tiedosto joka listaa tuotteet ryhmittäin. Kun klikkaa ryhmän nimeä, ryhmän tuotteet listataan ryhmän nimen alla. Kun tuotteen nimeä klikataan tuotteen tiedot ladataan ja siirrytään sivulle editProduct.php. Sivulta pääsee tuotteen tiedon lisäämislomakkeeseen addProduct.php painamalla ”lisää tuote” linkkiä ja tuotteen poistamiseen tarkoitetulta sivulle removeProduct.php. editProduct.php ja removeProduct.php sivulle välitetään GET-parametrissa tuotteen id.

editProduct.php

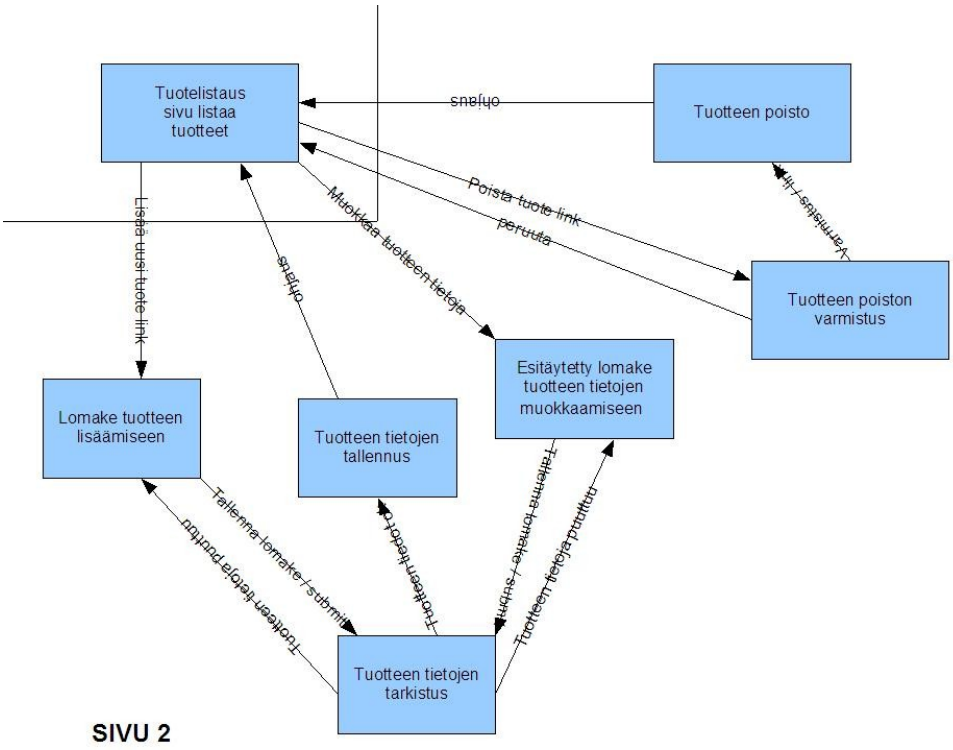
Sivulla näytetään tuotteen tiedot lomakkeessa. Tuotteen tiedot ladataan GET-parametrin mukaan. Kun tuotteen tietoja muokataan ja klikataan tallenna, tuotteen tiedot tarkistetaan ja jos pakolliset tiedot on syötetty, tuotteen tiedot tallennetaan. Tiedot välitetään html-lomakkeella POST metodilla editProduct.php-tiedostolle. Tallennuksen jälkeen siirrytään product.php-sivulle jossa näytetään tieto että tuotteen tiedot tallennettu. Jos kaikkia tuotteen pakollisia tietoja ei ole syötetty, palataan takaisin lomakkeeseen virheellisine tietoineen ja lomakkeen virheellisen kentän kohdalla tulostetaan virhetieto. Lomakkeesta pääsee sivulle product.php klikkaamalla ”Peruuta” nappia.

addProduct.php

Sivulla näytetään tyhjä lomake, johon voidaan syöttää tuotteen tiedot. Tuotteen lisäyksessä tehdään samat tarkistukset kuin editProduct.php-sivun kohdalla. Tarkistus ja tuotteen lisäys tehdään sivulla addProduct.php. Virhetilanteessa tulostetaan lomake, jossa edellisessä tilanteessa syötetyt tiedot ja virheteksti väärin syötettyjen kenttien kohdalla. Lomakkeesta pääsee sivulle product.php klikkaamalla ”Peruuta” nappia.

removeProduct.php

Sivulle tullaan product.php-sivulta. Tälle sivulle välitetään GET-parametreissa tuotteen id. Sivulla kysytään varmistus halutaanko tuote poistaa vai ei. Jos varmistusta ei anneta, käyttäjä ohjataan ei-linkillä takaisin product.php sivulle. Mikäli varmistus annetaan, tuote passivoidaan asettamalla tuote passiiviseksi. Sivulla annetaan tieto tuotteen passivoitumisesta. Sivulta pääsee takaisin product.php sivulle linkillä ”ei”.



Kuva 6: Tuotetietojen hallinta

user.php

Sivulla haetaan käyttäjiä tietokannasta. Lomakkeeseen annetaan käyttäjätunnuksen osa, jolla haetaan käyttäjää tietokannasta. Hakutuloksissa hakusana on lihavoitu jokaisen käyttäjän nimessä. Esim. Hakusana ”ayt” palauttaa tuloksinaan käyttäjät ”**ayt**oma”, ”k**ayt**taja” ja ”m**ayt**”, mikäli edellä mainitut käyttäjät löytyvät tietokannasta. Hakutulokset näytetään aakkosjärjestyksessä. Sivulta pääsee muokkaamaan, lisäämään ja poistamaan käyttäjiä. Muokkaus tapahtuu tiedostossa editUser.php, lisäys addUser.php ja poisto removeUser.php tiedostossa. Tiedostoon editUser.php ja removeUser.php mentäessä välitetään käyttäjän tunniste (id) GET muuttujissa, jolla käyttäjän tiedot ladataan tietokannasta.

addUser.php

Sivulla tulostetaan tyhjä lomake, joka tallennettaessa tarkistaa käyttäjän tiedot. Mikäli tiedoissa on jotain huomautettavaa, käyttäjä ohjataan takaisin tälle sivulle, jossa näytetään käyttäjän tiedot kentissä ja virheteksti väärin syötettyjen kenttien vieressä. Mikäli kaikki tiedot ovat kunnossa, käyttäjän tiedot tallennetaan addUser.php tiedostossa ja palataan user.php-sivulle jossa näytetään "käyttäjän tiedot lisätty" teksti. Lomakkeesta pääsee takaisin sivulle user.php klikkaamalla ”Peruuta” nappia.

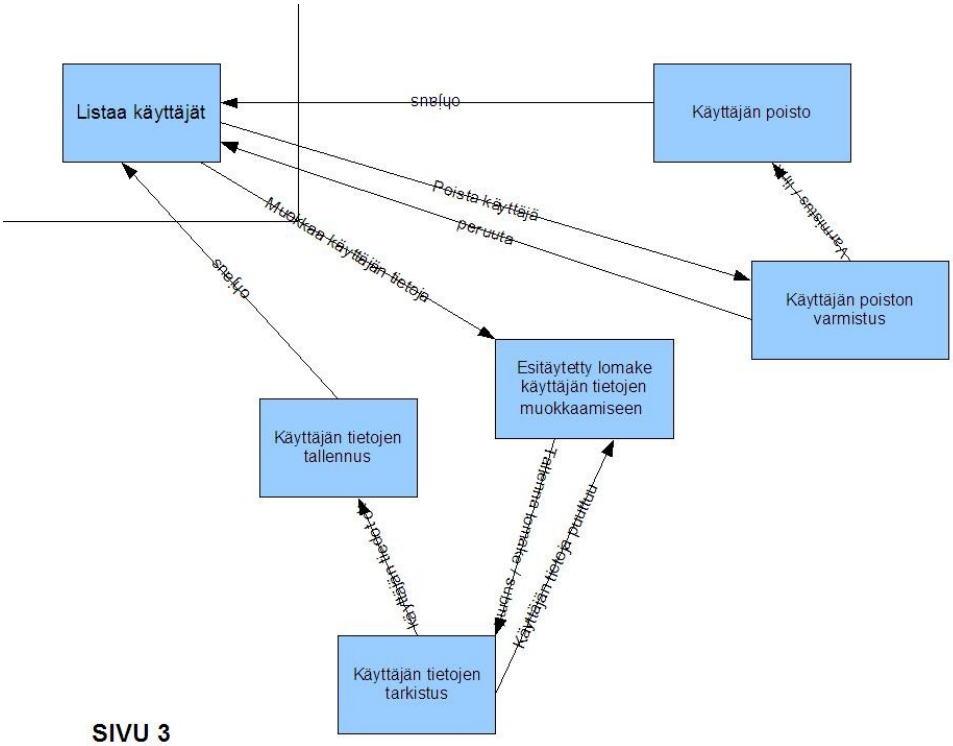
editUser.php

Sivulla näytetään käyttäjän tiedot tietokannasta ladattuna. Sivulle tullaan GET muuttujan username kanssa, joka viittaa käyttäjän tunnukseen. Sivulla voidaan muokata

käyttäjän tietoja ja mikäli niissä on virheitä, käyttäjä ohjataan takaisin tälle sivulle siten, että virheellisten tietojen kenttien vieressä näytetään virheteksti. Mikäli kaikki tiedot ovat kunnossa, tiedot tallennetaan tietokantaan ja käyttäjä ohjataan user.php-sivulle jossa näytetään infoteksti "käyttäjän tiedot tallennettu". Lomakkeesta pääsee sivulle user.php klikkaamalla ”Peruuta” nappia. Raportointiliittymään kirjautunut ei voi poistaa pääkäyttäjätunnuksiaan tällä sivulla.

removeUser.php

Sivulla kysytään varmistus käyttäjän poistamisesta. Mikäli käyttäjän poistosta saadaan varmistus, käyttäjän tiedot poistetaan tietokannasta. Sivulta on paluu linkki ”ei”, jota klikkaamalla pääsee takaisin user.php sivulle. Samoin kuin edtiUser.php, sivulle välitetään GET muuttujissa käyttäjätunnus username. Raportointiliittymään kirjautunut ei voi poistaa itseään tällä sivulla.



Kuva 7: Käyttäjätietojen hallinta

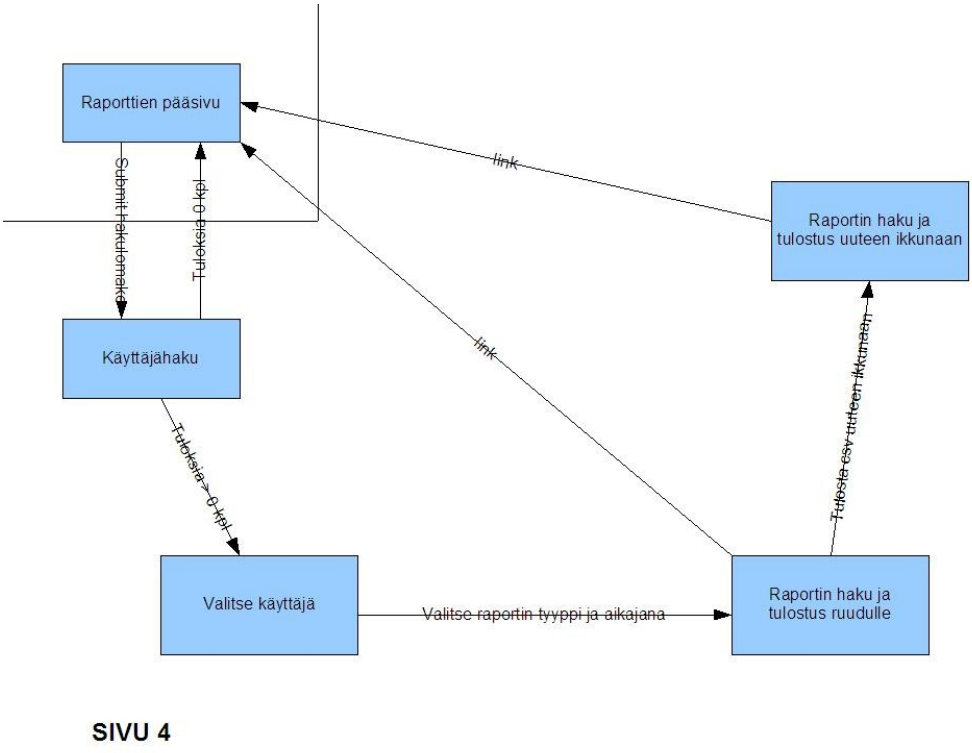
report.php

Sivulla haetaan käyttäjiä tietokannasta. Lomakkeeseen annetaan käyttäjätunnuksen osa, jolla haetaan käyttäjää tietokannasta. Hakutuloksissa hakusana on lihavoitu jokaisen käyttäjän nimessä. Esim. Hakusana ”ayt” palauttaa tuloksinaan käyttäjät ”**ayt**oma”, ”k**ayt**taja” ja ”m**ayt**”, mikäli edellä mainitut käyttäjät löytyvät tietokannasta. Hakutulokset näytetään aakkosjärjestyksessä. Ensin valitaan käyttäjät klikkaamalla valintalaatikoista käyttäjät, joiden tapahtumat halutaan listattavan. Kun käyttäjät on

valittu, valitaan alasveto valikoista raportin tulostuksen aloitusaika ja lopetusaika.

Kun lomake lähetetään, sivu näyttää tietokannasta raportin listauksena. Sivun yläreunaan tulostetaan linkki "csv-export", jota klikkaamalla aukeaa uusi ikkuna, joka tulostaa saman raportin csv-erotinmerkkien kanssa. Tältä sivulta käyttäjä voi tallentaa sivun omalle koneelleen ja katsella sitä esim. Excel-taulukkolaskentaohjelmassa.

Jokaiselle eri raportin tyypille tehdään oma sivu, joka näkyy raportointiliittymän alaotsikkona.



Kuva 8: Raporttien hallinta

instructions.php

Sivulla on ohjeet raportointiliittymän käytöstä, eli ns. pikaohje. Sivulta on myös linkit kokonaiseen käyttöohjeeseen joka on tallennettu pdf muodossa raportointiliittymän saataville.

6.4 Raportointiliittymän luokat

Raportointiliittymän luokat sisältävät raportointiliittymässä käytettävät monimutkaisemmat tieto-oliot. Näitä luokkia ovat, Language, Connection, Logger, Product, ProductGroup ja Report. Seuraavaksi esitellään jokainen näistä luokista ja niiden rakenne.

6.4.1 Language

Luokka Language pitää sisällään yhden kielen sanat raportointiliittymään. Jokainen sana on omassa string-muuttujassaan. Luokasta on kolme ilmentymää tiedostoissa include/fin.php, include/swe.php ja include/eng.php. Vain yksi luokka liitetään jokaisen php-tiedoston alussa. Luokan muuttujia kutsutaan raportointiliittymän php-tiedostoissa seuraavasti.

include('include/fin.php'); $lang = new Language(); echo $lang->s_login;

Luokalla ei ole metodeita, vaan kaikkia muuttujia kutsutaan suoraan edellä mainitulla tavalla. Luokan muuttujien arvot ovat julkisia muuttujia jotka sisältävät kyseisen kielen tekstit.

Kielivalinta pysyy sessiossa ja sitä käytetään seuraavasti

Vaihe 1: käyttäjän kielen tieto on saatu (fin, eng, swe)
Jos se on fin, sivun alussa "otetaan käyttöön" eli includataan tiedosto fin.php, jos tuo kieli on eng niin eng.php.

Nuo tiedostot fin ja eng ovat identtisiä paitsi että nuo luokan sisäiset muuttujat sisältävät fin.php tiedostossa suomen kielen sanoja ja eng.php tiedostossa englannin kielen sanoja.

Eli ensin luodaan ilmentymä luokasta Language (Tämä voidaan tehdä on sivulla "otettu käyttöön" fin.php tai eng.php, koska molemmissa on sama luokka)

sitten kun sivulla halutaan tulostaa esim. html lomake

SUOMALAISELLE:
<html>
Tunnus:
<input type=text value="">
Tunnus:
<input type=password value="">
</html>

ENGANNINKIELISELLE
<html>
Login:
<input type=text value="">
Password:
<input type=password value="">
</html>

TÄMÄ voidaan toteuttaa PHP:lla seuraavasti:

<?php
if($_SESSION[”language”] == "fin"){
include("include/fin.php");

```
}else if($_SESSION[”language”] == "eng"){
include("include/eng.php");
}
// luodaan kielestä luokka
$lang = new Language();

?>

<html>
<?echo $lang->s_login;?>:
<input type=text value="">
<?echo $lang->s_password;?>:
<input type=password value="">
</html>
```

TAI

```
<html>
<?echo $lang->gettext(”s_login”);?>:
<input type=text value="">
<?echo $lang->gettext(”s_password”);?>:
<input type=password value="">
</html>
```

Muuttujat

$s_login

Muuttuja pitää sisällään ”Kirjautuminen” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Login”.

$s_password

Muuttuja pitää sisällään ”Salasana” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Password”.

$s_report

Muuttuja pitää sisällään ”Raportti” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Report”.

$s_username

Muuttuja pitää sisällään ”Tunnus” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Login”.

$s_firstname

Muuttuja pitää sisällään ”Etunimi” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Firstname”.

$s_lastname

Muuttuja pitää sisällään ”Sukunimi” suomenkielisessä Language tiedostossa. Sama muuttuja Englanninkielisessä versiossa muuttuja pitää sisällään tekstin ”Lastname”.

$s_... ...

Muuttujia lisätään kielitiedostoon sitä mukaa kuin raportointiliittymässä lisätään uusia tekstejä.

Metodit

public String gettext(String text);

Metodi palauttaa halutun tekstin ja mikäli metodi ei löydä syötettyä muuttujaa, se palauttaa virheilmoituksen.

6.4.2 Connection

Luokka Connection hoitaa tietokantayhteyden ottamisen, yhteyden sulkemisen sekä kyselyiden välittämisen tietokannalle. Connection-tiedostosta tehdään kaksi versiota, toinen Postgres-kantaa ja toinen Oracle-kantaa varten. Luokka otetaan käyttöön nimeämällä tiedostot uudelleen. Käytössä oleva tiedosto on nimetty Connection.php. Luokasta on lisäksi kaksi muuta tiedostoa Connection_postgres.php ja Connection_oracle.php. Muiden tietokantojen lisääminen tapahtuu kirjoittamalla uusi Connection tiedosto joka toteuttaa metodit samalla tavoin kuin Connection_postgres.php ja Connection_oracle.php.

Luokan metodeita käytetään muiden luokkien sisällä jotka huolehtivat sql-kyselyiden oikeasta muodosta.

Muuttujat

$r_conn

Tietokantayhteys, jonka sql_connect() funktio palauttaa.

Metodit

public int startConnection()

Metodi avaa tietokantayhteyden ja palauttaa 1 onnistumisen merkiksi, muuten 0.

public void endConnection()

Metodi sulkee tietokantayhteyden

public list[] executeSelect(String query)

Metodi suorittaa annetun kyselyn PostgreSQL tai Oracle -tietokantafunktiolla ja palauttaa tulosrivit listana. Listaa käydään läpi koodissa foreach($row AS $row){} komennoilla. Eli funktio foreach käy rivit läpi, rivi kerrallaan ja saatuihin arvoihin voidaan viitata $row[”kantasarakkeennimi”].

public int executeDelete(String query)

Metodi suorittaa annetun tietokantarivin poistamiskyselyn ja palauttaa 1 onnistuessaan, muuten 0. Jos tietokantakysely epäonnistuu, eli kysely ei ole validi, metodi käyttää PHP:n die funktiota, jossa tulostetaan annettu kysely käyttäjälle.

public int executeUpdate(String query)

Metodi suorittaa annetun tietokantarivin poistamiskyselyn ja palauttaa 1 onnistuessaan, muuten 0. Jos tietokantakysely epäonnistuu, eli kysely ei ole validi, metodi käyttää

PHP:n die funktiota, jossa tulostetaan annettu kysely käyttäjälle.

public int executeInsert(String query)

Metodi suorittaa annetun tietokantarivin poistamiskyselyn ja palauttaa 1 onnistuessaan, muuten 0. Jos tietokantakysely epäonnistuu, eli kysely ei ole validi, metodi käyttää PHP:n die funktiota, jossa tulostetaan annettu kysely käyttäjälle.

6.4.3 Logger

Luokka Logger hoitaa raportointipuolen muutoksien kirjaamisen tietokantaan. Se myös palauttaa tarvittavan lokitiedon tietokannasta. Luokkaa kutsutaan productBeanissa, kun raakatuotteen arvoja muutetaan.

Muuttujat

$s_username

Käyttäjätunnus, jonka toimia kirjataan lokeihin.

Metodit

public void logRawProductValueLog(int $i_rawProductId, int $i_productSizeId, double$d_oldValue, double $d_newValue)

Metodi tallentaa uuden rivin tietokantaan, tauluun RawProductValueLog, annetuilla arvoilla, asetetulla käyttäjällä ja uudella id:llä, sekä nykyisellä aikaleimalla

public void logFinalProductValueLog(int $i_finalProductId, int $i_productSizeId, double$d_oldValue, double$d_newValue)

Metodi tallentaa uuden rivin tietokantaan, tauluun FinalProductValueLog, annetuilla arvoilla, asetetulla käyttäjällä ja uudella id:llä, sekä nykyisellä aikaleimalla

public void logGroupProductChangeLog(int $i_groupId, String $s_oldValue, String $s_newValue)

Metodi tallentaa uuden rivin tietokantaan tauluun GroupProductChangeLog annetuilla arvoilla, asetetulla käyttäjällä ja uudella id:llä, sekä nykyisellä aikaleimalla

public void logRawProductChangeLog(int $i_rawProductId, int $i_productSizeId, String $s_oldValue, String $s_newValue)

Metodi tallentaa uuden rivin tietokantaan tauluun RawProductChangeLog annetuilla arvoilla, asetetulla käyttäjällä ja uudella id:llä, sekä nykyisellä aikaleimalla

public void logFinalProductChangeLog(int $i_finalProductId, String $s_oldValue, String $s_newValue)

Metodi tallentaa uuden rivin tietokantaan tauluun FinalProductChangeLog annetuilla arvoilla, asetetulla käyttäjällä ja uudella id:llä, sekä nykyisellä aikaleimalla

public list[] getEvents(String $s_tableName, String $s_username, timestamp $t_acttime_start, timestamp $t_acttime_stop)

Metodi palauttaa annetusta taulusta rivejä arrayssa. Jos $s_username on annettu, metodi palauttaa vain sen käyttäjän rivit. Jos $t_acttime_start on annettu, metodi palauttaa siitä aikaleimasta eteenpäin kaikki rivit. Jos $t_acttime_stop on annettu, metodi palauttaa siitä aikaleimasta taaksepäin kaikki rivit. Jos kumpaakaan $t_acttime muuttujaa ei ole annettu, metodi palauttaa annetusta taulusta kaikki rivit.

6.4.4 Product

Luokka Product hoitaa tuotteeseen liittyvät tapahtumat. Sen kautta lisätään, muokataan ja poistetaan tuotteita tietokannasta. Se myös lataa tietokannasta tuotteen tiedot. Luokan kautta ladataan ja muokataan raakatuotteen tiedot ja lopullisen tuotteen tiedot. Luokassa käytetään Connection ja Logger luokkia.

Muuttujat

var $a_get

var $a_post

var $a_files

Metodit

public array[] loadFinalProducts()

Metodi lataa kaikki lopputuotteet tietokannasta ja palauttaa ne listana. Listassa ei ole mukana raakatuotteen tietoja, vaan ne saadaan toisella metodilla loadRawProducts().

public array[] loadRawProducts()

Metodi lataa kaikki raakatuotteet tietokannasta ja palauttaa ne listana.

public void setGetVars(array[] GET)

Metodissa asetetaan _GET muuttujat luokan sisään.

public void setPostVars(array[] POST)

Metodissa asetetaan _POST muuttujat luokan sisään.

public void setFileVars(array[] FILES)

Metodissa asetetaan _FILES muuttujat luokan sisään. Muuttujia tarvitaan kuvan lataamisessa.

public array[] loadFinalProduct(int id)

Metodi lataa tuotteen tietokannasta ja palauttaa tuotteen tiedot arrayssa. Jos tuotteen

lataus epäonnistui, palauttaa tyhjän arrayn.

public int saveFinalProduct()

Metodi tallentaa tuotteen tiedot POST muuttujien mukaan ja palauttaa 1 mikäli tuotteen tallennus onnistui, muuten 0..

public void removeFinalProduct(int id)

Metodi poistaa tuotteen, eli asettaa sen ei-aktiiviseksi.

public int addFinalProduct()

Metodi lisää tuotteen ja palauttaa 1 mikäli tuotteen lisäys onnistui, muuten 0.

public array[] validateFinalProduct()

Metodi tarkistaa annetut tuotteen tiedot luokan sisäisestä $a_post muuttujasta. Jos kaikki kentät ovat kunnollisia, palauttaa tyhjän listan. Jos metodi huomaa virheen jossain kentässä, se palauttaa kentän nimen kohdalla virheilmoituksen, joka kertoo mikä kentässä on vikana. Toisin sanoen, jos kenttä nimi on tyhjä, metodi palauttaa listassa kohdalla ["nimi"] = "Kenttä tyhjä";

public array[] loadRawProduct(int id)

Metodi lataa raakatuotteen tietokannasta ja palauttaa tuotteen tiedot arrayssa. Jos tuotteen lataus epäonnistui, palauttaa tyhjän arrayn.

public int saveRawProduct()

Metodi tallentaa raakatuotteen tiedot $a_post muuttujien mukaan.

public void removeRawProduct(int id)

Metodi poistaa raakatuotteen.

public int addRawProduct()

Metodi lisää raakatuotteen ja palauttaa 1 mikäli tuotteen lisäys onnistui.

public array[] validateRawProduct()

Metodi tarkistaa annetut raakatuotteen tiedot POST muuttujasta. Jos kaikki kentät ovat kunnollisia, palauttaa tyhjän listan. Jos metodi huomaa virheen jossakin kentässä, se palauttaa kentän nimen kohdalla virheilmoituksen, joka kertoo mikä kentässä on vikana. Toisin sanoen, jos kenttä nimi on tyhjä, metodi palauttaa listassa kohdalla ["nimi"] = "Kenttä tyhjä";

6.4.5 User

Luokassa hoidetaan käyttäjien tietojen muokkaus, poisto ja lataus. Luokassa **ei ole** uuden käyttäjän lisäämistä, koska tämä tapahtuu asiakasliittymässä. Käyttäjän käyttäjätunnusta ei voi muuttaa, eikä omia pääkäyttäjän oikeuksia voi poistaa. Myöskään itseään ei voi poistaa.

Muuttujat

var array[] $a_post

var array[] $a_get

Metodit

public void setGetVars(array[] GET)

Metodissa asetetaan _GET muuttujat luokan sisään.

public void setPostVars(array[] POST)

Metodissa asetetaan _POST muuttujat luokan sisään.

public array[] loadUser(int id)

Metodi palauttaa käyttäjän tiedot arrayssa annetun id:n perusteella. Jos tietoja ei löydy palauttaa tyhjän arrayn.

public int saveUser()

Metodi tallentaa käyttäjän tiedot $a_post muuttujien mukaan. Jos käyttäjä yrittää muokata itseltään pois pääkäyttäjä oikeuksia, palauttaa 0. Muut tiedot tallentuvat annettujen tietojen mukaan. Jos tallennus onnistuu, palautetaan 1, muuten 0.

public int removeUser(String username)

Käyttäjän tiedot poistetaan järjestelmästä, ja tieto että kyseinen käyttäjä on poistettu, lisätään tietokannan lokitauluun. Jos itsensä yrittää poistaa, palauttaa 0 ja poistoa ei tehdä.

public int addGroup()

Metodi lisää uuden ryhmän tietokantaan muuttujien $a_post mukaan. Jos lisäys onnistuu, metodi palauttaa 1, muuten 0.

public int saveGroup()

Metodi muokkaa ryhmän nimiä ja saldoa, $a_post muuttujien mukaan. Jos muokkaus onnistuu niin palauttaa 1, muuten 0.

public array[] loadGroup(int groupid)

Metodi palauttaa ryhmän tiedot arrayssa. Jos lataus epäonnistuu, palauttaa tyhjän arrayn.

6.4.6 Report

Luokka Report hoitaa raporttien tulostamisen ja hakemisen tietokannasta. Se tulostaa halutunlaisia raportteja ulos. Luokka hoitaa myös csv-muotoisen raportin tulostamisen.

Muuttujat

var array[] $a_post

var array[] $a_get

var String $c_colSeparator

csv-tulosteen sarake-erotinmerkki

var String $c_rowSeparator

csv-tulosteen rivierotinmerkki

var array[] $a_excludeFromReport

Muuttujassa on niiden sarakkeiden nimet, joita ei tulosteta raporttiin. Jos lista on tyhjä, kaikki sarakkeet tulostetaan raporttiin

Metodit

public void setGetVars(array[] GET)

Asetetaan GET-muuttujat, joiden mukaan raportteja tulostetaan.

public array[] getReport(String $s_tablename, timestamp $t_acttime_start, timestamp $t_acttime_stop, array[] $s_users)

Palauttaa valitun raportin listassa ilman erotinmerkkejä. Jos $t_acttime_start on annettu, palautetaan tapahtumat siitä ajasta eteenpäin. Jos $t_acttime_stop on annettu, palautetaan ennen sitä tapahtuneet tapahtumat. Jos molemmat acttime-muuttujat ovat tyhjät, palautetaan kaikki tapahtumat. $s_users arrayssa on raporttiin tulostettavien käyttäjien käyttäjätunnukset.

public String getSeparatorReport(String $s_tablename, timestamp $t_acttime_start,timestamp $t_acttime_stop)

Palauttaa valitun raportin tekstinä erotinmerkkien kanssa. Jos $t_acttime_start on annettu, palautetaan tapahtumat siitä ajasta eteenpäin. Jos $t_acttime_stop on annettu, palautetaan ennen sitä tapahtuneet tapahtumat. Jos molemmat acttime-muuttujat ovat tyhjät, palautetaan kaikki tapahtumat. $s_users arrayssa on raporttiin tulostettavien käyttäjien käyttäjätunnukset.

7 Asiakasliittymän luokkakuvaukset

Tässä osiossa kuvataan aiemmin esiteltyjen Asiakasliittymän komponenttien luokat. Luettavuuden parantamiseksi komponenttien luokkakaaviot ovat tämän dokumentin liitteenä. Luokat on laitettu paketteihin siten, että asiakasliittymän juuripaketti on nimeltään *cafe* ja eri komponentit ovat sen alaisissa paketeissa. Kunkin komponentin kohdalla on mainittu paketti, jossa sen luokat ovat.

7.1 MainProgram-komponentti

Paketti: main

MainProgram-komponentti koostuu kolmesta luokasta. Se toteuttaa UserInterfaceEvent-rajapinnan GUI-komponentin ja FingerprintEvent-rajapinnan FingerprintAuthentication-komponentin käytettäväksi.

MainProgram-komponentti itse käyttää GUI:n toteuttamaa UserInterface-rajapintaa, Database-komponentin Database-rajapintaa, Settings-komponentin toteuttamaa Configuration-rajapintaa, Authentication-komponentin toteuttamaa UserAuthentication-rajapintaa ja FingerprintAuthentication-komponentin toteuttamaa Fingerprint-rajapintaa.

Komponentin UML-kaaviossa (liite A, kuva 2) on esitelty sen luokkien väliset suhteet ja tärkeimmät metodit. Lisäksi siitä näkee helposti komponentin käyttämät ulkoiset luokat.

7.1.1 CoffeeTouch

CoffeeTouch on komponentin ja ohjelman pääluokka, joka käynnistää muut komponentit, ja lisäksi välittää tarpeen vaatiessa käyttöliittymälle virheilmoituksen.

Muuttujat

protected Settings settings;

Ilmentymä Settings-luokasta.

protected GuiMain gui;

Ilmentymä GUI-komponentin pääluokasta.

protected FingerprintAuthentication fpa;

Ilmentymä FingerprintAuthentication-komponentin pääluokasta.

protected DBCommunication db;

Ilmentymä Database-komponentin pääluokasta.

protected MainGuiHandler guiHandler;

Ilmentymä Main-komponentin apuluokasta.

protected MainFingerprintHandler fpaHandler;

Ilmentymä Main-komponentin apuluokasta.

Metodit

public CoffeeTouch(void);

Konstruktorissa luodaan ensin uusi ilmentymä Settings-luokasta, jolta haetaan DBSettings-olio. Tämän jälkeen luodaan ilmentymä DBCommunication-luokasta, jolle annetaan DBSettings parametrina. Seuraavaksi luodaan ilmentymä MainGuiHandler-luokasta, jolle annetaan parametrina Database-rajapinnan toteuttava DBCommunication-olio ja "**this**".

Seuraavaksi luodaan GuiMain-olio, joka saa MainGuiHandler-olion ja Settings-luokasta haettavan GUISettings-olion parametrina. Viimeiseksi luodaan MainFingerprintHandler (parametreina UserInterface-rajapinnan toteuttava GuiMain, Database-rajapinnan toteuttava DBCommunication ja "**this**"), ja FingerprintAuthentication, jolle MainFingerprintHandler ja Settings-luokasta haettava FingerprintSettings annetaan parametreina.

Konstruktorin on käsiteltävä Settings-luokasta mahdollisesti syntyvät poikkeukset (IOExcepion, FileNotFoundException tai InvalidPropertiesFormatException). Virheilmoitus näytetään käyttäjälle kutsumalla displayPopup-metodia. Muuta ohjelmaa ei käynnistetä, mikäli Settingsistä syntyy poikkeus. Lisäksi on käsiteltävä DBCommunicationilta tuleva DBConLostException, jos kantaan ei ole saatu yhteyttä.

public CoffeeTouch(String settingsFileUri);

Main-metodi kutsuu tätä konstruktoria, jos ohjelman käynnistämisen yhteydessä on annettu parametrina asetustiedoston osoite. Tämä toimii muuten kuten yllä kuvattu konstruktori, mutta Settings-luokasta kutsutaan konstruktoria, jolle asetustiedoston osoite voidaan antaa parametrina.

public void deleteFingerprint(int id);

Kutsuu FingerprintAuthentication-olion (fpa) samannimistä metodia (poistetaan sormenjälkitunniste välimuistista jäljen id-numeron perusteella).

public void addFingerprint(FPData data);

Kutsuu FingerprintAuthentication-olion (fpa) samannimistä metodia (lisätään uusi sormenjälkitunniste välimuistiin).

public void displayDBError(String messageid);

Mikäli MainGuiHandler tai MainFingerprintHandler saavat poikkeuksen tietokantaluokalta (ts. tietokantayhteys on rikki), ne kutsuvat tätä metodia. Metodissa kutsutaan ensin GuiMain-luokan stopForProblem(String problemstringid)-metodia antaen saatu parametri eteenpäin. Tämän jälkeen kutsutaan tietyn ajan välein Databasen reconnectDatabase()-metodia, kunnes se palauttaa "true". Lopuksi kutsutaan GUI:n continue()-metodia.

protected void displayPopup(String message);

Metodi toteuttaa virheilmoituksen piirtämisen näytölle ponnahdusikkunana. Virheilmoituksen teksti saadaan parametrina. Metodia käytetään, mikäli komponenteista tulee poikkeuksia ohjelman käynnistysvaiheessa.

public static void main(String[] args);

Luodaan uusi ilmentymä tästä luokasta ohjelman käynnistämiseksi. Parametrina voidaan antaa asetustiedoston osoite, joka välitetään Settingsin konstruktorille.

7.1.2 MainGuiHandler

Luokan tehtävä on toimia yhteytenä käyttöliittymän, tietokannan ja käyttäjätunnusten autentikoinnin välillä. Luokka toteuttaa UserInterfaceEvent-rajapinnan ja annetaan parametrina GuiMain-luokan konstruktorille, joka odottaa kyseisen rajapinnan toteuttavaa luokkaa. Luokan käyttämät (tai pikemminkin välittämät) ProductGroup, ImportProduct, CancelProduct ja ExportProduct on kuvattu luvussa 5.2, kuten myös UserInfo-luokka.

Tietokantaluokan metodeja kutsuttaessa pitää käsitellä tietokantaluokan mahdollisesti heittämä DBConLostException. Tällaisen sattuessa kutsutaan CoffeeTouch-luokan displayDBError-metodia.

Muuttujat

protected Database db;

Tässä muuttujassa on Database-rajapinnan toteuttavan tietokantaluokan ilmentymä, jolle tietokantakyselyt kohdistetaan.

protected UserAuthentication auth;

Tunnistuskomponentin UserAuthentication-rajapinnan toteuttavan luokan ilmentymä, jolta varmennetaan käyttäjätunnukset.

protected CoffeeTouch main;

Viittaus pääluokkaan tallennetaan tähän.

Metodit

public MainGuiHandler (Database db, CoffeeTouch main, AuthenticationSettings authSettings);

Konstruktori tallettaa parametrina saadun Database-rajapinnan toteuttavan luokan ja CoffeeTouchin, ja luo ilmentymän UserAuthentication-rajapinnan toteuttavasta Authentication-luokasta antaen sille AuthenticationSettingsit parametrina.

public boolean authenticateUser(String username, String password);

Kutsuu UserAuthentication-rajapinnan (auth) samannimistä metodia ja palauttaa saadun vastauksen.

public void saveUser(UserInfo user);

Kutsuu Database-rajapinnan (db) samannimistä metodia (käyttäjätietojen talletus).

public UserInfo loadUser(String username);

Kutsuu Database-rajapinnan (db) samannimistä metodia (käyttäjätietojen lataus käyttäjätunnuksen perusteella).

public void exportProduct(UserInfo user, ExportProduct product);

Kutsuu Database-rajapinnan (db) samannimistä metodia (tuotteen "ostaminen" eli vienti). Tietokanta voi heittää ProductNotFoundException-poikkeuksen, mikäli kyseinen tuote ei ole enää tietokannassa aktiivisena.

public void importProduct(UserInfo user, ImportProduct product, int amount);

Kutsuu Database-rajapinnan (db) samannimistä metodia (tuotteen tuonti). Tietokanta voi heittää ProductNotFoundException-poikkeuksen, mikäli kyseinen tuote ei ole enää tietokannassa aktiivisena.

public ExportProduct[] getProductList(void);

Kutsuu Database-rajapinnan (db) samannimistä metodia (haetaan taulu kaikista vietävistä tuotteista).

public void insertAlert(ImportProduct product);

Kutsuu Database-rajapinnan (db) samannimistä metodia (asetetaan hälytys jonkin tuotavan tuotteen loppumisesta). Tietokanta voi heittää ProductNotFoundException-poikkeuksen, mikäli kyseinen tuote ei ole enää tietokannassa aktiivisena.

public void removeAlert(ImportProduct product);

Kutsuu Database-rajapinnan (db) samannimistä metodia (poistetaan hälytys joltakin tuotteelta).

public ImportProduct[] getAlerts(void);

Kutsuu Database-rajapinnan (db) samannimistä metodia (haetaan taulu kaikista tuotteille asetetuista hälytyksistä).

public ProductGroup[] getProductGroups(void);

Kutsuu Database-rajapinnan (db) samannimistä metodia (haetaan kaikki tuoteryhmät).

public void deleteFingerprint(int id);

Kutsuu Database-rajapinnan (db) ja pääluokan (CoffeeTouch) samannimistä metodia (poistetaan sormenjälki kannasta jäljen id-numeron perusteella).

public CancelProduct cancelProductExport(UserInfo user);

Kutsuu Database-rajapinnan (db) samannimistä metodia (peruutetaan käyttäjältä viimeisin veloitustapahtuma).

7.1.3 MainFingerprintHandler

Luokan tehtävä on toimia yhteytenä sormenjälkitunnistuksen, tietokannan ja käyttöliittymän välillä. Luokka annetaan parametrina FingerprintAuthenticationin konstruktorille. Luokan välittämät FPData-luokan oliot on kuvattu luvussa 5.2. Luokka toteuttaa FingerprintEvent-rajapinnan.

Tietokantaluokan metodeja kutsuvien metodien pitää käsitellä tietokantaluokan mahdollisesti heittämä DBConLostException. Tällaisen sattuessa kutsutaan CoffeeTouch-luokan displayDBError-metodia.

Muuttujat

protected Database db;

Database-rajapinnan toteuttava luokka.

protected UserInterface gui;

UserInterface-rajapinnan toteuttava luokka.

protected CoffeeTouch main;

Viittaus pääluokkaan.

Metodit

public MainFingerprintHandler(UserInterface gui, Database db, CoffeeTouch main);

Konstruktori tallettaa parametrina saadut Database- ja UserInterface-rajapinnat toteuttavat luokat, sekä CoffeeTouch-pääluokan.

public boolean authenticatedFingerprint(int id);

Metodia kutsutaan, kun sormenjälkitunnistuskomponentti on onnistuneesti tunnistanut sormenjäljen. Parametrina saadulla id-luvulla kutsutaan Database-rajapinnan loadUser(int id):UserInfo-metodia. Lopuksi kutsutaan UserInterface-rajapinnan metodia authenticatedUser(UserInfo user, int fingerprintid) antaen saatu tietokannasta saatu UserInfo parametrina. Mikäli sormenjälkeä vastaavaa käyttäjää ei löytynyt enää tietokannasta, metodi palauttaa ”**false**”. Muuten palautetaan ”**true**”.

public void unAuthenticatedFingerprint(FPData data);

Metodia kutsutaan, kun sormenjälkitunnistuskomponentti ei kyennyt tunnistamaan lukijaan laitettua sormea. Ensin on kutsuttava GuiMainin metodia unknownUser(). Mikäli se palauttaa false, voidaan metodin suoritus lopettaa. Mikäli se taas palauttaa true, kutsutaan tämän luokan metodia saveFingerprint, jolle annetaan parametrina saatu FPData parametriksi. Lopuksi kutsutaan UserInterfacen metodia addFingerprint(id:int), jolle annetaan parametriksi saveFingerprintin palauttama int (sormenjäljen id-numero).

public FPData[] getFingerprintData(void);

Metodia kutsutaan, kun sormenjälkitunnistuskomponentti tarvitsee tietokannasta sormenjälkidataa vertailtavaksi skannattuun sormenjälkeen. Tämän metodin tehtävä on kutsua tietokantarajapinnan samannimistä metodia samoilla parametreilla ja välittää saatu vastaus takaisin.

protected int saveFingerprint(FPData data);

Metodi kutsuu tietokantarajapinnan samannimistä metodia samalla parametrilla, ja palauttaa vastauksena saadun int-muuttujan kutsujalle. Lisäksi tuo tietokannasta saatu id-numero lisätään FPData-olioon, ja kutsutaan CoffeeTouch-pääluokan metodia addFingerprint(FPData data), joka hoitaa sormenjälkitunnisteen välittämisen takaisin FingerprintAuthentication-komponentille.

7.2 Authentication-komponentti

Paketti: authentication

Komponentin tehtävä on tunnistaa käyttäjä TKTL:n käyttäjäksi vertailemalla syötettyjä tunnuksia TKTL:n vastaaviin käyttäen HTTP-protokollan basic authentication menetelmää. Komponentti käyttää apunaan ulkoista luokkaa Base64, joka hoitaa käyttäjätunnuksen ja salasanan base64-koodauksen HTTP viestiä lähetettäessä. Jotta komponentti toimisi, käyttäjän täytyy lisätä TKTL:n SSL-varmenne Java-virtuaalikoneen varmenteiden (ns. keystore) käyttämällä Javan mukana tulevaa keytool-työkalua.

Komponentin UML-luokkakaavio on dokumentin liitteenä (liite A, kuva 3).

7.2.1 Authentication

Luokan tehtävänä on tunnistaa käyttäjä HTTP-protokollan basic authentication menetelmän avulla. Toteuttaa UserAuthentication-rajapinnan.

Muuttujat

private AuthenticationSettings settings;

Sisältää URL-osoitteen, jota vasten käyttäjätunnistus tehdään, polun Java-virtuaalikoneen SSL-varmenteiden säilytystiedostoon, sekä tämän tiedoston lukemiseen vaadittavan salasanan.

private Socket mySocket;

Yhteyden muodostamisessa tarvittava Socket-olio.

private SSLSocketFactory sslfactory;

Factory-olio uusien SSLSockettejen luomista varten.

private SSLContext ctx;

Toimii factory-oliona Socket factoryille.

private KeyManagerFactory kmf;

Toimii Factory-oliona KeyManagereille.

private KeyStore ks;

Toimii varastona kryptatuille avaimille ja varmenteille.

private char[] passphrase;

Keystoren muokkaamiseen tarvittava salasana.

private TrustManagerFactory tmf;

Toimii Factory-oliona TrustManagereille.

private ConnectionData condata;

Toimii tietovarastona, jonne on tallennettu HTTP-vastauskoodi, sekä suojausalue.

private String domain;

Settingsin mukana saadun URL:n jäsennetty alkuosa (domain).

private String path;

Settingsin mukana saadun URL:n jäsennetty loppuosa.

private Scanner parser;

Scanner olio, jolla jäsennetään settingsissä saatu URL.

private String buffer;

Tähän merkkijonon luetaan tietoa httpWriterista.

private bufferedWriter httpWriter;

Käytetään HTTP-pyynnön lähettämiseen.

private bufferedReader httpReader;

Käytetään HTTP-vastauksen lukemiseen.

Metodit

public Authentication(AuthenticationSettings settings);

Konstruktori saa parametrinaan asetukset, joiden perusteella luokka tietää URL-osoitteen jonne yhteys luodaan, polun Java-virtuaalikoneen SSL-varmenteiden säilytystiedostoon, sekä tämän tiedoston salasanan.

public boolean AuthenticateUser(String username, String password);

Metodi saa parametreinaan käyttäjänimen ja salasanan String-olioina. Metodi palauttaa "**true**"-arvon, jos käyttäjän salasana ja käyttäjätunnus ovat oikein. Muussa tapauksessa metodi palauttaa "**false**"-arvon.

7.2.2 ConnectionData

Luokka toimii tietovarastona, jonne tallennetaan HTTP vastauskoodi, sekä vaikutusalue yhteyttä muodostaessa. Luokka jäsentää saadun vastauksen.

Muuttujat

private int response;

HTTP-vastauskoodi.

private String WWWauthenticate;

HTTP-vastauksesta jäsennetty suojausalue (realm).

private Scanner parser;

Käytetään HTTP-vastauksen jäsentämiseen.

Metodit

Public ConnectionData();

Luokan konstruktorissa asetetaan response nollaksi ja WWWauthenticate tyhjäksi merkkijonoksi.

void insertString(String data);

Metodi saa parametrinaan aina yhden rivin HTTP-vastauksesta, jonka se sitten jäsentää erottaen merkkijonoista HTTP-koodin ja suojausalueen.

public int getResponse();

Palauttaa HTTP-vastauskoodin.

public String getWWWauthenticate();

Palauttaa suojausalueen.

7.3 Settings-komponentti

Paketti: settings

Settings-komponentin tehtävä on kerätä ohjelman asetukset asetustiedostoista, ja välittää ne ohjelman muille komponenteille.

Komponentin UML-luokkakaavio on dokumentin liitteenä (liite A, kuva 4).

7.3.1 Settings

Komponentin ainoa luokka toteuttaa Configuration-rajapinnan.

Muuttujat

private String settingsFileUri;

Pääasetustiedoston tiedostonimi.

protected Properties mainSettings;

Pääasetustiedosto muunnettuna Properties-olioksi, joka luodaan Javan valmiista java.util.Properties-apuluokasta.

Metodit

public Settings(void) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Konstruktori kutsuu getProperties-metodia pääasetustiedoston nimellä ja tallettaa

saadun Properties-olion mainsettings-muuttujaan.

public Settings(String settingsFileUri) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Konstruktori saa parametrinaan pääasetustiedoston osoitteen ja kutsuu getProperties-metodia pääasetustiedoston nimellä ja tallettaa saadun Properties-olion mainsettings-muuttujaan.

public DBSettings getDBSettings(void) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Metodi kutsuu mainSettings-olion getProperty(String key)-metodia avaimilla "databaseuri", "dbuser", "dbpassword" ja "cancelTimeout". Lisäksi kutsutaan tämän luokan getDBQueries()-metodia. Saaduilla tiedoilla luodaan palautettava DBSettings-olio (kuvattu muualla).

public GUISettings getGUISettings(void) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Metodi kutsuu mainSettings-olion getProperty(String key)-metodia avaimilla "idletimeout", "paytimeout" ja "errortimeout". Lisäksi kutsutaan tämän luokan getLanguage()-metodia jokaisen kielen kohdalla. Saaduilla tiedoilla luodaan palautettava GUISettings-olio (kuvattu muualla).

public FingerprintSettings getFingerprintSettings(void);

FingerprintSettings-olion (kuvattu muualla) palauttamista varten kutsutaan mainSettings-oliolta getProperty(String key)-metodia avaimilla "nativelibpath" ja "licensepath".

public AuthenticationSettings getAuthenticationSettings(void);

Palauttaa AuthenticationSettings-olion (kuvattu muualla), jonka luomista varten kutsutaan mainSettings-oliolta getProperty(String key)-metodia avaimilla ”authenticationurl”, ”keystorepath” ja ”keystorepassword”.

protected Hashtable getDBQueries(void) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Metodi kutsuu mainSettingsin metodia getProperty(String key) parametrilla "queryfile" ja saa vastauksena tietokantakyselyt sisältävän tiedoston osoitteen. Tästä tiedostosta luodaan getProperties-metodilla Properties-olio, jonka propertyNames()-metodia kutsumalla saadaan kaikki avaimet sisältävä Enumeration.

Enumeration käydään läpi siten, että jokaisen avaimen kohdalla kutsutaan Properties-olion getProperty(String key)-metodia. Avain-arvo-parit laitetaan palautettavaan hajautustauluun.

protected Language getLanguage(String language) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Metodi kutsuu parametrina saadun kielen mukaisesti mainSettingsin getProperty-

metodia, esimerkiksi getProperty(stringfile-fi), ja saa vastauksena kyseisen kielen asetustiedoston osoitteen. Tämän jälkeen kutsutaan getProperties-metodia tuolla tiedostonimellä, ja saadaan vastaava Properties-olio. Tuolta Properties-oliolta kutsutaan propertyNames()-metodia, joka palauttaa Enumerationin, jossa on kaikki olion avaimet (key).

Jokaista avainta vastaava arvo kysytään Properties-oliolta, ja kielen merkkijonoja tarkoittavat avain-arvo-parit tallennetaan hajautustauluun. Lisäksi muodostetaan ImageIcon kielen kuvaan viittaavasta tiedostonimestä. Hajautustaulu ja kielen kuva annetaan parametrina Language-oliolle (kuvattu muualla), joka palautetaan metodin kutsujalle.

protected Properties getProperties(String filename) throws FileNotFoundException, IOException, InvalidPropertiesFormatException;

Ajatuksena on käyttää Javan valmista java.util.Properties-luokkaa, joka osaa hakea asetukset tietyn muotoisesta XML-tiedostosta. Metodin pitää luoda asetustiedostoon liittyvä InputStream, joka annetaan parametrina Properties-luokalle, josta saadaan asetukset sisältävä Properties-olio palautettavaksi.

7.3.2 XML-asetustiedostot

Komponentti hakee asiakasliittymän säädettävissä olevat asetukset XML-muotoisista tiedostoista. Ne noudattavat seuraavaa muotoa:

<PROPERTIES>
    <ENTRY KEY=”**avain**”>**arvo**</ENTRY>
</PROPERTIES>

Asetustiedostoja on viisi kappaletta. Pääasetustiedosto settings.xml:n osoite voidaan ohjelmalle antaa komentorivillä käynnistyksen yhteydessä. Muiden asetustiedostojen osoitteet on määritelty sen sisällä. Muut tiedostot ovat tietokantakyselyt sisältävä dbqueries.xml ja käyttöliittymän tekstit eri kielillä sisältävät english.xml, finnish.xml ja swedish.xml.

7.4 Database-komponentti

Paketti: database

Komponentti huolehtii asiakasliittymän tietokantayhteyden luomisesta ja sen tietokantakyselyistä. Sen tehtävänä on piilottaa tietokannan kanssa käytettävän SQL-kielen kyselyt muulta ohjelmalta ja muuttaa ne normaaliksi Javan metodirajapinnaksi.

Komponentin UML-luokkakaavio on dokumentin liitteenä (liite A, kuva 5).

7.4.1 DBCommunication

Luokan tarkoitus on välittää tietoa tietokannasta ohjelmalle ja ohjelmasta kantaan. Kun luokasta luodaan uusi ilmentymä, yhteys kantaan avataan ja sitä ei suljeta välissä. Kaikki kantaa käyttävät metodit koettavat käynnistää yhteyden kerran jos se on alhaalla, ja heittävät sitten vasta poikkeuksen.

Muuttujat

protected DBSettings settings;

Sisältää yhteystiedot kantaan, kantahaut sekä tiedon miten vanhan tapahtuman voi perua.

Metodit

public DBCommunication(DBSettings settings) throws DBConLostException;

Konstruktori luo tietokantayhteysolion, joka avaa yhteyden kantaan luontivaiheessa.

public UserInfo loadUser(int id) throws DBConLostException;

Palauttaa käyttäjätieto-olion sormenjäljen perusteella. Jos käyttäjää ei löydy kannasta, palautetaan null-arvo. Käyttäjä haetaan yhdistämällä userdata-taulussa olevat käyttäjätiedot fingerprint-taulussa olevaan sormenjälkitietoon finger-yhteystaulun avulla.

public UserInfo loadUser(String username) throws DBConLostException;

Palauttaa käyttäjätieto-olion tunnuksen perusteella. Jos käyttäjää ei löydy kannasta, palautetaan null-arvo. Käyttäjänimeä verrataan userdata-taulussa esiintyvään käyttäjänimeen.

public void saveUser(UserInfo user) throws DBConLostException;

Tallentaa käyttäjätieto-olion tiedot kantaan. Jos vastaavaa käyttäjää ei löydy, lisätään uusi rivi Userdata-tauluun ja lisätään yhteydet fingerprint-taulussa oleviin sormenjälkiin finger-tauluun.

public void exportProduct(UserInfo user, ExportProduct product) throws ProductNotFoundException, DBConLostException;

Lisää käyttäjälle ostotapahtuman outproduct-tauluun ja päivittää sekä käyttäjän että tuoteryhmän saldoja userbalance- ja groupdata-tauluissa. Jos tuote on poistettu käytöstä, heittää ProductNotFoundException-poikkeuksen.

public cancelProduct cancelProductExport(UserInfo user);

Peruuttaa käyttäjän viimeisimmän oston. Lisää käyttäjälle korvaavan tapahtuman outproduct-tauluun, sekä päivittää saldot userbalance- sekä groupdata-tauluissa korvatakseen tapahtuman.

public void importProduct(UserInfo user, Product product, int amount) throws ProductNotFoundException, DBConLostException;

Lisää käyttäjälle tuontitapahtumaan inproduct-tauluun. Päivittää sekä käyttäjän että tuoteryhmän saldoja userbalance- ja groupdata-tauluissa. Poistaa mahdollisen hälytyksen kyseisestä tuotteesta samalla.

public ExportProduct[] getProductList(void) throws DBConLostException;

Palauttaa listan ostettavista tuotteista. Tuotteet järjestetään käyttömäärien, jotka lasketaan outproduct-taulun tapahtumista, mukaan.

public void insertAlert(ImportProduct product) throws ProductNotFoundException, DBConLostException;

Lisää tuotteelle hälytyksen tuotteen loppumisesta. Koska järjestelmä ei pidä varastokirjanpitoa, käyttäjät voivat tällä ilmoittaa tuotteiden puutoksista. Muuttaa rawproduct-taulun alertia. Jos tuote on poistettu käytöstä, heittää ProductNotFoundException-poikkeuksen.

public void removeAlert(ImportProduct product) throws DBConLostException;

Poistaa hälytyksen tuotteesta. Jos tuotteessa ei ollut hälytystä, ei tee mitään.

public ImportProduct[] getAlerts(void) throws DBConLostException;

Palauttaa listan tuotteista, joille on asetettu hälytys.

public int saveFingerprint(FPData data) throws DBConLostException;

Tallentaa sormenjäljen tunnistustiedon kantaan ja palauttaa sormenjäljen saaman uuden id:n.

public FPData[] getFingerprintData(void) throws DBConLostException;

Palauttaa taulukossa kaikki sormenjälkien tunnistetiedot, eli FPData-olioita, jotka ovat sormenjälkien tunnistetiedon, id:n ja laatuparametrin sisältäviä kolmikoita.

public ProductGroup[] getProductGroups(void) throws DBConLostException;

Palauttaa kaikki tuoteryhmät ProductGroup-olioina taulukossa. Lisäksi näihin ProductGroup-olioihin sisällytetään kaikki kuhunkin ryhmään kuuluvat ImportProductit.

public void deleteFingerPrint(id:int) ;

Poistaa fingerprint-taulusta rivin johon kyseinen id-viittaa. Kanta poistaa myös finger-taulun rivin, jossa on viite tähän sormenjälkeen.

public boolean reconnectDatabase(void);

Yrittää avata yhteyden tietokantaan uudestaan. Palauttaa ”true” jos yhteys onnistui, ”false” jos ei.

7.4.2 DBConLostException

Tämä poikkeusluokka on sitä varten, että tällä ilmoitetaan tietokantayhteyden olevan kadonnut. Tämä luokka perii Exception-luokan.

public DBConLostException(void);

Kutsutaan yliluokan konstruktoria.

7.4.3 ProductNotFoundException

Tämä poikkeusluokka sitä varten että ilmoitetaan että tuote puuttuu, eli tuote on poistettu käytöstä. Viestiksi annetaan tuotteen nimi sekä id. Tämä luokka perii Exception-luokan.

public ProductNotFoundException(String id, String name);

Konstruktori yhdistää id:n ja nimen yhdeksi merkkijonoksi jossa erottimena kaksoispiste ja kutsuu yliluokan merkkijonolla kuormitettua konstruktoria.

7.5 FingerprintAuthentication-komponentti

Paketti: fingerprintauthentication

Komponentti huolehtii sormenjälkien lukemisesta ja tunnistamisesta, mihin käytetään Griaulen GrFinger -apuohjelmaa. Komponentti käyttää FingerprintEvent-rajapintaa, joka tarjoaa tarvittavat yhteydet Main-komponenttiin. UML-kaaviossa komponentista (Liite A, kuva 6) näkee helposti luokkien väliset yhteydet ja niiden metodit, sekä tärkeät sisäiset muuttujat.

7.5.1 FingerprintAuthentication

Komponentin pääluokka, joka ottaa vastaan käynnistysasetukset ja olion, joka toteuttaa FingerprintEvent-rajapinnan. Luokan tehtäviin kuuluu myös Griaulen GrFingerJava-kirjaston alustaminen ja käyttöönotto. Lisäksi se toteuttaa IStatusEventListener-tapahtumarajapinnan, jonka kautta sormenjälkitunnistuskirjasto ja -ajuri kertovat, onko koneessa kytkettynä sormenjälkitunnistinta vai ei.

Luokassa luodaan FingerprintCache-olio, jonka tehtävänä on tallettaa muistiin ohjelman tarvitsemat sormenjälkien tunnistustiedot, ja ReaderListener-olio jokaista sillähetkellä kytkettynä olevaa sormenjälki-lukijaa varten. Myös MatchingContext-olioiden luominen ja käyttäminen on tämän luokan tehtävänä. Nämä GrFingerJava-kirjastoon kuuluvat oliot tekevät varsinaisen sormenjälkikuvien vertailun. Toinen rajapinta, jonka luokka toteuttaa on FingerprintData. Tämän rajapinnan avulla ReaderListener-olio lähettää uuden sormenjälkienlukijalta tulleen kuvan tälle luokalle käsiteltäväksi.

Komponentin UML-luokkakaaviosta (Liite A, kuva 6) käyvät ilmi komponentin luokkien suhteet toisiinsa ja GrFingerJava-kirjastoon.

Muuttujat

private FingerprintSettings settings;

Tämä muuttuja varastoi asetukset tallentavan FingerprintSettings-olion.

private FingerprintEvent eventinterface;

Tämä muuttuja varastoi olion, joka toteuttaa FingerprintEvent-rajapinnan, jonka kautta komponentti ilmoittaa uusista joko tunnistetuista tai tunnistamattomista sormenjäljistä.

Rajapinnat

Fingerprint

Rajapinta on kuvattu komponenttikuvauksen yhteydessä. Sen tehtävänä on välittää tietoa muun onjelmiston muutoksista sormenjälkikomponentille.

FingerprintData

Rajapinta on kuvattu hieman myöhemmin ja on osa FingerprintAuthentication-komponenttia. Sen tehtävänä on välittää uusia sormenjälkiä ReaderListener-luokalta rajapinnan toteuttavalle luokalle.

Metodit

public FingerprintAuthentication(FingerprintSettings settings, FingerprintEvent eventinterface);

Luo uuden ilmentymän FingerprintAuthentication luokasta. Saa syötteekseen FingerprintSettings-olion ja FingerprintEvent-rajapinnan toteuttavan olion.

7.5.2 FingerprintCache

Tämän luokan tehtävänä on luoda välimuistipuskuri komponentissa käytettäville sormenjälkien tunnistustiedoille. Tämän pitäisi nopeuttaa ohjelman toimivuutta ja lisäksi vähentää ohjelman tietokannalle aiheuttamaa kuormaa. Ohjelman asetuksissa voidaan säätää sitä kuinka monta sormenjälkeä pidetään muistissa, tai voidaan myös asettaa, että kaikki sormenjäljet talletetaan muistiin. Luokka on erittäin vahvasti riippuvainen Fingerprint-luokasta, joiden muodostamana taulukkona välimuisti toteutetaan.

Muuttujat

private int cachesize;

Muuttujaan talletetaan tieto siitä, että kuinka monta kuvaa sormenjälkikuvien välimuistissa pidetään. Jos asetuksissa on annettu, että kaikki sormenjäljet, niin tähän muuttujaan tallennetaan kokona sormenjälkien määrä kannassa.

Metodit

public FingerprintCache FingerprintCache(int size);

Luokan konstruktori. Saa syötteenä sormenjälkien tallentamiseen käytettävän välimuistin koon. Ja luo sen perusteella taulukon, johon sormenjäljet tallennetaan.

public void insertFingerprint(int id, Template data);

Lisätään uusi sormenjäljen tunnistustieto taulukkoon. Syötteenä annetaan sormenjäljen id-numero ja sen kuva. Jos taulu on jo täynnä, niin silloin kasvatetaan taulun kokoa.

public void loadCache(int[] id, Template[] data);

Metodi tallentaa välimuistiin tietokannasta saadut sormenjäljet. Se saa syötteenä kaksi samankokoista listaa, joista toisessa on sormenjäljen id-numero ja toisessa sormenjäljen tunnistetieto Template-oliona. FingerprintCache-luo näistä sitten FingerprintCaheData-olioita.

public int[] getIdList(void);

Tällä metodilla haetaan kaikkien välimuistissa olevien sormenjälkien id-numerot listana. Tätä käytetään, kun kysellään kannasta välimuistissa olemattomia sormenjälkiä.

public Fingerprint getNextFingerPrint(void);

Tämä iteraattori metodi antaa seuraavan välimuistissa olevan olevan sormenjälki-id:n, jos seuraavaa sormenjälkeä ei löydy palauttaa NULL:n ja palauttaa iteraattorin taulun ensimmäiseen olioon.

public void resetIterator(void);

Tämä metodi palauttaa iteraattorin alkuun jotta taulun läpikäyminen voidaan aloittaa uudestaan.

7.5.3 FingerprintCacheData

Tämä luokka toimii komponentin sisällä sormenjälkien tallettajana, kun ne varastoidaan välimuistiin. Lisäksi se pitää kirjaa siitä, että kuinka monta kertaa mitäkin sormenjälkeä on oikeasti tarvittu, jotta käyttämättömät sormenjäljet voidaan siivota pois muistista. Tätä luokkaa käyttävät FingerprintCache ja FingerprintAuthentication -luokat. Sormenjälki on tallennettuna jo valmiiksi GrFingerJavan-tarvitsemaan FingerprintImage-muotoon.

Muuttujat

private int id;

Muuttuja pitää sisällään sormenjäljen id-numeron. Nämä id-numerot luodaan, kun sormenjäljen kuva tallennetaan tietokantaan. ID on nouseva kokonaisluku, joka nousee nollasta eteenpäin.

private int times;

Tämä muuttuja pitää sisällään tiedon siitä kuinka monta kertaa sormenjälkeä on onnistuneesti käytetty asiakkaan tunnistamisessa. Tämän tiedon perusteella päätetään siitä, että poistetaanko sormenjälki, jos niille varattu välimuistialue on täyttymässä.

private Date accessdate;

Muuttuja pitää sisällään tiedon siitä, että milloin viimeksi sormenjälkeä on onnistuneesti käytetty käyttäjän tunnistamiseen. Muuttujassa on javan Date-olio.

Metodit
public Fingerprint(int id, Template data);

Konstruktori, joka luo uuden sormenjälki-olion välimuistiin. Ja asettaa id:n ja data-olioiden:n arvot kohdalleen. Samalla myös times-muuttujan arvoksi asetetaan 1 ja accessdate-muuttujan arvoksi sen luomishetken ajankohta.

public int getID(void);

Metodi palauttaa sormenjäljen id:n.

public Template getImage(void);

Metodi palauttaa sormenjäljen vertaukseen tarvittavan Template-olion.

public void addTime(void);

Metodi lisää sormenjäljelle yhden käyttökerran. Ja asettaa käyttöajaksi käytön ajanhetken.

public int getTimes(void);

Metodi palauttaa tiedon, että kuinka monta kertaa sormenjälkeä on käytetty.

public Date getAccessDate(void);

Metodi palauttaa javan Date-olion, joka kertoo milloin sormenjälkitietoa on viimeksi käytetty onnistuneesti.

7.5.4 FingerprintData

Tämä rajapinnan avulla ReaderListener-luokka kertoo FingerprintAuthentication luokalle saaneensa uuden kuvan ja välittää sen jatkokäsiteltäväksi. Rajapinnassa on vain yksi metodi.

Metodit

public void newFingerprint(FingerprintImage image);

Metodi välittää kuvan uudesta sormenjälkikuvasta.

7.5.5 ReaderListener

Luokan tehtävänä on ottaa vastaan sormenjälkien kuvia sormenjälkilukijalta ja välittää niitä eteenpäin jatkokäsiteltäväksi. Jokaista järjestelmään liitettyä lukijaa varten luodaan yksi näitä luokkia ja se toimii tämän jälkeen itsenäisesti kuunnellen tapahtumia sormenjälkilukijalta. Saatuaan uuden kuvan luokka lähettää sen eteenpäin käyttäen FingerprintData-rajapinnan newFingerprint-metodia.

Muuttujat

private FingerprintData imagelistener;

Metodit

public ReaderListener(FingerprintData imagelistener);

public void deleteReaderListener(void);

7.6 GUI-komponentti

Paketti: gui ja sen alipaketit

GUI-komponentti toteuttaa järjestelmän asiakasliittymän graafisen käyttöliittymän. Monimutkaisuudestaan johtuen komponentti on jaettu useaan pakettiin, jotka on seuraavassa dokumentoitu omien alaotsikoidensa alla. Paketeissa käytetyt general-paketissa tai sen alipaketeissa olevat luokat on dokumentoitu vain kertaalleen omassa kohdassaan. Osa komponentissa käytettävistä olioista on dokumentoitu aikaisemmin tässä dokumentissa luvussa 5.3 (UserInfo ja GuiSettings sekä niiden apuluokat, ja rajapinnat UserInterfaceEvent ja UserInterface). GUI-komponentin sisäiset rajapinnat on dokumentoitu luvussa 5.3.8.

Jokainen käyttöliittymän ”nappula”, eli käyttäjän manipuloitavissa oleva osa, toteuttaa ActionListener-rajapinnan, joten sitä ei ole jokaisen kohdalla erikseen mainittu.

7.6.1 GuiMain-alikomponentti

Paketti: gui

Paketissa on käyttöliittymäkomponentin pääluokka sekä (kappaleessa 5 kuvatut) rajapinnat UserInterface ja UserInterfaceEvent, joiden kautta käyttöliittymä keskustelee muun järjestelmän kanssa.

7.6.1.1 GuiMain

Luokka perii JFrame-luokan, eli luo käyttöliittymän ikkunan. Se toteuttaa UserInterface- ja ViewEventInterface-rajapinnat. UserInterfaceEvent-rajapinnan avulla

luokka pystyy kutsumaan muuta järjestelmää tarpeen mukaan. Lisäksi luokka käyttää GUI-komponentin sisällä RegisterViewInterface, ShopViewInterface, ImportViewInterface, AlertViewInterface, StartViewInterface ja ConfigViewInterface -rajapintoja.

Julkiset muuttujat

public static final Color defaultColor;

Näkymissä käytettävä oletustaustaväri.

public static final Border defaultBorder;

Näkymissä käytettävä oletusreunaviiva.

public static final Border defaultEmptyBorder;

Näkymissä käytettävä läpinäkyvä reunaviiva.

public static final Border defaultMargins;

Kaikkien näkymien ympärille (niiden ja ikkunan reunan väliin) piirrettävä tyhjä alue.

public static final int FINNISH = 0;

Suomea vastaava vakio.

public static final int ENGLISH = 1;

Englantia vastaava vakio.

public static final int SWEDISH = 2;

Ruotsia vastaava vakio.

Metodit

public GuiMain(GuiSettings settings, UserInterfaceEvent eventinterface);

Konstruktori saa parametrinaan GUI-komponentin asetusolion ja UserInterfaceEvent-rajapinnan toteuttavan (MainProgram-komponentin) luokan.

GuiMainin muut metodit koostuvat edellä mainittujen rajapintojen toteuttamisesta. Näiden toiminta on kuvattu rajapintakuvauksissa luvussa 5, eikä siihen tässä perehdytä tarkemmin.

7.6.2 StartView-alikomponentti

Paketti: gui.startView

StartView-alikomponentti toteuttaa käyttäjälle ensimmäisenä näkyvän osan ohjelmasta. Tämän komponentin toteuttamassa näkymässä käyttäjä voi aloittaa rekisteröitymisen

ohjelman käyttäjäksi, aloittaa tuotteen loppumisesta kertovan varoituksen lisäämisen, poistaa tuotteen loppumisen varoitus tai sormenjäljen antamalla kirjautua sisään ohjelmaan. Toteuttaa StartViewInterface-rajapinnan.

7.6.2.1 StartPanel

Luokka toimii paketin pääluokkana. Se toteuttaa startViewInterface- ja AlertEvent-rajapinnat. AlertEvent-rajapinnan avulla sille kerrotaan, että käyttäjä haluaa lisätä tai poistaa hälytyksen tuotteelta. Tästä kerrotaan eteenpäin ViewEventInterface-rajapinnan avulla. Luokka perii JPanel-luokan.

Muuttujat

private JPanel CPanel;

Keskimmäinen paneeli, luodaan konstruktorissa.

private ViewEventInterface viewEventInterface;

Rajapinta viestintää varten, saadaan konstruktorissa.

Metodit

public StartPanel(ViewEventInterface viewEventInterface);

Konstruktorissa saadaan ViewEventInterface. Konstruktorissa luodaan CPanel. CPanelin String-parametrit saadaan ViewEventInterface-rajapinnan metodilta getGuiText(int id):String.

7.6.2.2 CPanel

Alkuruudun keskimmäinen paneeli. Luokka perii JPanel-luokan.

Muuttujat

private JLabel header;

Näytetään sivun otsikko ja sen kuva.

private JLabel guide;

Näytetään käyttäjälle ohje.

private JButton addWarning;

Tästä napista käyttäjä voi siirtyä varoituksen lisäämiseen.

private JButton register;

Tästä napista siirrytään rekisteröintinäkymään.

private String remove;

Teksti, joka annetaan RemovePanelille (poistonappulan teksti).

Metodit

protected CPanel(String header, String guide, String warning, String register, String remove, LanguageButton languageButton, alertEvent:AlertEvent);

Saaduista parametreista header ja guide asetetaan vastaavan nimisiin JLabeleihin, warning addWarning-nappiin ja register register-nappiin. Parametri remove tallennetaan muuttujaan välitettäväksi addAlert-metodissa luotaville WarningPanel-luokille, joille annetaan myös alertEvent-parametri. Konstruktorissa luodaan CsPanel ja näytetään se tämän paneelin alareunassa.

protected void addAlert(String message);

Luodaan WarningPanel-luokka, jolle annetaan parametreina message ja remove.

7.6.2.3 CsPanel

Luokka sisältää vain general.LanguageButton-luokan, jonka avulla käyttäjä voi vaihtaa kieltä. Luokka perii JPanel-luokan.

Muuttujat

private LanguageButton languageButton;

Nappi kielenvaihtoon.

Metodit

protected CsPanel(LanguageButton languageButton);

Konstruktorissa laitetaan paneeliin näkymään parametrina saatu languageButton.

7.6.2.4 WarningPanel

Paneelissa on teksti hälytyksestä ja nappi, jolla hälytyksen voi poistaa. Luokka perii Jpanel-luokan.

Muuttujat

private JLabel message;

Tässä näytetään hälytyksen teksti.

private JButton remove;

Napilla voi peruuttaa hälytyksen.

private AlertEvent alertEvent;

Rajapinta, saadaan konstruktorissa.

Metodit

protected WarningPanel(String remove, String message, AlertEvent alertEvent);

Konstruktorissa saadaan tekstit privaatteihin kenttiin ja AlertEvent-rajapinta, jonka avulla kerrotaan, kun halutaan poistaa jokin hälytys.

7.6.2.5 AlertEvent (rajapinta)

Rajapinnan avulla paketin sisällä välitetään tieto hälytyksien lisäämisestä ja poistamisesta.

Metodit

protected void addAlert(ImportProduct product);

Kutsutaan mikäli käyttäjä haluaa siirtyä lisäämään hälytyksen.

StartPanel kutsuu ViewEventInterface-rajapinnan metodia changeAlertView. Kun hälytys on tehty luodaan StartPanel uudelleen ja näin uusi hälytys tulee näkyviin.

protected void removeAlert(ImportProduct product);

Kutsutaan kun halutaan poistaa jokin hälytys.

StartPanel kutsuu ViewEventInterface-rajapinnan metodia removeAlert(product:ImportProduct). Mikäli poisto hyväksytään GuiMain luo StartPanelin uudelleen ja näin poistettu hälytys häviää näkyvistä.

7.6.3 RegisterView-alikomponentti

Paketti: gui.registerView

RegisterView-alikomponentti mahdollistaa uuden käyttäjän rekisteröitymisen. Siinä käyttäjä kirjoittaa TKTL:n käyttäjätunnuksen ja salasanansa. Jos nämä ovat oikein, käyttäjälle annetaan mahdollisuus syöttää itselleen etunimi ja sukunimi järjestelmään. Näiden tietojen antamisen jälkeen käyttäjä siirretään normaaliin ConfigView-alikomponentin näkymään. Toteuttaa RegisterViewInterface-rajapinnan

7.6.3.1 RegisterPanel

Luokka toteuttaa RegisterEvent- ja RegisterViewInterface-rajapinnat ja perii JPanel-luokan.

Muuttujat

private CPanel cPanel;

Keskimmäinen paneeli, luodaan konstruktorissa.

private ViewEventInterface viewEventInterface;

Rajapinta, saadaan konstruktorissa.

Metodit

public RegisterPanel(ViewEventInterface viewEventInterface);

Konstruktorissa luodaan Cpanel.

7.6.3.2 CPanel

Keskimmäinen paneeli, perii JPanel-luokan.

Muuttujat

private Lower lower;

Luodaan konstruktorissa tarpeen mukaan. Riippuu authOK-parametrista.

private Upper upper

Luodaan konstruktorissa. Disabloituna mikäli näytetään myös Lower

private RegisterEvent registerEvent;

Saadaan konstruktorissa.

private String[] texts;

Sisältää tekstit luotaville luokille. Järjestys: header, username, password, ok(upper), info1(upper), info2(upper), firstname, lastname, ok(lower), info1(lower), info2(lower), info3(lower)

Metodit

protected CPanel(RegisterEvent registerEvent, boolean authOK, String[] texts);

Konstruktorissa luodaan tarpeen mukaan Lower, mutta aina Upper, jonka aktiivisuus riippuu siitä, että luodaanko Lower.

7.6.3.3 Lower

Näyttää käyttäjälle kentät, joihin hän voi kirjoittaa etu- ja sukunimensä. Perii JPanel-luokan.

Muuttujat

private JLabel firstnameT;

Etunimi-teksti.

private JLabel lastnameT;

Sukunimi-teksti.

private JTextField firstname;

Kenttä, johon etunimi kirjoitetaan.

private JTextField lastname;

Kenttä, johon sukunimi kirjoitetaan.

private JButton ok;

Nappi, jolla hyväksytään annetut nimet ja siirrytään seuraavaan ikkunaan.

private JLabel info1;

1. ohjeteksti.

private JLabel info2;

2. ohjeteksti.

private JLabel info3;

3. ohjeteksti.

private RegisterEvent registerEvent ;

Rajapinta, jolle kerrotaan annetuista nimistä.

Metodit

protected Lower(RegisterEvent registerEvent, String firstname, String lastname, String ok, String info1, String info2, String info3);

Konstruktorissa saadut parametrit talletetaan vastaaviin muuttujiin.

7.6.3.4 Upper

Näyttää käyttäjälle kentät, joihin hän voi laittaa käyttäjätunnuksensa ja salasanansa. Perii JPanel-luokan.

Muuttujat

private JPassword password;

Salasanakenttä.

private JTextField username;

Tunnuskenttä.

private JLabel usernameT;

Käyttäjänimi-teksti.

private JLabel passwordT;

Salasana-teksti.

private JButton ok;

Nappi, jolla hyväksytään salasana-käyttäjätunnus pari.

private JLabel info1;

Ohjeteksti 1.

private JLabel info2;

Ohjeteksti 2.

private RegisterEvent registerEvent;

Rajapinta, jolle kerrotaan annetuista tunnuksista.

Metodit

protected Upper(RegisterEvent registerEvent, boolean authOK, String password, String username, String ok, String info1, String info 2);

Luo uuden Upper-paneelin ja tallettaa parametrit muuttujiin.

7.6.3.5 CnPanel

Näyttää sivun otsikon. Perii JPanel-luokan.

Muuttujat

private JLabel header;

Sivun otsikko.

Metodit

protected CnPanel(String header);

Konstruktorissa luodaan otsikko parametrin mukaisesti.

7.6.3.6 RegisterEvent (rajapinta)

Antaa mahdollisuuden kertoa rekisteröitymisyrityksestä ja nimen antamisesta.

Metodit

public void newUser(String username, String password);

Kutsutaan, kun käyttäjä syöttää käyttäjänimensä ja salasanansa.

public void userNames(String firstname, String lastname);

Kutsutaan, kun käyttäjä syöttää etu- ja sukunimensä.

7.6.4 ConfigView-alikomponentti

Paketti: gui.configView

ConfigView-alikomponentin tuottama näkymä mahdollistaa käyttäjän käyttäjätietojen muuttamisen ohjelmassa. Sen avulla käyttäjä voi vaihtaa vakiotuotettaan ja lisätä itselleen uusia, tai korvata vanhoja, sormenjälkiä, jotka mahdollistavat käyttäjän sisäänkirjautumisen sormenjälkilukijan avulla.

ConfigView-alikomponentin näkymästä käyttäjä voi siirtyä tuotteiden tuonti tai ostonäkymään. Alikomponentti toteuttaa ConfigViewInterface-rajapinnan.

7.6.4.1 ConfigPanel

Paketin pääluokka, joka toteuttaa ConfigViewInterface- ja FingerEvent-rajapinnat ja perii Jpanel-luokan.

Muuttujat

private NPanel npanel;

Navigointipaneeli.

private CPanel cPanel;

Keskimmäinen paneeli, luodaan konstruktorissa.

Metodit

public ConfigPanel ConfigPanel(ViewEventInterface viewEventInterface);

Metodissa luodaan CPanel ja NPanel.

7.6.4.2 CPanel

Paneli on olemassa järjestääkseen sen sisällä olevat paneelit parempaan järjestykseen. Perii JPanel-luokan.

Muuttujat

private CnPanel cnPanel;

Luodaan konstruktorissa.

private CcPanel ccPanel;

Luodaan konstruktorissa.

Metodit

protected CPanel(void);

Konstruktorissa välitetään parametrit eteenpäin ja vain luodaan yksityiset kentät.

7.6.4.3 CnPanel

Näyttää sivun otsikon. Perii JPanel-luokan.

Muuttujat

private JLabel header;

Sivun otsikko.

Metodit

protected CnPanel(String header);

Konstruktorissa laitetaan otsikko näkyviin.

7.6.4.4 CcPanel

Paneli on olemassa ainoastaan järjestääkseen sen sisällä olevat paneelit parempaan järjestykseen. Perii JPanel-luokan.

Muuttujat

private CccPanel cccPanel;

Luodaan konstruktorissa.

private CcwPanel ccwPanel;

Luodaan konstruktorissa.

private CcnPanel ccnPanel;

Luodaan konstruktorissa.

Metodit

protected CcPanel(void);

Konstruktorissa luodaan yksityiset muuttujat ja laitetaan ne näkyviin tähän paneeliin.

7.6.4.5 CccPanel

Jakaa keskimmäisen alueen vasempaan ja oikeaan puoleen. Perii Jpanel-luokan.

Muuttujat

private LeftSide leftSide;

Luodaan konstruktorissa.

private RightSide rightSide;

Luodaan konstruktorissa.

Metodit

protected CccPanel(void);

Konstruktorissa luodaan yksityiset muuttujat ja laitetaan ne näkyviin tähän paneeliin.

7.6.4.6 CcwPanel

Sisältää ShopableList-luokan.

Muuttujat

private ShopableList shopabelList;

Lista, joka näyttää tuotteet, joista voi valita suosikkituotteen.

Metodit

protected CcwPanel(void);

Konstruktorissa laitetaan ShopableList näkyviin tähän luokkaan.

7.6.4.7 CcnPanel

Näyttää ohjeen käyttäjälle ja tiedon mikä on hänen tämän hetkinen vakiotuotteensa.

Muuttujat

private JLabel info;

Ohje käyttäjälle.

privat JLabel favorite;

Valittu vakiotuote.

Metodit

protected CcnPanel(void);

Konstruktorissa luodaan ohjeen ja vakiotuotteen näyttävät JLabel-luokat ja näytetään ne.

7.6.4.8 RightSide

Näyttää paneelin sormienjälkien muokkaamiseen ja lisäämiseen.

Muuttujat

private FingerConfigPanel fingerConfigPanel;

Luokka joka luodessa tuottaa sormenjälkien muokkaamispaneelin.

Metodit

protected RightSide(void);

Konstruktorissa luodaan sormenjälkien muokkaamiseen tarkoitettu paneeli ja näytetään se.

7.6.4.9 LeftSide

Näyttää käyttäjän saldot tuoteryhmissä.

Muuttujat

private JLabel balance;

Käyttäjän saldot.

Metodit

protected LeftSide(void);

Konstruktorissa luodaan saldon näyttävä luokka.

7.6.4.10 FingerEvent (rajapinta)

Rajapinnan avulla voidaan kertoa kun sormenjälkeä halutaan muokata.

Metodit

public void addFinger(int finger);

Metodilla kerrotaan, että finger (0-4) sormea halutaan muokata.

public void cancelAdding(void);

Metodilla voi kertoa, että sormenjälkeä ei halutakaan muokata.

7.6.5 ShopView-alikomponentti

Paketti: gui.shopView

ShopView-alikomponentti tuottaa näkymän, jonka kautta käyttäjä voi merkitä ostaneensa tuotteita järjestelmästä. Näkymä tulee näkyville, kun käyttäjä joko syöttää ohjelman alkuruudussa sormenjälkensä tai siirtyy siihen jostain muusta näkymästä nappia painamalla.

Jos käyttäjä on syöttänyt sormenjälkensä alkuruudussa ja ei tee mitään muuta tämän jälkeen tai kirjautuu ulos järjestelmästä, veloitetaan häneltä asetuksissa määritelty vakiotuote. Tämän komponentin näkymästä käyttäjä voi siirtyä tuotteiden tuonti- tai omat tiedot -näkymiin. Alikomponentti toteuttaa ShopViewInterface-rajapinnan.

7.6.5.1 ShopPanel

Paketin pääluokka, joka keskustelee muun järjestelmän kanssa ViewEventInterfacen avulla. Toteuttaa ShopViewInterface- ja BuyEvent-rajapinnat. Luo luokat SPanel, CPanel ja NPanel. Perii JPanel-luokan.

Muuttujat

private CPanel cPanel;

Luodaan konstruktorissa.

private SPanel sPanel;

Luodaan konstruktorissa.

private ViewEventInterface viewEventInterface;

Saadaan konstruktorissa.

Metodit

public ShopPanel(void);

Luo paneelin ja tämän paneelin alipaneelit.

7.6.5.2 CPanel

Näyttää osto- ja peruutusnapin ja tiedot mitä tuotetta käyttäjä on ostamassa sekä ohjetekstin. Perii Jpanel-luokan.

Muuttujat

private BuyButton buyButton;

Ostonappi jolla hyväksytään vienti.

private CancelButton cancelButton;

Peruutusnappi, jolla saadaan lisää aikaa valikoimiseen.

private JLabel info1;

Ylin teksti.

private JLabel info2;

Alin teksti

private JLabel product;

Tuote, jota ollaan ostamassa.

Metodit

protected CPanel(void);

Konstruktorissa tehdään yksityiset muuttujat ja laitetaan ne näkymään tässä paneelissa.

7.6.5.3 SPanel

Näyttää käyttäjän saldon ruudun alareunassa. Perii JPanel-luokan.

Muuttujat

private JLabel balance:

Metodit

protected SPanel(String balance);

Näytetään saldo valitun tuotteen tuoteryhmässä.

7.6.5.4 CancelButton

Nappi, jolla saadaan lisäaikaa miettimiseen, eli peruutetaan tuotteen veloitus tietyn ajan kuluttua.

Muuttujat

private CancelEvent cancelEvent;

Rajapinta, jolle kerrotaan peruutus.

Metodit

protected CancelButton(void);

Konstruktorissa luodaan vain tämä nappi ja laitetaan rajapinta talteen.

7.6.5.5 BuyButton

Nappi, jolla vahvistetaan valittu tuote heti vietäväksi.

Muuttujat

private BuyEvent buyEvent;

Rajapinta, jonka avulla kerrotaan, että tuote on hyväksytty vietäväksi.

Metodit

protected BuyButton(void);

Konstruktorissa luodaan tämä nappi ja laitetaan rajapinta talteen.

7.6.5.6 BuyEvent (rajapinta)

Rajapinta, jonka avulla kerrotaan, että tuote on hyväksytty vietäväksi.

Metodit

public void buy(void);

Kerrotaan, että valittu tuote on hyväksytty vietäväksi.

7.6.5.7 CancelEvent (rajapinta)

Rajapinta, jonka avulla kerrotaan, että käyttäjä tarvitsee lisää aikaa miettimiseen.

Metodit

public void cancel(void);

Metodilla kerrotaan, että käyttäjä tarvitsee lisää aikaa.

7.6.6 ImportView-alikomponentti

Paketti: gui.importView

ImportView-alikomponentin tuottaman näkymän kautta käyttäjät pystyvät merkitsemään tiedon siitä, että ovat tuoneet tuotteita. Nämä merkinnät lisäävät käyttäjän saldoa järjestelmässä. Näkymästä pääsee osto- ja omat tiedot -näkymiin. Alikomponentti toteuttaa ImportViewInterface-rajapinnan.

7.6.6.1 ImportPanel

Paketin pääluokka, joka luodaan kun halutaan sivu näkyviin. Toteuttaa ImportEvent-rajapinnan, jonka avulla kerrotaan kun jokin tuote halutaan hyväksyä tuotavaksi. Perii JPanel-luokan.

Muuttujat

private SPanel sPanel;

Alaosan paneeli, luodaan konstruktorissa.

private CPanel cPanel;

Keskimmäinen paneeli, luodaan konstruktorissa.

private ViewEventInterface viewEventInterface;

Rajapinta, saadaan konstruktorissa.

private ImportableList importableList;

Tuotelista, saadaan konstruktorissa.

Metodit

public ImportPanel(ViewEventInterface viewEventInterface, ImportableList importableList);

Konstruktorissa luodaan SPanel ja CPanel. Laitetaan myös ImportableList omalle paikalleen sivun vasempaan reunaan.

7.6.6.2 SPanel

Luokka näyttää käyttäjän saldon sivun alareunassa.

Muuttujat

private JLabel balance;

Käyttäjän saldo.

Metodit

public SPanel(String balance);

Konstruktorissa luodaan käyttäjän saldo ja näytetään se tässä paneelissa.

7.6.6.3 CPanel

Luokka jakaa näkymän kahtia ja luo vasemmalle puolelle listan tuotavista tuotteista ja oikealle puolelle ohjeen ja hyväksymisnapin. Perii JPanel-luokan.

Muuttujat

private ImportList importList;

Lista tuotavista tuotteista. Vasemmanpuoleinen osa.

private RightSide rightSide;

Oikeanpuoleinen osa.

Metodit

public CPanel(ProductGroup[] productGroup, AlertEvent alertEvent, String ok, String guide);

Konstruktorissa luodaan vasen ja oikea puoli ja vain välitetään saatuja parametreja eteenpäin.

7.6.6.4 RightSide

Näytetään CPanelin oikealla puolella.

Muuttujat

private JButton ok;

Tuonnin hyväksymisnappula.

private JLabel guide;

Ohjeteksti tämän sivun käyttämiseen.

Metodit

public RightSide(AlertEvent alertEvent, String ok, String guide);

Näytetään ohje- ja tuonninhyväksymisnappula.

7.6.6.5 ImportEvent (rajapinta)

Rajapinnan avulla kerrotaan kun jokin tuote halutaan hyväksyä tuoduksi.

Metodit

public void ImportProduct(void);

Hyväksytään tällä hetkellä valittuna oleva tuote.

7.6.7 AlertView-alikomponentti

Paketti: gui.alertView

AlertView-alikomponentti tuottaa näkymän, jossa käyttäjä pystyy lisäämään varoituksen jonkin tuotavan tuotteen loppumisesta. Käyttäjän lisättyä varoituksen ohjelma siirtyy takaisin aloitusnäkymään. Alikomponentti toteuttaa AlertViewInterface-rajapinnan.

7.6.7.1 CPanel

Keskimmäinen paneeli, perii JPanel-luokan.

Muuttujat

private JLabel header;

Sivun otsikko.

private JLabel guide;

Ohje sivulla toimimiseen.

private JLabel product;

Valittu teksti.

private JButton ok;

Hälytyksen hyväksymisnappula.

Metodit

public CPanel(AlertEvent alertEvent, ProductGroup[] productGroup, String header, String guide, String product);

Konstruktorissa saadaan tekstit, jotka näytetään paneelissa ja AlertEvent, jonka avulla hyväksymisnappi kertoo, että nyt hälytyksen asettaminen hyväksytään.

7.6.7.2 AlertPanel

Paketin pääluokka, toteuttaa paketin rajapinnat ja keskustelee muun järjestelmän kanssa. Perii JPanel-luokan.

Muuttujat

private CPanel cPanel;

Keskimmäinen paneeli, luodaan konstruktorissa.

private ViewEventInterface viewEventInterface;

Rajapinta, saadaan konstruktorissa.

Metodit

public AlertPanel(ViewEventInterface viewEventInterface);

Konstruktorissa saadaan rajapinta, jonka avulla keskustellaan muun järjestelmän kanssa. Luodaan general.NPanel ja kutsutaan sen metodia alertView(). Luodaan myös CPanel.

7.6.7.3 AlertEvent (rajapinta)

Rajapinnan avulla kerrotaan kun halutaan lisätä hälytys valitusta tuotteesta.

Metodit

protected void addAlert(ImportProduct product);

Kerrotaan mikä tuote laitetaan hälytettävien tuotteiden listaan.

7.6.8 General-alikomponentti

Paketti: gui.general ja sen alipaketit

7.6.8.1 NPanel

Luokasta luodaan ilmentymä. Tämän jälkeen kutsutaan sopivaa metodia ja luokka näyttää tietyt navigointi nappulat. Luokka perii JPanel-luokan.

Muuttujat

private LogoutButton logout;

Uloskirjautumisnappi.

private ShopButton shop;

Ostonäkymänappi.

private ImportButton import;

Tuontinäkymänappi.

private ConfigButton config;

Asetusnäkymänappi.

private LanguageButton language;

Kielenvaihtonappi.

private CancelProductButton cancel;

Edellisen oston peruutusnappi.

private boolean isCancel;

Tieto siitä, onko edellisen oston peruutus sallittu.

private ViewEventInterface viewEventInterface;

Rajapinta, saadaan konstruktorissa.

Metodit

public NPanel(ViewEventInterface viewEventInterface, boolean isCancel);

Konstruktorissa luodaan ilmentymä luokassa ja laitetaan parametrit talteen yksityisiin kenttiin.

public void shopView(void);

Luodaan ja näytetään LogoutButton, ImportButton, ConfigButton ja LanguageButton. Tarvittaessa CancelProductButton.

public void importView(void);

Luodaan ja näytetään LogoutButton, ShopButton, ConfigButton ja LanguageButton. Tarvittaessa CancelProductButton.

public void registerView(void);

Luodaan ja näytetään LogoutButton ja LanguageButton.

public void startView(void);

Luodaan ja näytetään LanguageButton.

public void configView(void);

Luodaan ja näytetään LogoutButton, ImportButton, ShopButton ja LanguageButton. Tarvittaessa CancelProductButton.

public void alertView(void);

Luodaan ja näytetään LogoutButton ja LanguageButton.

7.6.8.2 LanguageButton

Kielenvaihtonappi, perii JButton-luokan.

Muuttujat

private ImageIcon image;

Lippu, joka näytetään napissa.

private ViewEvent viewEvent;

Rajapinta, jonka avulla kerrotaan napin painamisesta NPanelille.

Metodit

public LanguageButton(String text, ImageIcon image, ViewEvent viewEvent);

Konstruktorissa luodaan nappi ja annetaan sille kuva.

7.6.8.3 ImportButton

Nappi mahdollistaa siirtymisen tuotteiden tuonti-ikkunaan. Perii JButton-luokan.

Muuttujat

private ViewEvent viewEvent;

Rajapinta, jonka avulla kerrotaan napin painamisesta NPanelille.

Metodit

public ImportButton(String text, ViewEvent viewEvent);

Konstruktorissa luodaan nappi ja laitetaan sille teksti.

7.6.8.4 ConfigButton

Nappi mahdollistaa omien asetusten näkymään siirtymisen. Perii JButton-luokan.

Muuttujat

private ViewEvent viewEvent;

Rajapinta, jonka avulla kerrotaan napin painamisesta NPanelille.

Metodit

public ConfigButton(String text, ViewEvent viewEvent);

Konstruktorissa luodaan nappi ja laitetaan sille teksti.

7.6.8.5 LogoutButton

Nappi antaa mahdollisuuden siirtyä alkutilaan. Perii JButton-luokan.

Muuttujat

private ViewEvent viewEvent;

Rajapinta, jonka avulla kerrotaan napin painamisesta NPanelille.

Metodit

public LogoutButton(String text, ViewEvent viewEvent);

Konstruktorissa luodaan nappi ja laitetaan sille teksti.

7.6.8.6 CancelProductButton

Nappi antaa mahdollisuuden perua aikaisemmin tehty ostos. Mikäli peruutusta ei ole tai sitä ei sallita, nappi ei ole aktiivinen.

Muuttujat

private CancelEvent cancelEvent;

Rajapinta, jonka avulla kerrotaan, että käyttäjä haluaa perua ostoksen.

Metodit

public CancelProductButton(String text, CancelEvent cancelEvent);

Konstruktorissa luodaan nappi ja laitetaan sille teksti.

7.6.8.7 ViewEvent (rajapinta)

Rajapinnan avulla NPanelin navigointinapit kutsuvat NPanelia ja kertovat, että niitä on painettu.

Metodit

public void logout(void);

Metodilla kerrotaan, että käyttäjä haluaa alkutilaan.

public void shopView(void);

Käyttäjä siirretään ostotilaan.

public void importView(void);

Käyttäjä siirretään tuontitilaan.

public void ConfigView(void);

Käyttäjä siirretään omien tietojen muokkaustilaan.

public void RegisterView(void);

Kertoo, että halutaan siirtyä rekisteröitymisikkunaan.

7.6.8.8 CancelEvent (rajapinta)

Rajapinnan avulla kerrotaan, että käyttäjä haluaa perua ostoksensa.

Metodit

public void cancel(void);

Perutaan käyttäjän edellinen vienti.

Paketti: gui.general.productLists.importList

Komponentti jakaa tuotavat tuotteet kahteen JList-komponenttiin. Vasemman puoleisessa näytetään ryhmät, mihin tuotteita voi tuoda ja oikeanpuoleisessa näytetään valitun ryhmän sisältävät tuotteet ja niiden mahdolliset koot. Käyttäjä voi valita näistä listoista tuovansa tuotteen.

7.6.8.9 ImportableList

Luokka on paketin pääluokka joka luodaan kun halutaan luoda lista tuotavista tuotteista.

Muuttujat

private GroupList groupList;

JListin perivä luokka, jossa on tuotavat ryhmät.

Metodit

public ImportableList(ProductGroup[] productGroup);

Luodaan GroupList ja annetaan sille ryhmät.

7.6.8.10 GroupList

GroupList sisältää tuotavien tuotteiden ryhmät ja näyttää ne JList-elementteinä.

Muuttujat

Ei luokan muuttujia.

Metodit

public GroupList(ProductChangeEvent productChangeEvent, ProductGroup[] productGroup);

Luo olion ja tallettaa parametrina saadun rajapinnan ja tuoteryhmien listan.

7.6.8.11 ProductList

ProductList sisältää tuotavan ryhmän kaikki tuotteet ja niiden tuotavissa olevat koot.

Muuttujat

Ei luokan muuttujia.

Metodit

public ProductList(ProductChangeEvent productChangeEvent, ImportableProduct[] products);

Luo olion ja tallettaa parametrina saadun rajapinnan ja tuotavien tuotteiden listan.

7.6.8.12 ProductChangeEvent (rajapinta)

Käytetään kun tuote muuttuu listassa.

Metodit

public void selectedProduct(ImportableProduct importableProduct);

Kerrotaan mikä tuote valittiin.

Paketti: gui.general.productLists.shopableList

Näytetään kaikki mahdolliset tuotteet mitä käyttäjä voi viedä. Tuotteet ovat käyttäjän suosikkijärjestyksessä. Vakiotuote on kuitenkin ensimmäisenä. Perii Jlist-luokan.

7.6.8.13 ProductList

Lista kaikista tuotavista tuotteista.

Muuttujat

Ei luokan muuttujia.

Metodit

public ProductList(ExportProduct[] products);

Luodaan lista ja asetetaan siihen parametrina saadut tuotteet.

7.6.8.14 ShopableList

Muuttujat

private ProductList productList;

JList-luokan perivä luokka, jossa tuotteet näkyvät.

Metodit

public ShopableList(ExportProduct[] products);

Näytetään tuotteet Jlist-elementteinä.

7.6.8.15 ProductChangeEvent (rajapinta)

Rajapintaa käytetään kertomaan, että käyttäjä on valinnut jonkin tuotteen.

Metodit

public void selectedProduct(ExportProduct exportProduct);

Kerrotaan mikä tuote valittiin.

Paketti: gui.general.fingerConfig

Tarjoaa käyttäjälle mahdollisuuden nähdä minkä sormien sormenjäljet järjestelmään on lisätty. Mahdollistaa myös sormenjälkien lisäämisen ja muokkauksen.

7.6.8.16 FingerConfigPanel

Paketin pääluokka, perii JPanel-luokan.

Muuttujat

private CPanel cPanel;

Keskimmäinen paneeli.

Metodit

public FingerConfigPanel(FingerEvent fingerEvent, int[] fingerprints);

Konstruktori luo olion ja tallettaa parametrina saadun rajapinnan ja sormenjälkitaulukon.

7.6.8.17 NPanel

Näyttää otsikon.

Muuttujat

private JLabel header;

Otsikkoteksti.

Metodit

public NPanel(String header);

Luodaan tämä paneeli tietyllä otsikolla.

7.6.8.18 CPanel

Keskimmäinen paneeli, joka perii JPanel-luokan. Sisältää sormenjälkinapit.

Muuttujat

Metodit

public CPanel(FingerEvent fingerEvent, int[] fingerprints);

Metodin kuvaus

7.6.8.19 FingerButton

Nappi, jonka avulla käyttäjä kertoo mitä sormea hän haluaa muokata. Napin väri myös riippuu siitä onko siinä jo sormenjälki.

Muuttujat

private int finger;

Sormi johon nappi liittyy.

private FingerEvent fingerEvent;

Rajapinta, jonka avulla kerrotaan, että nappia on painettu.

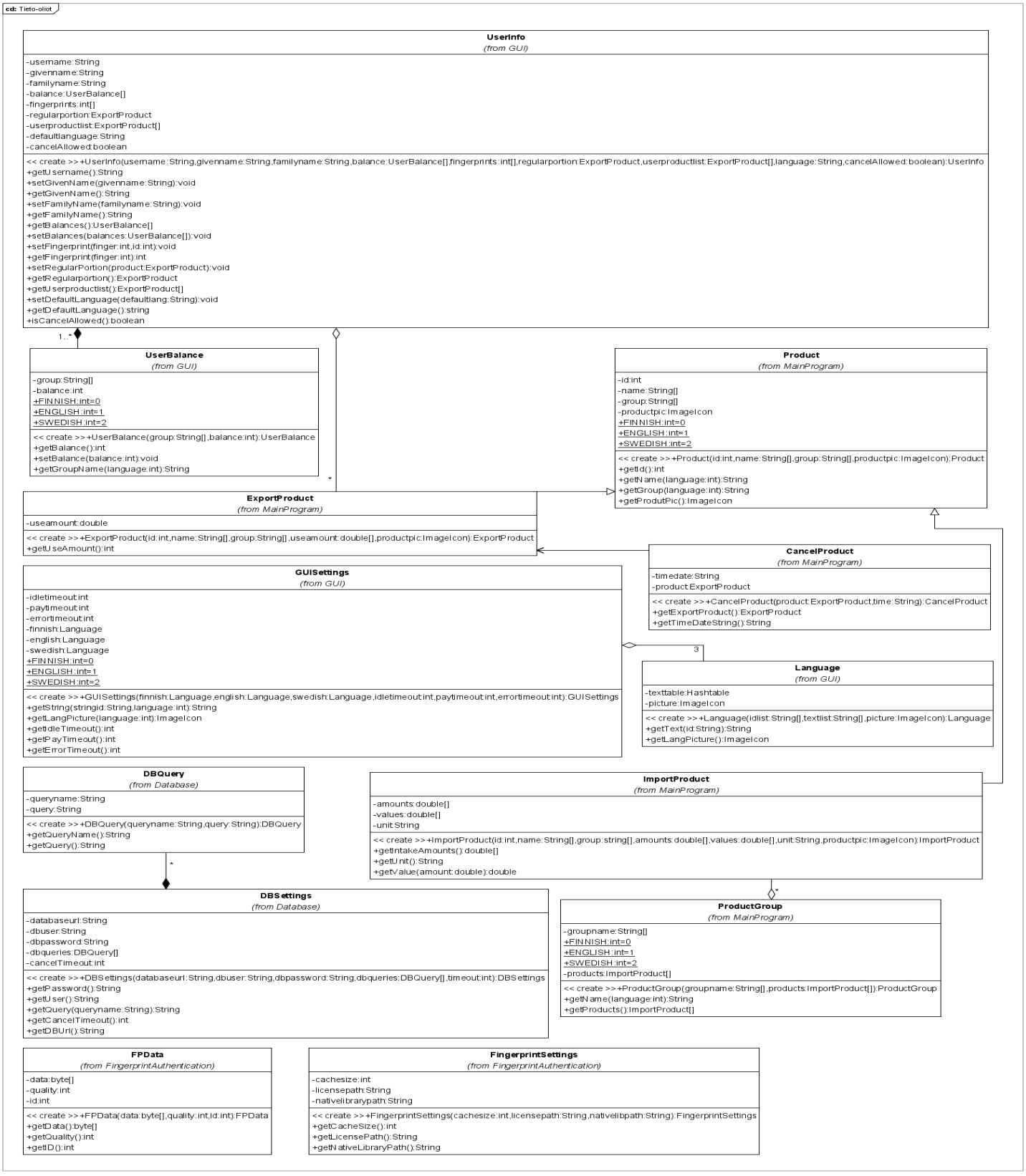
Metodit

public FingerButton(int finger, FingerEvent fingerEvent);

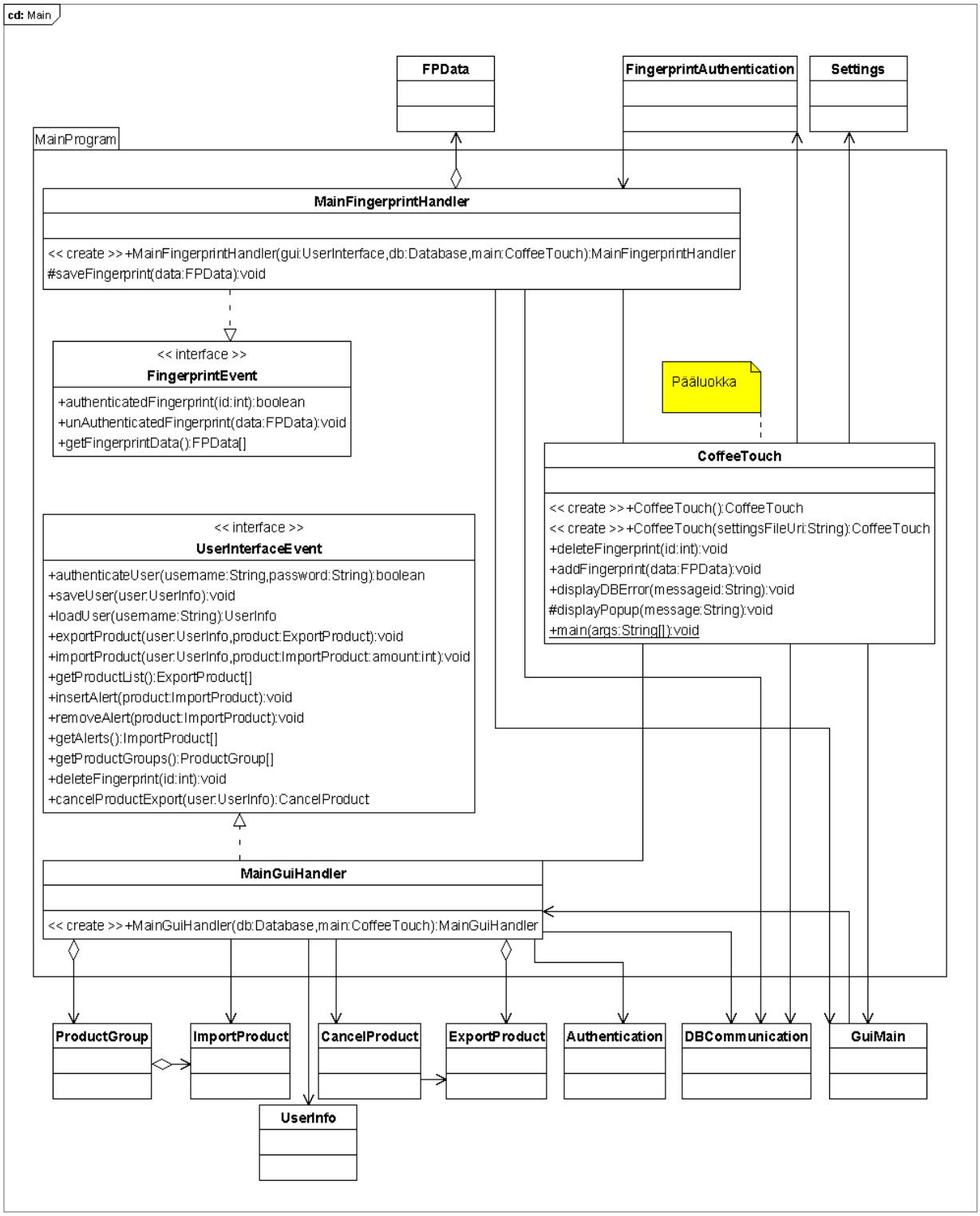
Luodaan nappi, jolla on yksikäsitteinen sormen id-numero (finger) ja rajapinta jonka avulla se keskustelee muun maailman kanssa (FingerEvent).

Liite A: Luokkakaaviot

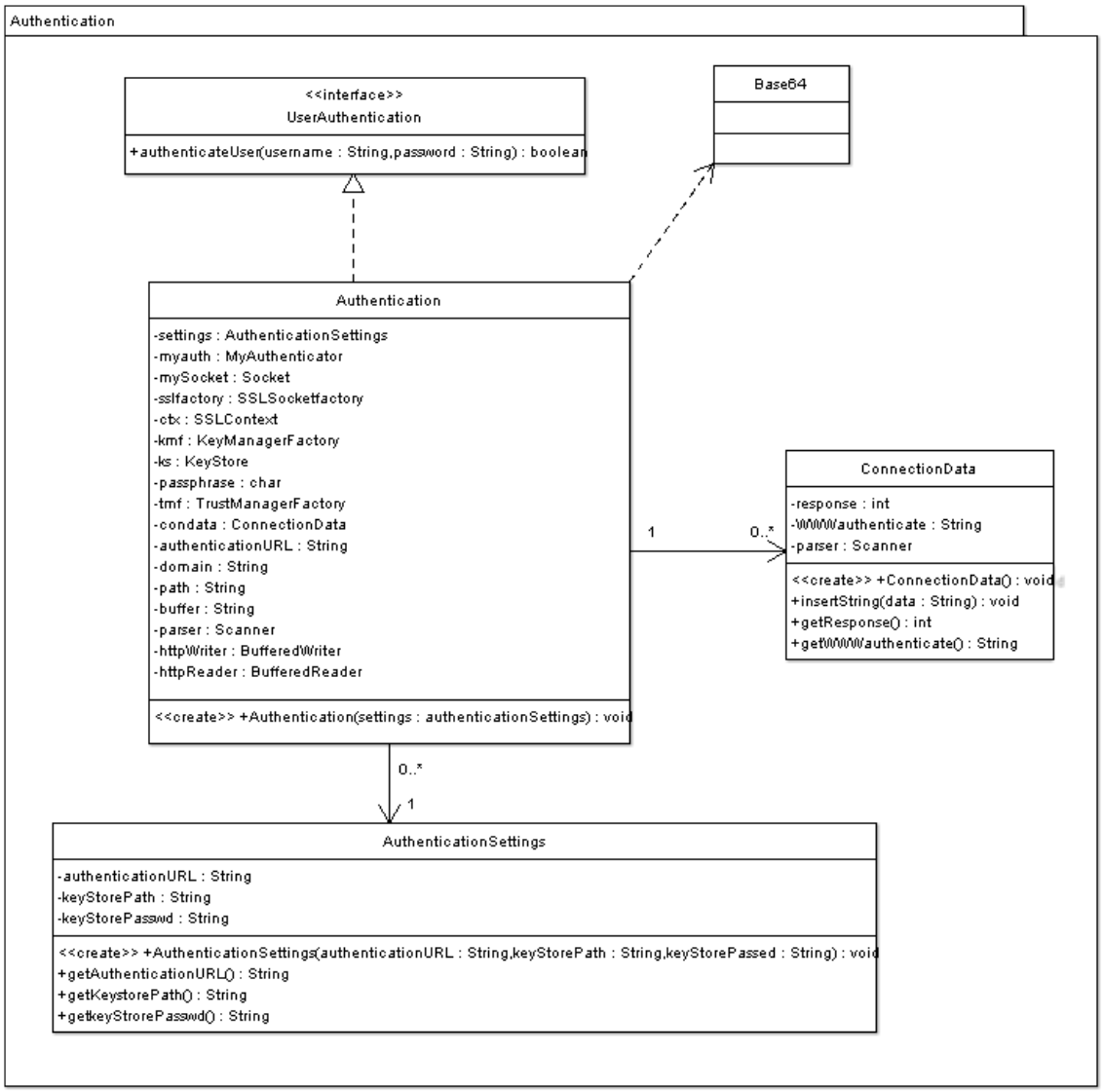
1. Asiakasliittymän tieto-oliot



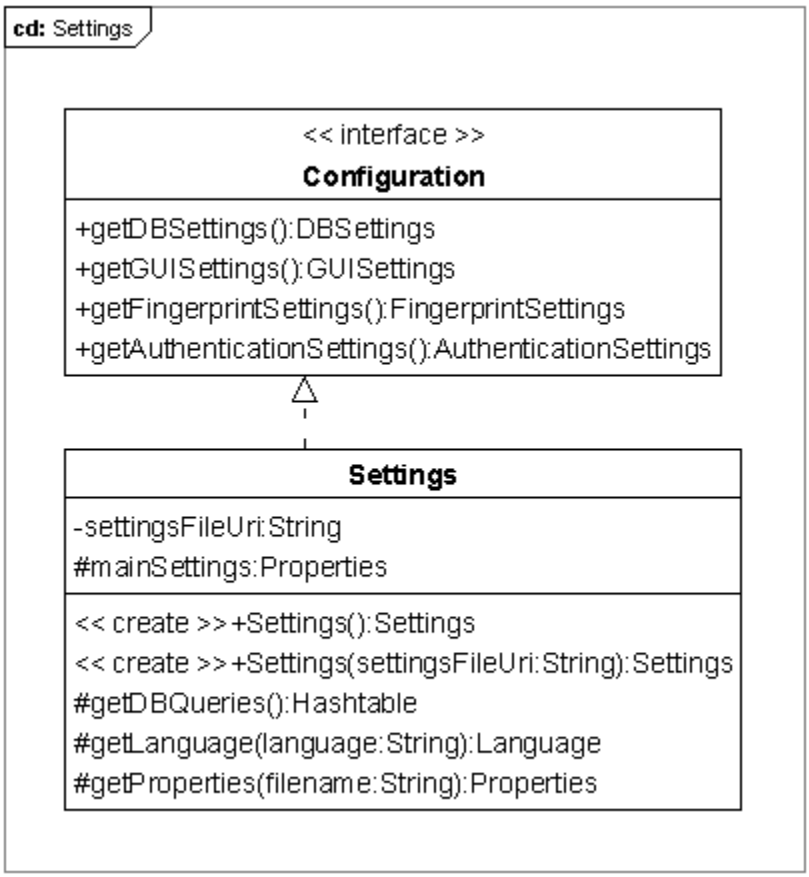
2. MainProgram-komponentin luokkakaavio



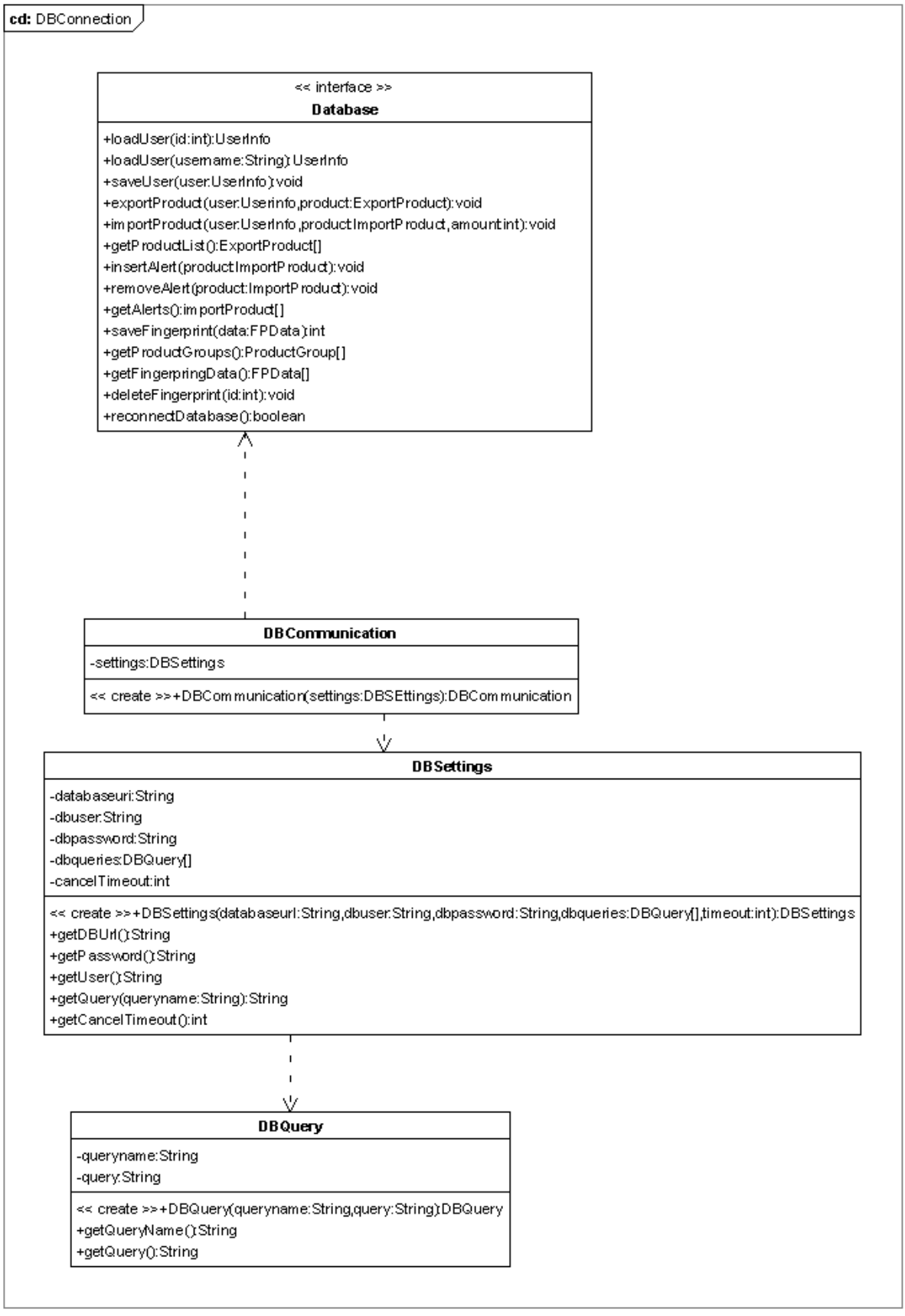
3. Authentication-komponentin luokkakaavio



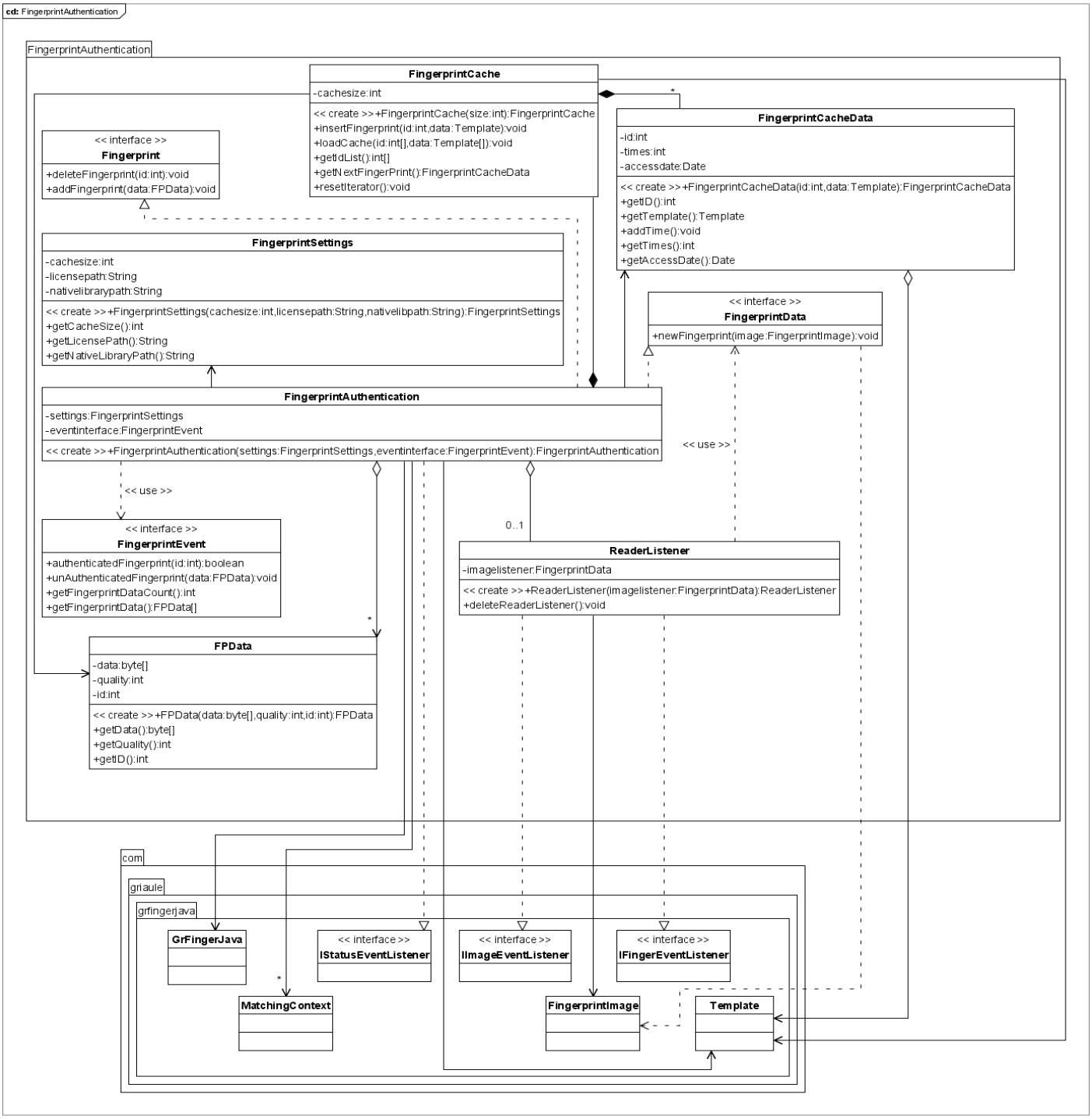
4. Settings-komponentin luokkakaavio.



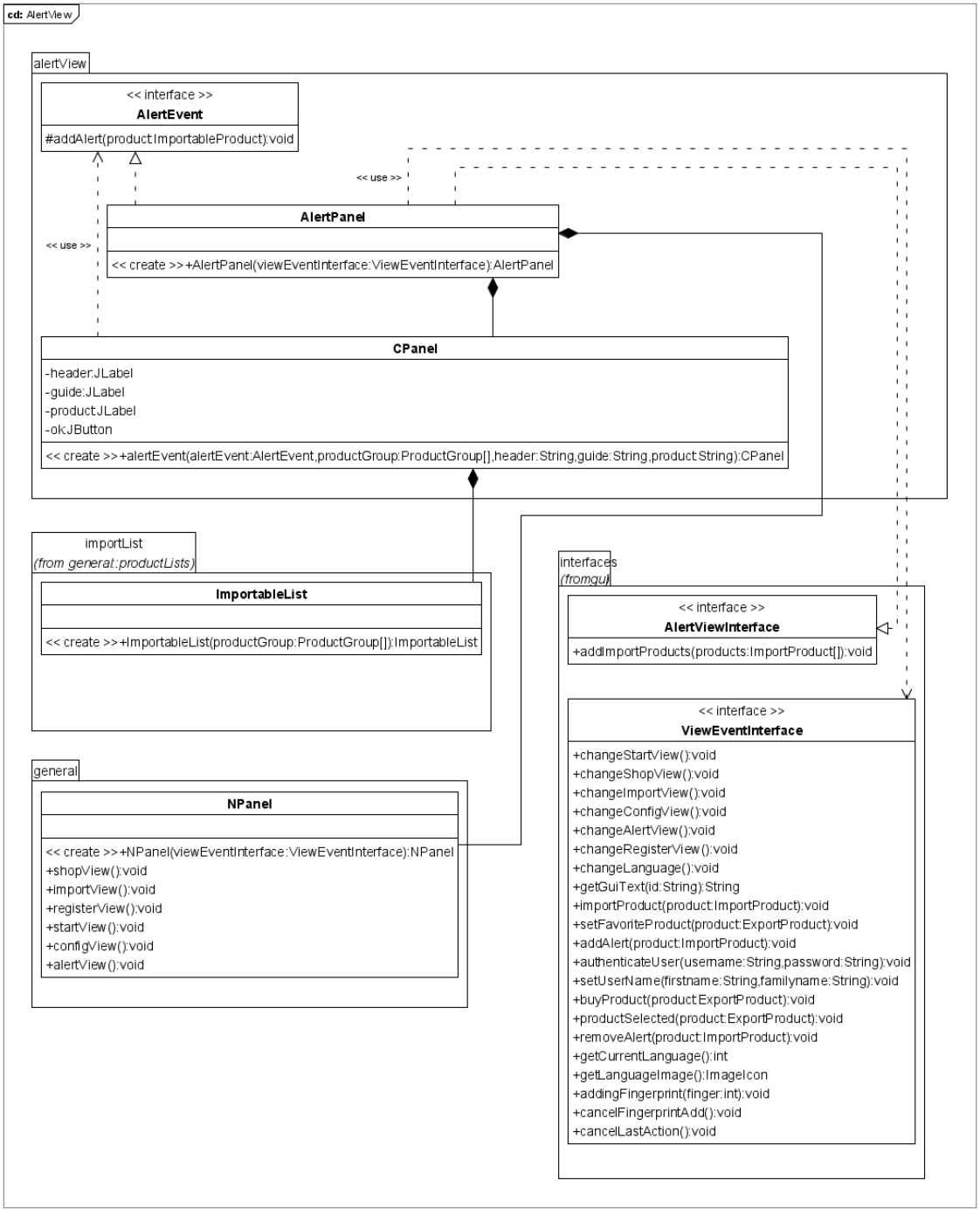
5. Database-komponentin luokkakaavio

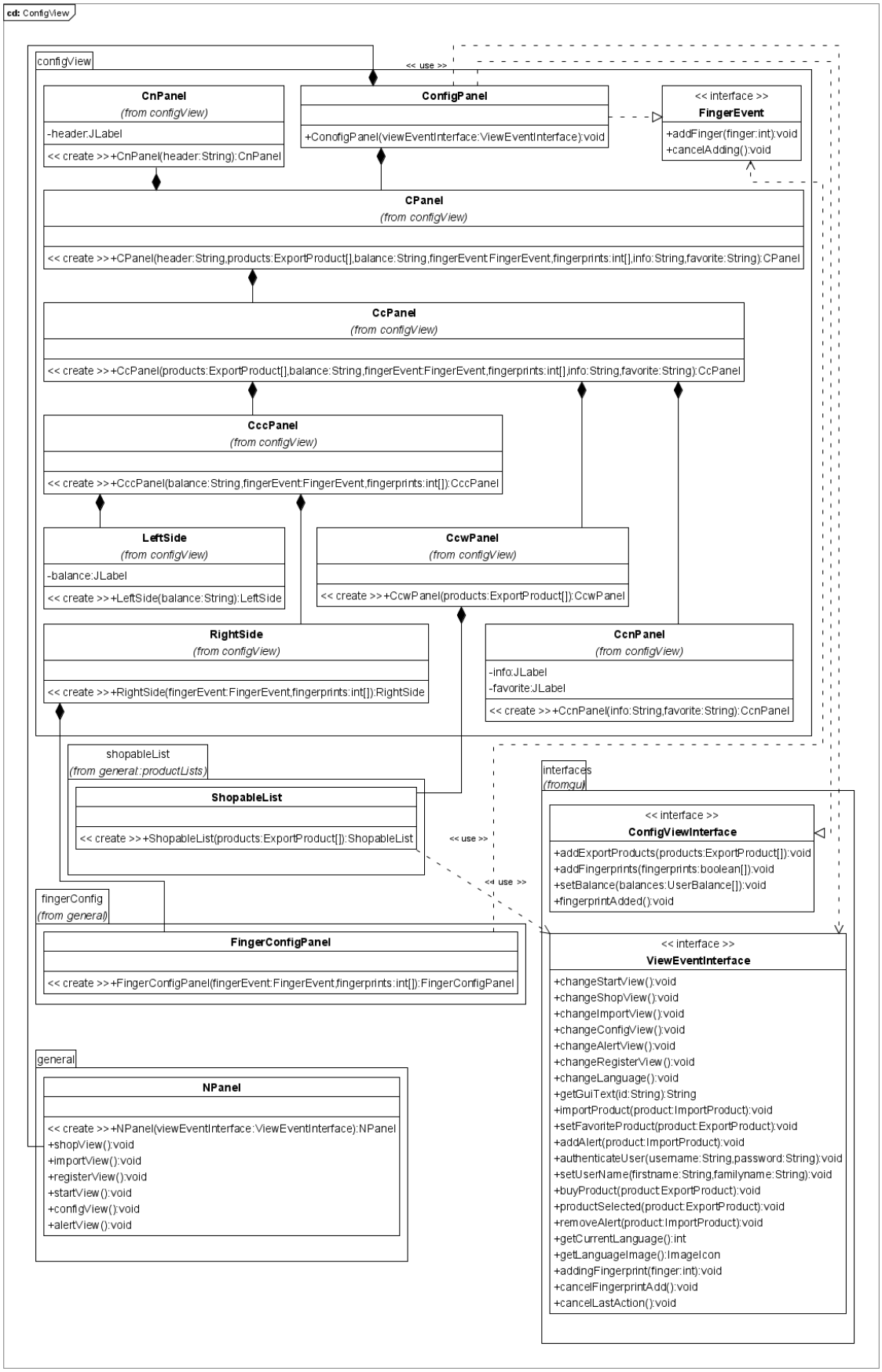


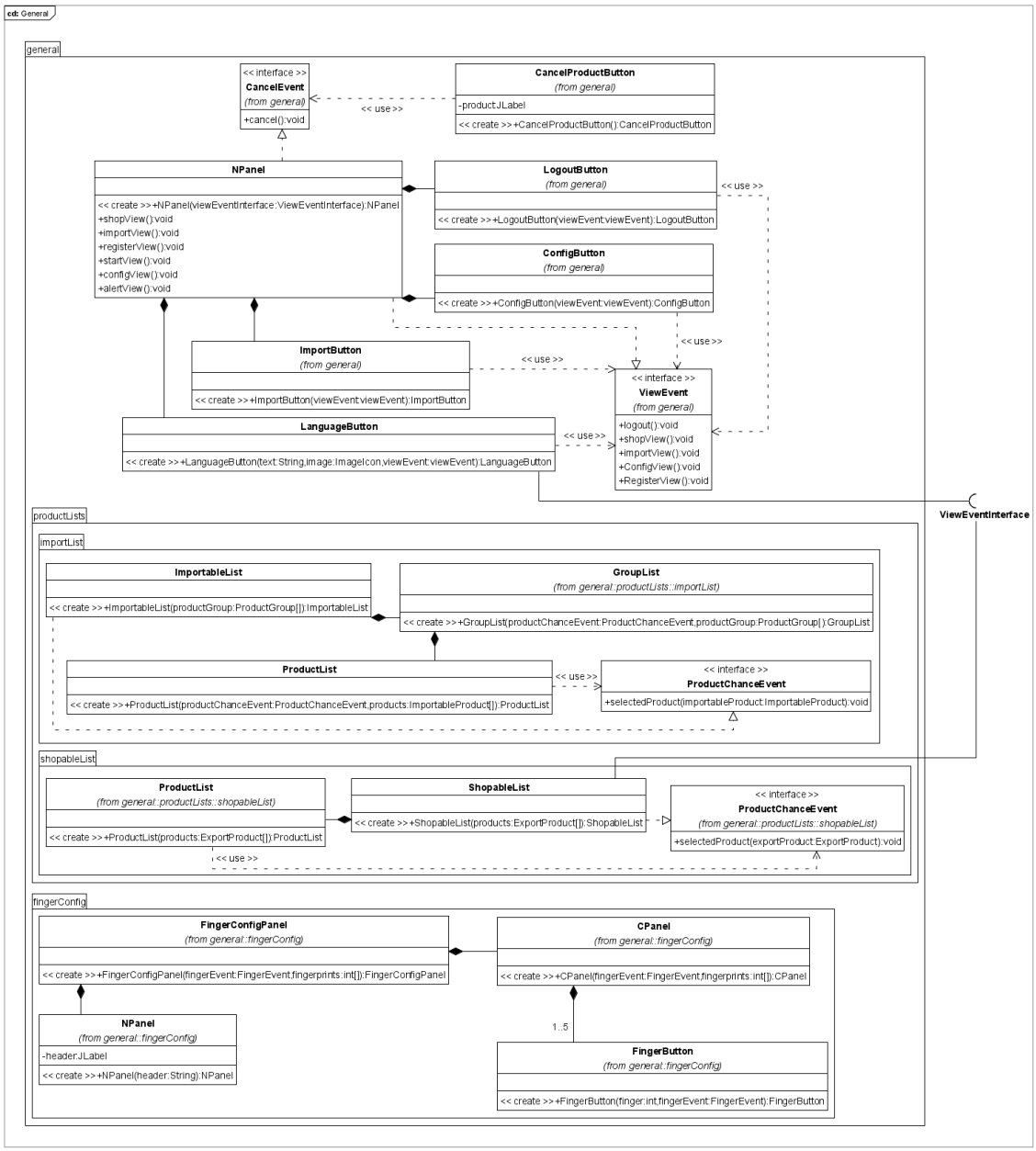
6. FingerprintAuthentication-komponentin luokkakaavio

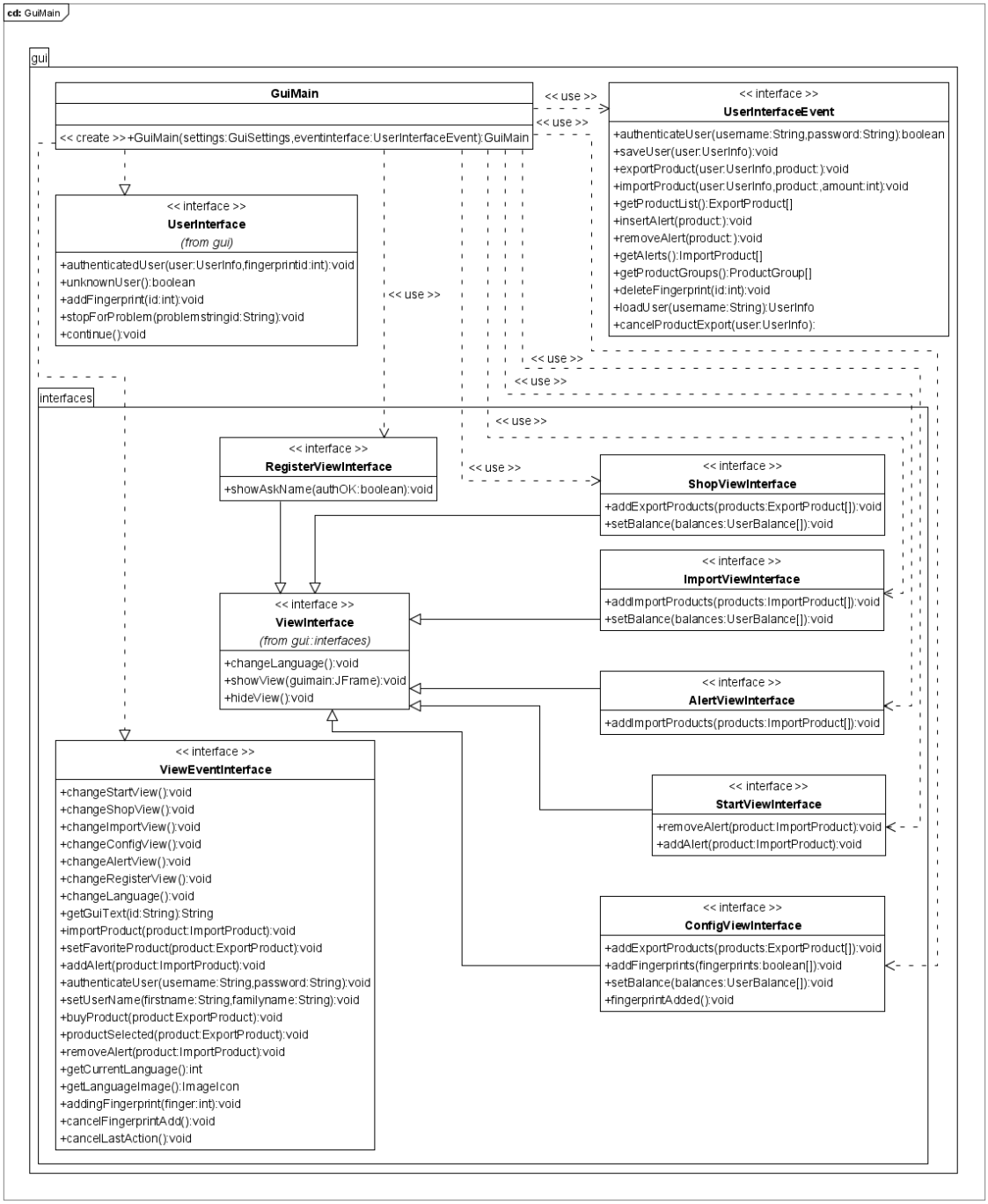


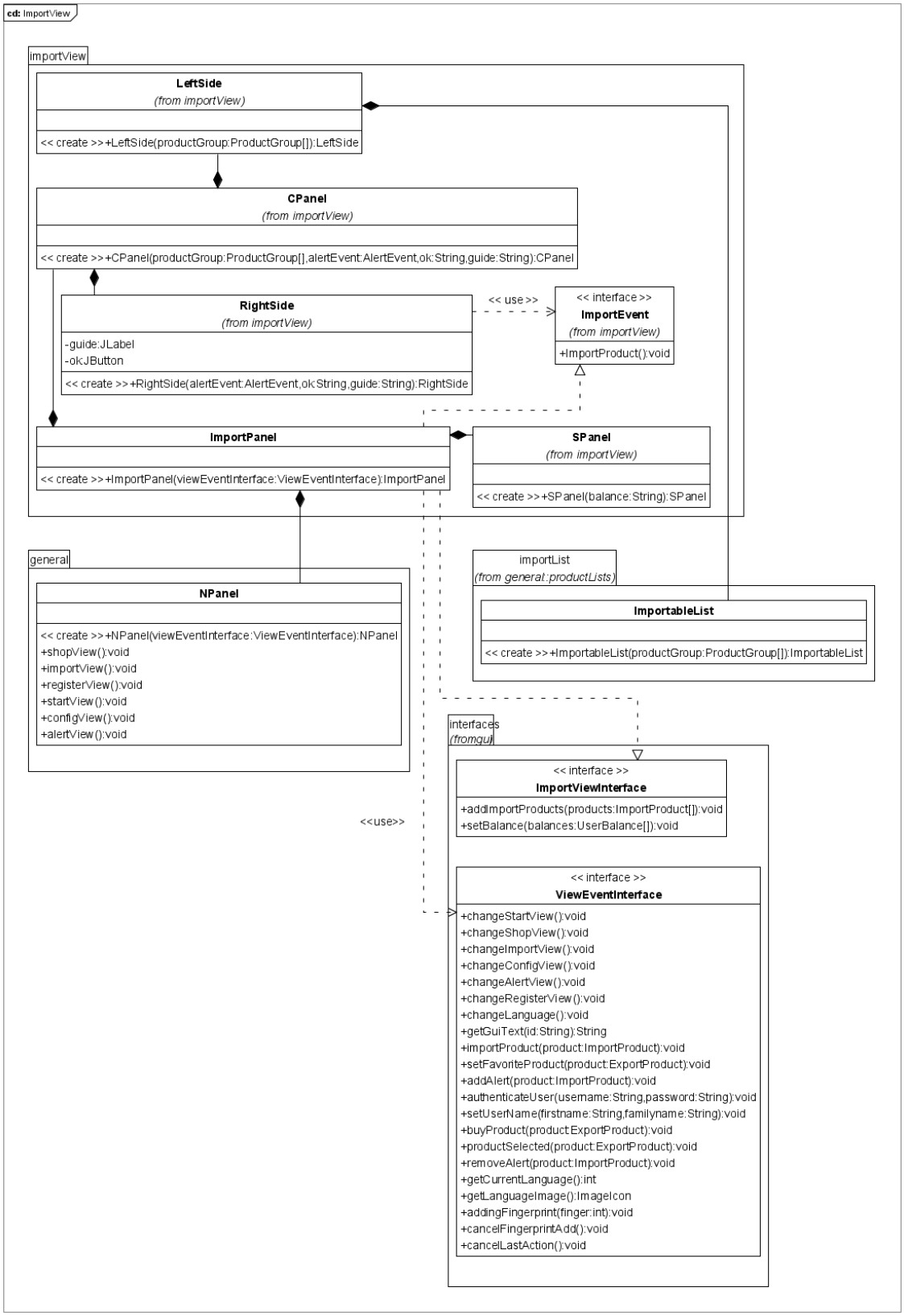
7. GUI-komponentin luokkakaaviot

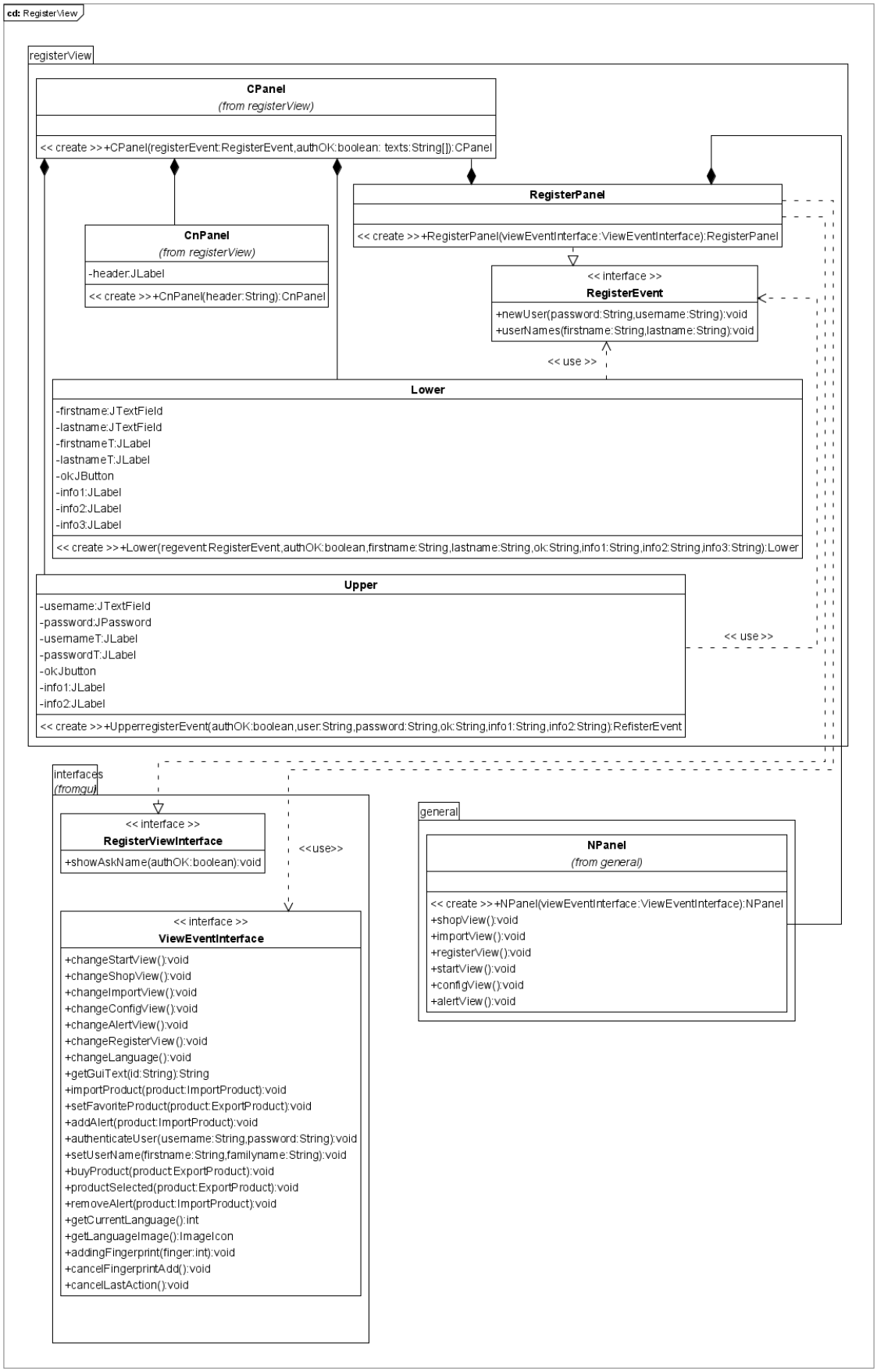


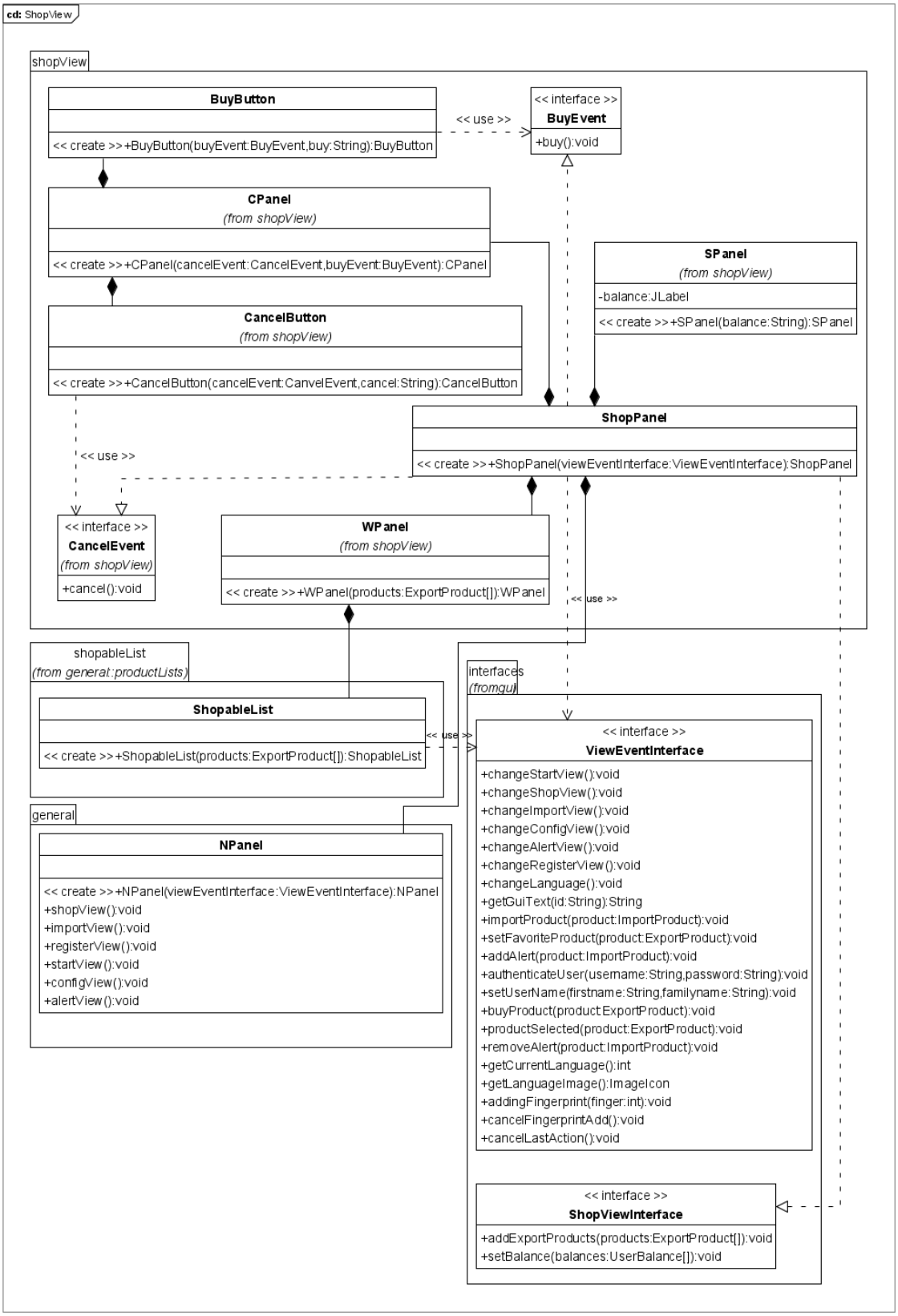

cd: ConfigView
configView
<< use >>
CnPanel
(from configView)
-header:JLabel
<< create >>+CnPanel(header:String):CnPanel
ConfigPanel
+ConofigPanel(viewEventInterface:ViewEventInterface):void
<< interface >>
FingerEvent
+addFinger(finger:int):void
+cancelAdding():void
CPanel
(from configView)
<< create >>+CPanel(header:String,products:ExportProduct[],balance:String,fingerEvent:FingerEvent,fingerprints:int[],info:String,favorite:String):CPanel
CcPanel
(from configView)
<< create >>+CcPanel(products:ExportProduct[],balance:String,fingerEvent:FingerEvent,fingerprints:int[],info:String,favorite:String):CcPanel
CccPanel
(from configView)
<< create >>+CccPanel(balance:String,fingerEvent:FingerEvent,fingerprints:int[]):CccPanel
LeftSide
(from configView)
-balance:JLabel
<< create >>+LeftSide(balance:String):LeftSide
CcwPanel
(from configView)
<< create >>+CcwPanel(products:ExportProduct[]):CcwPanel
RightSide
(from configView)
<< create >>+RightSide(fingerEvent:FingerEvent,fingerprints:int[]):RightSide
CcnPanel
(from configView)
-info:JLabel
-favorite:JLabel
<< create >>+CcnPanel(info:String,favorite:String):CcnPanel
shopableList
(from general::productLists)
ShopableList
<< create >>+ShopableList(products:ExportProduct[]):ShopableList
<< use >>
<< use >>
interfaces
(fromgui)
<< interface >>
ConfigViewInterface
+addExportProducts(products:ExportProduct[]):void
+addFingerprints(fingerprints:boolean[]):void
+setBalance(balances:UserBalance[]):void
+fingerprintAdded():void
fingerConfig
(from general)
FingerConfigPanel
<< create >>+FingerConfigPanel(fingerEvent:FingerEvent,fingerprints:int[]):FingerConfigPanel
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void
general
NPanel
<< create >>+NPanel(viewEventInterface:ViewEventInterface):NPanel
+shopView():void
+importView():void
+registerView():void
+startView():void
+configView():void
+alertView():void

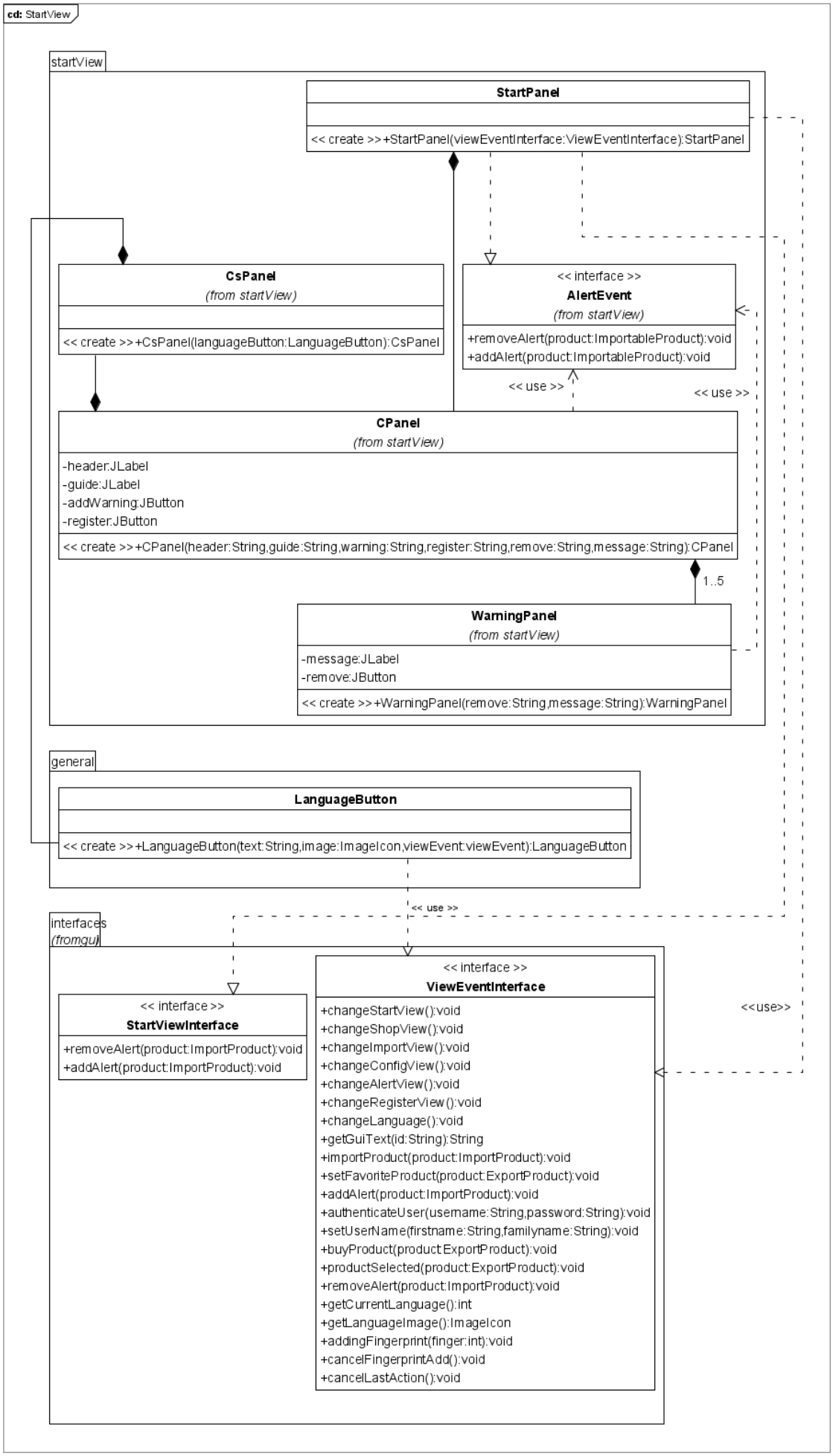

cd: General
general
<< interface >>
CancelEvent
(from general)
+cancel():void
<< use >>
CancelProductButton
(from general)
-productJLabel
<< create >>+CancelProductButton():CancelProductButton
NPanel
<< create >>+NPanel(viewEventInterface:ViewEventInterface):NPanel
+shopView():void
+importView():void
+registerView():void
+startView():void
+configView():void
+alertView():void
LogoutButton
(from general)
<< create >>+LogoutButton(viewEvent:viewEvent):LogoutButton
<< use >>
ConfigButton
(from general)
<< create >>+ConfigButton(viewEvent:viewEvent):ConfigButton
<< use >>
ImportButton
(from general)
<< create >>+ImportButton(viewEvent:viewEvent):ImportButton
<< use >>
<< interface >>
ViewEvent
(from general)
+logout():void
+shopView():void
+importView():void
+ConfigView():void
+RegisterView():void
LanguageButton
<< create >>+LanguageButton(text:String,image:ImageIcon,viewEvent:viewEvent):LanguageButton
<< use >>
ViewEventInterface
productLists
importList
ImportableList
<< create >>+ImportableList(productGroup:ProductGroup[]):ImportableList
GroupList
(from general::productLists::importList)
<< create >>+GroupList(productChanceEvent:ProductChanceEvent,productGroup:ProductGroup[]):GroupList
ProductList
<< create >>+ProductList(productChanceEvent:ProductChanceEvent,products:ImportableProduct[]):ProductList
<< use >>
<< interface >>
ProductChanceEvent
+selectedProduct(importableProduct:ImportableProduct):void
shopableList
ProductList
(from general::productLists::shopableList)
<< create >>+ProductList(products:ExportProduct[]):ProductList
<< use >>
ShopableList
<< create >>+ShopableList(products:ExportProduct[]):ShopableList
<< interface >>
ProductChanceEvent
(from general::productLists::shopableList)
+selectedProduct(exportProduct:ExportProduct):void
fingerConfig
FingerConfigPanel
(from general::fingerConfig)
<< create >>+FingerConfigPanel(fingerEvent:FingerEvent,fingerprints:int[]):FingerConfigPanel
CPanel
(from general::fingerConfig)
<< create >>+CPanel(fingerEvent:FingerEvent,fingerprints:int[]):CPanel
NPanel
(from general::fingerConfig)
-header:JLabel
<< create >>+NPanel(header:String):NPanel
1..5
FingerButton
(from general::fingerConfig)
<< create >>+FingerButton(finger:int,fingerEvent:FingerEvent):FingerButton


cd: GuiMain
gui
GuiMain
<< create >>+GuiMain(settings:GuiSettings,eventinterface:UserInterfaceEvent):GuiMain
<< use >>
<< interface >>
UserInterfaceEvent
+authenticateUser(username:String,password:String):boolean
+saveUser(user:UserInfo):void
+exportProduct(user:UserInfo,product:):void
+importProduct(user:UserInfo,product:,amount:int):void
+getProductList():ExportProduct[]
+insertAlert(product:):void
+removeAlert(product:):void
+getAlerts():ImportProduct[]
+getProductGroups():ProductGroup[]
+deleteFingerprint(id:int):void
+loadUser(username:String):UserInfo
+cancelProductExport(user:UserInfo):
<< interface >>
UserInterface
(from gui)
+authenticatedUser(user:UserInfo,fingerprintid:int):void
+unknownUser():boolean
+addFingerprint(id:int):void
+stopForProblem(problemstringid:String):void
+continue():void
interfaces
<< interface >>
RegisterViewInterface
+showAskName(authOK:boolean):void
<< interface >>
ShopViewInterface
+addExportProducts(products:ExportProduct[]):void
+setBalance(balances:UserBalance[]):void
<< interface >>
ImportViewInterface
+addImportProducts(products:ImportProduct[]):void
+setBalance(balances:UserBalance[]):void
<< interface >>
ViewInterface
(from gui::interfaces)
+changeLanguage():void
+showView(guimain:JFrame):void
+hideView():void
<< interface >>
AlertViewInterface
+addImportProducts(products:ImportProduct[]):void
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void
<< interface >>
StartViewInterface
+removeAlert(product:ImportProduct):void
+addAlert(product:ImportProduct):void
<< interface >>
ConfigViewInterface
+addExportProducts(products:ExportProduct[]):void
+addFingerprints(fingerprints:boolean[]):void
+setBalance(balances:UserBalance[]):void
+fingerprintAdded():void


cd: ImportView
importView
LeftSide
(from importView)
<< create >> +LeftSide(productGroup:ProductGroup[]):LeftSide
CPanel
(from importView)
<< create >> +CPanel(productGroup:ProductGroup[],alertEvent:AlertEvent,ok:String,guide:String):CPanel
RightSide
(from importView)
-guide:JLabel
-okJButton
<< create >> +RightSide(alertEvent:AlertEvent,ok:String,guide:String):RightSide
<< use >>
<< interface >>
ImportEvent
(from importView)
+ImportProduct():void
ImportPanel
<< create >> +ImportPanel(viewEventInterface:ViewEventInterface):ImportPanel
SPanel
(from importView)
<< create >> +SPanel(balance:String):SPanel
general
NPanel
<< create >> +NPanel(viewEventInterface:ViewEventInterface):NPanel
+shopView():void
+importView():void
+registerView():void
+startView():void
+configView():void
+alertView():void
importList
(from general::productLists)
ImportableList
<< create >> +ImportableList(productGroup:ProductGroup[]):ImportableList
interfaces
(fromgui)
<< interface >>
ImportViewInterface
+addImportProducts(products:ImportProduct[]):void
+setBalance(balances:UserBalance[]):void
<<use>>
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void


cd: RegisterView
registerView
CPanel
(from registerView)
<< create >> +CPanel(registerEvent:RegisterEvent,authOK:boolean: texts:String[]):CPanel
RegisterPanel
<< create >> +RegisterPanel(viewEventInterface:ViewEventInterface):RegisterPanel
CnPanel
(from registerView)
-header:JLabel
<< create >> +CnPanel(header:String):CnPanel
<< interface >>
RegisterEvent
+newUser(password:String,username:String):void
+userNames(firstname:String,lastname:String):void
<< use >>
Lower
-firstname:JTextField
-lastname:JTextField
-firstnameT:JLabel
-lastnameT:JLabel
-okJButton
-info1:JLabel
-info2:JLabel
-info3:JLabel
<< create >> +Lower(regevent:RegisterEvent,authOK:boolean,firstname:String,lastname:String,ok:String,info1:String,info2:String,info3:String):Lower
Upper
-username:JTextField
-password:JPassword
-usernameT:JLabel
-passwordT:JLabel
-okJbutton
-info1:JLabel
-info2:JLabel
<< create >> +UpperregisterEvent(authOK:boolean,user:String,password:String,ok:String,info1:String,info2:String):RefisterEvent
<< use >>
interfaces
(fromgui)
<< interface >>
RegisterViewInterface
+showAskName(authOK:boolean):void
<<use>>
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void
general
NPanel
(from general)
<< create >> +NPanel(viewEventInterface:ViewEventInterface):NPanel
+shopView():void
+importView():void
+registerView():void
+startView():void
+configView():void
+alertView():void


cd: ShopView
shopView
BuyButton
<< create >> +BuyButton(buyEvent:BuyEvent,buy:String):BuyButton
<< use >>
<< interface >>
BuyEvent
+buy():void
CPanel
(from shopView)
<< create >> +CPanel(cancelEvent:CancelEvent,buyEvent:BuyEvent):CPanel
SPanel
(from shopView)
-balance:JLabel
<< create >> +SPanel(balance:String):SPanel
CancelButton
(from shopView)
<< create >> +CancelButton(cancelEvent:CanvelEvent,cancel:String):CancelButton
ShopPanel
<< create >> +ShopPanel(viewEventInterface:ViewEventInterface):ShopPanel
<< use >>
<< interface >>
CancelEvent
(from shopView)
+cancel():void
WPanel
(from shopView)
<< create >> +WPanel(products:ExportProduct[]):WPanel
<< use >>
shopableList
(from general::productLists)
ShopableList
<< create >> +ShopableList(products:ExportProduct[]):ShopableList
<< use >>
interfaces
(fromgui)
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void
general
NPanel
<< create >> +NPanel(viewEventInterface:ViewEventInterface):NPanel
+shopView():void
+importView():void
+registerView():void
+startView():void
+configView():void
+alertView():void
<< interface >>
ShopViewInterface
+addExportProducts(products:ExportProduct[]):void
+setBalance(balances:UserBalance[]):void


cd: StartView
startView
StartPanel
<< create >>+StartPanel(viewEventInterface:ViewEventInterface):StartPanel
CsPanel
(from startView)
<< create >>+CsPanel(languageButton:LanguageButton):CsPanel
<< interface >>
AlertEvent
(from startView)
+removeAlert(product:ImportableProduct):void
+addAlert(product:ImportableProduct):void
<< use >>
<< use >>
CPanel
(from startView)
-header:JLabel
-guide:JLabel
-addWarning:JButton
-register:JButton
<< create >>+CPanel(header:String,guide:String,warning:String,register:String,remove:String,message:String):CPanel
1..5
WarningPanel
(from startView)
-message:JLabel
-remove:JButton
<< create >>+WarningPanel(remove:String,message:String):WarningPanel
general
LanguageButton
<< create >>+LanguageButton(text:String,image:ImageIcon,viewEvent:viewEvent):LanguageButton
<< use >>
interfaces
(fromgui)
<< interface >>
StartViewInterface
+removeAlert(product:ImportProduct):void
+addAlert(product:ImportProduct):void
<< interface >>
ViewEventInterface
+changeStartView():void
+changeShopView():void
+changeImportView():void
+changeConfigView():void
+changeAlertView():void
+changeRegisterView():void
+changeLanguage():void
+getGuiText(id:String):String
+importProduct(product:ImportProduct):void
+setFavoriteProduct(product:ExportProduct):void
+addAlert(product:ImportProduct):void
+authenticateUser(username:String,password:String):void
+setUserName(firstname:String,familyname:String):void
+buyProduct(product:ExportProduct):void
+productSelected(product:ExportProduct):void
+removeAlert(product:ImportProduct):void
+getCurrentLanguage():int
+getLanguageImage():ImageIcon
+addingFingerprint(finger:int):void
+cancelFingerprintAdd():void
+cancelLastAction():void
<<use>>