4.5.2017

Dokumentaatio: Pizzapalvelun tietokantasovellus nimeltään Gizza

Johdanto

Gizza on kuvitteelliselle Pizzeria Omertalle suunniteltu ja toteutettu tietojärjestelmä, jonka pääasiallinen tarkoitus on mahdollistaa pizzojen tilaus Internetin välityksellä. Asiakas siis käyttää Gizzaa selaimella. Hän voi valita tilaukseensa pizzojen lisäksi myös muita tuotteita kuten esim. juomia. Asiakkaan tekemään tilaukseen kuuluu myös tieto siitä, milloin ja mihin osoitteeseen pizzatilaus on tarkoitus toimittaa.

Gizzan ominaisuuksiin kuuluu asiakkaan identifioiminen ja asiakkaaseen liittyvien historiatietojen tallentaminen. Käytännössä tämä tarkoittaa sitä, että asiakkaan kannalta järjestelmän käyttäminen edellyttää asiakastilin luomista. Asiakkaan tekemät tilaukset tallennetaan järjestelmään. Sekä asiakas itse että pizzerian henkilökunta voivat myöhemmin tarkastella näitä tilaustietoja. Henkilökunnalla on myös mahdollisuus tallentaa ja tarkastella tietoja siitä, sujuiko tilauksen toimitus kuten piti vai ilmenikö ongelmia. Jokainen ongelma liittyy tiettyyn tilaukseen, ja toisaalta jokainen tilaus liittyy tiettyyn asiakkaaseen. Henkilökunta voi siis Gizzan avulla saada tiedon asiakkaista, joiden kanssa on ilmennyt ongelmia. (Gizzan tietokantaan tallennetut asiakkaat ovat puhtaasti kuvitteellisia henkilöitä, joten tässä yhteydessä ei ole tarpeen pohtia henkilötietolain asettamia rajoituksia.)

Gizzan toiminnallisuus määräytyy melko pitkälti sen mukaan, kirjaudutaanko sisään ylläpitäjänä (eli henkilökunnan jäsenenä) vai asiakkaana. Ylläpitäjä voi tarkastella ja muokata kaikkiin tilauksiin, asiakkaisiin ja asiakkaiden osoitekirjoihin liittyviä tietoja. Ylläpitäjä voi myös tehdä, muokata ja poistaa tilauksia asiakkaiden puolesta. Tämä mahdollistaa esim. puhelinasiakkaiden helpon palvelemisen Gizzan avulla.

Asiakas voi Gizzan avulla tehdä, muokata, katsella ja poistaa omia pizzatilauksiaan. Hän voi myös katsella ja muuttaa oman asiakastilinsä tietoja sekä halutessaan poistaa tilinsä. Kenellä tahansa on mahdollisuus luoda Gizzaan uusi asiakastili. Valitettava rajoitus asiakkaan kannalta on se, että hän ei itse voi lisätä osoitekirjaansa uusia osoitteita, vaan hänen täytyy pyytää ylläpitäjää tekemään se puolestaan (en ehtinyt toteuttaa kovin toiminnallista osoitekirjaa).

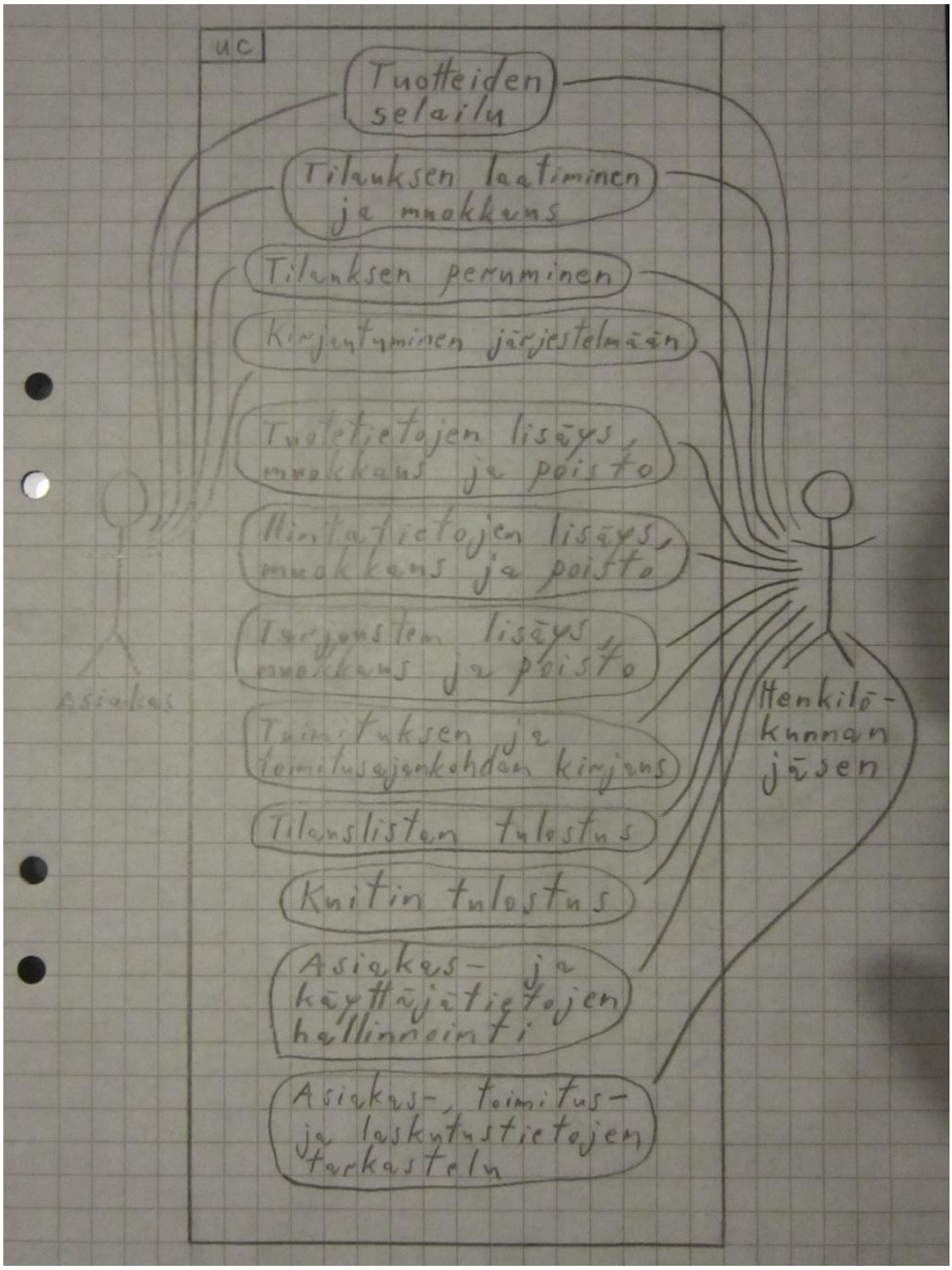
Olen pyrkinyt Gizzassa siihen, että käyttöoikeuksiin liittyvän tietoturvan perusasiat ovat kunnossa. Tavallisen asiakastilin kautta ei siis pitäisi olla mahdollista katsoa tai muuttaa toisiin asiakkaisiin tai heidän tilauksiinsa liittyviä tietoja. Vahva tietoturva ei kuitenkaan ole tämän kurssin kaikkein keskeisin aihe, joten en ole varmistanut tietoturvan toimivuutta millään kovin systemaattisella tavalla.

Gizza on web-sovellus, jonka alustajärjestelmän on tuettava PHP-ohjelmointikieltä ja PostgreSQL-tietokantaa. Oletus on, että sovellus ei tule olemaan helposti siirrettävissä (portattavissa) eri tietokantajärjestelmien välillä. Käyttäjän selaimelta ei vaadita erityistä tukea tietylle ohjelmointikielelle. Gizzan toteutus- ja toimintaympäristö on palvelin `users.cs.helsinki.fi` Apache-palvelun alla. Palvelimen `users` PHP-versio on 5.3.2-1ubuntu4.30 ja PostgreSQL-versio on 8.4.22.

Käyttötapaukset

Gizzan sidosryhminä ovat lähinnä asiakkaat ja henkilökunnan jäsenet. Ainakin jälkimmäinen ryhmä voitaisiin periaatteessa jakaa edelleen pienempiin osiin, kuten esim. johtoon, tilausten valmistajiin ja kuljettajiin. Tässä vaiheessa tuntuu kuitenkin järkevimmältä hahmottaa sidosryhmiksi vain asiakkaat ja henkilökunnan jäsenet. Asiakas on siis henkilö, joka tekee pizzeriaan tilauksen Gizzan välityksellä.

Seuraavassa Gizzan käyttötapauskaavion ensimmäinen versio. Siinä luetellut käyttötapaukset tuntuvat varsin helposti hahmotettavilta, joten kuvailen ne sanalliset vain lyhyesti.

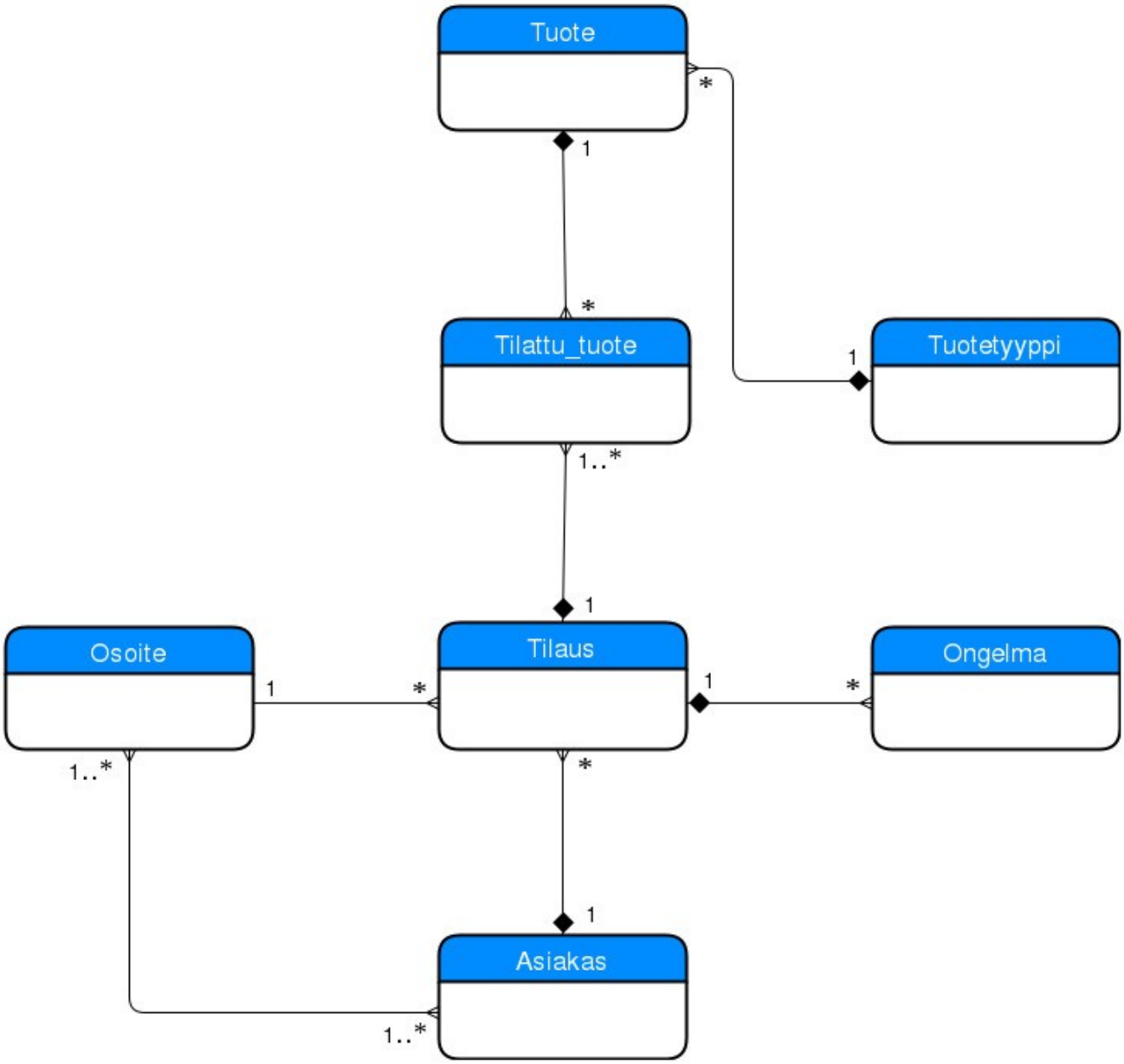


Seuraavassa taulukossa alleviivatut käyttötapaukset tarkoittavat niitä käyttötapauksia, jotka liittyvät sekä asiakkaisiin että henkilökunnan jäseniin. Huomaa, että kaikki käyttötapaukset liittyvät henkilökunnan jäseniin, mutta vain neljä käyttötapausta liittyy asiakkaisiin.

Käyttötapaus	**Kommentti**
Tuotteiden selailu	Sekä henkilökunta että asiakkaat voivat selailla pizzerian tuotevalikoimaa
Tilauksen laatiminen ja muokkaus	Asiakas voi muokata tilausta, jos sovittuun toimitusaikaan on yli tunti. Tuntuu järkevältä, että myös henkilökunnan jäsen voi laatia tilauksen esim. puhelinasiakkaan puolesta.
Tilauksen peruminen	Asiakas voi perua tilauksen, jos sovittuun toimitusaikaan on yli tunti
Kirjautuminen järjestelmään	Molemmat sidosryhmät tarvitsevat järjestelmän käyttöön käyttäjätunnuksen ja salasanan
Tuotetietojen lisäys, muokkaus ja poisto	Järjestelmä voisi automaattisesti pitää kirjaa siitä, paljonko tiettyä tuotetta (tai raaka-ainetta) on vielä jäljellä
Hintatietojen lisäys, muokkaus ja poisto	Hintatietoihin kuuluu myös tieto siitä, miten esim. vuorokaudenaika vaikuttaa hintaan
Tarjousten lisäys, muokkaus ja poisto	Tarjouksiin voi liittää myös ajankohdan, jolloin ne raukeavat. Tällöin tarjous ei enää näy asiakkaille.
Toimituksen ja toimitusajankohdan kirjaus	Toivotun ja todellisen toimitusajankohdan avulla järjestelmä voi laskea asiakkaan saaman myöhästymisalennuksen.
Tilauslistan tulostus	–
Kuitin tulostus	–
Asiakas- ja käyttäjätietojen tarkastelu sekä hallinnointi	Järjestelmällä voisi olla yksi käyttäjä ylläpitäjän valtuuksilla. Hän voisi jakaa muille henkilökunnan jäsenille näiden tarvitsemat oikeudet järjestelmän käyttöön.
Tilaus-, toimitus- ja laskutustietojen tarkastelu sekä hallinnointi	–

Järjestelmän tietosisältö

Seuraavassa on järjestelmän keskeinen tietosisältö kuvattu käsitekaavion avulla.



Ennen relaatiotietokantakaavion luomista käydään tietokohteet läpi tarkemmin kuvailemalla niiden tarkoitusta ja keskeisiä attribuutteja.

Tietokohde: Asiakas

Asiakas on Gizzassa tilausten tekijä ja hän on toisaalta se, jolle tilaukset toimitetaan ja joka ne maksaa. Perusidea on, että jokaisella asiakkaalla on käyttäjätunnus Gizzassa. Asiakas-tietokohteeseen kuuluu käsitteellisesti

jonkinlainen osoitekirja. Asiakas- ja osoitetaulujen välillä on monesta moneen -suhde, joka on toteutettu välitaulun avulla.

Tietokohde Asiakas sisältää tosiasiassa kaikki Gizzan käyttäjätunnukset eli myös ne, jotka kuuluvat henkilökunnan jäsenille. Asiakkaat ja pääkäyttäjät (henkilökunnan jäsenet) erottaa toisistaan attribuutti `on_paakayttaja`, jonka arvo on `true`, jos kyseessä on pääkäyttäjä.

Attribuutti	**Arvojoukko**	**Kuvaus**
ktunnus	Merkkijono. Saa sisältää vain pieniä kirjaimia. Pituus vähintään 3 ja enintään 20 merkkiä.	Pääavain. Käyttäjätunnus, jolla asiakas tai henkilökunnan jäsen kirjautuu Gizzaan.
on_paakayttaja	Boolean	Henkilökunnan jäsenet ovat pääkäyttäjiä.
Salasana	Merkkijono. Null-arvo ilmaisee, että käyttäjätunnus on poistettu (väliaikaisesti) käytöstä.	
etunimi	Merkkijono	
sukunimi	Merkkijono	
puhelinnumero	Merkkijono, mieluiten säännöllisen lausekkeen mukainen	
sahkopostiosoite	Kuten yllä	

Tietokohde: Osoite

Täsmälleen sama osoite (esim. Satukatu 1 A 2, 12345 Satukylä) ei saisi toistua taulussa. PostgreSQL:llä tällaisen toistumisen voi ilmeisesti estää create table -rakenteen rivillä UNIQUE(lahiosoite, postinumero, postitoimipaikka).

Attribuutti	**Arvojoukko**	**Kuvaus**
osoite_id	int	Pääavain, joka toimii muissa tauluissa kompaktina viiteavaimena
lähiosoite	Merkkijono	
postinumero	Merkkijono	
postitoimipaikka	Merkkijono	

Tietokohde: mm_Asiakas_Osoite

Tällä taululla toteutetaan monesta moneen -yhteys Asiakas- ja Osoite-taulujen välillä. Pääavain on siis kaksiosainen.

Attribuutti	**Arvojoukko**	**Kuvaus**
ktunnus	string	Toimii myös viiteavaimena Asiakas-tauluun
osoite_id	int	Toimii myös viiteavaimena Osoite-tauluun

Tietokohde: Tilaus

Tilaus- ja Osoite-taulujen välillä on suora yhteys. Tämä voi ensikatsomalta vaikuttaa turhalta, koska myös Asiakas-tauluun on suora yhteys. Asiakkaalla voi kuitenkin olla useampi kuin yksi osoite. Oletetaan, että jos tilaus on luovutettu asiakkaalle (eli kentän ts_tak_toteutunut arvo ei ole NULL), niin asiakas on myös maksanut tilauksen. Attribuutit ktunnus ja ts_tilauksen_teko on merkitty **vahvennetulla**. Tämä tarkoittaa sitä, että parin (ktunnus, ts_tilauksen_teko) on tarkoitus olla taulussa uniikki (unique constraint). Tämä ilmentää sitä, että tilauksen olemassaolo on riippuvainen sen tehneestä asiakkaasta.

Attribuutti	**Arvojoukko**	**Kuvaus**
tilaus_id	int	Pääavain
ktunnus	string	Viiteavain Asiakas-tauluun
ts_tilauksen_teko	Timestamp	Hetki, jona tilaus on jätetty järjestelmään
ts_tak_toivottu	Timestamp	Tak eli toimitusajankohta. Asiakkaan toivoma ajankohta toimitukselle. Arvo NULL tarkoittaa ”mahdollisimman pian”.
ts_tak_toteutunut	Timestamp	Ajankohta, jolloin toimitus luovutettiin sen maksaneelle asiakkaalle. Ennen luovutusta arvo on NULL.
osoite_id	int	Viiteavain Osoite-tauluun

Tietokohde: Ongelma

Tiettyyn tilaukseen liittyy nolla tai useampi ongelmaa, ja toisaalta jokainen ongelma liittyy johonkin tiettyyn tilaukseen. Ongelma-tietokohteen avulla on tarkoitus tallentaa järjestelmään tietoja lähinnä asiakkaiden kanssa tulleista

ongelmista. Esim. tilaukseen 1234 voi liittyä ongelmat ”tilauksen toimittaja pahoinpideltiin” ja ”asiakas kieltäytyi maksamasta tilausta vaikka otti sen vastaan”. Tuntuu järkevältä rajata ongelmien päätyypit esim. kolmeen: (1) väkivaltainen asiakas, (2) asiakasta ei tavoitettu, (3) asiakas ei ollut maksukykyinen. Varsinaisessa PostgreSQL-toteutuksessa tätä voisi vastata seuraava enum: `create type` **`Ongelma_enum`** `as enum( 'violence', 'customer_not_found', 'no_payment');`

Huomaa kaksiosainen pääavain. Avaimen ensimmäinen sarake yksilöi tilauksen. Toinen sarake on kolmiarvoinen enum, joten yksittäiseen tilaukseen voi liittyä korkeintaan kolme ongelmaa.

Attribuutti	**Arvojoukko**	**Kuvaus**
tilaus_id	int	Viiteavain Tilaus-tauluun
ongelman_tyyppi	Edellä mainittu Ongelma_enum, jolla kolme mahdollista arvoa	
ts_ongelma	Timestamp-arvo	Ongelman tapahtuma-ajankohta
ongelman_kuvaus	Merkkijono (hieman pidempi)	Vapaamuotoinen selvitys siitä, mitä tapahtui

Tietokohde: Tuotetyyppi

Samasta tuotteesta voi olla useita versioita. Esim. Americano-pizzaa voi olla isoa ja pientä kokoa. Tuotetyyppi on siis tässä tapauksessa Americano-pizza (itse tuote taas on tuotetyypin ja tuoteversion yhdistelmä). Jokaiseen tuotteeseen liittyy yksi tuotetyyppi; jokaiseen tuotetyyppiin liittyy yksi tai useampi tuote. Jokaiseen tuotetyyppiin liittyy tuotekategoria, esim. 'pizza' tai 'virvoitusjuoma'.

Attribuutti	**Arvojoukko**	**Kuvaus**
tuotetyyppi_id	int	Pääavain
tuotekategoria	Tuotekategoria_enum	Mahdollisina arvoina 'pizza', 'vegaanipizza', 'virvoitusjuoma', 'olut' ja 'muu'
tuotenimi	Merkkijono	Esim. 'Americano'
tuotekuvaus	Merkkijono	Tuotekuvaus on siis sama kaikille tuotetyypin versioille
kuva_tuotteesta	Merkkijono (tiedostopolku, esim. ~/images/kuva.jpg	

Tietokohde: Tuote

Tuotetta voi ajatella tuotetyypin eräänlaisena ”ilmentymänä”. Tuotteen olemassaolo on riippuvainen tietystä tuotetyypistä. Tuote yksilöidään sen tuotetyypin ja tuoteversion perusteella.

Attribuutti	**Arvojoukko**	**Kuvaus**
tuotetyyppi_id	int	Viiteavain Tuotetyyppi-tauluun
tuoteversio	Tuoteversio_enum	Arvoina 'pieni', 'tavallinen' ja 'iso'
hinta	Numeric(6,2), esim. 123.45	Tuotteen hinta

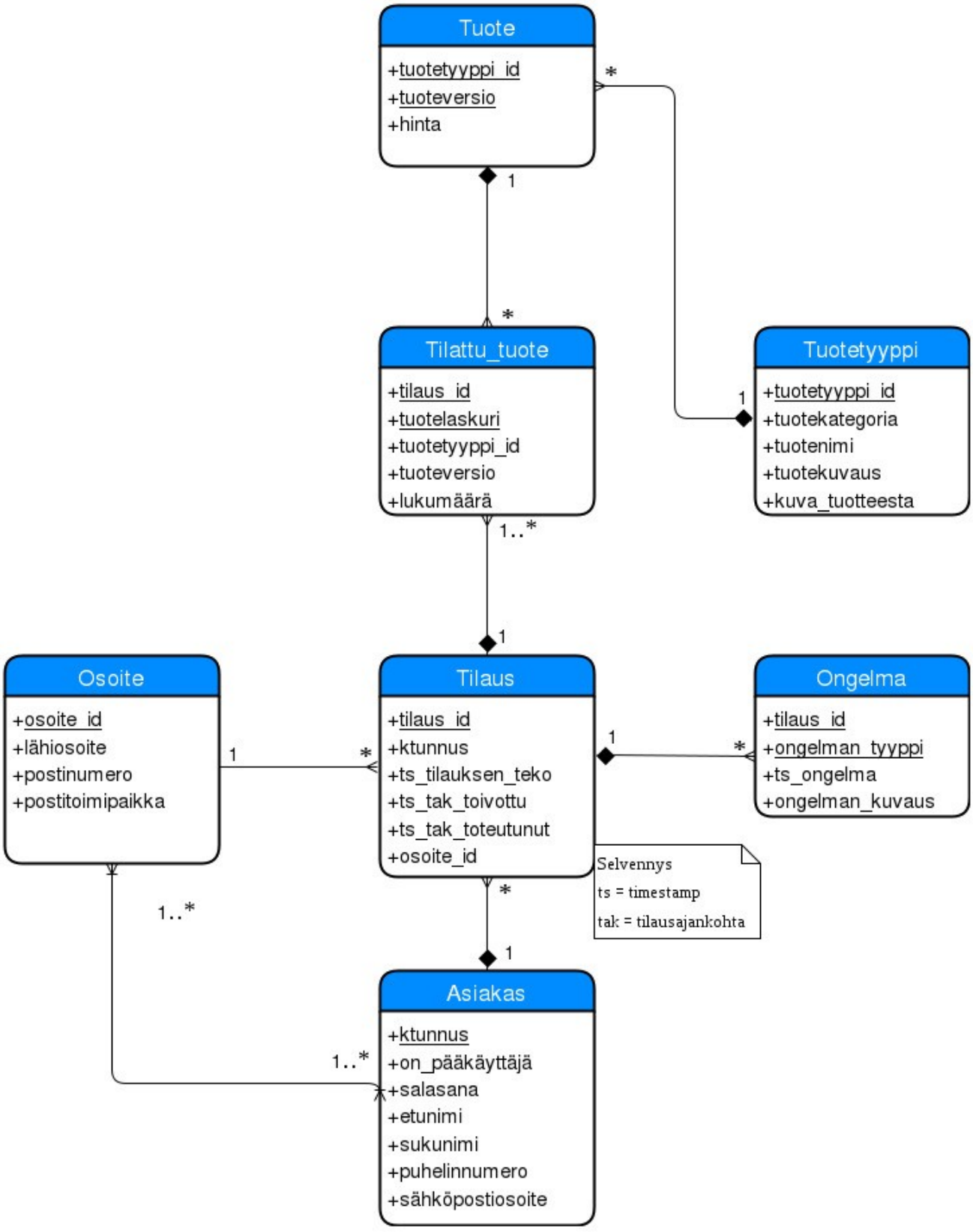
Tietokohde: Tilattu_tuote

Tilaus koostuu yhdestä tai useammasta tilatusta tuotteesta. Jokainen tilattu tuote liittyy täsmälleen yhteen tuotteeseen.

Attribuutti	**Arvojoukko**	**Kuvaus**
tilaus_id	int	Viiteavain Tilaus-tauluun
tuotelaskuri	Serial int	Tilatut tuotteet numeroidaan alkaen ykkösestä
tuotetyyppi_id	int	Yhdessä tuoteversion kanssa viiteavain tauluun Tuote
tuoteversio	Tuoteversio_enum	
lukumaara	int	Esim. 2 * Americano (iso)

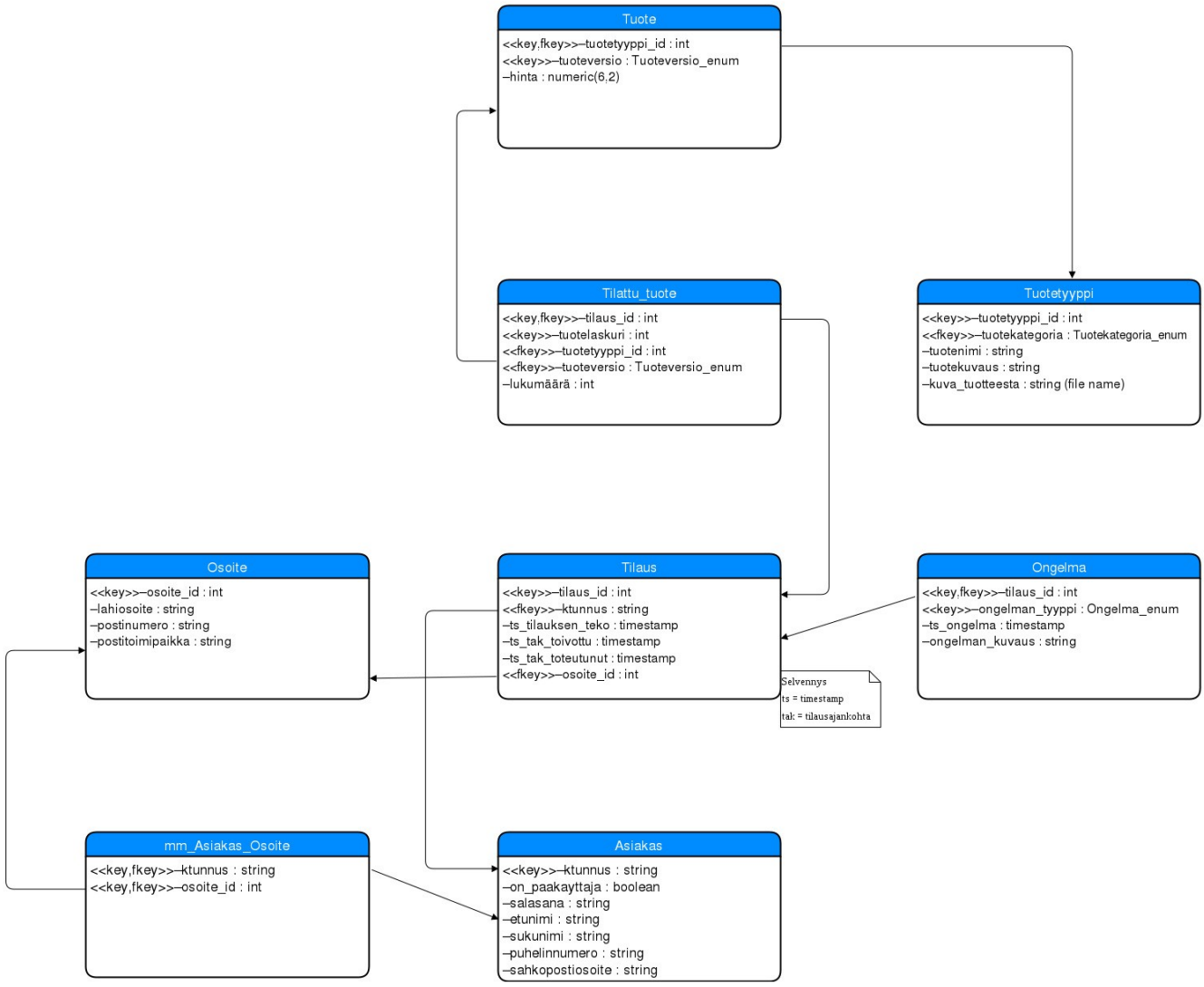
Attribuuteilla täydennetty käsitekaavio

Esitin jo aiemmin Gizzan tietosisällön käsitekaavion avulla. Seuraavalla sivulla on sama käsitekaavio attribuuteilla höystettynä.


Tuote
+tuotetyyppi_id
+tuoteversio
+hinta
Tilattu_tuote
+tilaus_id
+tuotelaskuri
+tuotetyyppi_id
+tuoteversio
+lukumäärä
Tuotetyyppi
+tuotetyyppi_id
+tuotekategoria
+tuotenimi
+tuotekuvaus
+kuva_tuotteesta
Osoite
+osoite_id
+lähiosoite
+postinumero
+postitoimipaikka
Tilaus
+tilaus_id
+ktunnus
+ts_tilauksen_teko
+ts_tak_toivottu
+ts_tak_toteutunut
+osoite_id
Ongelma
+tilaus_id
+ongelman_tyyppi
+ts_ongelma
+ongelman_kuvaus
Asiakas
+ktunnus
+on_pääkäyttäjä
+salasana
+etunimi
+sukunimi
+puhelinnumero
+sähköpostiosoite
Selvennys
ts = timestamp
tak = tilausajankohta
1
*
1..*

Relaatiotietokantakaavio

Lopuksi vielä relaatiotietokantakaavio, jossa näkyvät myös välitaulut ja viiteavaimet.



Asennustiedot

Sovellus on siis asennettuna palvelimelle `users.cs.helsinki.fi`. Mainitulta palvelimelta löytyy Apache-palvelu sekä PHP- ja PostgreSQL-tuki. Kurssiin liittyvän Tsoha Bootstrap -aloituspaketin mukana tuli työkaluja, jotka helpottivat sovelluksen asentamista usersille. En ole kokeillut asentaa Gizzaa muualle kuin usersille, mutta jos minulle tulisi tarve tehdä niin, menettelisin samaan tapaan kuin Tsoha Bootstrapin tapauksessa. Siis näin:

- Kloonataan Gizzan tiedostot GitHubista omalle koneelle komennolla `git clone https://github.com/morgulcore/Pizzapalvelu.git`
- Editoidaan tiedostoa `config/environment.sh`, jos on tarvetta muuttaa web-palvelimen käyttäjätunnusta
- Ajetaan komento `bash bootstrap.sh`. Jos homma ei toimi, avataan projektin juurikansiossa sijaitseva `bootstrap.sh` ja yritetään päätellä, mikä on vialla. Jos ongelma tuntuu liittyvän tietokantaan, vastauksia voisi ehkä hakea tiedostosta `config/database.php`. Sovelluksen on tarkoitus sijaita osoitteessa

`<käyttäjätunnus>.palvelin.fi/<projektihakemisto>`. Tämän vuoksi voi olla tarpeen tutkia ja säätää Apache-palvelun virtual host -asetuksia.

- Kun on editoinut omalla koneella sijaitsevia Gizzan tiedostoja, muutokset saa siirrettyä palvelimelle komennolla `bash deploy.sh && bash create_tables.sh && bash add_test_data.sh`. Jos ei ole editoinut hakemistossa `sql` sijaitsevia tiedostoja, pelkkä `bash deploy.sh` riittää.

Käynnistys- ja käyttöohjeet

Tietokantasovellus Gizza löytyy valmiiksi asennettuna osoitteesta http://xhexhex.users.cs.helsinki.fi/pizzapalvelu/. On periaatteessa kolme tapaa alkaa käyttää (testata) sovellusta: voi kirjautua sisään ylläpitäjänä, tavallisena asiakkaana tai luoda uuden asiakastilin ja kirjautua sisään sillä. Navigaatiopalkista löytyy linkki *Rekisteröidy* uuden asiakastilin luontia varten. Ennen kuin vastikään luodulla asiakastilillä voi tehdä tilauksia, pitää osoitekirjaan lisätä ainakin yksi osoite. Tässä tulee vastaan eräs ajanpuutteesta johtuva epäkohta Gizzassa: vain ylläpitäjä voi lisätä osoitteita asiakkaille.

Kahdella käyttäjätunnuksella on Gizzassa ylläpitäjän oikeudet: ne ovat **admin** ja **skurpitsa**. Molempien salasana on **Tsoh4**. Ylläpitäjä pääsee lisäämään osoitteen napsauttamalla navigaatiopalkista *Osoitteet*, laittamalla raksin Lisäystoiminto-ruutuun, valitsemalla pudotusvalikoista haluttu asiakas ja osoite ja lopuksi napsauttamalla Hae tiedot. Tässä yhteydessä on hyvä tietää se, että jos molemmissa pudotusvalikoissa on tähti ja lisäystoiminto on valittuna, Hae tiedot -painikkeen painaminen saa aikaan sen, että taulujen Asiakas ja Osoite karteesinen tulo lisätään tauluun mm_Asiakas_Osoite. Uusia rivejä lisätään silloin tietokantaan yli sata.

Olemassaolevia asiakastilejä ovat esim. **mruusu** ja **tmansikka**. Tässäkin tapauksessa molempien salasana on **Tsoh4**. Kaikkia asiakastilejä pääsee halutessaan tarkastelemaan myös tietokantayhteystestin avulla taulusta Asiakas.

Yleisesti ottaen Gizzan käytön pitäisi olla varsin helppoa ja intuitiivisesti omaksuttavissa. Asiakkaan esittelysivulla kannattaa varoa, ettei paina Poista-painiketta, jos on kirjautuneena sisään esittelysivua vastaavalla käyttäjätunnuksella. Mitään vakavaa ei tapahdu, jos tekee näin, mutta sovelluksen suoritus päättyy virheilmoitukseen.

Kannattaa selata useiden asiakkaiden esittelysivuja. Esittelysivun sisältö muuttuu jonkin verran asiakkaasta riippuen. Esim. jos asiakkaaseen liittyy ongelmia, esittelysivu tarjoaa mahdollisuuden siirtyä tarkastelemaan niitä. Myös tilausten esittelysivuja kannattaa katsoa useampia.

Järjestelmän yleisrakenne

Käytin työni pohjana kurssilla tarjottua aloituspakettia Tsoha-Bootstrap. Gizzan hakemistorakenne ja tiedostojen nimet ovat edelleen pitkälti samat kuin aloituspaketissa. Olen mahdollisimman tarkasti pyrkinyt toteuttamaan Gizzan MVC-mallin mukaisesti. Kuten aloituspaketissakin, mallit, näkymät ja kontrollerit löytyvät vastaavasti hakemiston `app` alikansioista `models`, `views` ja

`controllers`. Hakemiston `models` tiedostot on nimetty niin, että ne vastaavat tietokannan taulujen (eli järjestelmän tietokohteiden) nimiä – esim. taulua Asiakas vastaa tiedosto `Asiakas.php`. Perusidea on, että kaikki (tai ainakin useimmat) tauluun *X* liittyvät SQL-kyselyt löytyvät tiedostosta `app/models/`*X*`.php`. Olen välttänyt kokonaan sijoittamasta SQL-kyselyitä muualle kuin hakemiston `models` tiedostoihin.

Toisin kuin malleissa, kontrollereissa ei ole Gizzassa tarkkaa yksi yhteen -vastaavuutta tietokannan taulujen kanssa. Vaikka olen nimennyt kontrollerit taulujen mukaan (esim. `asiakas_controller.php`), ei kontrollerin toiminta välttämättä rajoitu yksittäiseen tietokohteeseen liittyviin asioihin. Jos Gizzaa haluaisi kehittää ja laajentaa, voisikin olla hyvä miettiä kontrollereille jokin järkevämpi nimeämiskäytäntö kuin se, mitä olen käyttänyt.

Näkymät löytyvät siis hakemistosta `app/views`. Suurin osa hakemiston HTML-tiedostoista perivät tiedoston `base.html`. Idea tässä on, että näkymän sivujen yhteinen/yhtenäinen rakenne on määritelty yhdessä paikassa. Olennaisin `base.html`:n sisältö lienee sivujen ylälaidasta löytyvä navigointivalikko. Hakemistosta `views` löytyy muutama tiedokohteiden mukaan nimetty alihakemisto. Alikansioiden HTML-tiedostojen nimeämiskäytännöstä on seuraava perusidea:

- `index.html` – Tietokohteen kaikkien rivien/olioiden listaus
- `esittely.html` – Yksittäisen olion esittelysivu (olio voidaan usein myös poistaa sivulta löytyvällä painikkeella)
- `muokkaa.html` – Sivu, jonka avulla tietokohteen oliota voidaan muokata.
- `uusi.html` – Uuden olion luomista varten (esim. uusi tilaus)

Alihakemistot `views/macros` ja `views/include` ovat makroja ja muuta toistuvasti käytettävää HTML-koodia varten. Tärkeitä juurihakemistosta löytyviä hakemistoja ovat `app`:in lisäksi ainakin `config`, `lib` ja `sql`. Viimeksi mainittu sisältää tarvittavat tiedostot tietokannan taulujen ja niiden alustavan sisällön luomiseen. Seuraavassa joitakin muita tärkeitä tiedostoja:

- `config/routes.php` – Jokaiseen sovelluksen polkuun (esim. /asiakas/kirjaudu) liittyy tietyn kontrollerin tietty metodi. Ne on määritelty tässä tiedostossa.
- `lib/base_controller.php` ja `lib/base_model.php` – Kaikkien Gizzan kontrollerien ja mallien yläluokat. Kaikille kontrollereille tai malleille yhteisiksi tarkoitetut funktiot ja attribuutit sijoitetaan näihin tiedostoihin.

Katsotaan vielä Gizzan keskeisiä istuntoihin liittyviä funktioita:

- `asiakas_controller.php`: `sisaankirjautumisen_kasittely()` – Jos todennus onnistuu, asetetaan superglobaalin muuttujan `$_SESSION[ 'user' ]` arvoksi asiakkaan käyttäjätunnus
- `Asiakas.php: todenna( $ktunnus, $salasana )` – Jos parametrit täsmäävät, palauttaa vastaavan Asiakas-olion. Annetun käyttäjätunnuksen ja salasanan oikeellisuuden tarkistus tapahtuu siis tämän funktion avulla.
- `base_controller.php` – Kaikki tämän luokan funktiot ovat olennaisia istuntojen tai käyttöoikeuksien tarkistamisen kannalta

Käyttöliittymä ja järjestelmän komponentit

Gizzassa on käytössä navigaatiopalkki, jonka avulla useimmilta sivuilta pääsee useimmille muille sivuille. Olen jättänyt merkitsemättä kaavioihin navigaatiopalkin luomat yhteydet sivujen välillä, koska tämä tekisi kaavioista tarpeettoman monimutkaisia.

Julkiset sivut

Seuraavat näkymät ovat käytettävissä myös ilman kirjautumista.

Asiakaskirjautumista vaativat sivut

Seuraavan kaavion näkymät ovat tarjolla asiakkaalle eli sellaiselle Gizzan käyttäjälle, jolla ei ole ylläpitäjän oikeuksia.

Huomaa, että tuotteen esittelysivu jäi toteuttamatta. Laitoin sen kuitenkin mukaan kaavioon, koska se on käsitteellisesti tärkeä. Onhan selvää, että asiakas haluaa tarkastella tuotetta sen omalla esittelysivulla ennen kuin tekee päätöksen sen tilaamisesta.

Ylläpitäjän kirjautumista vaativat sivut

Myös seuraavassa kaaviossa esiintyy tuotteen esittelysivu, jota todellisuudessa ei ole olemassa. Kaaviosta voi havaita myös, että ongelmia ei voi muuta kuin tarkastella niiden listaussivulla; niitä ei voi lisätä, poistaa tai muokata. Tämä on luonnollisesti epäkohta ja kehittämisen paikka Gizzassa.

Mitä tuli tehtyä eli mitä jäi tekemättä

Seuraavassa olen luetteloinut muutamia asioita, joita en ehtinyt Gizzaan liittyen tehdä tai toteuttaa. Luetteloa voi myös ajatella TODO-listana.

- Tilauksiin ja asiakkaisiin liittyviä ongelmia pitäisi pystyä lisäämään tietokantaan, ja olemassaolevia ongelmia pitäisi pystyä muokkaamaan ja poistamaan. Tällä hetkellä ongelmat voi vain listata.
- Tuotteille pitäisi saada esittelysivu. Tarkoitus on, että asiakas voisi tilausta tehdessään napsauttaa tuotteen nimeä, jolloin esittelysivu aukeaisi. Parasta olisi ehkä se, jos esittelysivu olisi toteutettuna modal boxina. Toteutusohjeet löysin jo, sääli ettei riittänyt aikaa.
- Asiakkaille pitäisi toteuttaa näkymä osoitekirjan editointiin. Nyt ainoa osoitteisiin liittyvä näkymä toimii vain ylläpitäjän oikeuksilla, ja sekin on hieman omalaatuinen monesta moneen -yhteyden demo. Sillä ei voi lisätä uusia osoitteita tietokantaan.
- Validointia voisi edelleen kehittää. Gizza kyllä jonkinasteisella tarkkuudella tarkistaa tekstikentistä saadut syötteet (esim. timestampit ja sähköpostiosoitteet). Kuitenkin esim. pudotusvalikoiden kautta annnettua dataa ei validoida mitenkään. Tulee mieleen, että tämä voisi

olla ongelmallista tuotantokäyttöön tarkoitetussa järjestelmässä. Joku voisi HTTP:n post-metodilla lähettää Gizzalle kaikenlaista epäilyttävää dataa tarkoituksenaan häiritä tai muuttaa järjestelmän toimintaa.

- Epäkohta Gizzan toteutuksessa on toisteisuus eli copy-paste-koodi. Varsinaisen toiston lisäksi tämä ilmenee siellä täällä liian pitkinä funktioina. Funktioistani tuli liian pitkiä, koska en ehtinyt tehdä niistä lyhyitä! Näkymien koodissa olen käyttänyt muutamia makroja ja includejä. Tämä on varmaankin hyvä alku, mutta parantamisen varaa olisi vielä paljon.

Testaus ja tunnetut bugit

En ole testannut Gizzaa systemaattisesti esim. yksikkötestauksella. Testaus on ollut puhtaasti ”manuaalista”. Olen editoinut koodia KDE:n Kate-editorilla, siirtänyt muutokset usersille ja kokeillut lähinnä Gizzan näkymien avulla, toimiiko kaikki, kuten pitää. Myös `Kint::dump()` -funktiosta on ollut hyötyä bugien etsinnässä. PHP-lähdekoodissani esiintyy melko paljon köyhän miehen asserteja, jotka on merkitty kommentilla `// Bugtrap`. Näistäkin on ollut jonkin verran apua bugien nopeassa löytämisessä.

Gizzassa ei ole yhtään tunnettua bugia. Epäilemättä siinä on kuitenkin huomattava määrä tuntemattomia bugeja.