Pizza-Palvelu

Johdanto:

Tarkoituksena on tehdä pizzerian varausjärjestelmä ja asiakkaiden hallintasovellus. Sovellus toimii pizzerian tarkoituksien mukaisesti varaus, laskutus, asiakkaiden tietokantana. Sovellus mahdollistaa pizzojen tilaamisen verkko-sivuston kautta. Sovellus mahdollistaa tietoverkkojen käytön pizzan tilaamisessa.

Totetutus ja toiminta-ympäristö ovat Linux -pohjaisten serveri-ratkaisun käyttö, ja Apache -palvelin sekä PostgreSQL -tietokanta-ohjelmiston varaan rakennettu sovellus. Sovellus toimii *armv7* -pohjaisessa palvelimessa. Sovellus kehitetään käyttäen normaaleja UNIX -pohjaisia sovelluksia kuten *vi* -editori.

Sovellus toteutetaan PHP -ohjelmointikielessä. Sovellus käyttää ***Twig*** -nimistä template -engineä. Sovellus käyttää **SLIM** -nimistä sovelluskehystä PHP:n päälle. Myös **PDO** -kirjastoja käytetään tiedon hakemista ja varastoimista varten tietokantaan. Sovellus on suunnitteltu alusta asti tietoturvalliseksi, esim. käyttäjien syötteet normalisoidaan aina. Sovellus käyttää Cookie:eita tukeakseen sessioita. Sovellus tehdään mahdollisimman vähän Javascript -kieltä käyttäen. Myöskään mitään Flash, HTML5, tai vastaavia sovelluksia ei käytetä. Sovellus toimii yhtä tietokantaa käyttäen.

Käyttötapaukset:

1. Järjestelmään rekisteröityminen
   Käyttäjät: asiakas
   Tavoite: saada käyttäjätunnus järjestelmään
   Käyttötapauksen kulku:
      Käyttäjä valitsee käyttäjätunnuksen ja sähköposti-osoitteen. Järjestelmä lähettää salasanan sähköpostitse käyttäjälle.

2. Tuotteiden selaaminen
   Käyttäjät: asiakas
   Tavoite: valita haluamansa tuote
   Käyttötapauksen kulku:
      Käyttäjä kirjatuu järjestelmään. Käyttäjälle esitettään listat tuotteista.

3. Tilauksen laatiminen
   Käyttäjät: asiakas
   Tavoite: tilata pizza
   Käyttötapauksen kulku:
      Käyttäjä valitsee haluamansa tuotteet. Sovellus esittää yhteenvedon tilauksesta. Käyttäjä vahvistaa tilauksensa.
   Poikkeuksellinen toimita:
      Käyttäjällä häiriöitä, jonka tapauksessa tilaus voidaan peruttaa.

4. Kuitin tulostus
   Käyttäjät: asiakas
   Tavoite: saada kuitti tilauksesta
   Käyttötapauksen kulku:
      Käyttäjä valitsee kuitin -tulostuksen joka esitetään PDF -tiedostona käyttäjälle.

5. Kuitin tulostus
   Käyttäjät: ylläpito
   Tavoite: kuitti tilauksesta
   Käyttötapauksen kulku:
      Pizza-palvelu tulostaa PDF -dokumentin kuitista toimitettavaksi toimituksen yhtydessä.

6. Ylläpidon järjestelmään kirjatuminen
   Käyttäjät: ylläpito
   Tavoite: Pizza-palvelun ylläpito
   Käyttötapauksen kulku:
      Käyttäjä kirjautuu järjestelmään pääkäyttäjänä etukäteen annetulla salasanalla.

7. Asiakkaiden tietojen selaaminen
   Käyttäjät: ylläpito
   Tavoite: hallinnoida asiakkaiden tietoja
   Käyttötapauksen kulku:
      Käyttäjä kirjatuu järjestelmään. Käyttäjä voi hallinnoida asiakkaiden tietoja.

8. Tilausten selaaminen

Käyttäjät: ylläpito
Tavoite: hallinnoida tilauksia
Käyttötapauksen kulku:
Käyttäjä kirjatuu järjestelmään. Käyttäjä voi hallinnoida aviomia ja suljettuja tilauksia.

9. Tilauslistan tulostus
Käyttäjät: ylläpito
Tavoite: tulostaa tilauksista tietoja
Käyttötapauksen kulku:
Käyttäjä kirjatuu järjestelmään. Käyttäjälle esitetään suljettujen ja avointen tilausten listat PDF -dokumenttina.

10. Aikakohtaisten hintojen lisäys, poisto ja muokkaus
Käyttäjät: ylläpito
Tavoite: ylläpitää aikaan liittyviä hintoja
Käyttötapauksen kulku:
Käyttäjä kirjautuu järjestelmään. Käyttäjä voi luoda, poistaa tai muokata aikakohtaisia tarjouksia.

11. Toimitusten kirjaus
Käyttäjät: ylläpito
Tavoite: lisätä tieto toimituksesta
Käyttötapauksen kulku:
Käyttäjä voi merkitä tilauksen toimitetuksi. Myös häiriö-merkinnät tehdään tässä.

12. Myöhästymisalennuksen kirjaus
Käyttäjät: ylläpito
Tavoite: merkitä kantaan annettu alennus myöhästyneestä toimituksesta
Käyttötapauksen kulku:
Käyttäjä merkitsee järjestelmään myöhästyneen toimituksen. Asiakas voi halutessaan perua tai muuttaa yli tunnin kestänyttä toimitusta.

13. Tarjousten kirjaus
Käyttäjät: ylläpito
Tavoite: merkitä järjestelmään tarjouksia
Käyttötapauksen kulku:
Käyttäjällä on mahdollisuus kirjata uusia tarjouksia, peruttaa tarjouksia tai muokata olemassaolevia.

14. Tilauksen peruminen häiriöiden vuoksi
Käyttäjät: ylläpito
Tavoite: peruuttaa tilaus jos asiakkalla häiriömerkintöjä
Käyttötapauksen kulku:
Käyttäjän häiriömerkinnät näkyvät käyttäjä-listassa sekä tilausvahvistuksessa, ja käyttäjä voi poistaa/peruuttaa tilauksen.

15. Tuottetietojen lisäys, muokkaaminen ja poisto
Käyttäjät: ylläpito
Tavoite: hallinnoida tuottetietoja
Käyttötapauksen kulku:
Käyttäjä voi muokata olemassaolevia tuotetietoja, poistaa niitä ja lisätä uusia.

16. Lisukkeiden lisäys, muokkaaminen ja poisto

Käyttäjät: ylläpito

Tavoite: hallinnoida lisuketietoja

Käyttötapauksen kulku:

Käyttäjä voi muokata olemassaolevia lisuke-tietoja, poistaa niitä ja lisätä uusia.

17. Tuoteryhmien hallinta

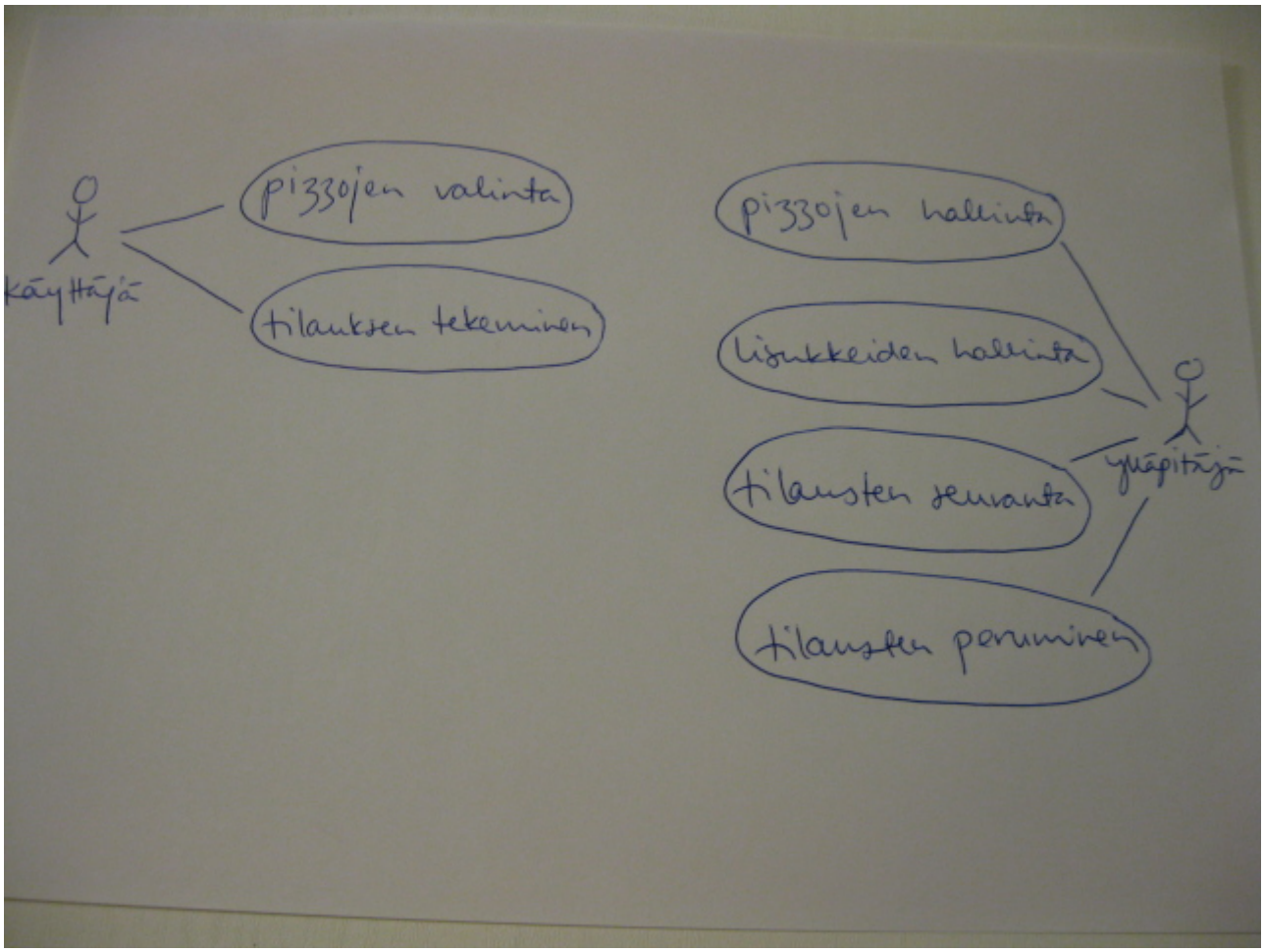
Käyttäjät: ylläpito

Tavoite: hallinnoida tuoteryhmiä

Käyttötapauksen kulku:

Järjestelmässä on eri tuoteryhmiä, esim. ”pizzat” ja ”kebabit”. Sovellus sallii tuotteen lisäyksen tuoteryhmään. Ensimmäisessä versiossa on ainoastaan yksi ”pizzat” -tuoteryhmä.

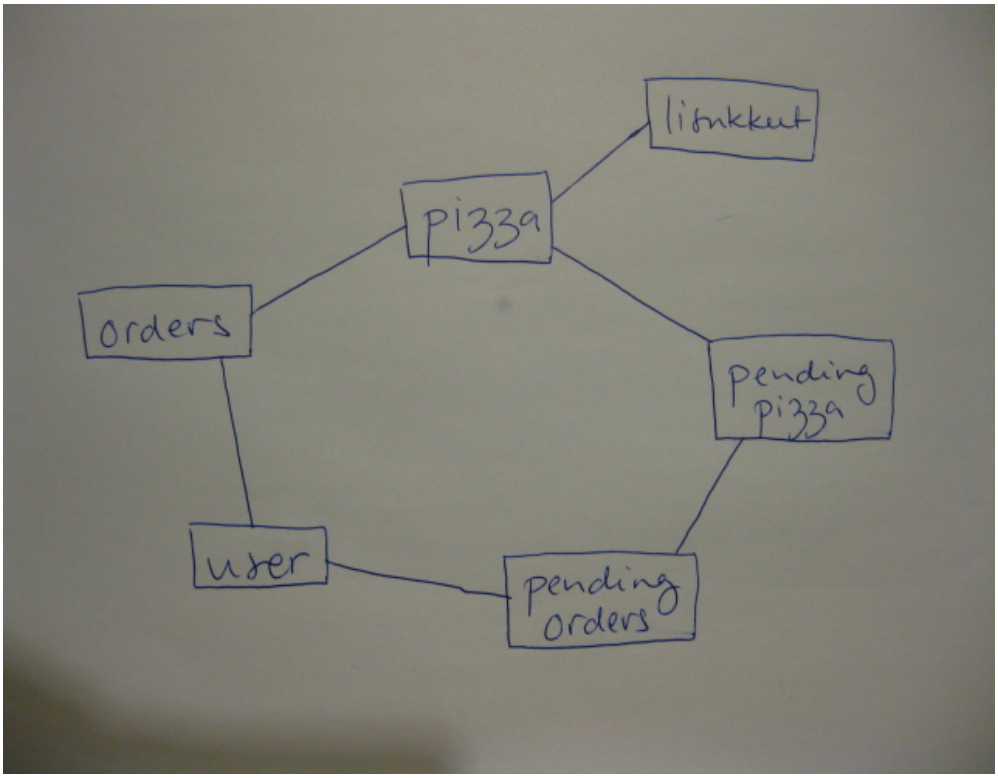
Kayttotapauskaavio:



Kayttajaryhmat:

- Anonyymi kayttaja
  - Anonyymilla tarkoitetaan keta tahansa joka Internet valityksella tulee kayttamaan Pizza-Palvelun tilauspalvelun sivuja.
- Yllapitaja
  - Yllapitajalla tarkotetaan Pizza-Palvelun pizzoja myyvan tahon salasanan takana suojatusta kayttajasta.

Järjestelmän tietosisältö:



Tietokohde: Pizza

Attribuutti	Arvojoukko	Kuvailu
Nimi	Merkkijono, maksimisssaa 255 merkkia pitka	Pizzan nimi

Tietokohde: Lisukkeet

Attribuutti	Arvojoukko	Kuvailu
Hinta	Kokonaisluku, suurempi kuin nolla ja pienempi kuin 6.	Pizzan lisukkeen hinta euroina.
Nimi	Merkkijono, maksimissaan 255 merkkia pitka.	Pizzan lisukkeen nimi.

Tietokohde: Tilaukset

Attribuutti	Arvojoukko	Kuvailu
Pizzan numero	Kokonaisluku, suurempi tai yhta kuin nolla.	Tilausken alla olevan pizzan numero-osoitin
Kayttajan tunnus	PHP session_id() -metoodin paluuarvo annetulle	Satunaismerkkijono joka on uniikki kayttajallle.

	anonyymille kayttajalle.	

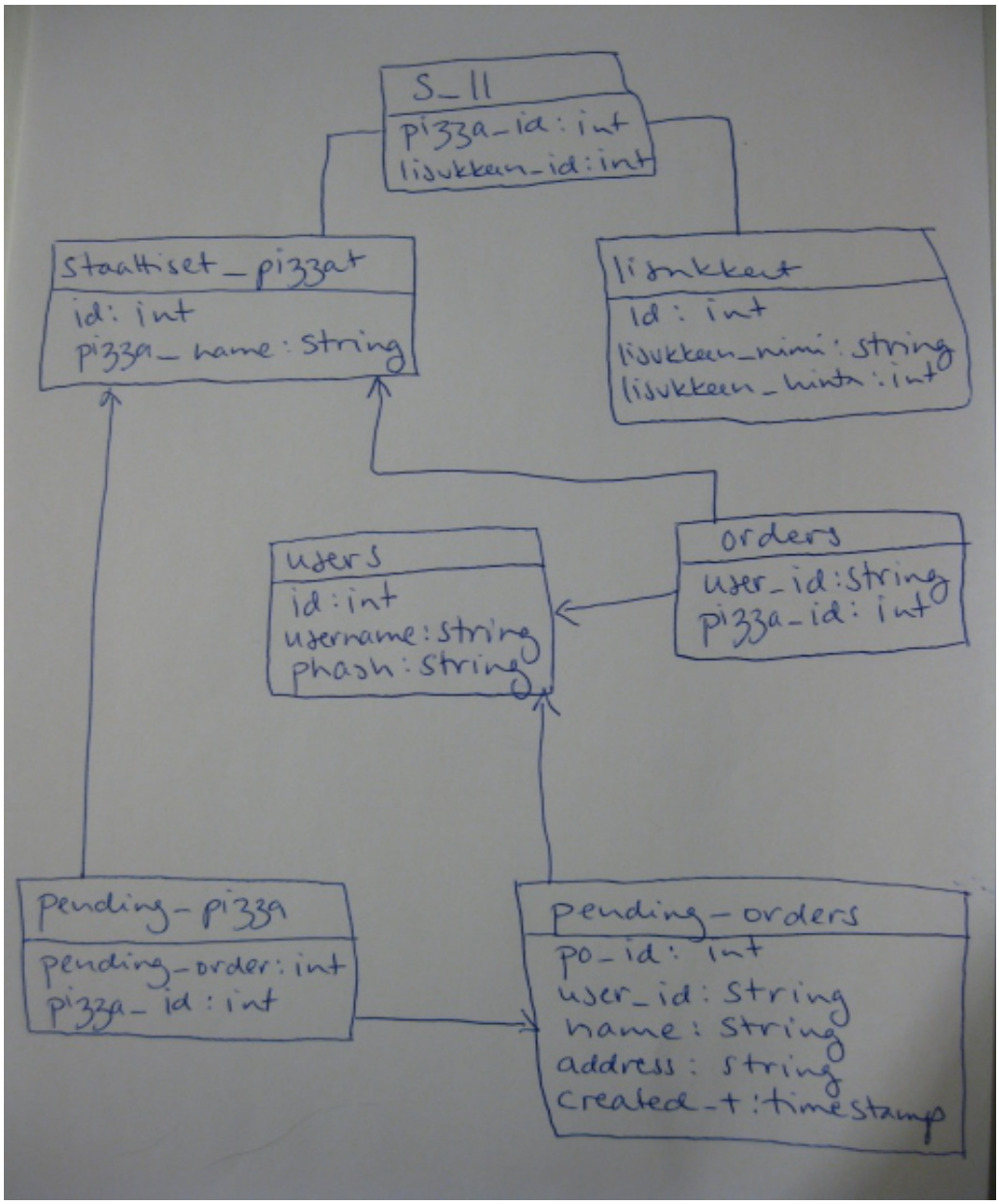
Tietokohde: Kayttaja

Attrbuutti	Arvojoukko	Kuvailu
Kayttajan kayttajatunnus	Merkkijono, maksimissaan 25 -merkkia pitka	Sisaankirjautuvan kayttajan kayttajatunnus. Talla hetkella ainoastaan *admin* -tunnus voi kirjautua sisaan.
Kayttajan salasana	Satunaismerkkijono, maksimissaan 65 -merkkia.	Kayttajan salasana.

Tietokohde: Odottava tilaus

Attribuutti	Arvojoukko	Kuvailu
Kaytttajan tunnus	Satunaismerkkijono maksimissaan 50 -merkkia pitka	Kayttajan session_id() -metoodin paluuarvo.
Kayttajan nimi	Merkkijono, maksimissaan 50-merkkia pitka.	Kayttajan oikea nimi.
Kayttajan osoite	Merkkijono, maksimisssaan 255 -merkkia pitka	Kayttajan osoite toimitusta varten.
Tilausajankohta	TIMESTAMP	Ajankohta jona kayttaja teki tilauksen.

Relaatiotietokantakaavio



Jarjestelman yleisrakenne

Sovellus on MVC -mallin mukaisesti tehty sovellus. Periaatteessa sovellus makaa **Slim** -nimisen sovelluskehyksen takana. Seuraavia asioita hoidetaan sovelluskehtyksen kautta:

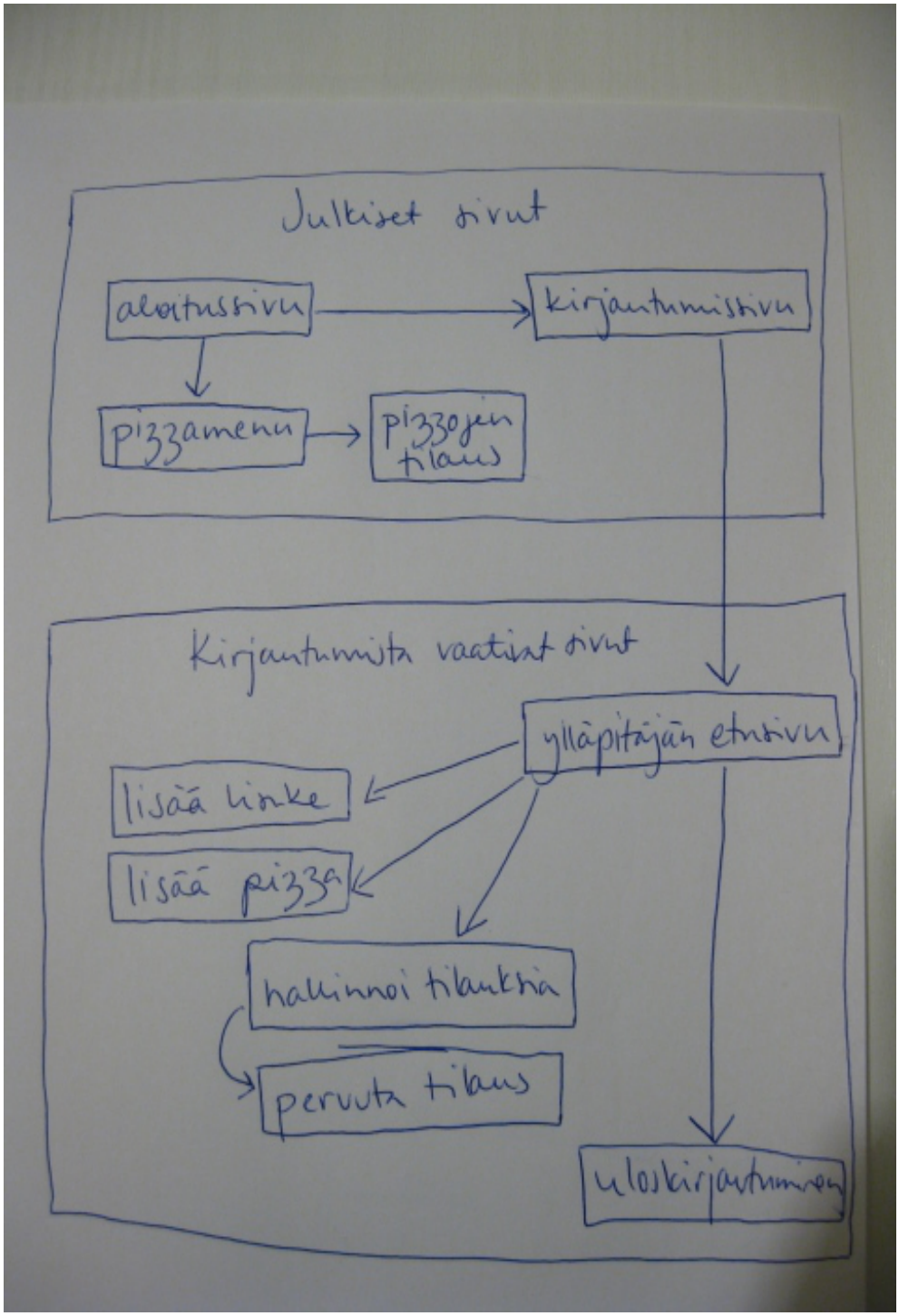
1. Tietokantayhteydet. Toimitetaan automaattisesti *DB::connect()* -metoodin kautta kahva jonka avulla syotetaan kaikki asiat tietokantaa ja tietokannasta. Myos salasana on maaritelty tassa kontekstissa.
2. Template -kieli on **Twig** -nimisen ohjelmiston toteuttama kayttoliittyman rakennus-kehys jossa on PHP:n oliomallia tukeva liittyma. Myos templaten kautta voidaan esittaa ns. *blockeja* -koodia.
3. *Kint* toimii debuggerina ongelmatilanteissa.
4. …

Sovellus toimii mallin kautta haetun tiedon perusteella controllerin kautta template-kieleen. Sovellusksen paa-piirteet ovat app/{models,controllers,views} -hakemistojen sisalta. Sovellus toteuttaa olio-rajapinnan models -luokkien paalle. Tama tarkoittaa sita etta sovellus:

1. Syottaa itsensa tietokannasta hakemien tietojen perusteella. Eli luokka populoi itsensa esim., avaimen perusteella tietokannasta, ja muodotettu olio palautetaan takaisin staattisen metoodin kautta ilmentymaksi.
2. Olio voi tayttaa itsensa annetuista arvoista, ja tyontaa arvot tietokantaan. Talloin voidaan helposti antaa esim., $olio->store() -kutsu, joka syottaa olion konseptuaalisesti kantaan.
3. Olio-malli toteuttaa avaimen kautta poistamisen tietokannasta.
4. Olio-malli toimii rekurssiivisesti, eli esimerkkina *pizza* voi sisaltaa *lisukkeita*, joten Pizza -malli kutsuu Lisuke-mallia ja pyytaa silta listan olioita liitettavaksi Pizza-olioon.

Sovelluksen objektimalli todettiin hyvin selkeasti ylivoimaiseksi ns., *ad hoc* -mallia vastaan verrattuna. Sovellus sisaltaa konseptuaaalisesti **Jarjestelman tietosisalto** -osan kohdalla mainituista olioista koostuvan mallin. Sovelluksen alkuvaiheissa hyodynettiin vain tiedon palauttamista suoraan ilman olio-mallia nakymalle suoritettavaksi, ja malli jatettiin pois laskuista. Ohjelma muodosti nopeaan tahtiin hankalasti hallittavaksi koska jokainen nakyma vaati oman mallinsa. Malli toteutettiin tietosisallon perusteella ja havaitiin ylivoimaikseksi tavaksi hallinnoida tietoa.

Kayttoliittyma ja jarjestelman komponentit



Asennustiedot

Sovellus on tehty toimimaan Linux pohjaisessa palvelimessa. Sovellus toimii:

1. Apache 2.4.10
2. Slim versio 2
3. mod_php5 5.6.26
4. PostgresSQL 9.4.9

Naiden sovellusten varassa toimiva https://github.com/Tsoha/Tsoha-Bootstrap Git -repository pitaa huolen myos sisaisesti etta sovellus toimii. Apachen konfiguraatiossa on tehty moduulien kaynnistys, lahina:

1. mod_php
2. mod_rewrite
3. …

Asennus ei ole kovin hyvin maarittelty sovelluskehitykseen kaytettyjen maarittelmien suhteen. Tarvittavat moduulit asennettuna ja saadettyna oikein, antaa ympariston joka esitetaan *NameVirtualHost* -maarityksen kautta, ja tsoha asennetaan siten etta sovelluksen tiedostot tulevat sellaiseen paikkaan etta ne saadaan kasiksi verkon kautta osoitteesta http://...../tsoha/. Tama johtuu siita etta hirmu useassa paikassa ollaan tehty oletus etta sovellus toimii /tsoha/ -hakemistosta kasin. Yksi tapa tehda tama toiminnallisuus on tehda /var/www/html -hakemistoon symboolinen linkki esim.:

 # cd /var/www/html
 # ln -s /home/$(whoami)/tsoha/deploy tsoha

Nain saadaan kayttoon /tsoha -hakemisto. **mod_rewrite** on tarpeellinen etta voidaan siirtaa pyyntojen kasittely suoraan tsoha/deploy/index.php -scriptin kasiteltavaksi.

Sovellus vaatii PostgreSQL -tietokannalta tukea toimiakseen. Se miten tietokanta luodaan, ja kuinka tehdaan kayttaja siihen on taman dokumentin laajuuden ulkopuolella.

Sovellus vaatii toimiakseen bootstrap.sh -tiedoston ajamisen. Ennen kun ajat boostrap.sh:n muista muuttaa $TSOHA_BASE ja $TSOHA_DEPLOY muuttujia bootstrap.sh -tiedostossa osoittemaan sinun Tsoha-Bootstrap -hakemistoon ja deploy -hakemistoon, esim.:

TSOHA_BASE=$HOME/tsoha/Tsoha-Bootstrap
TSOHA_DEPLOY=$HOME/tsoha/deploy

Editoi myos deploy.sh -tiedoston $TSOHA_BASE aj $TSOHA_DEPLOY muuttujia samanlaisiksi kuin boostrap.sh -tiedostossa. Tassa vaiheessa on hyva idea kokeilla toimiiko **psql** -ohjelma. Aja:

$ psql tietokannan_nimi

Miten tietokanta asetetaan toimivaksi ei ole taman dokumentin laajuudessa. Jos jarjestelma on maarittelty oikein pitaisi sinun saada *tietokannan_nimi=>* kehoite terminaaliin. Aja seuraavaksi:

$ sh create_tables.sh

Se luo sinulle taulut joita sovellus kayttaa. Nyt olet valmis ajamaan bootstrap.sh -tiedoston:

$ sh bootstrap.sh

Ja jos se onnistuu, aja:

$ sh deploy.sh

Ja tata kautta sovelluksen pitaisi olla kaytettavissa.

Kaynnistys- / kayttoohje

Jos seurasit tarkasti **asennustiedot** -kategorian ohjeita, pitaisi sinulla olla katyettava Pizza-palvelu osoitteessa **http://sinunpalvelimen-osoite/tsoha**.

Kehitetty sovellus toimii osoitteessa http://brute.havoc.fi:8800/tsoha ja jarjestelmaa voi testata kayttajatunnuksella ja salasanalla:

Kayttaja 1:
Kayttajatunnus: admin
Salasana: foobar

Testaus, tunnetut bugit, puuttet & jatkokehitysideat

Sovellusta ei ole kategoorisesti testattu. Sovelluksen laajuuden supistuminsen myota myos sovelluksen mahdollliset bugit ovat aika hyvin hallittuja. Sovelluksessa *tilausten seuranta* -sivulla nakyy liian suppeasti pizzat jotka liittyvat tilaukseen. Sovelluksen puutteita ovat tehottomuus. Sovellus tekee liikaa kyselyita hakeakseen esim. Pizza -mallin kanssa Lisukke-mallin olioita yksitellen, eli jos lisukkeita on monta, ja pizzoja on monta tehdaan sovelluksessa hurja maara erillisia SQL -kyselyita. Sovellus ajetaan **armv7** -pohjaisella palvelimella jossa tehottomuus tulee heti selville kun kannassa on muutama pizza ja lisuke.

Jatkokehitys-ideoita on paljon. Toimiva jarjestelma voisi tukea kayttaja-tunnuksia tilaajille. Nain saataisiin esim., toimitus -virheet kirjatttua kantaan, ja olisi helpompaa peruuttaa tilauksia jotka tulevat ongelmallisista osoitteista.

Omat kokemukset

Sovelluksen kattavuus on huikean iso jos sita siirtyisi tekemaan valmiiksi tuoteeksi pizzerioille. Testaus-mielessa sovellus tukee periaatteelisia asioita kuten pizzojen tilauksia ja tilauksten hallinnointia. Oikea sovellus sisaltaa huiman paljon erillaisia ominaisuuksia. Lahdin tekemaan harjoitustyota naivista lahtokohdasta, olettaen etta jarjestelma olisi helppo ja nopea ratkaista. Sovelluksen kehityksen aikana palautui mieleen Ohjelmistotekniikan menetelmat -kurssin opetukset, ja iteroin tiedon suorasta syottamisesta mallista nakymaan, mika osoittautui harvinaisen hankalaksi tavaksi hallinnoida tietoa. Siirtyminen olio-malliin helpotti tilannetta huomattavasti, ja osoittautui etta tietyt toiminnallisuudet sopivat hyvin yhteen selkean mallin kanssa.

Sovelluksen kehityksessa on tietoisesti oltu kayttamatta Javascript -koodia verkko-sivuilla.

p