ELEKTRONINEN PÄIVÄKIRJA PERHETUTKIMUKSESSA JA PERHETYÖSSÄ

eFamilyCoach

Laura Saari

Opinnäytetyö
Toukokuu 2010

Mediatekniikan koulutusohjelma
Teknologiayksikkö, ICT

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

JYVÄSKYLÄN AMMATTIKORKEAKOULU
JAMK UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖN
KUVAILULEHTI

Tekijä(t) SAARI, Laura	Julkaisun laji Opinnäytetyö	Päivämäärä 17.5.2010
	Sivumäärä 55	Julkaisun kieli suomi
	Luottamuksellisuus ( ) saakka	Verkkojulkaisulupa myönnetty (X)

Työn nimi
ELEKTRONINEN PÄIVÄKIRJA PERHETUTKIMUKSESSA JA PERHETYÖSSÄ

Koulutusohjelma
Mediatekniikka

Työn ohjaaja(t)
HUOTARI, Jouni

Toimeksiantaja(t)
JAMK/ Perheiden hyvinvoinnin ja terveyden painoala ja Jy Perhetutkimuskeskus

Tiivistelmä
Työn tavoitteena oli toteuttaa eFamilyCoach-prototyyppi tutkimuskäyttöön. Prototyypin avulla tuli voida Web-käyttöliittymän kautta luoda ja lähettää kyselyitä tekstiviestinä. Opinnäytetyössä suunniteltiin ja toteutettiin prototyypin tietokantaratkaisu sekä Web-käyttöliittymä. Näiden lisäksi prototyypin testaukseen käytettiin ulkopuolisen tahon tarjoamaa SMS gatewayta, jonka kautta viestit saatiin lähetettyä.

Työ aloitettiin suunnittelemalla käsitteiden pohjalta yksinkertainen tietokanta, jonka kehitystä jatkettiin työn edetessä. Tietokannan suunnittelun edetessä aloitettiin testaus yksinkertaisen käyttöliittymän avulla, jotta tietokantaa saatiin laajennettua prototyyppiin sopivaksi. Lopputulos sisälsi tietokannan ja sen päällä toimivan Web-käyttöliittymän, joita voidaan hyödyntää myös mahdollisessa jatkokehityksessä. Web-käyttöliittymästä pyrittiin tekemään mahdollisimman selkeä ja käytettävä, jotta sen hyödyntäminen eFamilyCoach-prototyypin pilotointivaiheessa onnistuisi ongelmitta. Lopputuloksena syntynyttä tietokantaratkaisua voitiin hyödyntää Web-käyttöliittymän ja SMS gatewayn avulla ensimmäisessä tutkimustestissä Toukokuussa 2010.

Työn lopputulosta tullaan hyödyntämään tutkimuksessa myös syksyllä 2010, jonka jälkeen tullaan mahdollisesti siirtymään jatkokehitysvaiheeseen, jonka aikana toteutetaan laajempaan käyttöön tarkoitettu eFamilyCoach-työkalu. Opinnäytetyössä toteutettua ratkaisua voidaan käyttää apuna selvitettäessä, kuinka hyvin Web-käyttöliittymän ja tekstiviestin hyödyntävä menetelmä soveltuu tutkimukseen ja tulosten pohjalta kehittää prototyyppiä haluttuun suuntaa. Syntynyttä prototyyppiä on mahdollista hyödyntää testauksen perusteella myös suunniteltua laajemmassa tutkimuksessa, eli se täyttää hyvin toimeksiantajan asettamat tavoitteet.

Avainsanat (asiasanat)
Tietokanta, Web-käyttöliittymä, SMS gateway

Muut tiedot

Author(s)
SAARI, Laura

Type of publication
Bachelor´s Thesis

Date
17.5.2010

Pages
55

Language
Finnish

Confidential

( ) Until

Permission for web publication
(X)

Title
ELECTRONIC DATA CAPTURE

Degree Programme
Media Engineering

Tutor(s)
HUOTARI, Jouni

Assigned by
JAMK University of Applied Sciences Health and Wellness Centre of Expertise and University of Jyväskylä Family Research Centre

Abstract
The goal of the thesis was to create eFamilyCoach prototype to be utilized in researches. With the help of the prototype researchers should be able to send and receive SMS messages from Web user interface. The execution of database solution and user interface was described in this Bachelor's Thesis. During the execution the prototype was tested with the help of an SMS gateway provided by a second party.

The project was started by planning a simple database solution, which expanded as the project continued. A simple user interface was created for the testing of the database solution. A result of the work included a database and a user interface which can be utilized also in further development. The goal of the user interface was to create a simple and usable tool which can be used in the pilot without problems. The database solution was utilized, with the support of a user interface and an SMS gateway in the first test pilot in May 2010.

The result will be utilized in a research in the autumn 2010, after which the next step may be further development. The goal of later development is to create further the eFamilyCoach tool. The result produced in this Bachelor's Thesis can be used to research how well a tool combining Web user interface and SMSs can be utilized in research and it can be developed with the help of the research results. The prototype developed in this project can be utilized also in larger researches than planned; thus it fulfils well the goals set by the assigners.

Keywords
database, web user interface, SMS gateway

Miscellaneous

SISÄLTÖ

KUVIOT

LYHENTEET

EDC	Electronic Data Capture, elektroninen tiedon tallennus. Tiedon tallennukseen elektronisessa muodossa tarkoitettu sovellus, jonka avulla kerätään tietoa esimerkiksi lääketieteellisissä kokeissa.
FK	Foreign Key, tietokannan taulun viiteavain.
ICT	Information and communication technologies
MD5	Message-Digest-algoritmi, viestitiivistealgoritmi.
MVC	Model-view-controller, ohjelmistoarkkitehtuurityyli, joka erottaa käyttöliittymä- ja sovellustiedostot.
PHP	Hypertext Preprocessor, web-palvelinympäristön dynaamisten web-sivujen luonnissa käytetty alustariippumaton ohjelmointikieli.
PK	Primary Key, tietokannan taulun perusavain.
SQL	Structured Query Language-kyselykieli, jonka avulla voidaan tehdä hakuja, muutoksia ja lisäyksiä relaatiokantaan.
SMS	Short Message Service, matkapuhelinten tekstiviestijärjestelmä.

1 TYÖN LÄHTÖKOHDAT

1.1 Toimeksiantaja

Toimeksiantajana opinnäytetyössä toimii Jyväskylän Ammattikorkeakoulun Perheiden hyvinvoinnin ja terveyden painoala, joka kehittää ja tuotteistaa uusia työmalleja, sosiaalisia innovaatioita ja palvelukonsepteja perheille ja palvelujen tuottajille. Tavoitteena on esimerkiksi auttaa henkilöstöä vastaamaan perheosaamisen uusiin haasteisiin ja tukea perhepalveluiden kehittämistä. Perheiden hyvinvoinnin ja terveyden painoala vahvistaa alueellista perhetyön, -hoitotyön ja -valmennuksen osaamista ja kansainvälistymistä kouluttamalla sekä tutkimus-, kehittämis- ja innovaatiotyöllä. (Perheiden hyvinvointi ja terveys 2010.)

1.2 Taustat ja lähtökohdat

Projekti esiteltiin opinnäytetyön ohjaaja Jouni Huotarille ja tekijä Laura Saarelle ensimmäisen kerran yhteistyöpalaverissa, jossa kerrottiin tavoitteista sekä tarpeesta. Projekti oli esitelty opiskelijoille mahdollisena toimeksiantona jo aiemmin Tietokantojen Suunnittelu- opintojaksolla, joten tavoitteena oli tarjota aluksi ainakin mahdollista tietokantaratkaisua toteutettavaksi opintojakson resurssien puitteissa. Ensimmäisessä palaverissa nousi kuitenkin esiin, että aiheesta saisi hyvän opinnäytetyöaiheen, mikä herätti kiinnostuksen asian tutkimiseen ja sitä kautta ratkaisuun toteuttaa aiheesta opinnäytetyö.

Ensimmäisessä yhteistyöpalaverissa esiteltiin prototyyppiin toivottavia ominaisuuksia sekä työtehtävien jako toteuttajien kesken. Aluksi tarkoituksena oli jakaa toteutus kahteen tai kolmeen osaan; mobiiliratkaisuun, tietokantaratkaisuun ja web-pohjaiseen käyttöliittymään. Ideana oli, että web-käyttöliittymä ja tietokantaratkaisu voitaisiin sisällyttää osaksi opiskelijaprojektin toteutusta. Mobiiliratkaisuun toteuttajaksi suunniteltiin jotakin Jyväskylän alueella toimivaa alan yritystä. Lisäksi projektiin oli alustavasti pyydetty mukaan käytettävyysasiantuntijaa, joka voisi ottaa kantaa prototyypin käyttöliittymän käytettävyyteen.

Projektin edetessä työnjakoa tarkennettiin ja mobiilialan asiantuntijoiden tapaamisen jälkeen päätettiin mobiiliratkaisu toteuttaa käyttäen vain erillistä tekstiviestipalvelua, kun aiemmin oli pidetty mahdollisena kokonaan erillistä ohjelmaa. Tämä ratkaisu mahdollisti prototyypin lähes koko toteutuksen siirtämisen opiskelijoille, tietokanta-asiantuntijan ja käytettävyysasiantuntijan toimiessa neuvoa-antavina osapuolina. Tietokantaratkaisu toteutettiin osana opintojaksoa, joten sen yhtenä tuloksena oli dokumentointi, joka käsittelee toteutuksen vaiheita ja lopputulosta. Projektin ollessa esimerkki Jyväskylän ammattikorkeakoulun ja yliopiston yhteistyöprojektista sekä ensimmäinen prototyyppi useamman vuoden kestävälle projektille, nousee projektin dokumentaatio tärkeäksi ja sitä kautta hyväksi lähtökohdaksi tälle opinnäytetyölle.

1.3 Tavoitteet ja tehtävät

Työn tavoitteena oli toteuttaa prototyyppi perhetutkimuksen ja perhetyön käyttöön tarkoitettuun elektroniseen päiväkirjaan. Elektronista päiväkirjaa on tarkoitus hyödyntää sekä tutkimusmenetelmänä että uudenlaisen arjen ohjauksen välineenä osana perhetyötä, -neuvontaa ja -terapiaa. Valmis prototyyppi on ohjelma, jonka välityksellä voidaan valita ja lähettää asiakkaan matkapuhelimeen kysymyksiä tekstiviestiksi esimerkiksi päivittäin ja tallentaa vastaukset tietokantaan. Käytännössä asiakas saa päivittäin kysymyksiä tekstiviestillä ja vastaa niihin esimerkiksi numeroin tai kirjoittamalla avoimen vastauksen. Kysymykset tallennetaan tietokantaan, josta työntekijä tai tutkija saa tiedon haluamassaan muodossa ja voi hyödyntää sitä työssään.

Tarve elektroniseen päiväkirjaan on syntynyt asiakasjonojen pidentyessä, sillä nykyiset resurssit eivät ole riittävät suhteessa asiakasmääriin. Elektronisen päiväkirjan avulla on mahdollista aktivoida ja seurata asiakkaan toimintaa päivittäin. Samalla myös osa vastuusta saadaan siirrettyä asiakkaille, jolloin asiakaskäyntien määrää voidaan vähentää ja sitä kautta muokata elektronisesta päiväkirjasta systemaattinen ja reaaliaikainen väline ohjauksessa ja palautteessa.

Elektronisen päiväkirjan kehittelytyö on aloitettu vuonna 2003 ottamalla pilotointi perhetutkimuksen metodiksi. Projektia on jatkettu hyödyntämällä ratkaisua aineistonkeruussa, jonka jälkeen aiheesta on kirjoitettu kirja ja metodiartikkeli.

2010 on siirrytty vaiheeseen, jossa prototyyppi on tarkoitus toteuttaa. Projekti jatkuu prototyypin valmistumisen jälkeen pilotoinnilla ja jatkokehittelyllä. Projektin nimeksi valittiin eFamily Coach ja jatkokehitystyö tulee sisältämään useamman eri työkalun, jotka on suunnattu eri kohderyhmille. Erot eri työkalujen välillä ovat lähinnä mukana tulevissa kysymyspatteristoissa, jotka on suunnattu esimerkiksi perhe- tai parisuhdetyöhön.

Prototyyppiprojektin tehtävät voitiin jakaa toteuttajien ja syntyvien tulosten mukaan ryhmiin. Prototyypin tietokantaratkaisun dokumentointeineen toteutti opiskelijaryhmä osana erillistä opintojaksoa.. Opinnäytetyö sisältää vaihtoehtoisten ratkaisujen vertailun, tietoturvaan perehtymisen ja varsinaisen prototyypin vaiheiden dokumentoinnin lisäksi Web-ratkaisun kuvauksen ja toteutuksen. Lisäksi yhtenä osana prototyyppiä voidaan pitää käytettävyyden arviointia ja kehitystä. Käytettävyyden osa-alueilla suunnittelun ja toteutuksen apuna käytettiin käytettävyysasiantuntijaa. Tuloksena oli Web-käyttöliittymän, tietokantaratkaisun ja tekstiviestipalvelun yhdistävä eFamily Coach- tuotteen prototyyppi, jonka käytettävyydessä ja tietoturvassa otetaan huomioon sovelluksen luonne osana perhetyötä ja -tutkimusta.

Varsinaista prototyyppiä tullaan hyödyntämään osana tutkimusta syksyllä 2010. Tutkimuksessa on tarkoituksena tutkia noin 60–90 lasta noin viikon ajan, minkä jälkeen tiedot kerätään yhteen ja käsitellään tutkijoiden hyödyntämillä tekniikoilla. Tutkimuksessa käytetään noin 10–12 kysymystä päivässä ja tutkimusjakso jaetaan luultavasti ryhmistä riippuen noin kolmeen osaan.

2 VERTAILUKOHTIA JA VAIHTOEHTOISIA TOTEUTUSTYÖKALUJA

2.1 Elektronisen päiväkirjan tekniikoita hyödyntävät aiemmat projektit

eFamily Coach - prototyypin toteutukseen käytettäviä tekniikoita on hyödynnetty aiemmin muissa projekteissa, joiden tavoitteena on ollut esimerkiksi kerätä tietoa mobililaitteen välityksellä. Projektit eivät ole täysin vastaavia, mutta niiden

tavoitteena on ollut tai on edelleen kerätä tietokantaan tietoa, jota on mahdollista analysoida esimerkiksi osana tutkimusta tai opetusta.

2.1.1 Etaitava

Etaitava on mobiilisovelluksen ja verkkopalvelun yhdistävä työkalu, jota hyödynnetään osana opiskelijoiden työssäoppimista. Järjestelmän idea on hyvin samankaltainen kuin elektronisessa päiväkirjassa: työssäoppija vastaa ennalta tehtyihin kysymyksiin tekstiviestillä tai Web-käyttöliittymän avulla ja vastaukset tallennetaan tietokantaan, josta opettajat voivat analysoida vastauksia monipuolisten kyselyiden avulla. Mobiilisovelluksen avulla voi myös esimerkiksi lähettää kuvia ja videoita opettajalle. Etaitava on kehitetty muokkaamaan työssäoppimisen ohjaamisesta säännöllistä ja tehokasta. (Etaitava 2010.)

Sovelluksen vastaanotto on ollut positiivista sekä opiskelijoiden että opettajien näkökulmasta. Opiskelijat ovat esimerkiksi kokeneet, että oppimisesta ollaan kiinnostuneita ja tiedot saadaan päivitettyä säännöllisesti. (Lehto 2007.)

eFamily Coachin ja Etaitavan erona on mobiilipuolen ratkaisu. Etaitavassa on käytössä erillinen puhelimeen asennettu sovellus, kun taas elektronisen päiväkirjan mobiiliratkaisuna käytetään vain tekstiviestipalvelua. Prototyypin toteuttaminen käyttäen erillistä asennettavaa mobiilisovellusta nähtiin liian raskaana ja monimutkaisena ratkaisuna. Päätökseen vaikuttivat esimerkiksi hyvin monet erilaiset mobiililaitteet, jotka vaatisivat useita eri versioita toteutettavasta mobiilisovelluksesta.

2.1.2 CERTAS

CERTAS on elektroninen tiedontallennustyökalu (EDC), jota hyödynnetään erityisesti lääketieteellisissä kokeissa ja tutkimuksissa. CERTAS sisältää Web-pohjaisen käyttöliittymän, josta voidaan valita kyselyitä tai yksittäisiä kysymyksiä ja lähettää niitä käyttäjien päätelaitteeseen kuten esimerkiksi matkapuhelimeen. Ratkaisussa voi luoda erilaisia kyselyitä, joiden on mahdollista reagoida käyttäjän vastaukseen uudella kysymyksellä. Samantyyppistä ominaisuutta, jossa

käyttäjän vastauksiin on olemassa ns. älykkäitä vastauksia, on suunniteltu hyödynnettävän myös elektronisessa päiväkirjassa. (CERTAS 2007.)

CERTAS ratkaisua voidaan verrata joiltakin osin eFamily Coachiin ainakin kohderyhmiltään ja ominaisuuksiltaan. Kokonaisuutena ratkaisu on verrattaessa prototyypin vastaaviin ratkaisuihin melko laaja, mikä johtuu useista eri käyttökohteista. Elektronista päiväkirjaa hyödynnetään ainakin alkuvaiheessa vain vastauksien saamisessa yksinkertaisessa muodossa, mikä vähentää esimerkiksi mobiiliratkaisun valintaa huomattavasti. Jos prototyyppiin olisi valittu toteutettavaksi erillinen mobiilisovellus, olisi toteutus luultavasti ollut lähempänä CERTAS kokonaisuutta. Lähteenä käytetyn tiedon ollessa ainakin osittain vanhentunutta, käytettiin tässä vaiheessa kyseessä olevaa esimerkkiratkaisua vain vertailukohteena terveydenhuollon alalta.

2.2 Prototyypin tekniikoita hyödyntävät avoimen lähdekoodin tuotteet

2.2.1 Yleistä

Yhtenä vaihtoehtona prototyypin toteuttamiseen voidaan pitää myös avoimen lähdekoodin (Open Source) ratkaisua. Prototyypin toteutuksessa olisi mahdollista hyödyntää esimerkiksi Web-sovelluksen ja mobiilisovelluksen yhdistävää ratkaisua, jolla olisi mahdollista luoda ja lähettää kyselyitä. Lisäksi vastaukset tulisi saada jossakin muodossa tallennettua tietokantaan. Tietokantaratkaisun ollessa erillinen opiskelijaprojekti tarvittavan ratkaisun ei tarvitsisi sisältää erillistä tietokantaosiota, vaan ainoastaan mahdollisuuden yhdistää ratkaisu erilliseen tietokantaan.

Mahdollisten ratkaisujen kartoitus ja valinta sisälsivät kaksi vaihetta. Ensimmäisessä vaiheessa kartoitettiin vapaan lähdekoodin sovelluksia, jotka joko suoraan tai jonkinlaisen muokkauksen kautta mahdollistaisivat tiedon keräämisen ja tallentamisen prototyypin edellyttämillä tekniikoilla. Tämän jälkeen lupaavimpia vaihtoehtoja tutkittiin löytyneiden tietojen ja mahdollisen demon avulla. Osittain projektin vaatimia ominaisuuksia sisältäviä ratkaisuja löytyi useita, vaikkakin suurin osa niistä oli liian monipuolisia tai eivät muulla tavalla täyttäneet vaati-

muksia. Projektin alkuvaiheessa valittiin toteutusmenetelmäksi maksullinen tekstiviestipalvelu, joten avoimen lähdekoodin sovellusten testaaminen jäi lähinnä vertailukohteeksi dokumentaatioon.

2.2.2 OpenXdata

OpenXdata on avoimen lähdekoodin ratkaisu, jolla on mahdollista luoda erityyppisiä kyselyitä ja hyödyntää niitä esimerkiksi terveydenhuollossa tai koulutuksessa. Sen käyttöön ei vaadita uusimpia puhelimia, joten sitä voidaan hyödyntää suhteellisen laajan käyttäjäkunnan kanssa. Mobiililaitteiden hyödyntäminen vaatii kuitenkin erillisen mobiiliohjelman, mikä asettaa rajoituksia mobiililaitteiden käyttöön. Tutkimuksista saatavaa tietoa pystytään käsittelemään usealla tavalla ja tutkimusten suunnittelu on tehty melko helpoksi openXdata-Manager ominaisuuden avulla. Tässä ratkaisussa kysymyksiä ja kyselyitä luodaan Web-käyttöliittymän kautta ja esimerkiksi kysymystyyppejä on useita. (OpenXdata 2009.)

OpenXdata-ratkaisun testaaminen tapahtui demon, dokumentaation ja esimerkkiasennuksen avulla. Asennus ja käyttöönotto olivat melko helppoja, eivätkä vaatineet paljoa osaamista. Demojen ja dokumentaation perusteella kyselyiden luomisen ei pitäisi olla monimutkaista ja sen tulisi onnistua ilman minkäänlaista kokemusta esimerkiksi ohjelmoinnista. Dokumentaatio oli melko selkeää ja vastauksia ainakin yleisimpiin kysymyksiin löytyi helposti. Testausta mobiililaitteessa ei tässä tapauksessa tehty. OpenXdata soveltuisi ainakin osittain prototyypin toteuttamiseen, mutta valinta vaatisi esimerkiksi hyvää arviota siitä, kuinka hyvin ratkaisu toimii Suomen matkapuhelinverkoissa. Lisäksi osalle käyttäjistä joutuisi kehittämään jonkin erilaisen ratkaisun, koska mobiilisovellus ei toimi kaikissa laitteissa.

2.2.3 FrontlineSMS

FrontlineSMS on hieman erityyppinen ratkaisu, kuin mitä prototyyppiin vaaditaan, mutta se on hyvin dokumentoitu avoimen lähdekoodin sovellus, joka sisältää samoja ominaisuuksia. Ratkaisu ei myöskään vaadi uusimpia mobiililaitteita, vaan sen kerrotaan toimivan kaikissa malleissa. Ratkaisu on helppoa asentaa ja

se skaalautuu hyvin suurillekin määrille käyttäjiä, mikä on yksi prototyypin vaatimuksista. FrontlineSMS on ratkaisu, jota on mahdollista hyödyntää monissa erityyppisissä ratkaisuissa ja sen käyttö on täysin sijainnista riippumatonta. Ratkaisu on ollut käytössä useissa suuria henkilömääriä sisältävissä projekteissa, joten sen voidaan olettaa toimivan riippumatta käyttäjämääristä. (FrontlineSMS 2010.)

FrontlineSMS eroaa toteutukseen valituista ratkaisuista toiminnaltaan, koska se ei varsinaisesti ole Web-käyttöliittymää hyödyntävä tutkimusväline. Ratkaisu on kuitenkin hyvä esimerkki siitä, että käytettävä sovellus toimii riippumatta käyttäjien laitteista tai sijainnista. Sovelluksen ollessa käytössä hyvinkin erityyppisissä ja erilaisille käyttäjille suunnatuissa toiminnoissa, voi sen olettaa olevan käytettävyydeltään ja luotettavuudeltaan hyvä ratkaisu. FrontlineSMS-ratkaisun arviointi tapahtui dokumentaation ja esimerkkidemojen avulla.

2.2.4 OpenClinica

OpenClinica on avoimen lähdekoodin Web-pohjainen ratkaisu, joka on tarkoitettu etupäässä lääketieteellisiin tutkimuksiin. Tuote mahdollistaa elektronisen tiedon keräämisen ja käsittelyn. Tuote on ilmainen ja laajennettavissa halutulla tavalla ja se soveltuu hyvin tiedon siirtämiseen tietokannan kautta haluttuun muotoon. Ratkaisussa on käytetty uusimpia tekniikoita ja sitä on hyödynnetty melko laajalti erilaisissa tutkimuksissa. Pääasiassa se on kuitenkin tarkoitettu keräämään tietoa potilaista osana kliinisiä kokeita. (OpenClinica 2010.)

OpenClinica-ratkaisun testaaminen ei onnistunut yhtä hyvin kuin muissa ratkaisuissa, joten varsinaiset kokemukset esimerkiksi asennuksesta jäivät puuttumaan. Tuote ei kuitenkaan vastaa prototyypin tarvitsemia ratkaisuja kuin osittain, joten sen testaaminen ei ollut projektin kannalta välttämätöntä.

3 PROTOTYYPPIIN VALITUT RATKAISUT

3.1 Palvelin ja muut hardware-ratkaisut

3.1.1 Yleistä

Varsinaista prototyyppiä tullaan hyödyntämään tutkimuksessa syksyllä 2010, jolloin käyttäjäryhmä on rajattu tutkijoihin ja noin 60–90 lapseen. Tällöin ratkaisu saadaan tarvittaessa toimimaan esimerkiksi vain yhden koneen avulla, jolloin kone toimisi samalla lähettävänä mobiililaitteena ja Web-käyttöliittymän hallinnoijana. Alusta asti on kuitenkin otettava huomioon, että kehitystä jatketaan ja lopputulosta tulee hyödyntämään moninkertainen määrä käyttäjiä. Lisäksi on huomattava, että tutkimusten ja tietojen luonne on salassapitomääräysten sitomaa, mikä nostaa vaaditun tietoturvan määrää.

Palvelimena tullaan perhetyössä käyttämään oletettavasti käyttäjäosapuolen valitsemaa ratkaisua, jota on esimerkiksi käytetty myös muiden suojattujen tietojen tallentamisessa. Prototyyppiä suunniteltaessa ja toteutettaessa otetaan huomioon laajennettavuus ja tehdään mahdollisesti ainakin suunnitelma siitä, kuinka esimerkiksi tietokanta ja lähetyspalvelin tulisi sijoittaa, jotta saavutetaan mahdollisimman hyvä tietoturva.

3.1.2 Toteutusympäristönä Labranet

Toteutusympäristönä projektissa käytettiin Jyväskylän ammattikorkeakoulun Labranet- verkkoa. Labranet on itsenäinen osa Jyväskylän ammattikorkeakoulun verkkoa ja sitä hyödynnetään ICT-tulosalueen opinnoissa, kuten esimerkiksi opintojaksojen harjoitustöissä. Labranet tarjosi prototyypissä hyödynnettävän SMS gateway – palvelimen sekä Web-käyttöliittymän palvelimen efamilycoach.labranet.jamk.fi.

3.2 Tietokantaratkaisu

3.2.1 Vaatimukset

Tietokantaratkaisu toteutettiin osaksi prototyyppiä, mutta samaa ratkaisua tullaan hyödyntämään myös versiossa, joka tulee käyttöön perhetyössä. Tietokantaratkaisussa oli siis alusta asti otettava huomioon skaalautuvuus. Ensimmäisessä versiossa, jolla elektronista päiväkirjaa pilotoidaan, kerätään tietoa vain noin 60–90 käyttäjältä viikon ajan, joten tietokantaan ei tule suuria määriä tietoa. Kehityksen jatkuessa käyttäjämäärät kasvavat ja sovellusta tullaan laajentamaan myös ominaisuuksiltaan, joten tietokantaa on voitava laajentaa helposti.

3.2.2 Tietokantaratkaisun toteuttajat

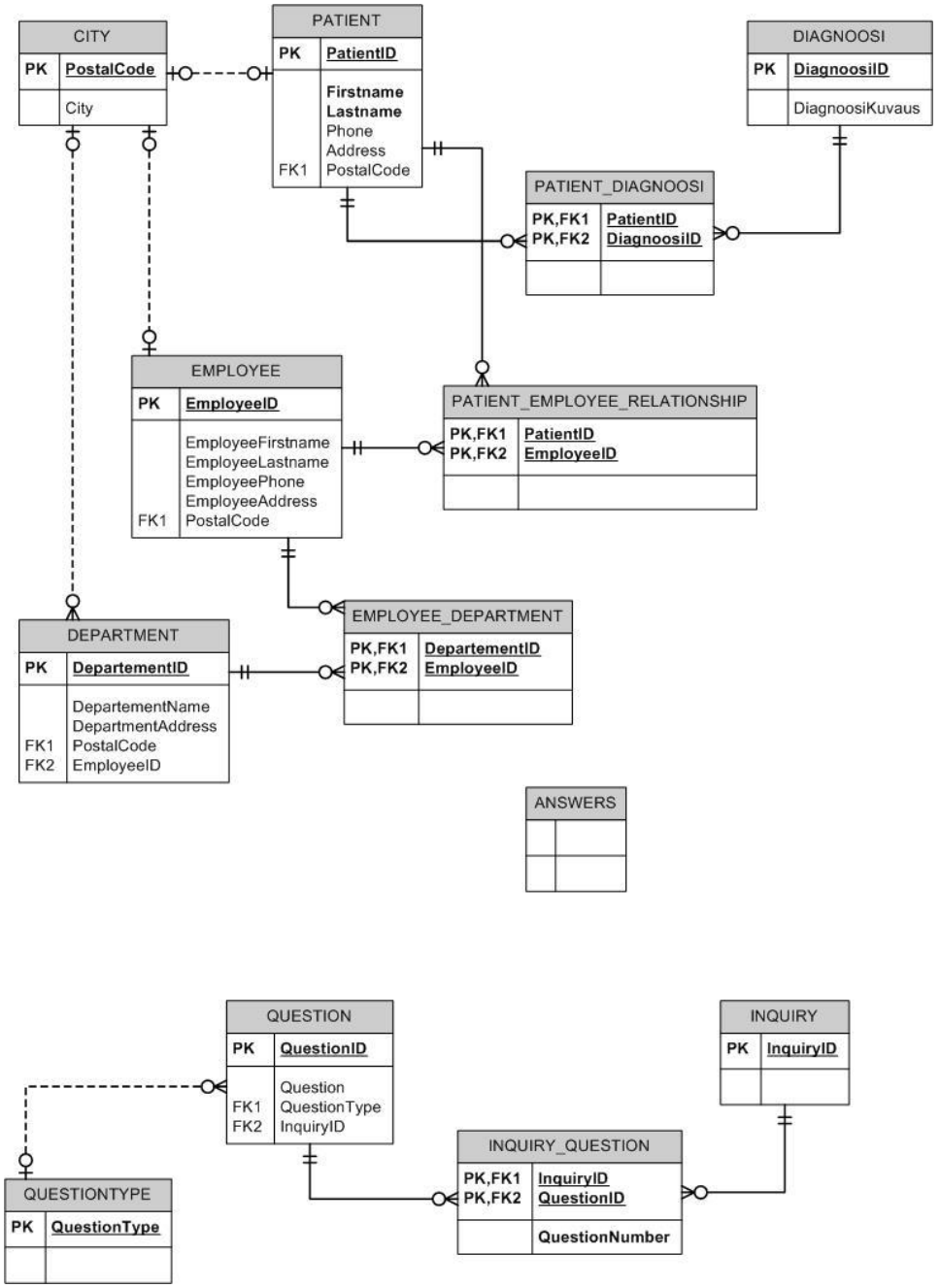
Tietokantaratkaisu toteutettiin harjoitustyönä Jyväskylän ammattikorkeakoulun ICT-yksikön tietokantojen suunnittelua käsittelevällä opistojaksolla. Työn toteutukseen osallistuivat opiskelijat Heikki Hyvönen, Leo-Matti Lehtonen ja Laura Saari. Tietokantaratkaisun toteutukseen käytettiin 80 tuntia ryhmän jäsentä kohti, eli yhteensä noin 240 tuntia. Toteutukseen sisältyi tietokannan suunnittelu ensimmäisestä asiakastapaamisesta ja vaatimusmäärittelystä normalisoituun tietokantasuunnitelmaan, josta voitiin generoida sql-komentosarja tietokannan luomista ja käyttöliittymän toteutusta varten.

3.2.3 Projektin vaiheet ja käytetyt työkalut

3.2.3.1 Suunnitteluvaihe ja tietokannan ensimmäiset versiot

Tietokantaratkaisun suunnittelu alkoi vaatimusmäärittelyn toteutuksena, jonka jälkeen tehtiin ensimmäinen käsitemalli käyttäen Microsoft Visio- työkalua. Ensimmäisessä käsitemallissa listattiin aluksi ainoastaan käsitteet asiakas, työntekijä ja tutkija, sekä yksinkertaisimmat muut käsitteet, kuten kysymykset ja vastaukset. (Ks. kuvio 1.) Version pohjalta toteutettiin myös käyttötapauskaavio ja yksinkertaistettu malli tietokannasta, joita voitiin hyödyntää esimerkiksi osana yhteistyökokousten esittelyjä. Tässä vaiheessa projektia eri käsitteet haluttiin erotella mahdollisimman selkeästi toisistaan, joten käsitettä asiakas ei vielä tässä vaiheessa käytetty, jotta varsinaisen prototyypin tilannutta asiakasta ei sekoi-

tettaisi työkalun käyttäjiä tarkoittavaan käsitteeseen. Myöhemmin siirryttiin käyttämään termiä client. Lisäksi käytössä on vielä termi diagnoosi, josta seuraavassa vaiheessa luovuttiin.



KUVIO 1. Ensimmäinen käsitemalli, toteutettu käyttäen Microsoft Visio 2007- ohjelmaa

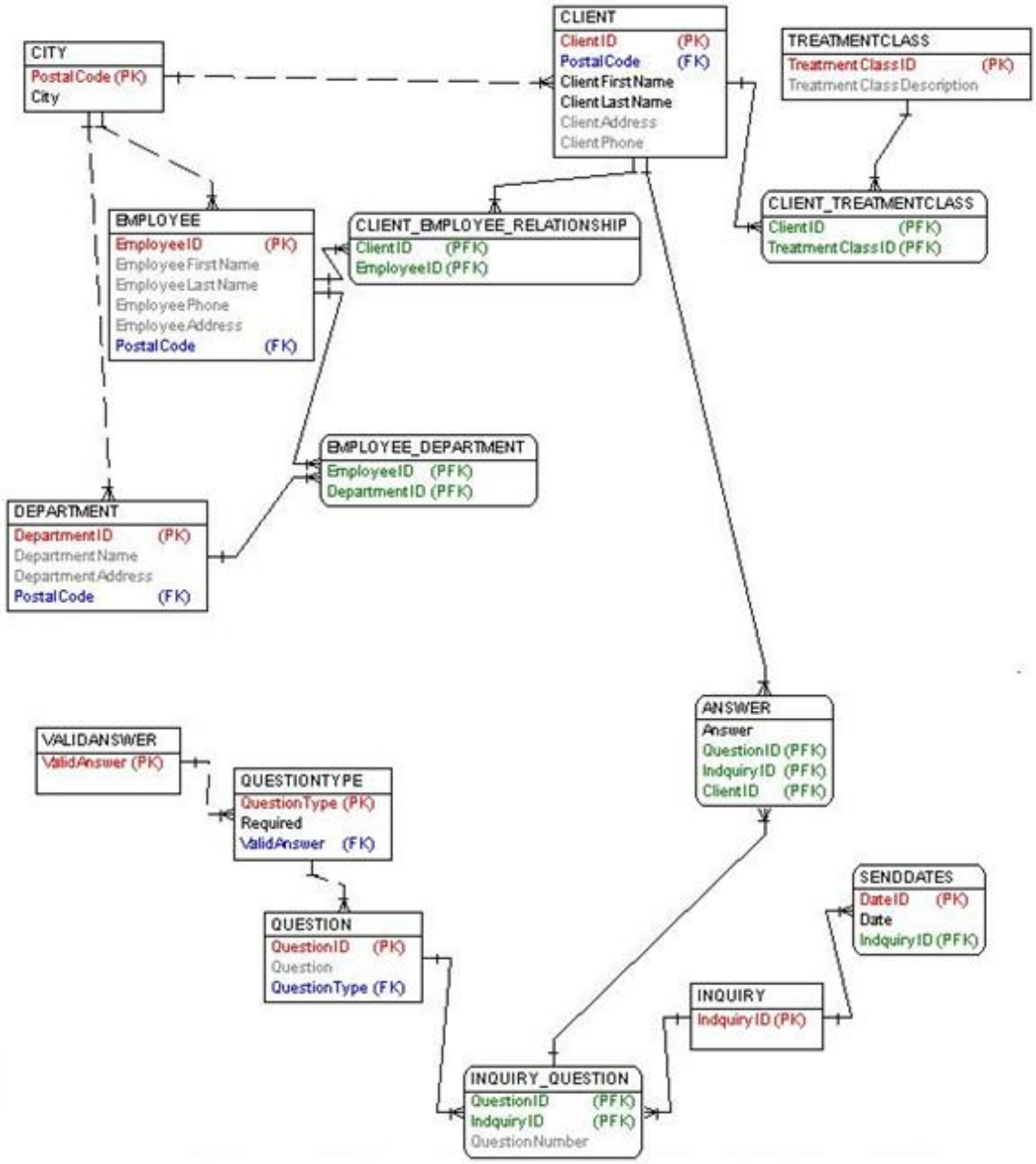
Tämän jälkeen siirryttiin purkamaan käsitteitä ja yhteyksiä niin, että saatiin ensimmäinen versio prototyyppiä varten. Tämä versio toteutettiin käyttäen Toad

Data Modeller-ohjelmaa ja siihen pyrittiin lisäämään selkeästi kaikki projektin vaiheen kannalta oleelliset käsitteet ja yhteydet. (Ks. kuvio 2.) Tässä vaiheessa projektia pyrittiin lisäämään käsitteitä niin, että version avulla olisi mahdollista luoda ensimmäinen versio tietokannasta ja sitä kautta prototyyppi käyttöliittymästä. Versioon lisättiin yhteydet esimerkiksi asiakkaan ja vastauksen välille ja esimerkiksi kysymyksiin liittyviä käsitteitä lisättiin ja laajennettiin. Tässä vaiheessa päätettiin jakaa kysymykset tyyppien mukaan kolmeen eri ryhmään: varsinaisiin kysymyksiin, ohjeita antaviin tiedotteisiin sekä muistutusviesteihin, jotka käyttäjä saa, mikäli ei vastaa kysymykseen tietyn ajan kuluessa.

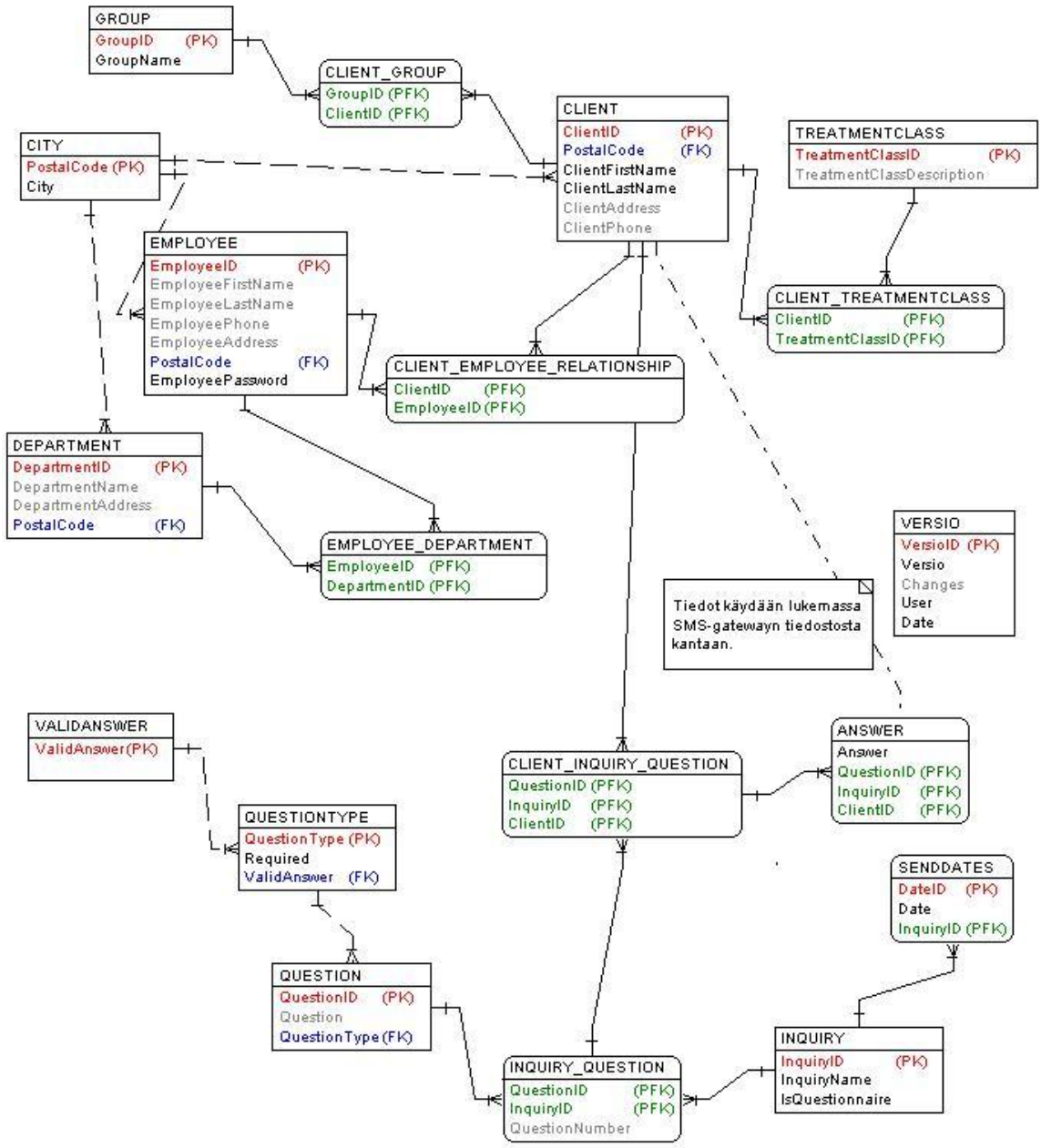
Ensimmäisen version avulla luotiin tietokanta ja sille yksinkertainen käyttöliittymä käyttäen Microsoft Office Access 2007-ohjelmaa. Ensimmäistä prototyyppiä käytettiin lähinnä kannan testaamiseen, koska varsinainen käyttöliittymä toteutettiin käyttäen muita työkaluja.

Tietokannan taulut client (asiakas) ja inquiry question (kyselyn kysymys) yhdistettiin käyttäen answer (vastaus)-taulua, joka aiemmassa suunnitelmassa oli vielä erillinen taulu (Ks. kuvio 1.). Asiakkaan lähettämän vastauksen katsottiin siis yhdistävän tietyn asiakkaan tiettyyn kysymykseen niin, että kysymykset ja vastaukset saataisiin luokiteltua oikealle henkilölle. Ongelmana koettiin kuitenkin oikean vastauksen yhdistäminen oikeaan kysymykseen, koska data- ja asiakasmäärien kasvaessa ja vastausten ollessa osassa kysymyksiä samoja, on erittäin tärkeää, että kysymys-vastaus-parit yhdistetään oikein. Ongelman ratkaisemiseksi suunnittelua jatkettiin ja tietokantasuunnitelmasta tehtiin uusi versio.

Kolmanteen versioon pyrittiin lisäämään kaikki tarvittavat taulut sekä ratkaisemaan ongelma kysymyksen ja vastauksen yhdistämisessä. Tietokantasuunnitelmaan lisättiin client inquiry question-taulu, johon tallentuu tieto asiakkaan ja kyselyn yhteydestä. Tämän lisäksi vastaus taulu yhdistettiin sekä asiakkaaseen ja uuteen client inquiry question- tauluun, jolloin kysymyksen ja vastauksen mukana kulkevat tiedot asiakkaasta ja kysymyksestä. (Ks. kuvio 3.) Sovelluslogiikassa kysymys ja vastaus tullaan vielä järjestämään lähetys- ja vastausajan mukaan, jotta kysely saadaan oikeaan järjestykseen ja vastaukset on varmasti yhdistetty oikeaan kysymykseen.



KUVIO 2. Ensimmäisen prototyypin taulut ja yhteydet, toteutettu käyttäen Toad Data Modeller-ohjelmaa



KUVIO 3. Tietokantasuunnitelman kolmas versio

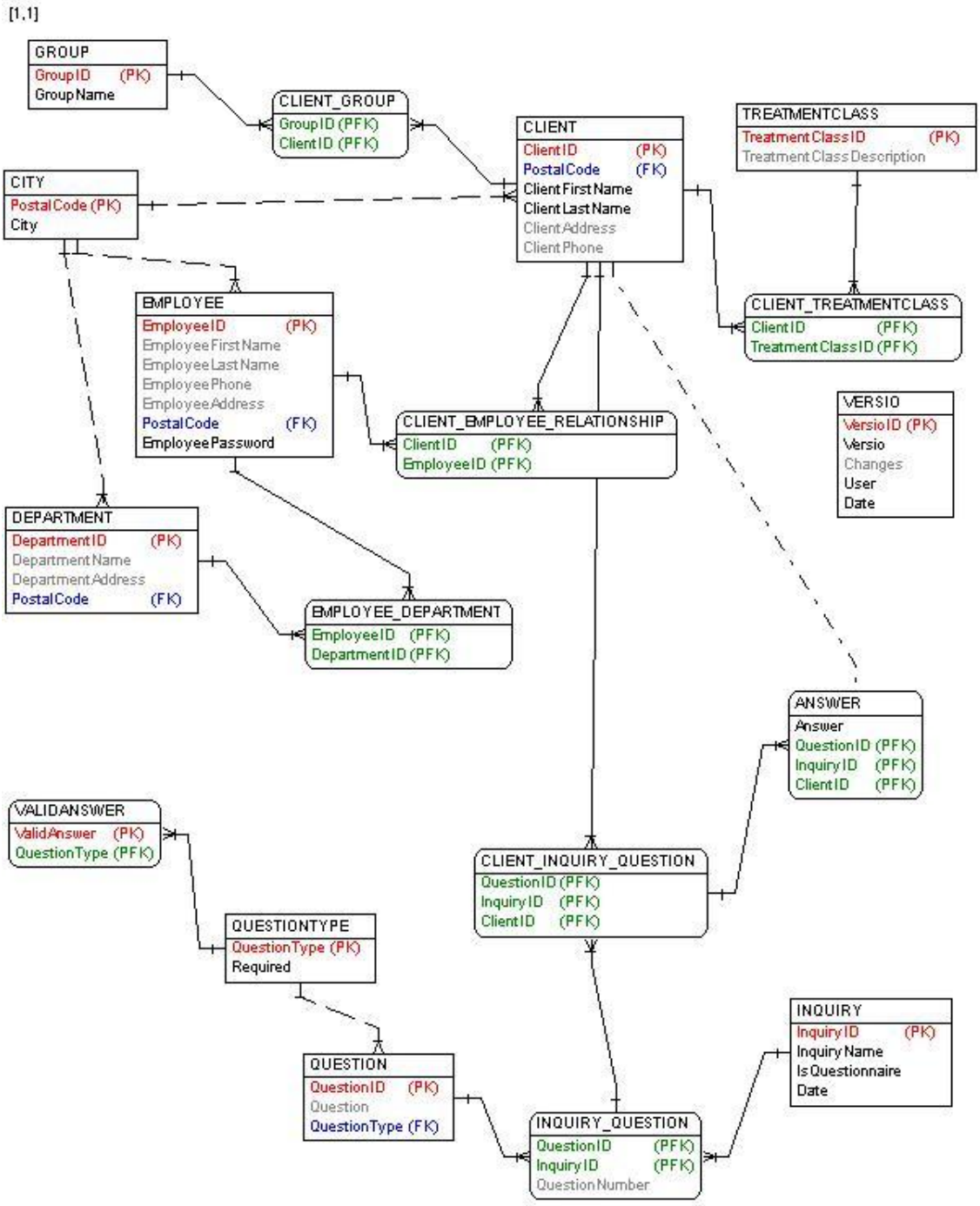
3.2.3.2 Tietokannan viimeistely ja normalisointi

Kolmannen version jälkeen suunnitelmalle tehtiin tarkastukset ja se pyrittiin muokkaamaan kolmanteen normaalimuotoon. Ensimmäisen normaalimuodon mukaan toistuvat ryhmät ja moniarvoiset sarakkeet tulee erottaa omiksi käsitteiksi. Suunnittelu oli alusta asti pyritty tekemään mahdollisimman pitkälle normalisointisääntöjä noudattavaksi, joten ensimmäistä normalisointisääntöä ei rikottu. Toisen normalisointisäännön mukaan jokaisen ei-avaintiedon tulee olla riippuvainen koko perusavaimesta, mikä ei ole missään vaiheessa ollut aiheellis-

ta toteutetussa tietokantaratkaisussa. Kolmannen normalisointisäännön mukaan tietokannan tauluista tulee poistaa sisäiset riippuvuudet. Tarkastuksen ja testauksen jälkeen myöskään tätä sääntöä rikkovia käsitteitä tai tauluja ei ollut, joten kanta on normalisoitu kolmanteen normaalimuotoon.

Tietokannan rakenteessa taulut, joiden käsitteistä yli puolet on samoja, olisi selkeyden kannalta usein hyvä yhdistää samaksi tauluksi. Toteutetussa kannassa asiakas- ja työntekijätaulut (client ja employee) sisältävät käytännössä kaikki samat käsitteet. Molempia tauluja käytetään henkilötietojen tallentamiseen, eli ne sisältävät käsitteet etunimi, sukunimi, osoite, puhelinnumero sekä perusavaimena käytetyn id-käsitteen, joka molemmissa tauluissa on automaattisella laskurilla generoitava, henkilön yksilöivä tunnus. Taulujen sisällön yhdistäminen tutkittaessa pelkästään käsitteitä näyttää käytännössä paremmalta ratkaisulta kuin niiden pitäminen erillään. Kahden erillisen taulun käyttämiseen on kuitenkin päädytty kahden henkilöryhmän eriävien luonteiden takia. Työntekijä/tutkija ja asiakas on pidettävä sovelluksen luonteen takia aina erillään sekä tietoturvan, että käytettävyyden kannalta. Lopullisessa käytössä, esimerkiksi osana perhetyötä, kaikki työntekijät eivät saa nähdä kaikkien asiakkaiden tietoja, joten asiakkaat täytyy yksilöidä myös heitä hoitavan työntekijän avulla. Tämä on tietokantaa suunniteltaessa ratkaistu tekemällä erillinen asiakkaan ja työntekijän suhdetta kuvaavalla taululla client employee relationship. Lisäksi taulut on erotettu myös siitä syystä, että työntekijät kirjautuvat järjestelmään salasanoja käyttäen, mutta asiakkailla ei vahingossakaan voi olla pääsyä samoihin tietoihin kuin työntekijällä.

Tietokannan tarkastuksessa ja viimeistelyssä suunnitelmasta poistettiin vielä lähetyspäivämäärä-taulu (send dates) ja lisättiin päivämäärä osaksi kyselytaulua (inquiry). Toisena muutoksena kysymystyyppi- (questiontype) ja validi vastaus (valid answer) -taulujen välinen yhteys vaihdettiin toisin päin, eli niin, että yhdellä kysymystyypillä on useita valideja vastauksia, mutta samat vastaukset eivät voi olla usealla eri kysymystyypillä (Kuvio 4.). Tähän ratkaisuun päädyttiin, koska käytännössä kysymystyyppejä on alle kymmenen ja niiden tärkeimmät erot ovat nimenomaan valideissa vastauksissa. Näiden muutosten lisäksi asiakas-, työntekijä- ja osastotauluihin lisättiin sarake lisätiedot (info).

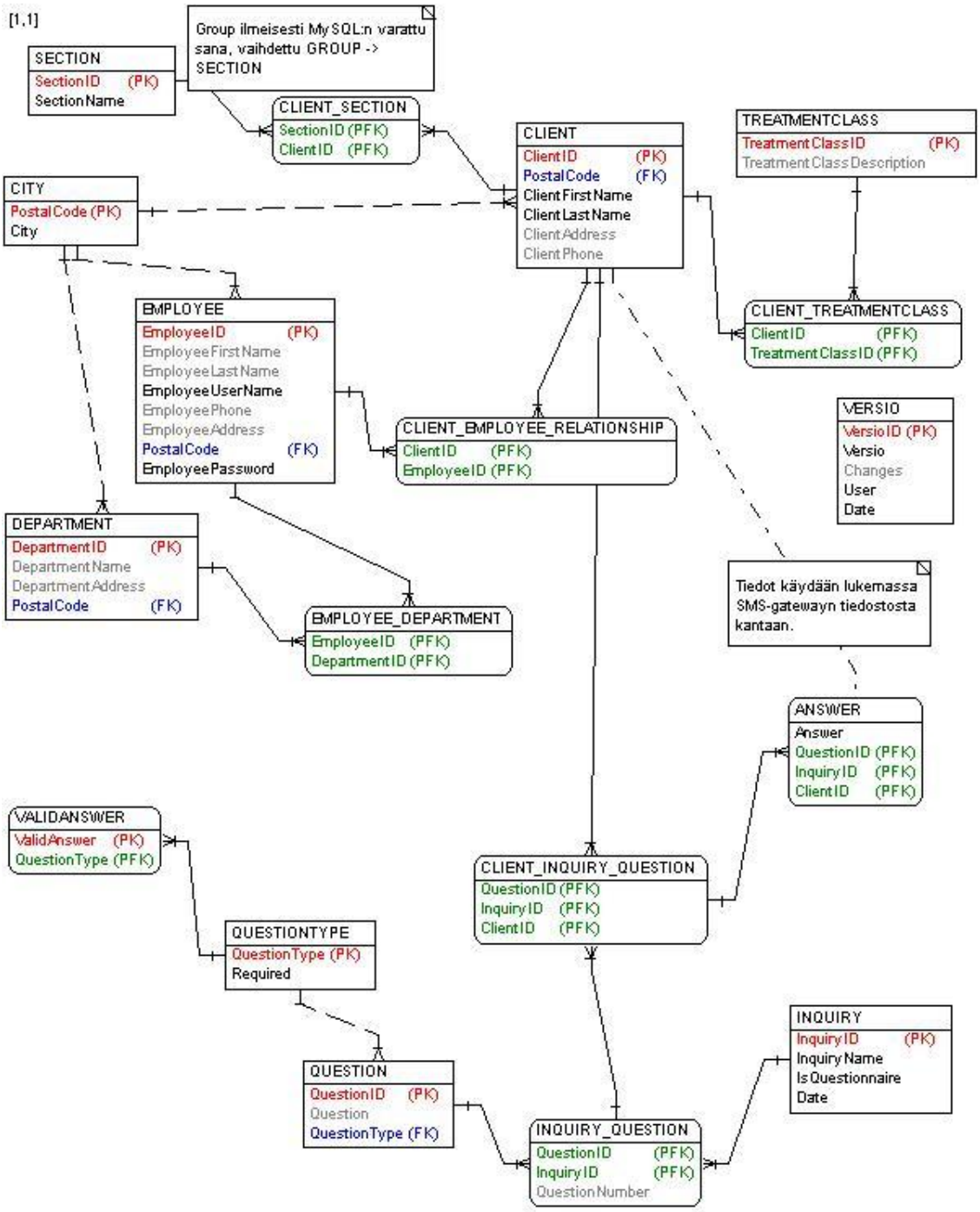


KUVIO 4. Tietokannan suunnitteluvaiheen lopullinen versio ennen käyttöliittymän testausta

3.2.3.3 Tietokannan toteutus ja testaus

Tietokannan toteutus aloitettiin suunnittelun ollessa vielä kesken, jotta testaaminen voitiin aloittaa mahdollisimman aikaisessa vaiheessa. Toad Data Modeller-ohjelmalla suunnitellusta loogisesta mallista generoitiin sql-scripti, jonka avulla toteutettiin varsinainen tietokanta. Taulujen muuttuessa luotiin uusia kantoja, joita testattiin käyttöliittymää suunniteltaessa.

Käyttöliittymän toteutuksessa käytettiin CodeIgniter-frameworkkiä, joka on toteutettu mvc-mallin mukaan. Tietokanta voitiin siis toteuttaa erikseen labranetin mysql-tietokannan avulla ja näin tauluja voitiin muuttaa samalla, kun käyttöliittymä kehittyi ja huomattiin esimerkiksi puutteita suunnittelussa. Esimerkkinä kantaan tehdyistä muutoksista käyttöliittymän toteutusvaiheen aikana on Group-taulun nimen vaihtaminen. Ensimmäisissä testauksissa käytettiin vain muutamia tauluja ja kun myöhemmin siirryttiin käyttämään koko kantaa, huomattiin, että ”group” on sql-kielessä varattu sana ja se jouduttiin korvaamaan sanalla ”section”. Koko kannan käyttöönoton aikana lisättiin myös työntekijä (employee) tauluun sarake EmployeePassword, jota käytetään käyttäjän kirjautuessa järjestelmään. (Kuvio 5.)



KUVIO 5. Tietokannan versio, jota käytettiin käyttöliittymän alkuvaiheen testauksessa

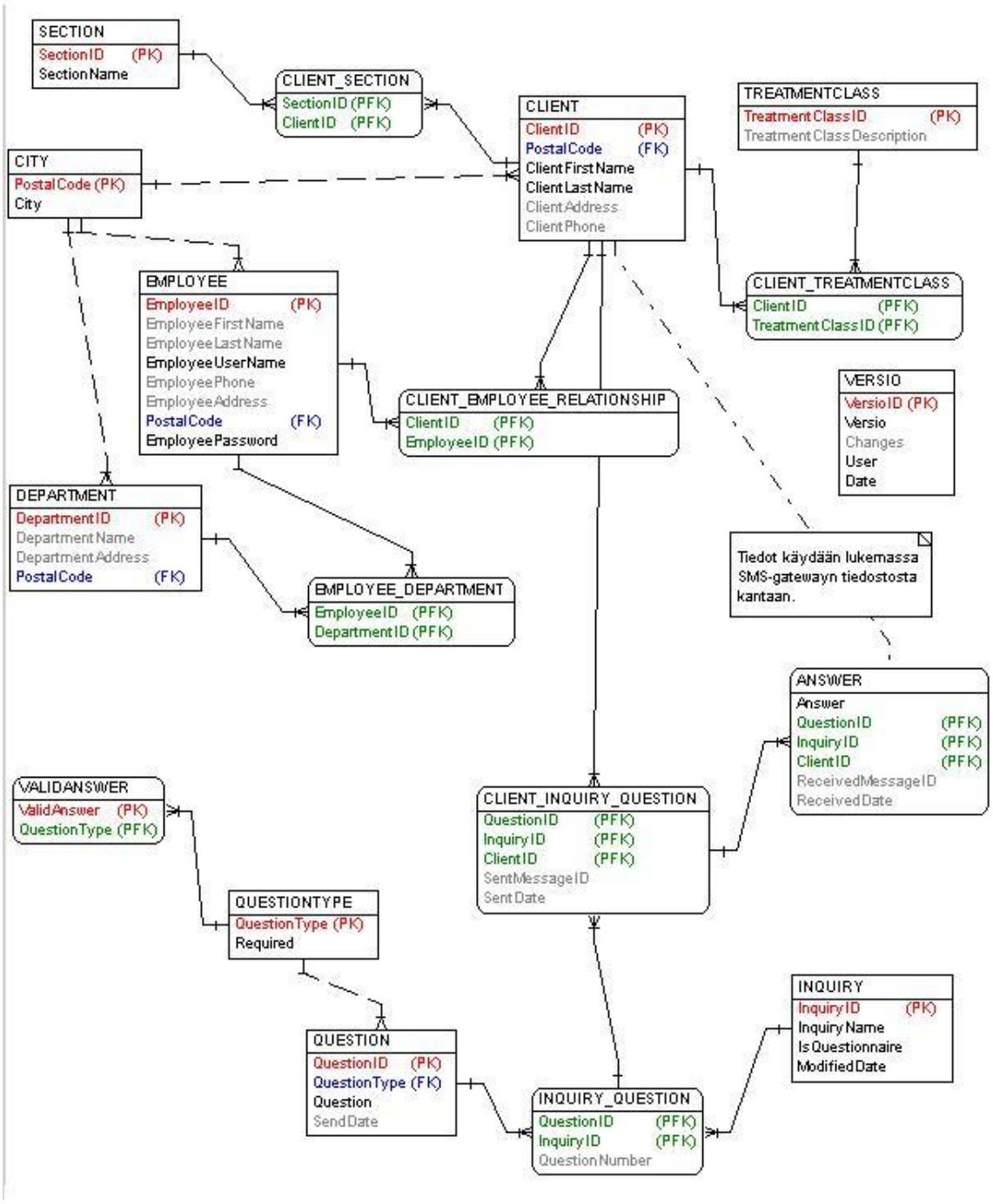
Käyttöliittymän toteutusta jatkettiin edellä esiteltyjen muutosten jälkeen lisäämällä esimerkiksi viestin lähetyksen mahdollistava toiminto. Samaan aikaan huomattiin, että tietokannasta puuttui joitakin sarakkeita, joiden hyödyntäminen helpottaisi käyttöliittymän toteuttamista. Prototyypin toteutuksessa käytetty SMS Gateway sisältää tietokannan, johon tallennetaan gatewaylle lähetetyt viestit ja sinne saapuneet viestit. Kanta käyttää niille tunnistetta MessageId, jonka avulla voidaan esimerkiksi tutkia viestin lähetyksen jälkeen, mikä sen tila on kyseisellä hetkellä, eli onko se esimerkiksi lähetetty vastaanottajalle. SMS Gatewayn gene-

roiman MessageId:n hyödyntäminen nähtiin tarpeelliseksi esimerkiksi ongelmatilanteissa, joten se päätettiin lisätä osaksi prototyypin tietokantaa. Sekaannukset lähetetyn kysymyksen ja saapuneen vastauksen välillä pyrittiin välttämään lisäämällä Clien Inquiry Question ja Answer -tauluihin eri nimiset sarakkeet, jotka tallentavat Gatewaylta saadun tiedon viestin Id:stä. Client Inquiry Question-tauluun lisättiin sarake SentMessageID ja Answer- tauluun sarake Received-Message ID (Kuvio 6).

Viestin tunnisteen lisäksi käyttöliittymän toteutusvaiheessa huomattiin, että aiemmin sovelluslogiikassa toteutettavaksi suunniteltu viestin ja vastauksen yhdistäminen lähetys- ja saapumisajan perusteella vaatisikin aikojen tallennuksen kantaan. Ratkaisuna päätettiin lisätä Question- tauluun sarake SendDate, johon tallennetaan tieto viestin lähetyspäivästä ja kellonajasta. SendDate-sarake kertoo sen päivämäärän ja ajan, jolloin viesti lähetetään ensimmäisen kerran asiakkaalle. Sekaannusten välttämiseksi tässä vaiheessa vaihdettiin myös kyselytaulun Date-sarakkeen nimeksi ModifiedDate, eli siitä selviää, milloin kyselyä on muokattu. (Kuvio 6.)

Viestin lähetysaika tallentuu myös Client_Inquiry_question-tauluun, eli käyttäjä asettaa ajan kysymykseen (Question-taulu), joka pyritään lähettämään asiakkaalle asetettuun aikaan. viestin lähetyksen onnistuessa sen aika tallentuu Client_Inquiry_Question-tauluun. Käytännössä aika on siis sama molemmissa tauluissa, mikäli viesti saadaan lähetettyä ensimmäisellä kerralla niin, että asiakas vastaa siihen. Mikäli vastausta ei saada, tallentuu uusi lähetysaika (esimerkiksi kaksi tuntia edellisen jälkeen) vain Client_Inquiry_Question-tauluun. Client_Inquiry_Question taulussa lähetysaika tallentuu SentDate-sarakkeeseen. (Kuvio 6)

Käyttöliittymän toiminnallisuutta toteutettaessa huomattiin myös, että vastauksen saapumisaika olisi hyvä tallentaa tietokantaan. Tähän ratkaisuun päädyttiin, jotta kysymyksen ja vastauksen yhdistäminen helpottuisi. Tietokannan answer-tauluun lisättiin sarake ReceivedDate, johon tallennetaan tieto saapumispäivämäärästä ja -kellonajasta. (Kuvio 6.)



KUVIO 6. Tietokannan toteutusvaiheessa päivitetty versio

3.2.4 Lopputulos

3.2.4.1 Tietokannan taulut

Tietokantaratkaisun suunnittelun aikana pyrittiin toteuttamaan tietokanta, jota voidaan hyödyntää myös prototyypin jatkokehityksessä. Kannan suunnittelu aloitettiin neljästä taulusta: työntekijä, asiakas, kysymys ja vastaus. Suunnittelu ete-

ni ratkaisuun, jossa on 12 varsinaista taulua, neljä moni-moneen- yhteyden purkavaa taulua ja niiden lisäksi versio-taulu, johon voidaan tallentaa tietokannan versiotiedot.

3.2.4.1.1 Employee

Employee-tauluun tallennetaan sovellusta käyttävien henkilöiden, eli tutkijoiden tai työntekijöiden, tiedot. Taulun perusavain on EmployeeID, joka generoidaan automaattisesti, kun uusi työntekijä luodaan, eli se on tietotyypiltään integer, autoincrement. Pakollisia tietoja taulussa ovat työntekijän käyttäjätunnus (EmployeeUserName) ja salasana (EmployeePassword), jotka syötetään käyttöliittymään uutta tunnusta luotaessa. Näiden pakollisten tietojen lisäksi taulu sisältää kentät etunimi, sukunimi, puhelinnumero ja osoite, johon haetaan kaupunki postinumero-taulusta.

Tauluun voidaan siis tallentaa tarvittaessa vain käyttäjätunnus ja salasana, mutta sinne voidaan tallentaa myös enemmän tietoa. Käyttäjätunnuksen tulee olla yksilöllinen, mikä tarkastetaan käyttäjätunnusta luotaessa Web-käyttöliittymän avulla. Salasana täytyy tunnusta luotaessa kirjoittaa kahdesti ja sen tulee olla vähintään kuusi merkkiä pitkä. Salasana koodataan kantaan MD5 viestitiivistealgoritmia.

3.2.4.1.2 Department

Department-tauluun tallennetaan tiedot osastosta, johon käyttäjät voivat kuulua. Tämä taulu on suunniteltu hyödynnettäväksi silloin, kun käyttäjät ovat esimerkiksi jonkin isomman yrityksen työntekijöitä ja eFamilyCoach-sovellusta hyödyntävät useiden eri yksikköjen työntekijät. Tämän taulun perusavain on automaattisesti generoituva DepartmentID, joka luodaan uutta osastoa lisättäessä. Lisäksi taulussa on kentät osaston nimi (DepartmentName) ja osaston osoite (DepartmentAddress), johon haetaan oikea kaupunki postinumeron perusteella Postal Code-taulusta.

3.2.4.1.3 Client

Client-tauluun tallennetaan tietoja sovellusta käyttävien työntekijöiden tai tutkijoiden asiakkaista. Tauluun tallennetaan siis niiden henkilöiden tiedot, joille sovelluksen avulla lähetetään kysymyksiä. Asiakastaulua voidaan pitää kannan tärkeimpänä tauluna, koska se sisältää esimerkiksi puhelinnumeron, johon viesti tulee lähettää. Samalla se kuitenkin sisältää tietoa, joka on pystyttävä pitämään hyvin salassa, kuten esimerkiksi asiakkaan henkilötietoja. Asiakastietoja saavat nähdä vain ne työntekijät, jotka on yhdistetty asiakkaaseen Client_employee_relationship-taulun avulla.

Taulun perusavain on ClientID, joka generoidaan automaattisesti uutta asiakasta tallennettaessa. Taulu sisältää pakolliset kentät asiakkaan etunimi ja asiakkaan sukuni (ClientFirstName, ClientLastName) sekä puhelinnumero, joka on sovelluksen käytön kannalta tärkein kenttä. Asiakkaan nimen yhdistämistä puhelinnumeroon harkittiin tietokannan suunnitteluvaiheessa, koska tiedot eivät saa näkyä kuin tietyille ihmisille ja siksi harkittiin esimerkiksi puhelinnumeron yhdistämistä vain johonkin koodiin, jonka yhdistäminen asiakastietoihin olisi jossakin erillisessä tiedostossa. Prototyyppivaiheessa tiedot jätettiin kuitenkin samaan tauluun puhelinnumeron kanssa, joten päätös tietoturvan kannalta parhaasta ratkaisusta jää jatkokehitysvaiheeseen.

Asiakas-tauluun voidaan tallentaa tietoa Web-käyttöliittymän avulla. Työntekijän tai tutkijan täytyy kirjautua sovellukseen omilla tunnuksillaan, minkä jälkeen uusi asiakas voidaan lisätä lomakkeen avulla. Lisäysvaiheessa voidaan lisätä kaikki asiakas-taulun tiedot, eli myös osoite (kaupunki haetaan tässäkin tapauksessa postinumero-taulusta), tai ainoastaan nimi ja puhelinnumero.

3.2.4.1.4 Treatmentclass

Hoitoluokka-tauluun voidaan tallentaa tietoja asiakkaan hoidosta, suunnittelun alkuvaiheessa taulun nimi oli diagnoosi, mutta se vaihdettiin suunnittelun edetessä neutraalimmaksi ja sopivammaksi myös tutkimuskäyttöön. Tauluun voidaan tallentaa lisätietoja tai esimerkiksi hoitohistoriaa ja sen erottaminen omaksi taulukseen mahdollistaa useampien tietojen ja historian tallentamisen.

3.2.4.1.5 Section

Ryhmä-taulun lisäämisen idea on syntynyt toimeksiantajan toiveesta voida erotella käyttäjiä erilaisiin ryhmiin. Käytännössä ryhmä voi olla esimerkiksi tietty perhe, johon kuuluu vanhempia ja lapsia. Tietoturvan kannalta henkilöiden yhdistäminen samaksi perheeksi voi kuitenkin olla huono valinta, mutta päätöksen tekeminen taulun käytöstä jäänee jatkokehitykseen. Prototyypin pilotointivaiheessa ryhmä-taulua voidaan hyödyntää esimerkiksi yhdistämään samalla luokalla tai samassa koulussa olevia henkilöitä. Taulun perusavain on SectionID, joka generoituu automaattisesti ryhmää lisättäessä ja sen lisäksi taulussa on sarake ryhmän nimi, joka on pakollinen tieto.

3.2.4.1.6 Inquiry

Kysely-taulua käytetään tallentamaan kysymysryhmiä, jotka voidaan lähettää asiakkaille. Käytännössä taulu sisältää kyselyn otsikkotiedot ja yhteyden kysymyksiin, jotka on yhdistetty samaan kyselyyn. Käyttäjät voivat siis sovelluksen kirjautumisen jälkeen siirtyä luomaan kyselyä, johon voidaan luoda kokonaan uusia kysymyksiä tai valita sovellukseen aiemmin tallennettuja kysymyksiä.

Taulun perusavain on InquiryID, joka generoidaan automaattisesti uutta kyselyä luotaessa. Sen lisäksi taulussa on kolme kenttää: kyselyn nimi (InquiryName), päivämäärä (Date) ja kenttä, joka kertoo, onko kysely kooste vai lähetetty kysely (IsQuestionnaire). Kyselyn nimi on sarake, johon voidaan tallentaa kyselyä kuvaava nimi, esimerkiksi ”perhetutkimus 01/2011”. Päivämäärä kertoo, milloin kysymys on luotu ja kenttä ”IsQuestionnaire” onko luotu kysely lähetettäväksi tarkoitettu valmis kysely vai vain kooste samantyyppisiä kysymyksiä. IsQuestionnaire on tietotyypiltään boolean ja arvo 1 tarkoittaa, että kysely on lähetettäväksi tarkoitettu ja arvo 0, että se on vain kooste kysymyksiä. Lisäksi taulussa on sarake ModifiedDate, johon tallennetaan tieto kyselyn viimeisimmästä muokkausajankohdasta.

3.2.4.1.7 Inquiry question

Kyselyn kysymys on taulu, jonka avulla voidaan yhdistää kysymyksiä tiettyyn kyselyyn. Taulu saa ID-tiedot kysely- ja kysymys-tauluista ja niiden lisäksi määri-

tetään kysymyksen numero, eli monentenako kyselyn kysymyksenä se lähetetään asiakkaalle. Tauluun tallennettavat tiedot määritetään Web-käyttöliittymässä työntekijän luodessa uutta kyselyä. Käyttäjä luo kyselyn (tallentuu inquiry-tauluun) ja valitsee tai luo siihen haluamiansa kysymyksiä lomakkeen avulla, josta voidaan suoraan määrittää taulussa oleva kenttä kysymyksen numero (QuestionNumber). Käyttäjä määrittää siis valitessaan kysymyksien lähetysjärjestyksen kysymyksen järjestysnumeron kyselyssä. Taulussa on kaksiosainen avain, joka muodostuu kysely- ja kysymystaulun perusavaimista.

3.2.4.1.8 Question

Kysymys-tauluun tallennetaan käyttäjien luomia kysymyksiä, joita voidaan lähettää asiakkaille. Kysymysten lähettäminen asiakkaille tapahtuu valitsemalla kysymyksiä kyselyyn (Inquiry-taulu), jonka tietojen avulla lähetetään tiettyjä kysymyksiä valittuun aikaan valittu määrä. Käyttäjä voi kirjautua sovellukseen ja luoda uusia kysymyksiä, jotka voidaan yhdistää suoraan haluttuun kyselyyn, joka lähetetään asiakkaalle. Toinen vaihtoehto on kysymyksen tallentaminen yksittäisenä, jolloin sitä ei yhdistetä mihinkään kyselyyn, vaan se voidaan myöhemmin valita käytettäväksi jossakin kyselyssä. Käyttäjän on myös mahdollista tallentaa kysymys johonkin kyselykokonaisuuteen, esimerkiksi lapsille tarkoitetut kysymykset. Kyselykokonaisuus on kysely-taulun avulla tallennettu joukko kysymyksiä, joka erotetaan lähetettävästä kyselystä sarakkeen ”IsQuestionnaire” arvolla 0.

Kysymys-taulun perusavain on QuestionID, joka generoidaan automaattisesti uutta kysymystä luotaessa. Taulussa on sarake Question, johon kysymys tallennetaan. Sen lisäksi kysymykselle valitaan kysymystyyppi, johon vaihtoehdot haetaan QuestionType-taulusta. Kysymys lähetetään asiakkaalle osana kyselyä, jossa määritellään jokaisen kysymyksen lähetysaika. Valittu lähetysaika tallentuu kysymys-taulun sarakkeeseen SendDate.

3.2.4.1.9 Questiontype

Questiontype-tauluun tallennetaan eri tyypit, joita käyttäjien luomat kysymykset voivat edustaa. Prototyyppivaiheessa kysymystyyppejä on ainakin neljä: avoin, numeroarvo, muistutus ja info. Normaaleja kysymyksiä, joihin siis odotetaan käyttäjältä vastausta, ovat avoin kysymys ja numeroarvo. Avoimeen kysymykseen odotetaan vastauksena avointa vastausta, eli yhtä tai useampaa sanaa tai lausetta. Numeroarvo on kysymys, jonka vastausvaihtoehdot ovat numerot yhdestä seitsemään (Likert-asteikko). Muistutus ja info ovat työntekijältä tai tutkijalta tulevia viestejä, joihin ei odoteta vastausta.

Taulun perusavain on QuestionType, joka on luonnollinen avain määrittäen samalla kysymystyypin nimen, eli esimerkiksi ”numeroarvo”. Perusavaimen lisäksi taulussa on sarake required, joka määrittää, odotetaanko kysymystyyppiä edustavaan kysymykseen vastausta. Required on tietotyyppiä boolean, joka arvo 1 tarkoittaa, että vastausta odotetaan, arvo 0 että kyseiseen kysymykseen ei odoteta vastausta.

3.2.4.1.10 Valid Answer

Kysymystyypille voidaan käyttäjän niin halutessa määrittää validi vastaus. Esimerkiksi vastaus voidaan haluta Likert-asteikon arvoilla 1-7, jolloin kysymyksen valideja vastauksia ovat 1,2,3,4,5,6 ja 7. Tässä tapauksessa käyttäjä on siis lisännyt kysymystyypin ”Likert” QuestionType-tauluun ja asettanut sen valideiksi vastauksiksi edellä mainitut arvot ValidAnswer-tauluun.

Taulun perusavain on ValidAnswer, joka on luonnollinen avain määrittäen samalla validin vastauksen arvon, esimerkiksi ”1”. Taulussa ei ole muita sarakkeita, ainoastaan yhteys siihen kysymystyyppiin, johon määritetty vastaus se on.

3.2.4.1.11 Answer

Vastaus-tauluun tallennetaan viestit, jotka asiakas on lähettänyt saatuaan jonkin kysymyksen. Tauluun tallennetaan ainoastaan saatu vastaus, joka yhdistetään lähetettyyn kysymykseen client_inquiry_question-taulun kautta. Kysymyksen ja vastauksen oikeaa yhdistämistä tutkitaan sovelluslogiikassa vertaamalla lähe-

tys- ja saapumisajankohtaa. Taululla on moniosainen avain, joka koostuu taulujen asiakas, kysely ja kysymys perusavaimista. Niiden lisäksi taulussa on sarake answer, johon vastaus tallennetaan ja sarake ReceivedMessageID, johon tallennetaan Gatewayn generoima, saapuneen viestin yksilöivä tunniste. Tunnisteen tallentaminen tietokantaan helpottaa käyttöliittymän toiminnallisuuden toteutusta. Tunnistetietojen lisäksi tauluun tallennetaan tieto siitä, milloin viesti on asiakkaalta vastaanotettu. Tieto tallennetaan sarakkeeseen ReceivedDate.

3.2.4.1.12 City

Kaupunki on tauluihin työntekijä, asiakas ja osasto liitetty taulu, joka määrittää osoitteeseen oikean kaupungin postinumeron perusteella. Esimerkiksi postinumeroa ”40100” vastaava arvo on ”Jyväskylä”. Taulu on lisätty tietokantaan, jotta vältytään virheiltä osoitteissa ja samalla sama postinumero ei saa useita arvoja. Tauluun voidaan esimerkiksi käyttöönoton aikana tallentaa valmiiksi postinumerot ja niitä vastaavat kaupungit.

Taulun perusavain on PostalCode, joka on luonnollinen perusavain, koska postinumero on uniikki ja sitä voidaan suoraan käyttää perusavaimena. Perusavaimen lisäksi taulussa on pakollinen tieto city, joka kertoo postinumeroa vastaavan kaupungin.

3.2.4.2 Moni-moneen-yhteyden purkavat taulut

Tietokannassa on aiemmin kuvattujen taulujen lisäksi niiden välisten moni-moneen-yhteyksien purkavia tauluja. Yhteyksien purkavat taulut sisältävät vain tiedot yhdistettävien taulujen perusavaimista.

3.2.4.2.1 Employee Department

Employee_Department-taulu purkaa työntekijän ja osaston välisen moni-moneen-yhteyden. Työntekijä voi siis kuulua useampaan osastoon ja yhdessä osastossa voi olla useita työntekijöitä. Tähän ratkaisuun päädyttiin, koska esimerkiksi sama työntekijä voi jossakin vaiheessa toimia useammalla osastolla yhtä aikaa ja samalla mahdollistetaan myös historiatietojen tallennus.

3.2.4.2.2 Client Employee Relationship

Client_Employee_Relationship-taulu purkaa asiakkaan ja työntekijän välisen moni-moneen-yhteyden. Yhteys asiakkaan ja työntekijän välillä tarvitaan, koska sovellukseen kirjautunut käyttäjä saa nähdä vain omat asiakkaansa, eli asiakkaista nähdään vain ne, jotka on yhdistetty tietokannassa kirjautuneeseen työntekijään. Yhteys mahdollistaa sen, että yhdellä työntekijällä voi olla useita asiakkaita ja samalla yhdestä asiakkaasta voi vastata useampi työntekijä.

3.2.4.2.3 Client Treatmentclass

Client_Treatmentclass-taulu purkaa moni-moneen-yhteyden asiakkaan ja hoitoluokan välillä. Tällä ratkaisulla on mahdollistettu useampien lisätietojen tallentaminen esimerkiksi asiakkaan hoidosta. Samalla erillinen taulu mahdollistaa hoitohistorian tallentamisen tietokantaan.

3.2.4.2.4 Client Section

Client_Section-taulu purkaa moni-moneen-yhteyden asiakas- ja ryhmätaulun välillä. Asiakas voi siis kuulua useampaan ryhmään ja samaan ryhmään voi kuulua useampi asiakas.

3.2.4.2.5 Client Inquiry Question

Client_Inquiry_Question-taulu purkaa moni-moneen-yhteyden asiakkaan ja kyselyn kysymys-taulujen välillä. Samalla sen avulla voidaan yhdistää asiakas, kysymys ja vastaus. Asiakkaalle voidaan siis tallentaa useita kysymyksiä (kysely sisältää yleensä aina useamman kuin yhden kysymyksen) ja sama asiakas voidaan lähettää usealle asiakkaalle. Taulussa on asiakas-, kysely- ja kysymystunnisteen lisäksi sarake SentMessageID, johon on tallennettu SMS Gatewayn generoima, viestin yksilöivä tunniste. Viesti lähetetään asiakkaalle ensimmäisen kerran kysymys-taulussa määriteltyyn aikaan (SendDate-sarake). Lähetysaika tallentuu myös Client_Inquiry_Question-taulun SentDate-sarakkeeseen. Mikäli asiakas ei vastaa lähetettyyn kysymykseen ensimmäisellä kerralla, lähetetään hänelle muistutusviesti ja sen jälkeen uudelleen sama kysymys, jolloin kysymyksen SentDate-tieto päivittyy vain Client_Inquiry_Question-tauluun.

3.3 Web-käyttöliittymä

3.3.1 Vaatimukset

Web-käyttöliittymä on tutkijoille ja työntekijöille tarkoitettu työkalu, jonka avulla voidaan hallita asiakkaita sekä kyselyitä. Käyttäjä kirjautuu sovellukseen, jonka jälkeen hänen on helposti voitava lisätä, muokata ja poistaa käyttäjiä, valita ja muokata kysymyksiä kyselyihin sekä tutkia saatuja vastauksia erilaisten raporttien avulla. Käyttöliittymän tärkein osa-alue on käytettävyys, koska sovelluksen sisältämät tiedot eivät saa joutua väärille käyttäjille ja esimerkiksi tuloksien raportoinnin on oltava täysin luotettavaa.

Käytettävyyden lisäksi on otettava huomioon tietoturva myös osana sivun sisällön suunnittelua, koska joissakin tapauksissa asiakas ja työntekijä voivat yhdessä luoda kysymyksiä tai kyselyitä. Näissä tapauksissa asiakas ei missään tapauksessa saa nähdä muiden asiakkaiden tietoja. Tämä ongelma pyrittiin ratkaisemaan jakamalla kyselyn luontiprosessi osiin niin, että työntekijä valitsee ensimmäiseksi kyseessä olevan asiakkaan halutusta ryhmästä ja kysymyksen luodaan vasta sen jälkeen. Tässä tapauksessa käyttäjä pääsee mukaan luontiprosessiin vasta siinä vaiheessa, kun työntekijä on jo siirtynyt pois asiakkaiden tietoja sisältävältä sivulta. Tietoturvan parantamiseksi sovellus näyttää kyselyn luontivaiheessa asiakastiedoissa ainoastaan käyttäjän nimen ja puhelinnumeron. Muita asiakkaiden tietoja hallinnoidaan erillisellä asiakkaiden hallinta - välilehdellä, jossa voidaan selata, lisätä, poistaa ja muokata käyttäjien tietoja.

3.3.2 Prototyypin Web-käyttöliittymän toteuttajat

Ensimmäisen prototyypin Web-käyttöliittymä toteutettiin osana tietokantatoteutusta, sillä se kuului osaksi Jyväskylän Ammattikorkeakoulun Tietokannan hallinta-opintojaksoa. Web-käyttöliittymän suunnitteluun osallistuivat kaikki projektiryhmän jäsenet ja toteutukseen Leo-Matti Lehtonen ja Laura Saari.

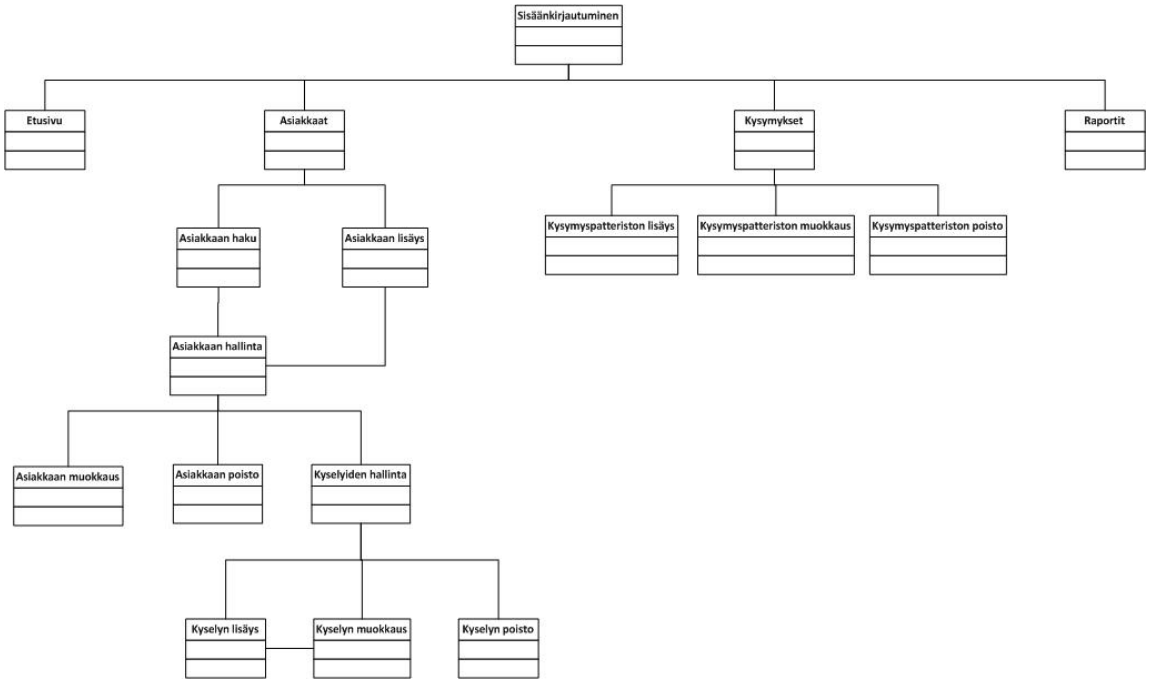
3.3.3 Käytetyt työkalut

Käyttöliittymän graafisen ilmeen suunnittelussa käytettiin Adobe Photoshop Creative Suite 4-ohjelmaa. Toiminnallisuuden toteutukseen käytettiin CodeIgniter-kehystä. CodeIgniter on avoimen lähdekoodin Web-sovellusten tekoon tarkoitettu framework, jolla voidaan toteuttaa dynaamisia Web-sivuja. Se on toteutettu pääasiassa PHP-kielellä ja lisäksi sivujen toteutuksessa olivat käytössä html-, xml- ja css-kielellä.

3.3.4 Web-käyttöliittymän toteutus

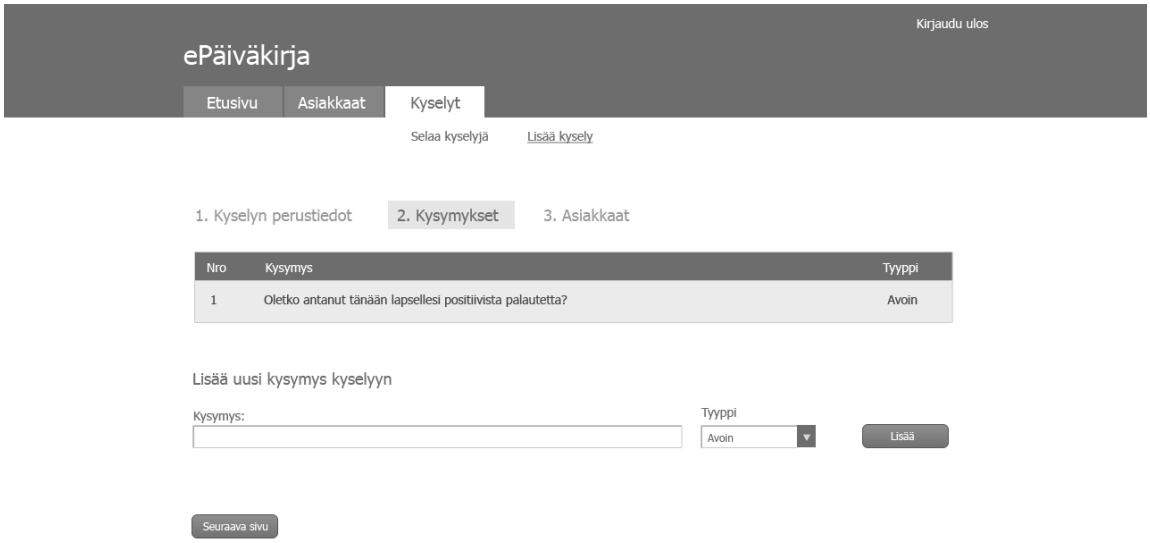
Web-käyttöliittymän toteutus aloitettiin tietokannan suunnittelun jälkeen hahmottelemalla käyttäjän toimintoja sekä graafista ilmettä. Käytettävyys on jo prototyyppivaiheessa hyvin tärkeä osa-alue, joten pyrittiin suunnittelemaan mahdollisimman yksinkertainen ja selkeä kokonaisuus. Käyttöliittymän suunnittelussa nousi tärkeäksi huomioksi myös se, että esimerkiksi työntekijän ja hänen asiakkaansa on pystyttävä luomaan kysymyksiä yhdessä, mutta asiakas ei missään vaiheessa saisi nähdä kenenkään muun käyttäjän tietoja.

Käyttöliittymän suunnittelun aikana elektronisen päiväkirjan suunnitelmaa muutettiin niin, että tuleva sovellus tulisi jakautumaan useampaan tuotteeseen kohderyhmien mukaisesti. Esimerkkeinä käytettiin esimerkiksi eroja perheiden ja parisuhteen ohjauksen välillä. Uuden suunnitelman pohjalta käyttäjät tulisivat hankkimaan tarkoituksiinsa sopivimman tuotteen ja saisivat sen mukana tietyn kysymyspatteriston. Näiden tietojen perusteella Web-käyttöliittymän ensimmäiseksi sivuksi valittiin kirjautuminen, jonka jälkeen siirrytään joko suoraan sovellukseen tai mahdolliselle välisivulle, josta valittaisiin haluttu työkalu. Käyttäjä voi siis olla hankkinut useampia kokonaisuuksia, joista valitsee kyseiseen tilanteeseen sopivimman. Tämän jälkeen siirrytään varsinaiseen sovellukseen, joka on jaettu viiteen eri välilehteen: etusivuun, kysymysten hallintaan, asiakkaiden hallintaan, kysymysten luontiin ja raporttien hallintaan. Näistä sivuista prototyyppi käsittää muut lukuun ottamatta raporttien hallintaa. (Kuvio 7.)



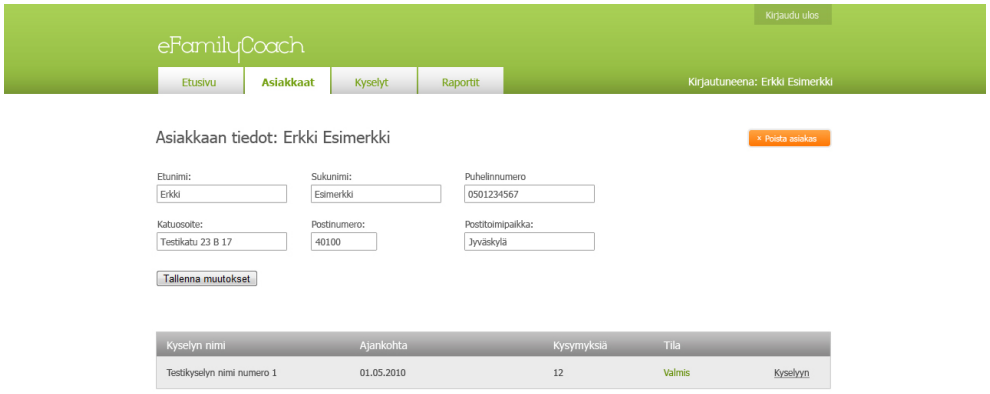
KUVIO 7. Web - käyttöliittymän sivurakenne

Käyttöliittymän graafisen ilmeen suunnitteli Leo-Matti Lehtonen. Ulkoasusta pyrittiin tekemään mahdollisimman selkeä ja yksinkertainen. (Kuvio 8.) Navigointi sivujen välillä sijoitettiin sivuston ylälaitaan ja lomakkeet avautuvat jokaisella sivulla samaan kohtaan. Esimerkiksi kuvassa esitetty kyselyt-välilehti avautuu useampisivuiseksi lomakkeeksi, jossa sivujen välillä navigointi tapahtuu samalla tavalla, kuin pääsivujen välillä.



KUVIO 8. Käyttöliittymän ensimmäinen versio

Ensimmäistä käyttöliittymäversiota käytettiin sovelluksen testaamiseen. Käyttöliittymän toiminnallisuuden testauksen edetessä suunniteltiin uusi layout, joka oli navigoinniltaan hyvin samanlainen, mutta erosi aiemmasta lähinnä värityksel-tään. Tämä käyttöliittymän versio jäi lopulliseksi versioksi prototyyppiin. (Kuvio 9.)



KUVIO 9. Käyttöliittymä

Toteutusvaiheessa käyttöliittymä jaettiin vaiheisiin näyttö kerrallaan, jotta prosessi etenisi mahdollisimman selkeästi ja vältyttäisiin virheiltä, jotka aiheutuvat huonosta suunnittelusta. Ensimmäisenä toteutettiin sisään kirjautuminen niin, että käyttäjä voi joko kirjautua valmiilla tunnuksellaan (kannassa EmployeeUserName) tai luomalla itselleen tunnuksen. Sisään kirjautuessa hyödynnetään siis kannassa olevaa Employee-taulua, johon on tallennettu käyttäjien tiedot. Käytössä oleva CodeIgniter framework on toteutettu mvc-arkkitehtuurin mukaisesti, eli se voidaan jakaa kolmeen osaan: malli, näkymä ja ohjain (model, view, controller). Malli huolehtii prototyypissä tiedon tallentamisesta tietokantaan, eli esimerkiksi käyttäjän luodessa tunnuksen tiedot tallennetaan kannan Employee-tauluun. Seuraava esimerkki on employee-model, eli malli, jota käyttäen tallennetaan tietoa employee-tauluun.

```
<?php

class Employee_model extends Model {

function validate()

        {
$this->db->where('EmployeeUserName', $this->input-
>post('username'));
$this->db->where('EmployeePassword', md5($this->input-
>post('password')));
$query = $this->db->get('EMPLOYEE');
if($query->num_rows == 1)
        {
        return true;
        }
}
function create_employee()
 {
 $new_employee_insert_data =
 array('EmployeeFirstName' => $this->input-
>post('first_name'),
        'EmployeeLastName' => $this->input-
```

```
>post('last_name'),
'EmployeeUserName' => $this->input->post('username'),

'EmployeePhone' => $this->input->post('phone'),
'EmployeeAddress' => $this->input->post('address'),
'PostalCode' => $this->input->post('postalcode'),
'EmployeePassword' => md5($this->input->post('password')));
$insert = $this->db->insert('EMPLOYEE',
$new_employee_insert_data);
return $insert;}
}
```

Näkymä-osiossa huolehditaan ulkoasun määrittämisestä, eli prototyypin tietokannasta hakeman tiedon esittämisestä ja graafisesta ilmeestä. Ohjain vastaanottaa käyttäjän antamat käskyt ja ohjaa mallia ja näkymää niiden mukaisesti. Käyttäjä esimerkiksi syöttää tiedot kirjautumislomakkeeseen, jonka jälkeen malli tarkastaa, että tiedot vastaavat kannassa olevia tietoja ja riippuen siitä, ovatko ne oikein vai väärin, ohjaa käyttäjän joko sovelluksen etusivulle (kirjautuminen onnistuu) tai virheestä ilmoittavalle sivulle (tiedot eivät vastaa kannassa olevia tietoja).

3.3.5 Lopputulos

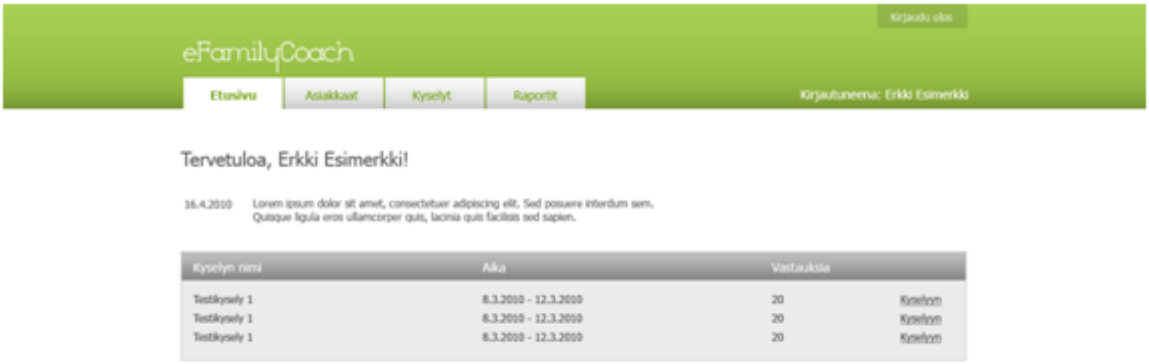
Web-käyttöliittymän toteutuksen aikana testattiin sekä tietokannan, että käyttöliittymän toimivuutta ja päädyttiin prototyypin lopputulokseen, joka on pyritty saamaan mahdollisimman helppokäyttöiseksi käyttäjien näkökulmasta, mutta samalla myös mahdollisimman hyvin skaalautuvaksi jatkokehitystä ajatellen. Tietokannan muutokset tullaan jatkokehityksessä tekemään mahdollisesti hyödyntäen prototyyppiin valmistunutta tietokantaa, kun taas käyttöliittymä tullaan luultavasti uudistamaan täysin. Tästä syystä käyttöliittymän suunnittelussa keskityttiin lähinnä toteuttamaan demoihin ja ensimmäiseen tutkimukseen tarkoitettu selkeä, mutta helposti toteutettava käyttöliittymä. Jatkokehityksessä hyödynnetään huomattavasti ensimmäistä prototyyppivaihetta useammin esimerkiksi käytätettävyysasiantuntijoita, joten ensimmäisen varsinaisen version käyttöliittymä tullaan suunnittelemaan huomattavasti tarkemmin.

3.3.6 Näytöt ja toiminnot

Web-käyttöliittymä jakautuu neljään osaan: etusivuun sekä asiakkaat-, kyselyt- ja raportit-välilehtiin.

3.3.6.1.1 Etusivu

Etusivulla kirjautunut käyttäjä näkee uusimmat tiedotteet ja viimeisen käynnin aikana tai sen jälkeen lähetettyjen kyselyiden tilan. Käyttäjä näkee, kuinka paljon vastauksia viimeisimpiin kysymyksiin on saatu ja voi siirtyä katsomaan vastauksia. Etusivulla, kuten kaikilla muillakin sivuilla, kirjautuneen käyttäjän tiedot näkyvät navigoinnin vieressä oikealla. Uloskirjautuminen löytyy oikeasta yläkulmasta. (Kuvio 10.)

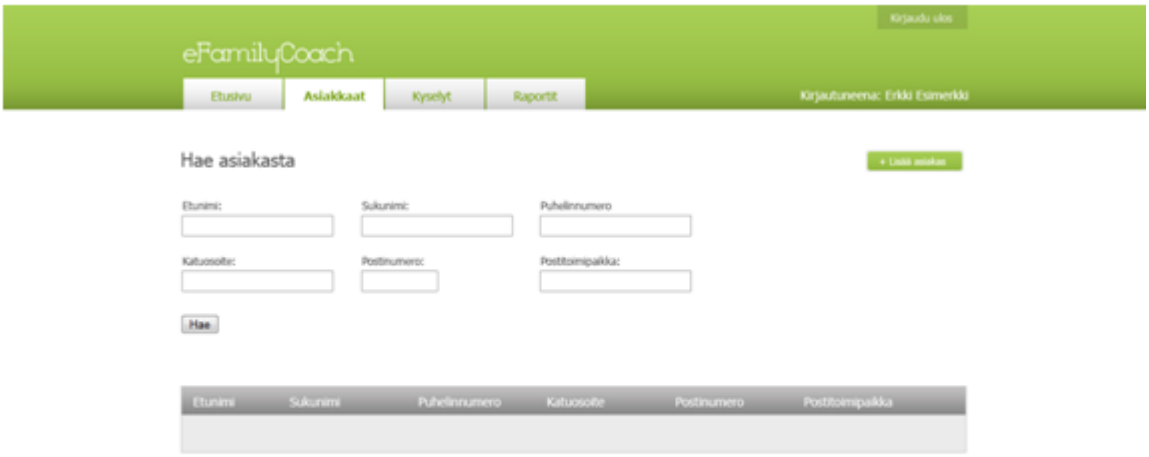


KUVIO 10 Web-käyttöliittymän etusivu

3.3.6.1.2 Asiakkaat

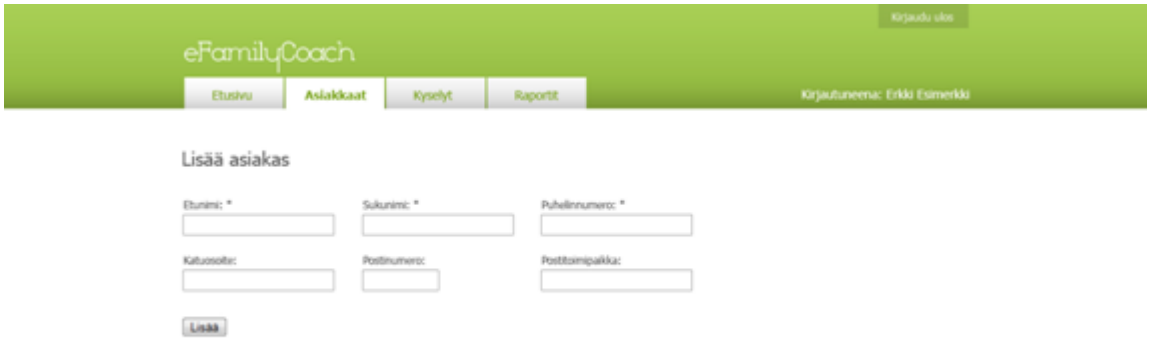
Asiakkaat-välilehden ensimmäisellä sivulla käyttäjä voi hakea asiakkaitaan lomakkeen avulla. Haussa voi käyttää kaikkia kenttiä, esimerkiksi kirjoittamalla postitoimipaikaksi ”Jyväskylä” saadaan näkyviin kaikki kirjautuneen käyttäjän asiakkat, joiden postitoimipaikaksi on tallennettu Jyväskylä. Sivulta voidaan siirtyä myös asiakkaan lisäykseen oikeassa reunassa olevan ”lisää asiakas”-painikkeen kautta. (Kuvio

11.)



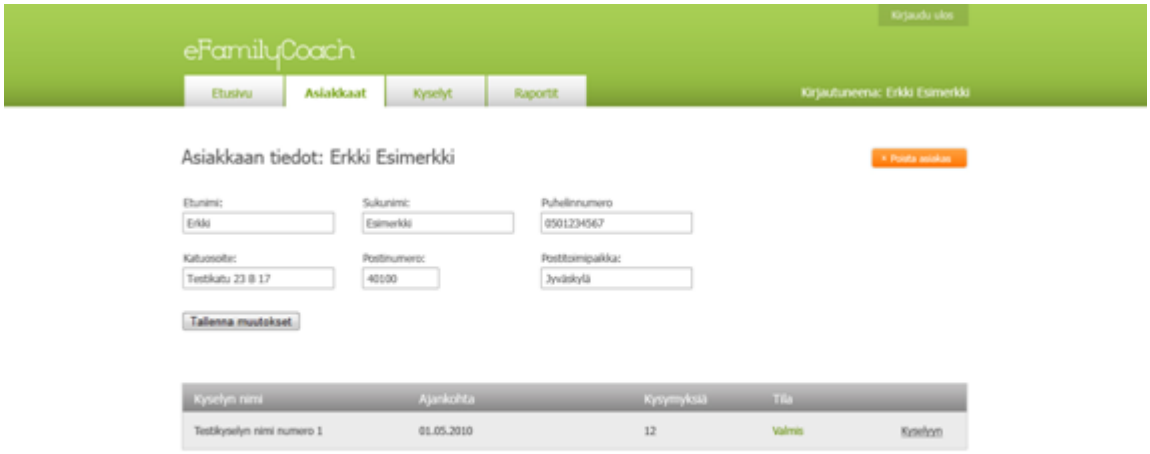
KUVIO 11 Asiakkaat-välilehti

Asiakkaan lisääminen tapahtuu samanlaisella lomakkeella, kuin asiakkaan haku. (Kuvio 12.)



KUVIO 12 Asiakkaan lisääminen

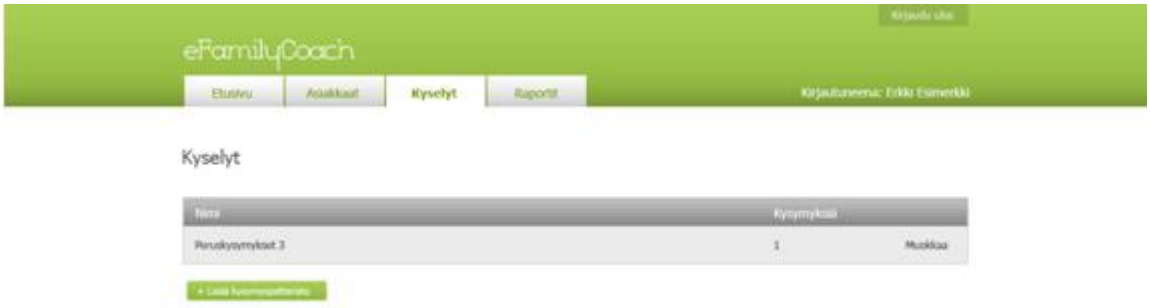
Asiakkaat-välilehden haun tuloksista voidaan siirtyä muokkaamaan haluttua asiakasta. Tiedot aukeavat lomakkeeseen, jossa kaikkia asiakkaan tietoja voidaan muokata. Muokkauksen yhteydessä näkyvät myös asiakkaan senhetkiset kyselyt. (Kuvio 13.)



KUVIO 13 Asiakkaan tietojen muokkaus

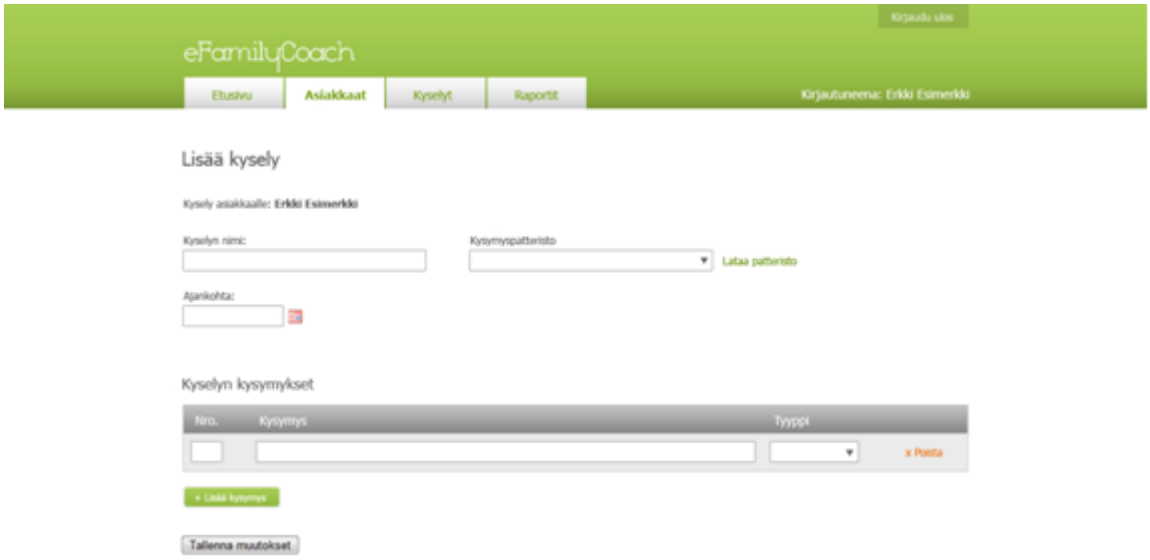
3.3.6.1.3 Kyselyt

Kyselyt-välilehdellä voidaan lisätä, muokata ja poistaa kyselyitä ja kysymyksiä. Välilehden etusivulla kirjautunut käyttäjä näkee kaikki hänelle tallennetut kysymyspatteristot, joista voidaan valita kysymyksiä asiakkaille tehtäviin kysymyksiin. Uuden patteriston lisäykseen päästään listauksen alla olevan painikkeen kautta. (Kuvio 14.)



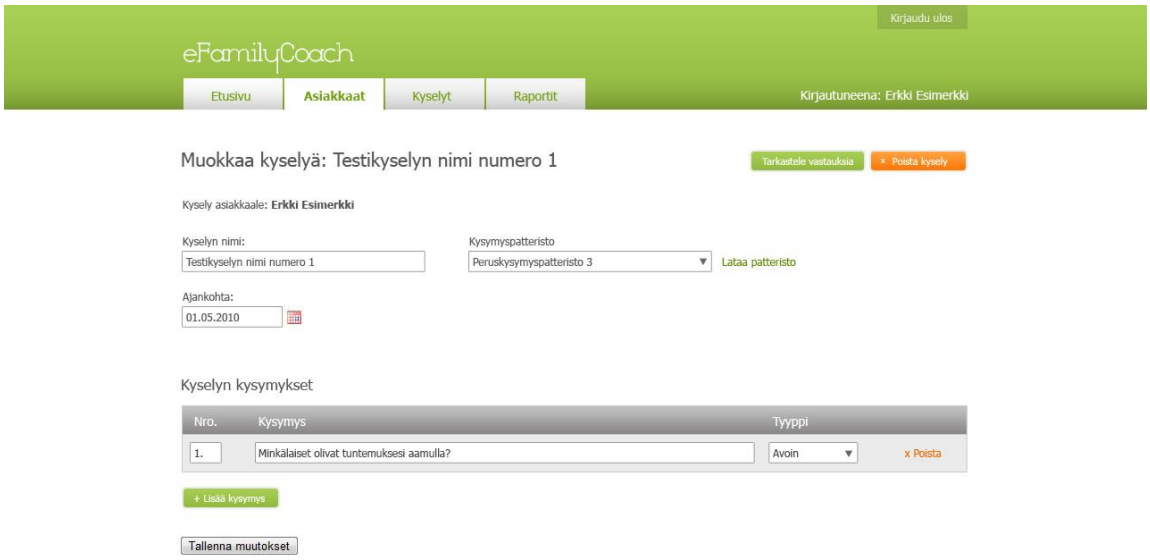
KUVIO 14 Kyselyt-välilehti

Kyselyn lisäykseen voidaan siirtyä sen jälkeen, kun kyselyn vastaanottava asiakas on valittu. Asiakkaan tiedot näkyvät lomakkeen yläreunassa ja muut tiedot täytetään lomakkeen kautta. Kyselyn kysymykset voidaan lisätä joko suoraan valmiista patteristosta tai lisäämällä kysymyksiä sivulle avautuneen lomakkeen avulla. (Kuvio 15.)



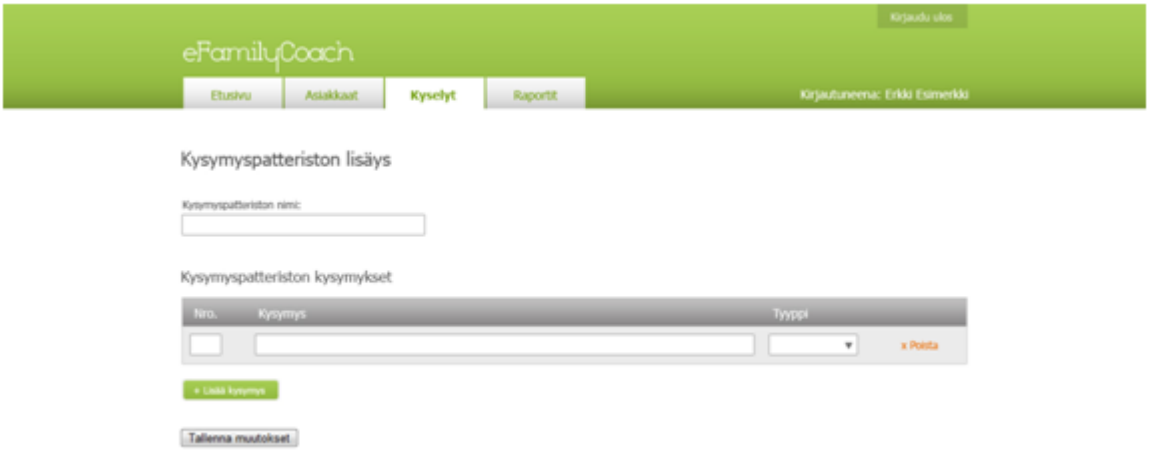
KUVIO 15 Kyselyn lisääminen

Valmista kyselyä voidaan muokata samanlaisella lomakkeella. Muokattavan kyselyn tiedot avautuvat lomakkeeseen, minkä jälkeen niitä voi muuttaa. (Kuvio 16.)



KUVIO 16 Kyselyn muokkaus

Kysymyspatteriston lisäys tapahtuu yksinkertaisemman lomakkeen kautta. Lomakkeessa nimetaan patteristo ja lisätään siihen haluttu määrä kysymyksiä. Uuden rivin kysymyksiä varten saa lisää kysymys-painikkeen avulla. (Kuvio 17.)



KUVIO 17 Kysymyspatteriston lisäys

3.3.6.1.4 Raportit

Raportit-välilehden sisältö ei prorotyypissä ole tärkeässä osassa, koska tuloksia tullaan tarkastelemaan erillisten ohjelmien avulla. Tuloksien esittämiseksi mahdollisimman selkeästi ja loogisesti, esimerkiksi niin, että niitä voidaan hyödyntää perhetyössä, tulee olemaan yksi jatkokehityksen osa-alue. Yksinkertaisimmillaan raportit-välilehdellä näkyvät lähetetyt kysymykset ja niihin saadut vastauksen asiakaskohtaisesti.

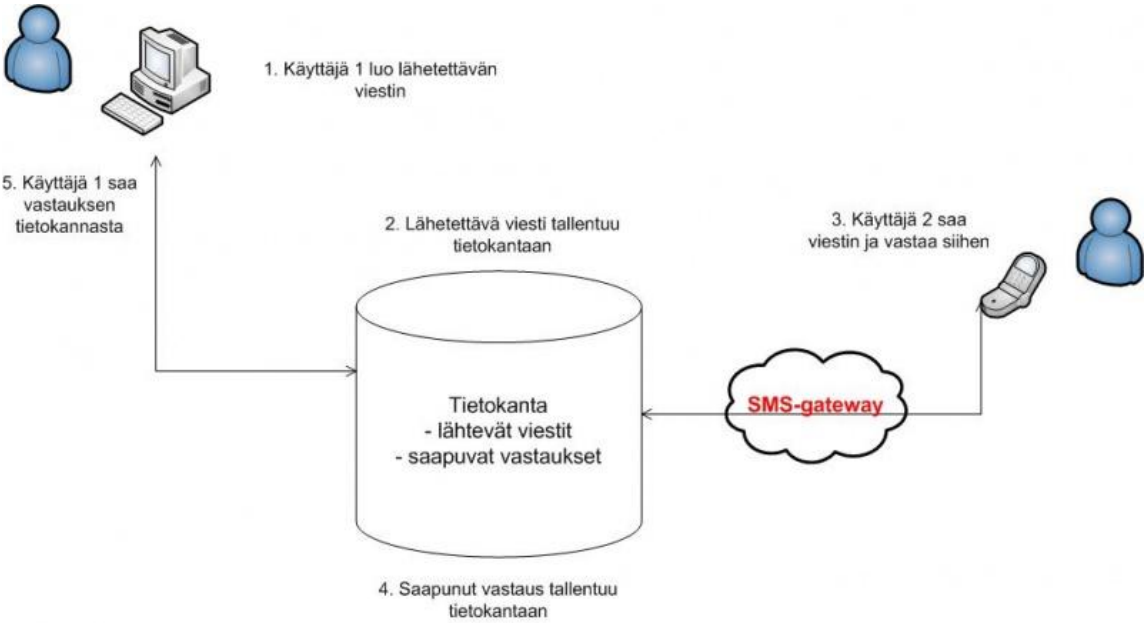
3.4 Mobiili

Vastuu prototyypin testauksessa ja ensimmäisessä pilotissa käytettävä SMS gatewaystä kuului projektin erilliselle osapuolelle. Käytetty gateway saatiin toimintaan projektin testausvaiheessa ja sen jälkeen ajoittain projektin edetessä. Valittu ratkaisu oli yksinkertainen tietylle koneelle käynnistettävä työkalu, joka lähetti Web-käyttöliittymän avulla lähetetyt viestit edelleen vastaanottajalle ja tallensi saapuneet viestit odottamaan, kunnes ne haettiin tietokantaan.

3.4.1 Toiminta

SMS gateway on viestinlähetyspalvelu, joka muuttaa muusta mediasta tulleen viestin mobiiliverkon liikenteeksi mahdollistaen tekstiviestien lähetyksen ilman mobiililaitetta. Viestinlähetyspalvelu käyttää Signalin System 7 -protokollaa, jonka avulla välitetään viesti mobiilioperaattorin SMS keskukseen (SMS-C). Palveluiden käyttöönotto on vaatinut useita sopimuksia operaattoreiden kanssa ja siksi palvelu on yleensä helpointa hankkia valmiina pakettina joltakin palveluntarjoajalta.

SMS-gateway toimii esimerkiksi käyttäjän tekemän sovelluksen, kuten Web-käyttöliittymän ja GMS-verkon välissä. Käytännössä Suomessa SMS gatewayn voi kytkeä mihin tahansa matkapuhelinverkkoon. Suomessa palvelua tarjoavilta yrityksiltä voi ostaa pääasiassa kahta kokonaisuutta: yksi- tai kaksisuuntaista SMS gatewayta. Yksisuuntainen gateway mahdollistaa viestin lähettämisen halutusta järjestelmästä haluttujen henkilöiden matkapuhelimiin, mutta viestiin ei odoteta vastausta. Mikäli viestin saanut asiakas vastaa saamaansa viestiin, lähetetään hänelle ilmoitus, joka kertoo virheestä. Kaksisuuntainen SMS gateway mahdollistaa viestiin vastaamisen, eli palvelun ostanut käyttäjä voi halutessaan esimerkiksi tallentaa vastausviestejä. Kaksisuuntainen gateway mahdollistaa myös laskutuksen viestiin vastanneelta asiakkaalta.



KUVIO 18 SMS gateway ja tietokanta

Suomessa ja monissa muissa maissa toimii useita SMS gateway palvelua tarjoavaa yritystä, joilta on mahdollista ostaa valmis palvelu tai sen osia. Yleisimmin palveluntarjoaja huolehtii tekstiviestien vastaanottamisesta, viestin muuntamisesta HTTP-kutsuksi, reitityksestä ja laskutuksesta. Ongelmana SMS-gatewayn käyttöönotossa esimerkiksi osana tietokantasovellusta voi olla se, että viestin lähetys käyttäen gatewayta on yleensä aina maksullista. Mikäli siis haluaa omaan sovellukseensa tekstiviestin lähetyksen mahdollistavan ominaisuuden, tulee joko sovelluksen omistajan tai käyttäjien maksaa palvelusta.

Joissakin palveluissa SMS gateway käyttää reititykseen viestin alussa olevaa tunnistetta, esimerkiksi jotakin merkkijonoa. Näissä tapauksissa samaa palvelua käyttää useampi tilaaja ja haluttu toiminto tai vastaus saadaa tunnistamalla viestin vastaanottaja viestin alkuun lisättävästä tunnisteesta. Suomessa tälläisiä palveluita on käytössä useita, esimerkiksi auton rekisterinumerolla auton omistajan tiedot lähettävä palvelu.

Yksinkertaisimmillaan SMS-gatewayn saa käyttöön ostamalla sen palveluna joltakin yritykseltä. Toinen mahdollisuus on yhdistää sim-kortti gatewayhin ja käyttää sitä tietokoneen avulla, jolloin voi määritellä käytettävän ohjelman avulla laajemmin haluttuja ominaisuuksia. Tälläinen käyttö vaatii sim-kortin, joka mahdollistaa viestien lähetyksen ainakin yleisimpiin matkapuhelimiin.

4 TIETOTURVA

4.1 Sovelluksen luonne

Prototyyppiä käytetään osana tutkimusta, jossa kerätään tietoa lasten arjen iloisista ja ikävistä hetkistä. Prototyyppiä suunniteltaessa on pyritty siihen, että asiakkaan henkilötietojen yhdistäminen kysymyksiin olisi mahdollistettu vain tutkijoille tai työntekijöille. Viestejä lähetettäessä ainoa tunnistetieto on asiakkaan puhelinnumero, mutta tietokantaan voi tarvittaessa tallentaa muitakin tietoja asiakkaasta. Tietokantaan tallennettavat viestit ja niiden vastaukset eivät sinäällään sisällä arkaluontoista tietoa, mutta tulevan jatkokehityksen ja laajemman käytön kannalta on tärkeää, että asiakkaat tietävät sisällön olevan vain heidän ja heitä hoitavan työtekijän välistä tietoa. Toinen huomioitava asia on se, että kaikki asi-

akkaat eivät välttämättä halua muiden tietävän heidän esimerkiksi olevan terapiahoidossa, eli ulkopuolisten ei tulisi voida nähdä asiakastietoja.

Seuraavissa kappaleissa kuvataan, kuinka tietoturva on pyritty hoitamaan prototyypin pilotointivaiheessa. Prototyyppiä suunniteltaessa tai toteutettaessa ei ole varsinaisesti voitu ottaa kantaa siihen, kuinka esimerkiksi palvelimen tietoturva tullaan hoitamaan jatkokehityksen aikana ja sen jälkeen varsinaisessa käytössä. Myöhemmin mahdollisesti kehitettävän varsinaiseen asiakaskäyttöön tulevan sovelluksen tietoturvan suunnittelu on siis osa jatkokehitystä.

4.2 Tietoturvaratkaisut

4.2.1 Palvelin

Prototyyppivaiheessa Web-käyttöliittymä ja tietokantapalvelin on sijoitettu Jyväskylän Ammattikorkeakoulun Labranet-verkkoon. Tietoturva on Labranetissä pyritty takaamaan palomuurisuojauksella ja sillä, että palvelintiloihin on pääsy vain rajatulla määrällä käyttäjiä. Verkon käyttöä monitoroidaan ja mikäli sovelluksessa huomataan tietomurto, sen aiheuttaja pyritään selvittäämään mahdollisimman pian. Labranet-verkossa olevista palvelimista huolehtivat henkilöt noudattavat tarkkoja tietoturvakäytänteitä, mikä on nähty riittäväksi valittaessa toteutusympäristöksi Labranet.

4.2.2 Tietokanta

Tietokanta on suunniteltu niin, että normaali käyttäjä, esimerkiksi tutkija, ei voi tehdä kantaan muutoksia kirjautumatta ensin sovellukseen. Tällä pyritään rajaamaan käyttäjiä ja samalla pystytään estämään ulkopuolisten henkilöiden pääsy kantaan ilman tunnuksia. Kirjautuneella käyttäjällä on pääsy ainoastaan niiden asiakkaiden tietoihin, joista hän on vastuussa, eli tietojen leviämisen ulkopuoliselle on riippuvainen siitä, kuinka hyvin käyttäjät huolehtivat tunnuksistaan.

Prototyyppivaiheessa tietokannan käyttö on sallittu vain kolmelle tutkijalle, jotka osallistuvat pilotointiin. Heidän lisäkseen kannalle tullaan luultavasti nimeämään

tietokannan hoitaja, jonka vastuulla on huolehtia kannan yleisestä toimivuudesta ja esimerkiksi turvallisuudesta. Asiakkaat yhdistetään viesteihin puhelinnumerolla, joka on ainakin prototyyppivaiheessa ainoa tunnistetieto. Myöhemmin saatu materiaali yhdistetään asiakkaisiin täysin erillisessä tutkimuskäytössä.

4.2.3 Mobiili

Tieton siirtäminen kannasta SMS-gatewayn kautta käyttäjän puhelimeen ja takaisin on yksi tietoturvariski, joka sovellukseen sisältyy. Kuitenkin lähetetyt kysymykset ja niiden vastaukset eivät sisällä muita tunnistetietoja, kuin asiakkaan puhelinnumeron ja tietoja käytetystä gatewaystä. Viestien sisältöä ei voida hyödyntää kovinkaan helposti verrattuna esimerkiksi henkilötietoihin, joten esimerkiksi tekstiviestin sisällön paljastuminen täysin ulkopuoliselle henkilölle on henkilön tietoturvan kannalta melko vaaratonta. Tietenkin esimerkiksi mahdollisessa myöhemmässä asiakaskäytössä olisi tärkeää, että viestien sisältökin olisi mahdollisimman salattua. Asiakkaiden rooli osana tietoturvaa nousee myös esiin tässä kohtaa, sillä helpoin keino selvittää viestien sisältö on lukea ne suoraan asiakkaan puhelimesta. Prototyyppivaiheessa tutkittavia lapsia kehotetaankin poistamaan lähetetyt viestit.

5 TYÖN JA TULOSTEN ARVIOINTI

5.1 Opinnäytetyön tekeminen

Opinnäytetyön tekeminen alkoi suunnittelemalla tietokantaa, jonka jälkeen siirryttiin vähitellen testauksen kautta toteutukseen. Tietokannan suunnittelun alkaessa käsitteitä oli vain muutama (asiakas, työntekijä, kysymys), mutta työn edetessä tietokanta laajeni käsittämään lähes 20 taulua. Suunnittelu sisälsi alusta asti paljon testausta, mikä aiheutti sen, että tietokanta muuttui useaan kertaan työn edetessä. Loppuvaiheessa oli kuitenkin jo täysin selvää, että hyvin suunniteltu tietokanta helpotti toteutusta ja syntynyttä tietokantaa voidaan hyödyntää ainakin osittain myös jatkokehityksessä.

Toteutusvaiheessa ongelmat koskivat enimmäkseen toiminnallisuuden toteutusta, eivätkä niinkään tietokantaa. Verrattaessa suunnittelu- ja toteutusvaiheita

huomattiin, että hyvä suunnittelu helpotti toteutusta ja jatkuvat testaus vähensi ongelmia loppuvaiheessa. Eniten aikaa työssä vei tietokannan suunnittelu ja muokkaus, jota tehtiin lähes koko projektin ajan, koska toteutusvaiheessakin huomattiin joitakin puutteita.

Kokonaisuutena suunnittelu ja toteutus veivät suunnilleen saman ajan, kun asioiden dokumentointi opinnäytetyöhön. Opinnäytetyö suunnittelun ja toteutuksen rinnalla selkeytti projektin etenemistä ja esimerkiksi ongelmien dokumentointia voitiin hyödyntää useita kertoja nopeuttamaan työn etenemistä, koska samoja ongelmia ei tarvinnut ratkaista useita kertoja.

5.2 Ongelmakohdat

Prototyypin suunnittelun ja toteutuksen aikana syntyneiden ongelmien kuvaukset on jaettu suunnittelun tai toteutuksen ongelmiin, eli ongelmat suunnittelussa-kappale käsittelee tietokantaan suunnitellessa syntyneitä ongelmia ja ongelmat toteutuksessa-kappale käyttöliittymää toteutettaessa syntyneitä ongelmia.

5.3 Ongelmat suunnittelussa

5.3.1 Puhelinnumeron sijoittaminen kantaan

Prototyyppi tehdään sovellukseen, jossa työntekijä kommunikoi asiakkaan kanssa mobiililaitteen välityksellä. Asiakkaasta ei kuitenkaan saa tallentaa tietokantaa juuri mitään muita tietoja, kuin puhelinnumeron. Ongelmana voitiin alkuvaiheessa pitää sitä, että asiakkaan puhelinnumeroita saattaa olla useita ja missään tapauksessa viestit eivät saa mennä väärään numeroon. Puhelinnumeron sijoittamista suunniteltaessa pidettiin yhtenä mahdollisena vaihtoehtona toteuttaa kokonaan erillinen taulu, johon numero sijoitettaisiin. Ajatuksesta kuitenkin luovuttiin, koska kannan käytön kannalta järkevintä oli sijoittaa puhelinnumero asiakas – tauluun ja näin jokaiselle asiakkaalle voidaan tallentaa vain yksi puhelinnumero, mikä vähentää mahdollisia sekaannuksia tai väärään numeroon lähetettävän viestin mahdollisuutta.

5.3.2 Asiakkaan ja vastauksen yhdistäminen

Suunnitteluvaiheessa eniten aikaa kului asiakkaan ja asiakkaan lähettämän vastauksen yhdistämisen suunnitteluun. Asia ei varsinaisesti ollut ongelma, mutta se oli suhteessa moneen muuhun yhteyteen eniten suunnittelua vaativa kohta. Ratkaisuksi valittiin asiakkaalle lähetetyn kysymyksen (client inquiry question) yhdistäminen asiakkaaseen ja vastaukseen. Asiakkaan ja vastauksen välille luotiin vielä informatiivinen yhteys ja toteutusvaiheessa kysymyksen ja vastauksen yhdistäminen toteutetaan tutkimalla lähetys- ja saapumisaikoja.

5.4 Ongelmat toteutuksessa

5.4.1 Group- taulu

Siirryttäessä kannan suunnittelusta toteutukseen huomattiin, että kannassa oleva taulu, joka määrittelee, mihin ryhmään asiakas kuuluu, aiheutti ongelmia scriptiä ajettaessa. Taulun nimi oli group, mikä on varattu sana, eikä taulua voitu luoda. Muutoksen jälkeen ryhmä-taulun nimeksi muutettiin section.

5.4.2 Send date

Tietokantaan ei alun perin suunniteltu tallennettavaksi tietoa siitä, milloin viestit lähetetään asiakkaalle, vaan ajastus oli tarkoitus toteuttaa sovelluslogiikassa. Käyttöliittymän toteutuksen edetessä huomattiin kuitenkin, että tiedon tallentaminen kantaan olisi hyödyllistä ja kysymyksen ja vastauksen yhdistäminen helpottuisi, mikäli tieto olisi kannassa. Send date – sarake lisättiin Question – tauluun. Myöhemmin työn edetessä huomattiin kuitenkin, että tietoja viestin lähetys- ja saapumisajankohdasta tarvitaan muissakin tauluissa. Ongelma ratkaistiin lisäämällä client inquiry question – tauluun sarake SentDate ja answer – tauluun sarake ReceivedDate.

5.4.3 MessageId

SMS gateway sisältää tietokannan, joka luo jokaiselle lähetetylle ja saapuneelle viestille Id:n, jonka perusteella voidaan myöhemmin tutkia esimerkiksi viestin

tilaa. Suunniteltuun tietokantaan ei aiemmin erikseen lisätty tietoa viestin Id:stä, mutta toteutuksen aikana se päätettiin lisätä kantaan, jotta viestien hallinta helpottuisi. Client Inquiry Question – tauluun lisättiin sarake SentMessagaID ja Answer- tauluun ReceivedMessageID.

5.5 Työskentely projektiryhmän ja toimeksiantajan kanssa

Opinnäytetyön toteutuksen aikana työskentelyyn osallistuivat projektiryhmän jäsenet, ohjaaja sekä toimeksiantajan edustajat. Projektiryhmän jäsenet pyrkivät toteuttamaan halutun ratkaisun ohjaajan tukemana ja toimeksiantajan edustajilta saatiin haluttujen ominaisuuksien kuvauksen lisäksi arvokasta palautetta testauksesta.

Projektiryhmä jäsenten kanssa työskentely onnistui alusta asti hyvin, ryhmä toimi hyvin yhdessä ja jäsenten kompetensseja voitiin hyödyntää ongelmanratkaisussa. Suunnitteluvaiheessa useamman henkilön osallistuminen selkeytti kokonaisuutta ja auttoi löytämään parhaat ratkaisut. Erilaiset näkökulmat asioihin toivat lopputulokseen hyviä ratkaisuja ja paransivat käytettävyyttä.

Ohjaaja ja toimeksiantajan edustajat toimivat hyvänä tukena prototyypin toteutuksessa ja testauksessa. Palavereissa saatiin rakentavaa palautetta, mikä helpotti prototyypin kehitystä. Toimeksiantajat suhtautuivat hyvin projektin toteutukseen opintojaksojen harjoitustyönä ja projektiryhmä sai hyvää kokemusta asiakastyöstä.

6 YHTEENVETO

Prototyypistä pyrittiin tekemään mahdollisimman selkeä ja käytettävä. Prototyyppiä tullaan käyttämään osana melko pientä tutkimusta ja sen avulla voidaan myös tutkia, kuinka käytetty tekniikka soveltuu tutkimukseen. Sovelluksesta olisi voinut kehittää ominaisuuksiltaan monipuolisemman ja lisätä siihen esimerkiksi enemmän testattavia ominaisuuksia, mutta testauksen kannalta helpompi ratkaisu oli toteuttaa selkeä ja melko yksinkertainen sovellus. Kokonaisuutena projekti oli kuitenkin melko monipuolinen ja siinä hyödynnettiin useampaa erilaista tekniikkaa.

Prototyypin kehitys aloitettiin tietokannan suunnittelusta, mikä oli hyvä lähtökohta myös testaukselle. Sovelluksen suunnittelu alkoi vaatimusmäärittelystä, joka olikin ensimmäinen projektin selkeästi kuvaava dokumentti. Ennen vaatimusmäärittelyä sovellusta ei oltu tarkasti määritelty, vaan tiedossa oli ainoastaan se, että asiakkaille tulisi voida lähettää viestejä ja vastauksia tulisi voida analysoida jollakin tavalla. Projektin etenemisen ja mahdollisimman yksinkertaisen ja selkeän sovelluksen kehityksen kannalta olikin ehkä hyvä, että määrittely oli alussa melko epämääräinen. Tietokannan suunnittelun edetessä ei jouduttu ongelmatilanteisiin esimerkiksi sen takia, että sovellukseen olisi haluttu jotakin tiettyjä, tarkasti määriteltyjä ominaisuuksia. Suunnittelun edetessä huomattiin mahdollisuuksia ja rajoituksia ja sitä kautta vältettiin ehkä joitakin ongelmia.

Käyttöliittymän kehitys aloitettiin tietokannan ollessa ensimmäisessä testausvaiheessa, joten testaus voitiin tehdä käyttöliittymälle ja kannalle yhtä aikaa. Tämän menettelytavan kautta käyttöliittymän ominaisuuksia saatiin kehitettyä tietokannan rinnalla ja pystyttiin jättämään erilaisia laajennusmahdollisuuksia jatkokehitystä ajatellen. Jatkokehityksessä voidaan siis hyödyntää sekä tietokantaa, että käyttöliittymää esimerkiksi testauksessa ja vaikka käyttöliittymä tullaankin luultavasti suunnittelemaan uudelleen, voidaan prototyypin ominaisuuksien avulla testata myös uusia ominaisuuksia. Erityisesti tietokanta on suunniteltu niin, että sitä voidaan suoraan hyödyntää jatkokehityksessä esimerkiksi lisäämällä mahdollisuuden ryhmitellä asiakkaita tai vaatia kyselyihin vastauksia oikeassa muodossa.

Keskeisimmäksi huomioksi työssä nousi tietokannan hyvä suunnittelu ja sen hyödyntäminen muiden ominaisuuksien testauksessa. Tietokanta muuttui koko prosessin ajan, mutta alussa hyvin suunnitellut taulut säilyivät käytössä koko toteutuksen ajan. Web-käyttöliittymä on sovelluksessa melko pieni osa verrattuna tietokantaan, jota luultavasti tullaan hyödyntämään enemmän jatkokehityksessä. Kokonaisuutena projekti antoi kuitenkin hyvän kuvan ohjelmistoprojektista, joka hyödyntää tietokantaa, web-käyttöliittymää ja mobiilitekniikkaa.

LÄHTEET

CERTAS. 2007. Viitattu 16.2.2010.
http://www.certas.com/dFeaturesBenefits.aspx.

Etaitava. 2010. Viitattu 16.2.2010.
http://etaitava.fi/fin/index.php?page=info.

FrontlineSMS. 2010. Viitattu 17.2.2010.
http://www.frontlinesms.com/what/.

Lehto, T. 2007.Kännykkä tukee työssäoppijaa. Arkistoitu artikkeli Etaitava -hankkeesta. Viitattu 16.2.2010.
http://www.tietokone.fi/uutiset/2007/kannykka_tukee_tyossaoppijaa.

OpenClinica. 2010. Viitattu 17.2.2010.

http://www.openclinica.org/section.php?sid=1.

OpenXdata. 2009. Viitattu 17.2.2010.
http://www.openxdata.org/Main/AboutopenXdata.

Perheiden hyvinvointi ja terveys. 2010. Viitattu 2.4.2010.
http://www.jamk.fi/yrityksille/sosiaalijaterveysala/perheidenhyvinvointi,

LIITTEET

Liite 1. Tietokannan luontiscripti

```
/*

Created 22.2.2010
Modified 23.4.2010
Project eFamilyCoach
Author Heikki Hyvönen, Leo-Matti Lehtonen, Laura Saari
Version 1.0
Database mySQL 5
*/
/* Asiakas-taulu, asiakkaan nimi, osoite ja puhelinnumero, perusavain ClientID */
Create table CLIENT (
 ClientID Int NOT NULL AUTO_INCREMENT,
 PostalCode Char(5) NOT NULL,
 ClientFirstName Varchar(100) NOT NULL,
 ClientLastName Varchar(100) NOT NULL,
 ClientAddress Varchar(100),
 ClientPhone Char(30),
 Primary Key (ClientID)) ENGINE = MyISAM;
 /* Asiakkaan hoito yms. tiedot, perusavain TreatmentClassID */
Create table TREATMENTCLASS (
 TreatmentClassID Int NOT NULL AUTO_INCREMENT,
 TreatmentClassDescription Longtext,
 Primary Key (TreatmentClassID)) ENGINE = MyISAM;
 /* Asiakkaan ja hoitotietojen yhdistäminen, kaksiosainen avain, clientID ja TreatmentClassID */
Create table CLIENT_TREATMENTCLASS (
 ClientID Int NOT NULL,
 TreatmentClassID Int NOT NULL,
 Primary Key (ClientID,TreatmentClassID)) ENGINE = MyISAM;
 /* Kaupunki, määrittää postinumeroa vastaavan postitoimipaikan, perusavain PostalCode, kaikki tiedot
pakollisia
*/
Create table CITY (
 PostalCode Char(5) NOT NULL,
 City Varchar(100) NOT NULL,
 Primary Key (PostalCode)) ENGINE = MyISAM;
 /* Työntekijän henkilö- ja kirjautumistiedot, pakollisia tietoja käyttäjätunnus, salasana, joista käyttäjä-
tunnuksen tulee olla uniikki. Perusavain EmployeeID, postinumeroa vastaava kaupunki haetaan City-
taulusta
*/
Create table EMPLOYEE (
 EmployeeID Int NOT NULL AUTO_INCREMENT,
 EmployeeFirstName Varchar(100),
 EmployeeLastName Varchar(100),
 EmployeeUserName Varchar(20) NOT NULL,
```

```
 EmployeePhone Varchar(20),
 EmployeeAddress Varchar(100),
 PostalCode Char(5) NOT NULL,
 EmployeePassword Char(32) NOT NULL,
 UNIQUE (EmployeeUserName),
 Primary Key (EmployeeID)) ENGINE = MyISAM;
/* Yksikkö, johon työntekijä voi kuulua (voi kuulua useampaankin), perusavain DepartmentID
*/
Create table DEPARTMENT (
 DepartmentID Int NOT NULL AUTO_INCREMENT,
 DepartmentName Varchar(100),
 DepartmentAddress Varchar(100),
 PostalCode Char(5) NOT NULL,
 Primary Key (DepartmentID)) ENGINE = MyISAM;
/* Työntekijän ja yksikön yhdistävä taulu, kasiosainen avian, DepartmentID ja employeeID
*/
Create table EMPLOYEE_DEPARTMENT (
 EmployeeID Int NOT NULL,
 DepartmentID Int NOT NULL,
 Primary Key (EmployeeID,DepartmentID)) ENGINE = MyISAM;
/* Työntekijän ja asiakkaan yhdistävä taulu, kaksiosainen avain, ClientID ja EmployeeID
*/
Create table CLIENT_EMPLOYEE_RELATIONSHIP (
 ClientID Int NOT NULL,
 EmployeeID Int NOT NULL,
 Primary Key (ClientID,EmployeeID)) ENGINE = MyISAM;
/* Kysymysten tallentamiseen tarkoitettu taulu, perusavain QuestionID, pakollisia tietoja kysymys ja
kysymystyyppi, joka haetaan questionType-taulusta
*/
Create table QUESTION (
 QuestionID Int NOT NULL AUTO_INCREMENT,
 QuestionType Varchar(100) NOT NULL,
 Question Varchar(255) NOT NULL,
 SendDate Datetime,
 Primary Key (QuestionID)) ENGINE = MyISAM;
/* Kysymystyyppi, perusavain QuestionType, eli tyypin nimi, pakollinen tieto Required, joka määrittelee,
odotetaanko vastausta
*/
Create table QUESTIONTYPE (
 QuestionType Varchar(100) NOT NULL,
 Required Bool NOT NULL,
 UNIQUE (QuestionType),
 Primary Key (QuestionType)) ENGINE = MyISAM;
/* Kysely-taulu, tiedot kysymyksiä sisältävästä kyselystä, perusavain InquiryID, pakollisia tietoja kyselyn
nimi, muokkaupäivämäärä ja arvo Isquestionnaire, joka määrittelee, onko kysely patteristo vai lähe-
tettävä kysely
*/
Create table INQUIRY (
 InquiryID Int NOT NULL AUTO_INCREMENT,
 InquiryName Varchar(250) NOT NULL,
 IsQuestionnaire Bool NOT NULL,
 ModifiedDate Datetime NOT NULL,
 Primary Key (InquiryID)) ENGINE = MyISAM;
/* Kyselyn ja kysymyksen yhdistävä taulu
*/
Create table INQUIRY_QUESTION (
 QuestionID Int NOT NULL,
 InquiryID Int NOT NULL,
 QuestionNumber Int,
```

```
 Primary Key (QuestionID,InquiryID)) ENGINE = MyISAM;
/* Validi vastaus, taulua käytetään tarkastamaan, onko haluttu vastaus oikeantyyppinen,   esimerkiksi numero. Tietylle kysymystyypille on siis yksi tai useampi validi vastaus.Perusavain ValidAnswer, eli vastaus
*/
Create table VALIDANSWER (
 ValidAnswer Char(20) NOT NULL,
 QuestionType Varchar(100) NOT NULL,
 Primary Key (ValidAnswer,QuestionType)) ENGINE = MyISAM;
/* Versionhallinta */
Create table VERSIO (
 VersioID Int NOT NULL AUTO_INCREMENT,
 Versio Varchar(20) NOT NULL,
 Changes Varchar(250),
 User Varchar(50) NOT NULL,
 Date Datetime NOT NULL,
 Primary Key (VersioID)) ENGINE = MyISAM;
/* käyttäjäryhmä */
Create table SECTION (
 SectionID Int NOT NULL AUTO_INCREMENT,
 SectionName Varchar(100) NOT NULL,
 Primary Key (SectionID)) ENGINE = MyISAM;
/* käyttäjäryhmän ja asiakkaan yhdistävä taulu */
Create table CLIENT_SECTION (
 SectionID Int NOT NULL,
 ClientID Int NOT NULL,
 Primary Key (SectionID,ClientID)) ENGINE = MyISAM;
/* vastaus  moniosainen avain muodostuu kysymys-, kysely- ja asiakas-Id:stä receivedMessageID saadaan gatewaylta
*/
Create table ANSWER (
 Answer Varchar(160) NOT NULL,
 QuestionID Int NOT NULL,
 InquiryID Int NOT NULL,
 ClientID Int NOT NULL,
 ReceivedMessageID Varchar(255),
 ReceivedDate Datetime,
 Primary Key (QuestionID,InquiryID,ClientID)) ENGINE = MyISAM;
/* asiakkaan, kysymyksen ja vastauksen yhdistävä taulu, moniosainen avain muodostuu sarkkeista QuestionID, InquiryID ja ClientID. SentMessageID saadaan gatewaylta
*/
Create table CLIENT_INQUIRY_QUESTION (
 QuestionID Int NOT NULL,
 InquiryID Int NOT NULL,
 ClientID Int NOT NULL,
 SentMessageID Varchar(255),
 SentDate Datetime,
 Primary Key (QuestionID,InquiryID,ClientID)) ENGINE = MyISAM;


Alter table CLIENT_TREATMENTCLASS add Foreign Key (ClientID) references CLIENT (ClientID) on delete restrict on update restrict;
Alter table CLIENT_EMPLOYEE_RELATIONSHIP add Foreign Key (ClientID) references CLIENT (ClientID) on delete restrict on update restrict;
Alter table CLIENT_SECTION add Foreign Key (ClientID) references CLIENT (ClientID) on delete restrict on update restrict;
Alter table CLIENT_INQUIRY_QUESTION add Foreign Key (ClientID) references CLIENT (ClientID) on delete restrict on update restrict;
Alter table CLIENT_TREATMENTCLASS add Foreign Key (TreatmentClassID) references TREATMENTCLASS (TreatmentClassID) on delete restrict on update restrict;
```

```
Alter table CLIENT add Foreign Key (PostalCode) references CITY (PostalCode) on delete restrict on update restrict;
Alter table EMPLOYEE add Foreign Key (PostalCode) references CITY (PostalCode) on delete restrict on update restrict;
Alter table DEPARTMENT add Foreign Key (PostalCode) references CITY (PostalCode) on delete restrict on update restrict;
Alter table EMPLOYEE_DEPARTMENT add Foreign Key (EmployeeID) references EMPLOYEE (EmployeeID) on delete restrict on update restrict;
Alter table CLIENT_EMPLOYEE_RELATIONSHIP add Foreign Key (EmployeeID) references EMPLOYEE (EmployeeID) on delete restrict on update restrict;
Alter table EMPLOYEE_DEPARTMENT add Foreign Key (DepartmentID) references DEPARTMENT (DepartmentID) on delete restrict on update restrict;
Alter table INQUIRY_QUESTION add Foreign Key (QuestionID) references QUESTION (QuestionID) on delete restrict on update restrict;
Alter table QUESTION add Foreign Key (QuestionType) references QUESTIONTYPE (QuestionType) on delete restrict on update restrict;
Alter table VALIDANSWER add Foreign Key (QuestionType) references QUESTIONTYPE (QuestionType) on delete restrict on update restrict;
Alter table INQUIRY_QUESTION add Foreign Key (InquiryID) references INQUIRY (InquiryID) on delete restrict on update restrict;
Alter table CLIENT_INQUIRY_QUESTION add Foreign Key (QuestionID,InquiryID) references INQUIRY_QUESTION (QuestionID,InquiryID) on delete restrict on update restrict;
Alter table CLIENT_SECTION add Foreign Key (SectionID) references SECTION (SectionID) on delete restrict on update restrict;
Alter table ANSWER add Foreign Key (QuestionID,InquiryID,ClientID) references
CLIENT_INQUIRY_QUESTION (QuestionID,InquiryID,ClientID) on delete restrict on update restrict;
/* Indeksit */
/* SELECT ClientID, ClientFirstName, ClientLastName, ClientPhone, ClientAddress, PostalCode FROM
CLIENT WHERE PostalCode = '40100'  ORDER BY ClientLastName;
*/
CREATE INDEX i_PostalCode_ClientLastName_ClintID_ClientFirstName_ClientPhone
ON CLIENT(ClientID, ClientFirstName, ClientLastName, ClientPhone, ClientAddress, PostalCode)
```