Kevät 2015

Tietokantaohjelmoinnin harjoitustyö
Vaihe II

Emmi Siitonen & Eetu Suonpää
Ryhmä 26

Sisällysluettelo

Yleistä....1
Ohjelman ominaisuudet....1
Kuvaus toteutuksesta....1
Ohjelman käytöstä....2
Kirjautuminen:....2
Opiskelijan puoli:....2
Tuutorin puoli:....3
Ylituutorin puoli:....3
Muutokset 1. vaiheeseen....3
Oma arvio....4
Työnjako....4
Mikä oli vaikeaa?....4
Mitä puutteita jäi?....4

Yleistä

Työssä yhdistyvät tietokantaohjelmoinnin ja www-ohjelmoinnin harjoitustyöt, ja sen takia harjoitustyön pohjana on käytetty www-ohjelmoinnin kurssiframeworkkia.

http://www.sis.uta.fi/~es98580/harjoitustyo/index.php

Ohjelman ominaisuudet

Harjoitustyöhön on toteutettu käyttäjäryhmien oikeuksien valvontaa, eli käyttäjät pääsevät käyttämään vain niitä palvelun osa-alueita jotka ovat heille tarkoitettuja. Palveluun kirjaudutaan sisään. Palvelussa opiskelijat voivat täyttää hops-lomakkeita ja tarkkailla aikaisemmin palautettuja lomakkeita ja opintosuorituksia. Opettajat voivat lähettää raportteja ja seurata opiskelijoidensa tietoja. Ylituutorit voivat kaiken edellämainitun ohella lisätä ja poistaa tuutoreita ja hopsryhmiä, muodostaa omia raporttejaan ja jakaa opiskelijoita ryhmiin. Kaikki käyttäjäryhmät voivat myös muokata omia henkilötietojaan.

Käyttöliittymäominaisuudet on toteutettu laajasti (käytetty paljon html5:sta ja CSS:ää). Master templatin avulla on saatu aikaiseksi sulava ja yhtenäinen käyttöliittymä.

Testiaineistoa on saatavilla laajalti (ks. erillinen tiedosto) testauksen helpottamiseksi.

Lisätoiminnoiksi olemme listanneet mm. Master Templaten, Frameworkin ja RBACin käytön koska vaikka niitä ei tehtävänannossa vaadittu, koimme niiden olevan tärkeitä osa-alueita.

Harjoitustyössä on toteutettu myös kaikki raportit R1-R6.

Kuvaus toteutuksesta

Harjoitustyö on toteutettu PHP-kielellä, käyttäen WWW-ohjelmoinnin tekniikka -kurssilta saatua frameworkia. Tarkempi framework-dokumentaatio löytyy osoitteesta www.www-programming-with-php.com viikkoharkkojen luota, joten sitä ei tässä dokumentaatiossa käsitellä kovinkaan tarkasti.

Framework (lyh. FW) käyttää MVC-mallia, joten harjoitustyö on toteutettu mallin mukaan. Järjestelmä koostuu pääosin neljästä FW-liitännäisestä, jotka ovat account, hops, profile ja login (kansiossa content). Account hoitaa kirjautumisen tarkistuksen, sekä opettajan lisäyksen (”rekisteröi” opettajan) ja login hoitaa kirjautumisen. Nämä kaksi löytyvät FW:stä itsestään, joten niihinkin dokumentaatio on tarkemmin em. sivulla.

Hops-liitännäinen on harjoitustyön ydin ja hoitaa kaiken hops-järjestelmän toiminnallisuuden. Profile hoitaa käyttäjän profiiliin liittyvät asiat, esim. tietojen katselun ja päivityksen, sekä opettajan poistamisen tietokannasta. Kummatkin liitännäiset koostuvat kolmesta osasta; kontrollereista, malleista ja näkymistä. Esimerkiksi hops-liitännäisen

kansiossa on jokaiselle omat kansiot, sekä front controller, joka ohjaa liitännäisen toimintaa. Front controller hakee tarvittavat kontrollerit ”controller”-kansiosta ja kutsuu halutun kontrolleriluokan execute()-metodia. Metodi lukee URL-parametrin action (tai view, joka on action-parametrin alias), ja kutsuu reflektoiden parametrin nimistä funktiota. Funktiossa (esim. display(), joka on oletusmetodi, mikäli parametria ei annettu) haetaan ”model”-kansiosta haluttu malli, tehdään mallilla tarvittava datan haku/käsittely ja annetaan se ”view”-kansiosta haetulle näkymälle. Tämän jälkeen kutsutaan näkymän display()-metodia, jolle voi antaa parametrin. Näkymä taas hakee joko default -tai parametrin nimisen esitettävän php-tiedoston tmpl-alikansiosta. Tällä tavoin toimivat muutkin liitännäiset, pienillä eroilla esim. haettuihin malleihin ja näkymiin.

Käytetty FW tarjoaa monia valmiita palveluita, joita harjoitustyössä käytetään. Tietokanta toimii FW:n omalla PDO-oliolla, joka haetaan tehdasmetodikutsulla SomeFactory::getDBO(). Tämän avulla tehdään melko perinteisiä tietokantakyselyitä. Ainoastaan account-liitännäisessä, uutta käyttäjää luotaessa, hyödynnetään FW:n tarjoamaa Active Record-mallia, jossa tietokannan perustoimintoja (CRUD + List) saadaan helpotettua. Toinen paljon käytetty FW:n tarjoama ominaisuus on sen käyttäjäominaisuus, jolla mm. tarkistetaan käyttäjän oikeuksia erinäisissä kohdissa. Kirjautunut käyttäjä saadaan kutsulla SomeFactory::getUser(), joka palauttaa SomeUser-luokan singleton -olion. Käyttäjille on määritelty tietokantatauluun id:t ja käyttäjätunnukset, sekä rooli, joka voidaan tarkistaa vertaamalla SomeUser-luokan staattisiin roolimuuttujiin. Täyttä potentiaalia ei saavuteta, sillä FW tarjoaisi erikseen RBAC (Role Based Access Control) -tyyppisen ratkaisun, jota ei liiallisten tietokantamuutosten takia otettu käyttöön.

Ohjelman käytöstä

Kirjautuminen:

Palveluun kirjaudutaan joko opiskelija- tai tuutoritunnuksella (esim. 95776 tai 104567). Salasana on testauksen helpottamiseksi määritetty kaikille samaksi, ja se on 'salasana'. Kirjautumisen jälkeen palvelu ohjaa käyttäjän hänelle määrätylle etusivulle.

Opiskelijan puoli:

Opiskelijat ohjataan etusivulle jossa häntä tervehditään nimellä. Yläpalkista löytyvät vaihtoehdot '*Etusivu*', '*Omat tiedot*', '*Hops-lomakkeet*' ja '*Suoritukset*'. '*Etusivu*' ohjaa luonnollisesti käyttäjän etusivulle (jolla hän tässä vaiheessa on) ja '*Omat tiedot*' taas vie hänet käyttäjän omaan profiiliin. Siellä hän voi tietojen lukemisen lisäksi halutessaan muokata yhteystietojaan painamalla 'muokkaa' painiketta. Muutokset tallentuvat tietokantaan, kun käyttäjä painaa 'tallenna' nappulaa.
'*Hops-lomakkeet*' välilehden alta löytyvät linkit käyttäjän jo aiemmin täyttämiin lomakkeisiin ja mahdolliseen yhä avoimeen lomakkeeseen. Linkeistä opiskelija voi käydä katsomassa palautettujen hopsien sisältöä ja kurssisuunnitelmiaan. Lomaketta täyttäessään häntä pyydetään kirjaamaan suunniteltujen kurssiensa lisäksi jonkinlaista palautetta

edelliseltä vuodelta ja tieto siitä meinaako opiskelija käydä töissä opintojen ohella ja jos kyllä, niin millaisissa. Lomake lähetetään tallenna painiketta painamalla.
'*Suoritukset*' välilehden alta löytyvät tiedot opiskelijan jo tähän mennessä loppuun asti suoritetuista kursseista arvosanoineen ja opintopisteineen. Samalla opiskelija myös näkee kuinka paljon hänellä on kertynyt opintopisteitä milloinkin ja mistä kursseista.

Uloskirjautuminen onnistuu sivun oikean ylälaidan ”Kirjaudu ulos” painikkeella.

Tuutorin puoli:

Kuten opiskelijat, myös tuutorit ohjataan etusivulle tervehdyksen kera. Yläpalkissa on valikot '*Etusivu*', '*Omat tiedot*', '*Hops-ryhmät*' ja '*Raportit*', joista '*Etusivu*' ja '*Omat tiedot*' toimivat samoin, kuten opiskelijan näkymässä. '*Hops-ryhmät*' -valikosta tulee näkyviin kyseisen tuutorin hops-ryhmät vuositason mukaan. Ryhmiin kuuluvien opiskelijoiden opiskelijatunnukset, nimet, sekä linkit opiskelijoiden profiileihin on listattu ryhmien alle. Profiili-linkeistä pääsee tarkastelemaan yksittäisen opiskelijan profiilia, jossa tuutori voi myös muokata opiskelijan tietoja.

'*Raportit'*-valikosta klikatessa esiin tulee näkymä, josta tuutori voi valita haluamansa raportin. Tuutorille valittavana ovat 3. vuoden opiskelijoiden sähköpostiosoitteet, sekä lukukauden päätösraportti. Sähköpostiraportti listaa kolmannen vuoden opiskelijoiden nimet, tunnukset, sekä sähköpostit. Päätösraportissa on listattu jokaisen ryhmän kohdalle päätösraportti ja sivun alalaidassa on ”Lähetä raportti” -painike, joka tallettaa raportin tietokantaan.

Ylituutorin puoli:

Ylituutori-näkymään kirjaudutaan kuten muihinkin näkymiin. Näkymän valikot ovat '*Etusivu*', '*Omat tiedot*', '*Tuutorit ja hops-ryhmä*', sekä '*Raportit*'. Kaksi ensimmäistä toimivat jälleen opiskelijanäkymässä kuvatulla tavalla. '*Tuutori ja hops-ryhmät*'-välilehti listaa kaikki ylituutorin alaisuudessa toimivat tuutorit, heidän hops-ryhmänsä opiskelijoineen, sekä opiskelijat, joilla ei ole vielä hops-ryhmää. Valikossa on mahdollista katsella opiskelijoiden ja tuutoreiden profiileja, sekä muokata niitä. Lisäksi ylituutori voi lisätä uuden opettajan, sekä uuden ryhmän, jolle annetaan ryhmätunnus ja määrätään opettaja. Opiskelijoiden ryhmiin allokointi tapahtuu erillisessä näkymässä, jossa jokaiselle opiskelijalle, kenellä ei vielä ole hops-ryhmää, voidaan sellainen määrätä.

'*Raportit'*-valikossa on mahdollista katsella niin ylituutorin omia, kuin tuutoreiden raportteja. Ylituutori voi katsoa tuutoreiden tuutoroitavien lukumääriä, sekä hops-ryhmien koossapysymistä, tai katsella tuutoreiden lähettämiä raportteja.

Muutokset 1. vaiheeseen

Hops-ryhmästä poistettiin sarake alkup_koko. Se tuli turhaksi kun alkuperäinen koko pitikin päätellä ekan hopsin palauttaneiden lukumäärästä.

Opiskelija-taulusta poistettiin sarakkeet oppisteet ja status koska ne olivat turhia.

Osallistuu taulukon pvm sarake jakaantui kahteen osaan → vuosi ja kausi, jotta kyselyiden

tekeminen ajankohdan perusteella olisi helpompaa. Sama tehtiin myös aikoo_suorittaa ja on_suorittanut taulujen pvm sarakkeille.

Hops-pohjaan primary keyksi muodostui lopulta vuositason ja lukuvuoden yhdistelmä. Tämä johtaa myös siihen, että hops_lomakkeenkin primary key muodostuu nyt opnrosta, vuositasosta ja lukuvuodesta.

Oma arvio

Työ ei ollut kovinkaan vaikea, mutta aiempia harjoitustöitä reilusti laajempi. Tämä saattaa olla paikoitellen nähtävissä koodia lukiessa, sillä aikaa siistimiseen ja optimointiin ei juuri jäänyt.

Työnjako

Työnjako meni suhteellisen tasapuolisesti. Kummatkin tekivät ulkoasua ja ohjelmointi sujui pääosin pariohjelmointina. Selkein työnjako oli ehkä siinä, että Emmi teki enemmän tietokantaan liittyviä asioita siinä missä Eetu hoiti muita koodillisia asioita ja frameworkiin liittyviä asioita.

Mikä oli vaikeaa?

Eniten vaikeuksia aiheutti mm. frameworkin asioiden sisäistäminen, sekä tietokantanäkymien muodostaminen. Selkeästi suurin ongelma oli kuitenkin suhteellisen epäselvä tehtävänanto, joka tuntui erittäin hajanaiselta. Osa asioista esiteltiin hops-toiminnan yleisessä kuvauksessa, muttei vaadittu valmiissa toteutuksissa, joka vaikutti erittäin epäloogiselta. Tehtävänanto oli myös yleisellä tasolla melko sekava, eikä kaikkia vaadittuja asioita kunnolla kerrottu. Tästä seuraa se, että emme ole edelleenkään täysin varmoja olemmeko toteuttaneet harjoitustyössä kaikki vaaditut osa-alueet. Toisaalta se on myös hyvää harjoitusta tulevaisuutta varten, sillä kaikki asiakkaat eivät varmasti osaa tuoda omia ajatuksiaan ja haluamiaan vaatimuksia heti esille.

Mitä puutteita jäi?

Työhön ei juurikaan jäänyt puutteita, ei ainakaan kovin selviä sellaisia. Suurimmat ja selvimmät puutteet ovat framework-puolen puutteita, eivätkä niinkään liity tämän kurssin harkkatyöhön. Harjoitustyön vaatimuksista jäi puuttumaan historiatietojen tallennus, sekä samanaikainen käyttö (jonka toimivuutta ei testattu, joten oletetaan, ettei se toimi).