Tietokantasovellus

Toteutusdokumentti

Salon Lulun ajanvarausjärjestelmä + hallintapaneeli

Mikko Nylén [mikko.nylen@gmail.com](mailto:mikko.nylen@gmail.com)

Helsingin yliopisto, tietojenkäsittelytieteen laitos

Ohjaaja: Tia Määttänen

Sisällysluettelo

[Johdanto 4](#_Toc239143396)

[1.1 Toimintaympäristö 4](#_Toc239143397)

[1.2 Muutokset suunnitteludokumenttiin 4](#_Toc239143398)

[2 Ohjelmiston yleisrakenne 5](#_Toc239143399)

[2.1 Ajanvarausjärjestelmä 5](#_Toc239143400)

[2.2 Hallintapaneeli 6](#_Toc239143401)

[3 Järjestelmän komponentit 7](#_Toc239143402)

[3.1 Ajanvarausjärjestelmä 7](#_Toc239143403)

[3.1.1 Varausprosessivalidointi 7](#_Toc239143404)

[3.1.2 Aloitussivu 7](#_Toc239143405)

[3.1.3 Palvelun valintalomake 8](#_Toc239143406)

[3.1.4 Työntekijän- ja ajan valintasivu 8](#_Toc239143407)

[3.1.5 Yhteystietolomake 10](#_Toc239143408)

[3.1.6 Vahvistussivu 11](#_Toc239143409)

[3.1.7 Yhteenvetosivu 11](#_Toc239143410)

[3.1.8 Varauksen peruutussivu 12](#_Toc239143411)

[3.1.9 Varauksen peruutuksen vahvistussivu 13](#_Toc239143412)

[3.1.10 Varauksen peruutuksen onnistuminen -ilmoitussivu 13](#_Toc239143413)

[3.2 Hallintapaneeli 14](#_Toc239143414)

[3.2.1 Kirjautumissivu 14](#_Toc239143415)

[3.2.2 Hallintapaneelin etusivu 14](#_Toc239143416)

[3.2.3 Työntekijöiden hallinta 15](#_Toc239143417)

[3.2.4 Työntekijän lisääminen 15](#_Toc239143418)

[3.2.5 Työntekijän tiedot 16](#_Toc239143419)

[3.2.6 Työvuoron lisäys 17](#_Toc239143420)

[3.2.7 Työvuoron poisto 18](#_Toc239143421)

[3.2.8 Työntekijän työlista 18](#_Toc239143422)

[3.2.9 Palvelulistaus 19](#_Toc239143423)

[3.2.10 Palvelun lisäys / muokkaus 20](#_Toc239143424)

[3.2.11 Viikkoraportti 21](#_Toc239143425)

[4 Tietokanta 22](#_Toc239143426)

[4.1 Tallennetut proseduurit 25](#_Toc239143427)

[4.1.1 gen\_shifts() returns integer 25](#_Toc239143428)

[4.1.2 gen\_sess\_id() returns char(32) 26](#_Toc239143429)

[4.1.3 gen\_ cancel\_key() returns char(5) 26](#_Toc239143430)

[5 Asennusohje 27](#_Toc239143431)

[5.1 Asennushakemiston luominen 27](#_Toc239143432)

[5.2 Tietokannan pystyttäminen 27](#_Toc239143433)

[5.2.1 Hallintapaneelin sisäänkirjautumistietojen lisääminen 28](#_Toc239143434)

[5.3 Asetusten säätö 28](#_Toc239143435)

[5.3.1 $SETTINGS[’time\_period\_duration’] 28](#_Toc239143436)

[5.3.2 $SETTINGS[’reservation\_timeout’] 28](#_Toc239143437)

[5.3.3 $SETTINGS[’session\_timeout’] 28](#_Toc239143438)

[5.4 public\_html näkyväksi www-palvelimelle 28](#_Toc239143439)

[5.5 include\_path:n asettaminen 28](#_Toc239143440)

[6 Testiasennus 30](#_Toc239143441)

# Johdanto

Tässä dokumentissa kuvataan kesän 2009 tietokantasovellus –kurssille toteutettua Salon Lulu –parturikampaamon ajanvarausjärjestelmää ja siihen liittyvää, kampaamon työnjohdon käyttöön tarkoitettua, hallintapaneelia.

Asiakaspuolelle näkyvän ajanvarausjärjestelmän tarkoituksena on tarjota mahdollisuus ajanvarauksen tekemiseen käyttäjän valitsemalle parturille/kampaajalle ja ajankohdalle. Asiakas valitsee haluamansa toimenpiteen (“palvelu”), parturin/kampaajan sekä ajankohdan yksinkertaisen “ajanvarausvelhon” avulla sekä syöttää tämän jälkeen yhteystietonsa mahdollisia yhteydenottoja tai lisäselvityksiä varten. Varauksen jättämisen lisäksi asiakkaiden käytössä on ajanvarauksen peruutuslomake, jolla aiemmin tehdyn ajanvarauksen voi peruuttaa.

Salon Lulun työnjohtoa varten on toteutettu asiakkailta piilotettu hallintapaneeli. Hallintapaneelista voidaan hallita työntekijöitä, työvuoroja, palveluita sekä työntekijöiden henkilökohtaisia palveluvalikoimia. Lisäksi hallintapaneelista voidaan tulostaa päivittäiset työlistat henkilöstölle ja tutkia ajanvarausjärjestelmän viikottaisia tilastotietoja.

## Toimintaympäristö

Ajanvarausjärjestelmä ja hallintapaneeli toimivat samassa ympäristössä ja ovat toteutettu samoilla tekniikoilla.

Vaatimukset suoritusympäristölle ovat seuraavat:

* PHP 5.3.0 tai uudempi
* PostgreSQL 8.2 tai uudempi
* HTTP-palvelin (Apache tms.)

Harjoitustyö ei ole riippuvainen alla olevasta käyttöjärjestelmästä, joten se voidaan ottaa käyttöön millä tahansa yllä olevat vaatimukset täyttävällä järjestelmällä.

Harjoitustyön käyttöönotto ja toimivuus on testattu seuraavilla kokoonpanoilla:

* Windows 2008 Server, Apache 2.2.11, PHP 5.3.0, PostgreSQL 8.3.7
* Fedora Core 10, Apache 2.2.9, PHP 5.3.0, PostgreSQL 8.2.13

## Muutokset suunnitteludokumenttiin

Muutoksia suunnitteludokumentissa määriteltyihin ominaisuuksiin tai rajoituksiin ei ole tehty toteutusvaiheessa.

Työ on toteutettu kokonaisuudessaan suunnitteludokumentin mukaisesti.

# Ohjelmiston yleisrakenne

## C:\Users\Administrator\Desktop\Ohjelmiston yleisrakenne.pngAjanvarausjärjestelmä

## C:\Users\Administrator\Desktop\Ohjelmiston yleisrakenne, hallintapaneeli.pngHallintapaneeli

Jotkin sivut edellyttävät tietyn tiedon löytymistä kannasta. Jos tätä tietoa ei löydy, näytetään käyttäjälle ”404 – Page Not Found” –ilmoitus.

# Järjestelmän komponentit

Tässä on kuvattu ainoastaan www-selaimella haettavissa olevat sivut. Käytetyt kirjastot ja muut apuohjelmat on kuvattuna liitteessä 2 olevassa kuvauksessa.

## Ajanvarausjärjestelmä

Kaikki ajanvarausjärjestelmän sivut jakavat saman navigoinnin, joka on sivun yläosassa. Navigoinnista pääsee seuraaville sivuille:

* Aloitussivu ”Home” (index.php)
* Uuden varauksen aloitussivu ”Make reservation” (selectService.php)
* Varauksen peruutussivu ”Cancel reservation” (cancelReservation.php)

### Varausprosessivalidointi

Jokaisessa varauksen vaiheessa tulee suorittaa sarja tarkistuksia, jotta varausprosessi voidaan saattaa varmuudella luotettavasti loppuun.

1. Tarkistetaan, missä vaiheessa ollaan. Jos joitakin vaiheita puuttuu välistä, uudelleenohjataan käyttäjän selain puuttuvaan vaiheeseen.
2. Jos istuntomuuttuja *reservation.step1* on asetettu, varmistetaan, että varauksen ensimmäisessä vaiheessa valittu palvelu löytyy yhä tietokannasta. Jos näin ei ole, näytetään virheilmoitus, jossa selitetään tilanne asiakkaalle.
3. Jos istuntomuuttuja *reservation.step2* on asetettu, varmistetaan, että varauksen toisessa vaiheessa valittu työntekijä löytyy yhä tietokannasta ja että työntekijä on yhä käytettävissä valittuna aikana. Jos näin ei ole, näytetään virheilmoitus, jossa selitetään tilanne asiakkaalle.

### Aloitussivu

**Kontrolleri:** public\_html/index.php

Näyttää templaten.

**Template:** templates/index.php

Staattinen HTML-sivu, jossa toivotetaan käyttäjä tervetulleeksi Salon Lulun www-sivustolle. Ei ota muuttujia.

### Palvelun valintalomake

**Lomakemalli:** forms/ServiceSelectionForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **service** | int | Valitun palvelun ID-tunniste. |

Validoitaessa varmistetaan kentän *service* määrittämän palvelun löytyvän tietokannasta.

**Kontrolleri:** public\_html/selectService.php

Validoi ajanvarausprosessin sivua noudettaessa. Jos kontrolleria pyydettiin GET-pyynnöllä, näytetään templaten. Muussa tapauksessa validoidaan lomake.

Olettaen, että lomakkeen tarkistuksessa ei esiintynyt virheitä, asetetaan istuntomuuttuja *reservation.step1* trueksi sekä talletetaan valitun palvelun tiedot istuntomuuttujaan *reservation.service*, jonka jälkeen käyttäjä uudelleenohjataan työntekijän ja ajan valinnan sivulle.

Jos lomaketta validoitaessa esiintyi virheitä, näytetään template uudelleen.

**Template:** templates/selectService.php

Generoitu HTML-sivu, jossa on listattuna kaikki varattavissa olevat palvelut. Sivun tehtävänä on toimia ensimmäisenä askeleena uuden varauksen teossa.

Sivulla näytetään lomakemallia kuvaava HTML-lomake.

Ottaa muuttujat:

*$form* – palvelunvalintalomake

### Työntekijän- ja ajan valintasivu

**Kontrolleri:** public\_html/calendar.php

Validoi ajanvarausprosessin sivua noudettaessa.

**Parametrit:**

*employee* - työntekijän ID-tunniste

Oletusarvoisesti ensimmäinen löydetyn palvelua suorittavan työntekijän ID-tunniste.

*week* – kalenterissa näytettävä viikko, muodossa YYYY-WV

Oletusarvoisesti ensimmäinen sellainen viikko, jolloin työntekijällä on työvuoroja, joille voidaan tehdä varauksia.

*datetime* **-** valitun varausajan Unix-aikaleima

Oletuksena ei arvoa.

Jos kontrolleria pyydetään GET-pyynnöllä, validoidaan mahdollisesti annetut parametrit. Jos parametreja ei ole annettu, asetetaan ne oletusarvoihinsa ja näytetään template.

Jos kontrollerille on lähetetty dataa POST-pyynnöllä, validoidaan parametrit kuten GET-pyynnönkin tapauksessa. Jos lisäksi *datetime* parametri on asetettu, talletetaan tehty työntekijävalinta *reservation.employee*-istuntomuuttujaan ja aikavalinta *reservation.*datetime*-*istuntomuuttujaan. Käyttäjä uudelleenohjataan yhteystietosivulle.

**Template:** templates/calendar.php

Generoitu HTML-sivu, jossa näytetään kalenteri kontrollerista saadulle työntekijälle. Sivulta voidaan vaihtaa työntekijää sekä kalenterin näyttämää viikkoa ”previous week” ja ”next week” –linkeillä.

Jos kontrolleri on asettanut *picketDateTime*-muuttujan, näytetään käyttäjälle vahvistuslomake, joka lähettää kontrollerille POST-pyynnön sivua haettaessa annetuilla parametreilla.

Ottaa muuttujat:

*$service* – varattavan palvelun tiedot

*$employee* – varauksen suorittavan työntekijän tiedot

*$employees* – lista palvelua suorittavista työntekijöistä

*$pickedDateTime –* valittu ajankohta (tai null, jos ei vielä valittu)

*$week* – kalenterissa näytettävä viikko muodossa YYYY-WV

*$nextWeek* – seuraava viikko muodossa YYYY-WV

*$prevWeek* – edellinen viikko muodossa YYYY-WV (tai null)

*$errors* – mahdolliset virheet

*$calendar* –Calendar-olio

### Yhteystietolomake

**Lomakemalli:** forms/ContactInfoForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **cust\_fname** | string | Asiakkaan etunimi. |
| **cust\_lname** | string | Asiakkaan sukunimi. |
| **cust\_email** | string | Asiakkaan sähköpostiosoite. |
| **cust\_phone** | string | Asiakkaan puhelinnumero. |

Kaikki kentät ovat pakollisia. Etu- ja sukunimi voivat olla mitä tahansa ei-tyhjiä merkkijonoja. Sähköpostiosoitteen ja puhelinnumeron tulee olla lähdekoodissa määritetyn säännöllisen lausekkeen mukaisia.

**Kontrolleri:** public\_html/contactInfo.php

Validoi ajanvarausprosessin sivua noudettaessa.

Jos kontrolleria on pyydetty GET-pyynnöllä, näytetään template. Jos kontrollerille on lähetetty dataa POST-pyynnöllä, validoidaan lomake.

Olettaen, että lomakkeen tarkistuksessa ei esiintynyt virheitä, talletetaan lomakkeen kentät *reservation.*-etuliitteellä istuntoon ja asetetaan *reservation.step3*-istuntomuuttujan arvoksi true, jonka jälkeen käyttäjä siirretään vahvistussivulle.

**Template:** templates/contactInfo.php

Generoitu HTML-sivu, jossa asiakas antaa yhteystietonsa tehdessään varausta.

Näyttää lomakemallin mukaisen HTML-lomakkeen sekä mahdolliset virheilmoitukset oikeilla paikoillaan.

Ottaa muuttujat:

*$form* – yhteystietolomake

*$employee* – varaukseen liittyvän työntekijän tiedot

*$service* – varattavan palvelun tiedot

*$datetime* – ajanvarauksen ajankohta

### Vahvistussivu

**Kontrolleri:** public\_html/confirm.php

Validoi ajanvarausprosessin sivua noudettaessa.

Jos kontrolleria on pyydetty GET-pyynnöllä tai POST-pyynnöllä ilman *confirm­*-parametria, näytetään template.

Jos kontrollerille on lähetetty POST-pyynnöllä *confirm*-parametri, tallennetaan tietokantaan uusi ajanvarausrivi istunnosta haetuilla tiedoilla. Uudesta rivistä otetaan talteen ID-tunniste, joka sijoitetaan *reservation.id*-istuntomuuttujaan. Aiemmin asetetut varaukseen liittyvät istuntomuuttujat poistetaan. Käyttäjä ohjataan yhteenvetosivulle.

**Template:** templates/confirm.php

Generoitu HTML-sivu, jolla näytetään tehdyt valinnat ja tarjotaan mahdollisuus palata aiempiin vaiheisiin muuttamaan valintoja.

Vahvistus tapahtuu painamalla sivulla olevaa ”Confirm”-nappia.

Ottaa muuttujat:

*$service* – varattavan palvelun tiedot

*$employee* – varaukseen liittyvän työntekijän tiedot

*$datetime* – varauksen ajankohta

### Yhteenvetosivu

**Kontrolleri:** public\_html/summary.php

Etsii kannasta istuntomuuttujassa *reservation.id* määritetyn varauksen tiedot kannasta. Jos istuntomuuttujaa ei ole asetettu, varausta ei löydy tai varaus on peruttu, uudelleenohjataan käyttäjä palvelun valintasivulle.

Muussa tapauksessa näytetään template.

**Template:** templates/reservationSummary.php

Generoitu HTML-sivu, jolla näytetään tehdyn varauksen tiedot sekä peruutusavain, jolla asiakas voi myöhemmin peruuttaa tehdyn varauksen. Ottaa muuttujat:

*$reservation* – varauksen tiedot

*$employee* – varaukseen liittyvän työntekijän tiedot

*$service* – varatun palvelun tiedot

### Varauksen peruutussivu

**Lomakemalli:** forms/CancelReservationForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **cust\_email** | string | Varatessa annettu sähköpostiosoite. |
| **cancel\_key** | string | Peruutusavain. |

Kaikki kentät ovat pakollisia. Annetuilla tiedoilla tulee löytyä tietokannasta varaus, joka on vielä peruutettavissa.

**Kontrolleri:** public\_html/cancelReservation.php

Jos kontrolleria pyydetään GET-pyynnöllä, näyttää templaten. Jos kontrollerille on lähetetty dataa POST-pyynnöllä, validoidaan lomake.

Olettaen, ettei lomakkeen validoinnissa tule ongelmia, ohjaa käyttäjän selaimen varauksen peruutuksen vahvistussivulle lomakkeen kentät kyselyparametreiksi muutettuna.

Virheellisen lomakkeen tapauksessa näytetään template uudelleen.

**Template:** templates/cancelReservation.php

Generoitu HTML-sivu, jossa on lomakemallin mukainen HTML-lomake. Ottaa muuttujat:

*$form* – varauksen peruutuslomake

### Varauksen peruutuksen vahvistussivu

**Kontrolleri:** public\_html/confirmCancellation.php

**Parametrit:**

*cust\_email* - varatessa annettu sähköpostiosoite

*cancel\_key* – peruutusavain

Validoidaan parametrit käyttäen *forms/CancelReservationForm.php -*lomakemallia.

Jos kontrollerille on lähetetty dataa POST-pyynnöllä ja *confirm*-parametri on asetettuna, asetetaan varaus peruutetuksi ja tallennetaan tiedot kantaan. Ohjataan käyttäjä sivulle, jolla kerrotaan varauksen peruuttamisen onnistuneen.

Muussa tapauksessa näytetään template.

**Template:** templates/confirmCancellation.php

Generoitu HTML-sivu, jolla näytetään varauksen tiedot sekä pyydetään käyttäjää vahvistamaan ajanvarauksen peruutus ”Confirm”-nappia painamalla. Ottaa muuttujat:

*$service* – varatun palvelun tiedot

*$employee* – varatun työntekijän tiedot

*$reservation* – varauksen tiedot

### Varauksen peruutuksen onnistuminen -ilmoitussivu

**Kontrolleri:** public\_html/reservationCancelled.php

Näyttää templaten.

**Template:** templates/reservationCancelled.php

Staattinen HTML-sivu, joka kertoo varauksen peruutuksen onnistuneen. Kehottaa käyttäjää tekemään uuden varauksen. Ei ota muuttujia.

## Hallintapaneeli

Hallintapaneelin jokaisella sivulla (kirjautumissivua index.php lukuunottamatta) tarkistetaan käyttäjän istunnon tila. Jos käyttäjä ei ole sisäänkirjautunut, uudelleenohjataan selain kirjautumissivulle index.php. Tätä ei mainita erikseen kuvausten yhteydessä.

Istunnoissa on säädettävissä oleva aikakatkaisu. Jos istunto on ollut käyttämättömänä liian kauan, kirjautuu käyttäjä automaattisesti ulos. Tämä johtaa seuraavalla muun kuin kirjautumissivun latauskerralla käyttäjän uudelleenohjaamiseen takaisin kirjautumissivulle.

### Kirjautumissivu

**Lomakemalli:** forms/LoginForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **username** | string | Käyttäjänimi. |
| **password** | string | Salasana. |

Kaikki kentät ovat pakollisia.

**Kontrolleri:** public\_html/admin/index.php

Tarkistaa käyttäjän sisäänkirjautumistilanteen. Jos käyttäjä on valmiiksi sisäänkirjautunut, uudelleenohjataan käyttäjä hallintapaneelin etusivulle.

Jos kontrolleria on pyydetty GET-pyynnöllä, näytetään template. Muussa tapauksessa validoidaan lomake ja tarkistetaan *lib/auth.php*-tiedostosta löytyvällä *auth()*-funktiolla onko annetut sisäänkirjautumistiedot kelvolliset. Olettaen, että *auth()-*funktio palauttaa arvon *true*, ohjataan käyttäjä eteenpäin hallintapaneelin etusivulle. Muussa tapauksessa näytetään template uudelleen.

**Template:** templates/admin/login.php

Generoitu HTML-sivu, jossa on sisäänkirjautumista varten lomake. Ottaa muuttujat:

*$form* – sisäänkirjautumislomake

*$errorMessage –* käyttäjälle näytettävä virheilmoitus

### Hallintapaneelin etusivu

**Kontrolleri:** public\_html/admin/admin.php

Näyttää templaten.

**Template:** templates/admin/admin.php

Staattinen HTML-sivu, jossa listataan ja selostetaan hallintapaneelin eri osiot. Ei ota muuttujia.

### Työntekijöiden hallinta

**Kontrolleri:** public\_html/admin/employees.php

Hakee tietokannasta listan kaikista työntekijöistä ja välittää listauksen templatelle.

**Template:** templates/admin/employees.php

Generoitu HTML-sivu, joka näyttää listauksen työntekijöistä. Linkkaa jokaisen työntekijän työntekijän tiedot –sivulle sekä sisältää linkin uuden työntekijän lisäämissivulle. Ottaa muuttujat:

*$employees –* lista työntekijöistä Employee-olioina

### Työntekijän lisääminen

**Lomakemalli:** forms/EditEmployeeForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **first\_name** | string | Työntekijän etunimi. |
| **last\_name** | string | Työntekijän sukunimi. |
| **active** | boolean | Onko työntekijä aktiivinen? |

Kaikki kentät ovat pakollisia (tyhjä arvo boolean-tyyppisellä kentällä tarkoittaa ”false”).

**Kontrolleri:** public\_html/admin/addEmployee.php

Jos kontrolleria on pyydetty GET-pyynnöllä, näyttää templaten.

Muussa tapauksessa validoidaan lomake. Olettaen, ettei lomakkeen tarkistuksessa esiinny virheitä, tallennetaan uuden työntekijän tiedot kantaan ja ohjataan käyttäjä edelleen juuri lisätyn työntekijän tiedot-sivulle.

**Template:** templates/admin/addEmployee.php

Generoitu HTML-sivu, jossa lomakemallin mukainen lomake työntekijöiden lisäämiseen. Ottaa muuttujat:

*$form* – työntekijän lisäyslomake

### Työntekijän tiedot

**Lomakemalli:** forms/EditEmployeeForm.php

Sama kuin ylempänä “työntekijän lisääminen” –sivulla.

**Kontrolleri:** public\_html/admin/employee.php

**Parametrit:**

*id –* työntekijän ID-tunniste

*workShiftAdded –* jos asetettu, näytä työvuoron lisäyksen onnistumisviesti

*workShiftDeleted* – jos asetettu, näytä työvuoron poiston onnistumisviesti

*employeeAdded –*jos asetettu,näytä työntekijän lisäyksen onnistumisviesti

Ainoa pakollinen parametri on *id*. Ainoastaan yksi muista mahdollisista parametreista otetaan huomioon (vain yksi viesti per pyyntö).

Parametrina annetulla työntekijän ID-tunnisteella tulee löytyä tietokannasta työntekijän tiedot. Jos näin ei ole, näytetään 404 – Page Not Found –virheilmoitus.

Jos kontrollerille on lähetetty dataa POST-pyynnöllä, validoidaan lomakkeen tiedot ja kenttien arvojen ollessa hyväksytyt talletetaan muutetut työntekijän tiedot tietokantaan. Käyttäjälle näytetään tämän jälkeen template, jossa ilmoitus onnistuneesta tallennuksesta.

Muussa tapauksessa näytetään template, jolle annettu parametrina mm. työntekijän tiedot ja työntekijän tulevat työvuorot.

**Template:** templates/admin/employee.php

Generoitu HTML-sivu, joka näyttää työntekijän tiedot ja tarjoaa lomakkeen näiden muuttamiseen. Lisäksi näytetään tulevat työvuorot. Ottaa muuttujat:

*$employee* – työntekijän tiedot Employee-oliona

*$form –* työntekijän muokkauslomake

*$workShifts* – lista työntekijän tulevista työvuoroista WorkShift-olioina

*$message* – käyttäjälle näytettävä viesti

Jokaiselle työvuorolle on linkki työvuoron poistamiseen sekä työvuoron yksityiskohtaiseen työlistaukseen. Lisäksi sivulla on linkki työvuorojen lisäykseen.

### Työvuoron lisäys

**Lomakemalli:** forms/EditWorkShiftForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **date** | date | Päivämäärä muodossa YYYY-MM-DD |
| **start\_time** | time | Kellonaika muodossa HH:MM |
| **end\_time** | time | Kellonaika muodossa HH:MM |

Kaikki kentät ovat pakollisia.

**Kontrolleri:** public\_html/admin/addWorkShift.php

**Parametrit:**

*employee\_id* – työntekijän ID-tunniste

Kontrolleria kutsuttaessa validoidaan, että parametrin *employee\_id* arvolla löytyy tietokannasta työntekijä. Jos näin ei ole, näytetään HTTP 404 – Page Not Found –virhesivu.

Pyydettäessä kontrolleria GET-pyynnöllä, näytetään template.

Jos lomake on lähetetty POST:ina, tarkistetaan onko POST-parametri ’cancel’ asetettu. Jos on, siirretään käyttäjä takaisin työntekijän tietoihin. Muussa tapauksessa validoidaan lomake. Olettaen, että lomakkeen tarkistuksessa ei esiinny virheitä, tallennetaan uuden vuoron tiedot kantaan. Riippuen painetun submit-napin nimestä ja siis asetetuista POST-parametreista, tehdään jokin seuraavista:

* Jos ’saveAndAddNew’ asetettu, tyhjennä lomake & näytä template
* Jos ’save’ asetettu, uudelleenohjaa työntekijän tietoihin

**Template:** templates/admin/addWorkShift.php

Generoitu HTML-sivu, jossa lomake työntekijän tietojen asettamiseen. Ottaa seuraavat muuttujat:

*$employee* – työntekijän tiedot Employee-tyyppisessä oliossa

*$form* – lomake

*$message* – käyttäjälle näytettävä viesti

### Työvuoron poisto

**Kontrolleri:** public\_html/admin/deleteWorkShift.php

**Parametrit:**

*id –* työvuoron ID-tunniste

Kontrolleria kutsuttaessa tarkistetaan, että annetulla *id*-parametrilla löytyy kannasta työvuoro.

Jos kontrolleria on pyydetty GET-pyynnöllä tai sille on lähetetty dataa POST-pyynnöllä ilman ’confirm’-parametrin asettamista, näytetään template.

Muussa tapauksessa haetaan kaikki työvuoron aikana olevat työntekijälle tehdyt varaukset ja perutaan ne. Tämän jälkeen poistetaan työvuoron tiedot tietokannasta ja uudelleenohjataan käyttäjä takaisin työvuoron tietoihin.

**Template:** templates/admin/confirmDeleteWorkShift.php

Generoitu HTML-sivu, joka pyytää vahvistusta työvuoron poistoon. Näyttää työvuoron aikana tapahtuvien varausten määrän. Ottaa seuraavat muuttujat:

*$shift* –työvuoron tiedot WorkShift-tyyppisessä oliossa

### Työntekijän työlista

**Kontrolleri:** public\_html/admin/workSchedule.php

**Parametrit:**

*employee\_id* – työntekijän ID-tunniste

*date* – näytettävä päivä muodossa YYYY-MM-DD

Kontrolleria kutsuttaessa tarkistetaan, että annetulla *employee\_id*-parametrilla löytyy kannasta työntekijä. Lisäksi tarkistetaan päivämäärän muoto.

Haetaan kaikki työntekijän varaukset määritetylle päivälle ja näytetään template.

**Template:** templates/admin/workSchedule.php

Generoitu HTML-sivu, joka näyttää työntekijän työlistauksen valitulle päivälle. Ottaa seuraavat muuttujat:

*$employee* – työntekijän tiedot Employee-oliona

*$reservations* – lista päivän työvuoroista Reservation-olioina

*$date* – valittu päivämäärä DateTime-oliona

### Palvelulistaus

**Kontrolleri:** public\_html/admin/serviceList.php

Kontrolleri hakee tietokannasta kaikkien palveluiden tiedot ja näyttää templaten.

**Template:** templates/admin/serviceList.php

Generoitu HTML-sivu, joka listaa kaikkien palveluiden tiedot HTML-taulukkona. Ottaa seuraavat muuttujat:

*$services –* lista palveluista Service-olioina

Jokaisen palvelun kohdalla on linkki palvelun muokkaussivulle. Lisäksi sivulla on linkki uuden palvelun lisäämiseen.

### Palvelun lisäys / muokkaus

**Lomakemalli:** forms/EditServiceForm.php

|  |  |  |
| --- | --- | --- |
| Nimi | Tyyppi | Kuvaus |
| **name** | string | Palvelun nimi. |
| **description** | string | Palvelun kuvaus. |
| **price** | float | Palvelun hinta muodossa xxx.xx |
| **duration** | int | Palvelun kesto kokonaisina minuutteina. |
| **available** | boolean | Onko palvelu varattavissa? |
| **perf\_emp** | array | Suorittavat työntekijät, monivalinta. |

Kaikki lomakkeen kentät ovat pakollisia *perf\_emp*-kenttää lukuunottamatta. Mahdolliset arvot *perf\_emp*-kentän alkioille saadaan hakemalla tietokannasta kaikkien työntekijöiden ID-tunnisteet.

**Kontrolleri:** public\_html/admin/editService.php

**Parametrit:**

*id –* palvelun ID-tunniste

Jos *id* ei ole asetettu, kontrolleri olettaa, että kyseessä on uuden palvelun lisäys. Muussa tapauksessa validoidaan, että annetulla *id-*parametrilla löytyy tietokannasta palvelu. Jos ei löydy, näytetään HTTP 404 – Page Not Found –virheilmoitus.

POST:lla dataa lähettäessä kontrolleri validoi lomakkeen tiedot. Olettaen, ettei lomakkeen tarkistuksessa esiinny virheitä, luodaan joko uusi palvelurivi tietokantaan tai päivitetään vanhaa. Tämän jälkeen näytetään template viestillä *”Saved”*.

Muussa tapauksessa näytetään template, jossa lomakkeen tiedot ovat joko esitäytettyinä (palvelun muokkaus / virhe validoinnissa) tai tyhjinä.

**Template:** templates/admin/editService.php

Generoitu HTML-sivu, jolla lomake palvelun muokkaamiseen tai uuden palvelun lisäämiseen. Ottaa muuttujat:

*$form –* muokattava lomake

*$servid* – muokattavan palvelun ID (tai null, jos uusi palvelu)

*$service* – muokattava palvelu Service-oliona (tai null, jos uusi palvelu)

*$message* – käyttäjälle näytettävä viesti

### Viikkoraportti

**Kontrolleri:** public\_html/admin/weeklyReport.php

**Parametrit:**

*week* – viikko muodossa YYYY-WV (esim. 2009-W21 viikolle 21/2009)

Jos ei asetettu tai jos epäkelpo, käytetään oletuksena nykyistä viikkoa.

Kontrolleri luo raportin valitulle viikolle käyttäen *lib/Report.php*-tiedostosta löytyvää raportointiluokkaa. ja näyttää templaten.

**Template:** templates/admin/weeklyReport.php

Generoitu HTML-sivu, jossa viikkoraportin sisältö. Lisäksi edellinen/seuraava –viikko linkit, jotka muuttavat kontrollerille lähetetyn *week*-parametrin arvoa. Ottaa muuttujat:

*$report* – raportti Report-oliona

# Tietokanta

Tietokannan yleiskuva on pysynyt samana suunnitteludokumentista. Alla tietokannan luomisessa käytettävät CREATE TABLE –lauseet, jotka löytyvät myös tiedostosta *database/new\_schema.sql*

-- employees table

DROP SEQUENCE IF EXISTS employees\_id\_seq CASCADE;

DROP TABLE IF EXISTS employees CASCADE;

CREATE SEQUENCE employees\_id\_seq;

CREATE TABLE employees

(

id BIGINT NOT NULL DEFAULT nextval('employees\_id\_seq'),

first\_name VARCHAR(255) NOT NULL,

last\_name VARCHAR(255) NOT NULL,

active BOOLEAN NOT NULL DEFAULT TRUE,

PRIMARY KEY (id)

);

-- work\_shifts table

DROP SEQUENCE IF EXISTS work\_shifts\_id\_seq CASCADE;

DROP TABLE IF EXISTS work\_shifts CASCADE;

CREATE SEQUENCE work\_shifts\_id\_seq;

CREATE TABLE work\_shifts

(

id BIGINT NOT NULL DEFAULT nextval('work\_shifts\_id\_seq'),

employee\_id BIGINT NOT NULL,

start\_time TIMESTAMP NOT NULL,

end\_time TIMESTAMP NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (employee\_id) REFERENCES employees

);

-- service\_catalog table

DROP SEQUENCE IF EXISTS service\_catalog\_id\_seq CASCADE;

DROP TABLE IF EXISTS service\_catalog CASCADE;

CREATE SEQUENCE service\_catalog\_id\_seq;

CREATE TABLE service\_catalog

(

id BIGINT NOT NULL DEFAULT nextval('service\_catalog\_id\_seq'),

name VARCHAR(50) NOT NULL,

description TEXT,

duration SMALLINT NOT NULL,

price DOUBLE PRECISION NOT NULL,

available BOOLEAN DEFAULT TRUE,

PRIMARY KEY (id)

);

-- employee\_service\_catalog table

DROP SEQUENCE IF EXISTS employee\_service\_catalog\_id\_seq CASCADE;

DROP TABLE IF EXISTS employee\_service\_catalog CASCADE;

CREATE SEQUENCE employee\_service\_catalog\_id\_seq;

CREATE TABLE employee\_service\_catalog

(

id BIGINT NOT NULL DEFAULT nextval('employee\_service\_catalog\_id\_seq'),

service\_id BIGINT NOT NULL,

employee\_id BIGINT NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (service\_id) REFERENCES service\_catalog,

FOREIGN KEY (employee\_id) REFERENCES employees

);

-- reservable\_services view

DROP VIEW IF EXISTS reservable\_services;

CREATE VIEW reservable\_services AS

SELECT \* FROM service\_catalog sc

WHERE sc.available = TRUE AND

(SELECT COUNT(id) FROM employee\_service\_catalog esc

WHERE esc.service\_id = sc.id) > 0;

-- reservations table

DROP SEQUENCE IF EXISTS reservations\_id\_seq CASCADE;

DROP TABLE IF EXISTS reservations CASCADE;

CREATE SEQUENCE reservations\_id\_seq;

CREATE TABLE reservations

(

id BIGINT NOT NULL DEFAULT nextval('reservations\_id\_seq'),

employee\_id BIGINT NOT NULL,

service\_id BIGINT NOT NULL,

start\_time TIMESTAMP NOT NULL,

end\_time TIMESTAMP NOT NULL,

price DOUBLE PRECISION NOT NULL,

cancel\_key CHAR(5) NOT NULL,

cancelled BOOLEAN NOT NULL DEFAULT FALSE,

cust\_fname VARCHAR(50) NOT NULL,

cust\_lname VARCHAR(50) NOT NULL,

cust\_email VARCHAR(100)NOT NULL,

cust\_phone VARCHAR(15) NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (employee\_id) REFERENCES employees,

FOREIGN KEY (service\_id) REFERENCES service\_catalog

);

-- managers table

DROP SEQUENCE IF EXISTS managers\_id\_seq CASCADE;

DROP TABLE IF EXISTS managers CASCADE;

CREATE SEQUENCE managers\_id\_seq;

CREATE TABLE managers

(

id BIGINT NOT NULL DEFAULT nextval('managers\_id\_seq'),

username VARCHAR(50) NOT NULL,

password CHAR(32) NOT NULL,

PRIMARY KEY (id)

);

-- sessions table

DROP TABLE IF EXISTS sessions;

CREATE TABLE sessions

(

id CHAR(32) NOT NULL,

manager\_id BIGINT NOT NULL,

last\_activity TIMESTAMP NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (manager\_id) REFERENCES managers

);

-- gen\_sess\_id()

CREATE OR REPLACE FUNCTION gen\_sess\_id() RETURNS char(32) AS $$

DECLARE

i smallint;

generated\_key char(32);

temp char(32);

count\_rec RECORD;

BEGIN

i := 0;

generated\_key := '';

WHILE i < 32 LOOP

temp := generated\_key || chr( cast(32 + random()\*93 as integer) );

generated\_key := temp;

i := i+1;

END LOOP;

SELECT INTO count\_rec id FROM sessions WHERE id = generated\_key;

IF FOUND THEN

RETURN gen\_sess\_id();

ELSE

RETURN generated\_key;

END IF;

END;

$$ LANGUAGE plpgsql;

-- gen\_cancel\_key()

CREATE OR REPLACE FUNCTION gen\_cancel\_key() RETURNS char(5) AS $$

DECLARE

i smallint;

generated\_key char(5);

temp char(5);

count\_rec RECORD;

BEGIN

i := 0;

generated\_key := '';

WHILE i < 5 LOOP

temp := generated\_key || cast(random()\*9 as integer);

generated\_key := temp;

i := i+1;

END LOOP;

SELECT INTO count\_rec id FROM reservations WHERE

cancel\_key = generated\_key AND

cancelled = FALSE AND

start\_time > NOW();

IF FOUND THEN

RETURN gen\_cancel\_key();

ELSE

return generated\_key;

END IF;

END;

$$ LANGUAGE plpgsql;

-- gen\_shifts()

CREATE OR REPLACE FUNCTION gen\_shifts() RETURNS integer AS $$

DECLARE

emp\_rec RECORD;

curr\_date date;

end\_date date;

begin\_hour interval;

last\_shift RECORD;

BEGIN

curr\_date := current\_date;

end\_date := cast(curr\_date + '30 days'::interval as date);

FOR emp\_rec IN SELECT id FROM employees WHERE active = TRUE LOOP

SELECT INTO last\_shift DATE(MAX(start\_time)) AS max FROM work\_shifts WHERE employee\_id = emp\_rec.id;

IF FOUND THEN

IF last\_shift.max IS NOT NULL THEN

curr\_date := cast(last\_shift.max + '1 day'::interval as date);

END IF;

END IF;

WHILE curr\_date <= end\_date LOOP

IF extract(dow from curr\_date) BETWEEN '1' AND '5' THEN

begin\_hour := ((cast( (random() \* 4) as integer) + 8) ||

‘ hours')::interval;

INSERT INTO work\_shifts (employee\_id, start\_time, end\_time) VALUES (

emp\_rec.id, ( curr\_date + begin\_hour ), ( curr\_date + begin\_hour +

'8 hours'::interval ));

END IF;

curr\_date := cast(curr\_date + '1 day'::interval as date);

END LOOP;

curr\_date := current\_date;

END LOOP;

RETURN 0;

END;

$$ LANGUAGE plpgsql;

## Tallennetut proseduurit

Tietokannassa on kolme tallennettua proseduuria:

### gen\_shifts() returns integer

Luo satunnaisesti työvuoroja kaikille kannassa oleville työntekijöille. Työvuorot sijoitetaan satunnaisesti alkamaan välillä 8 – 12 ja loppumaan välillä 16 – 20.

Työvuoroja generoidaan seuraavan 30 päivän ajalle, kuitenkin niin, että viikonloppuisin työvuoroja ei ole. Jos työntekijällä on kannassa valmiiksi työvuoroja, ensimmäinen generoitu työvuoro on aikaisintaan seuraavana päivänä klo. 8.00 viimeisimmästä rekisteröidystä työvuorosta.

### gen\_sess\_id() returns char(32)

Luo satunnaisen istuntotunnisteen, joka talletetaan sisäänkirjautumisen yhteydessä sessions-tauluun. Luodut istuntovarmisteet ovat uniikkeja.

### gen\_ cancel\_key() returns char(5)

Luo satunnaisen peruutusavaimen, jota voidaan käyttää varauksen perumiseen. Luodut peruutusavaimet ovat uniikkeja tällä hetkellä peruttavissa olevien varausten peruutusavaimien kanssa.

# Asennusohje

Ennen asennusta varmista kohdejärjestelmän täyttävän luvussa 1 (Johdanto) mainitut vähimmäisvaatimukset.

Tässä ohjeistetaan järjestelmän asentaminen Apache –palvelimelle.

## Asennushakemiston luominen

Asennuksen ensimmäisessä vaiheessa luodaan asennushakemisto. Tämä voidaan luoda esimerkiksi terminaalista tai käyttäen graafista tiedostomanageria (Windows Explorer / Nautilus Gnomessa / tms.).

Pura salonlulu.zip-tiedosto asennushakemistoon. Asennushakemiston sisältö puurakenteena pitäisi olla nyt suurinpiirtein seuraavanlainen:

.

|-- config

|-- database

|-- doc

| `-- diagrams

|-- forms

|-- lib

| |-- db

| `-- forms

|-- models

|-- public\_html

| |-- admin

| | `-- media

| | `-- css

| `-- media

| `-- css

`-- templates

`-- admin

Www-palvelimella tulee olla luku- ja suoritusoikeus public\_html-hakemiston sisällä oleviin hakemistoihin. Tämä voidaan saavuttaa seuraavalla komennolla Unix-komentotulkista:

$ find public\_html –type d –exec chmod 755 \{} \;

Varmista myös, että public\_html:llä on suoritus- ja lukuoikeus:

$ chmod 755 public\_html

## Tietokannan pystyttäminen

Luo haluamallesi PostgreSQL-palvelimelle haluamasi tietokanta CREATE DATABASE –lauseella ja yhdistä tähän tietokantaan.

Suorita seuraavat komennot interaktiivisessa PostgreSQL-tulkissa:

tietokanta=# CREATE LANGUAGE plpgsql;

tietokanta=# \i [asennushakemisto]/database/new\_schema.sql

Jos haluat testausta varten luoda kantaan valmiiksi palvelut, työntekijät, työntekijöiden palveluvalikoimat sekä generoidut työvuorot, voit suorittaa lisäksi seuraavan komennon:

tietokanta=# \i [asennushakemisto]/initial\_data.sql

### Hallintapaneelin sisäänkirjautumistietojen lisääminen

Hallintapaneelin sisäänkirjautumistiedot tulee lisätä *managers*-tauluun, jonka username-sarakkeeseen tulee haluttu käyttäjänimi sekä password-salakkeeseen PostgreSQL:n MD5()-funktiolla salasanasta laskettu tarkistussumma. Esimerkiksi:

tietokanta=# INSERT INTO managers (username, password) VALUES (’admin’, MD5(’demo’));

Voit lisätä niin monta riviä kuin haluat, kunhan käyttäjänimet eroavat toisistaan.

## Asetusten säätö

Avaa haluamallasi tekstieditorilla tiedosto [asennushakemisto]/config/settings.php ja muuta tietokannan yhteysasetukset oikeiksi.

Muista tallettaa tiedosto tehtyäsi muutokset!

### $SETTINGS[’time\_period\_duration’]

Kalenterissa näytettävän rivin kesto minuutteina. Oletus 15 min.

### $SETTINGS[’reservation\_timeout’]

Varauksen aikakatkaisu. Estää varausprosessin jatkumisen määritetyn ajan jälkeen. Oletus 30 min.

### $SETTINGS[’session\_timeout’]

Istunnon aikakatkaisu minuutteina. Kirjaa käyttäjän ulos hallintapaneelista, jos käyttäjä ei ole ollut aktiivinen määritetyn aikavälin sisällä. Oletus 30 min.

## public\_html näkyväksi www-palvelimelle

Siirrä [asennushakemisto]/public\_html www-palvelimen DocumentRootin tai oman käyttäjäsi www-hakemiston juureen ja nimeä se haluamallasi tavalla (esim. ”salonsulu”).

Jos palvelin osaa seurata symlinkkejä, voit myös tehdä symlinkin salonsulu -> [asennushakemisto]/public\_html

## include\_path:n asettaminen

Ennen kuin sovellus voidaan käynnistää, täytyy asettaa PHP:n include\_path. PHP:n asennustavasta riippuu, miten tämä tehdään. Jos PHP on asennettu Apachen moduuliksi, voit luoda .htaccess nimisen tiedoston edellisessä vaiheessa luomaasi hakemistoon:

php\_value include\_path .:[asennushakemisto]

Windows-järjestelmissä polkuerottimena käytetään ;-merkkiä:

php\_value include\_path .;[asennushakemisto]

Voit myös asettaa include\_path:n suoraan php.ini-tiedostosta tai www-palvelimen konfiguraatiotiedostosta, jos sinulla on tähän oikeudet. Tätä menetelmää ei selosteta täällä.

Jos et pysty asettamaan include\_pathia, sinun tulee käydä jokainen asennushakemistosta löytyvä PHP-tiedosto läpi ja lisätä seuraava rivi alkuun:

set\_include\_path(“.:[asennushakemisto]”); // muista oikea polkuerotin!

Asennus on nyt valmis. Voit testata asennuksen avaamalla www-selaimeesi public\_html:n alta löytyvän index.php tiedoston. Ongelmatilanteissa ota yhteyttä Mikko Nyléniin, sähköposti mikko.nylen@gmail.com.

# Testiasennus

Osoitteessa

http://db.cs.helsinki.fi/u/mnylen/tsoha/ toimii järjestelmän testiasennus.

Hallintapaneeliin pääsee käsiksi osoitteesta

http://db.cs.helsinki.fi/u/mnylen/tsoha/admin/

Käyttäjätunnus: admin

Salasana: demo

Voin alustaa kannan uudelleen testidatalla pyynnöstä. Lähetä tässä tapauksessa sähköpostia osoitteeseen mikko.nylen@gmail.com