TIETOKANTASOVELLUS: PALKANLASKENTA JA TUNTIKIRJANPITO

SISÄLLYSLUETTELO

1. **JOHDANTO**

Tavoitteena on rakentaa kevyt tuntikirjanpito ohjelmisto pk-yrityksille. Järjestelmä tulee yrityksen sisäiseksi työkaluksi, joten sen käyttäjiä ovat tietenkin kaikki yrityksen työntekijöitä, jotka kuitenkin tässä tapauksessa jaotellaan pääkäyttäjiin (rank=1) ja työntekijöihin (rank=0). Järjestelmä helpottaa ja nopeuttaa pienten yritysten työntekijöiden tuntien seurantaa.

Projektin aion toteuttaa perinteisen LAMP stackin päälle eli tässä tapauksessa Linux, Apache, MySQL ja PHP. Tämä pystytetään Linux virtuaalipalvelimelle (UpCloud) siten, että tietokantapalvelin ja itse web-palvelin ovat eriytetty – harjoittelen siis samalla vähän web-sovellus arkkitehtuureja. Projektin tietokantapalvelin on käytettävissä julkiverkon puolelta, joten pystyn helposti kehittämään sovellusta koneella kuin koneella lokaalisti, kunhan konfiguroin tietokannan tunnukset manuaalisesti.

Projektin palvelimet ovat jo pystyssä ja toiminnassa:

- DB: (185.20.136.223:3306) – vaatii salasanan
- Web-palvelin: (185.20.137.199:80) – vaatii heti sisäänkirjautumista

2. **YLEISKUVA JÄRJESTELMÄSTÄ**
2.1 KUVAUS

Usein etenkin IT-alalla henkilöstökulut ovat yrityksille suhteellisen suuri ja tärkeä kuluerä – tämän takia on tärkeää tietää työntekijöiden työtunneista ja palkasta mahdollisimman paljon. Ilman IT-järjestelmää tämä voi olla hidasta ja hankalaa. Tämän projektin ydin onkin työntekijöiden tuntikirjanpito, jossa otetaan huomioon erilainen työ (yötyö/pyhäpäivät/normaalit tunnit/asiakkaalle laskutettat tunnit/jne.).

Perusidea on, että työntekijät kirjaavat tuntinsa järjestelmään, josta heidän esimiehensä voivat helposti seurata tehtyjä tunteja.

Ohjelmaan voidaan myös merkitä asiakkaalta laskutettavat tunnit, joissa on monesta moneen suhde käyttäjien ja asiakastuntien välillä – asiakaille tehdyissä töissä kuitenkin saattaa olla useampia työntekijöitä. Ohjelma perustuu hyvin pitkälti CRUD toimintoihin ja resurssien näkymiin (resurssikohtaiset index näkymät).

2.2. KÄYTTÖTAPAUKSET

2.2.1 KÄYTTÖTAPAUSKAAVIO

Käyttötapauskaaviosta on jätetty pois kirjautuminen ja rekisteröityminen, sillä järjestelmään ei kirjautumatta pääse ollenkaan. Rekisteröitymistäkään ei tule, vaan esimiehet voivat luoda käyttäjätilejä työntekijöille.

2.2.2 KÄYTTÄJÄT

Työntekijä:

Saa muokata omia tietojaan sekä merkata tehtyjä työtuntejaan. Näkee vain omat tietonsa työtuntien osalta. Asiakaslaskutustunnit näkyvät kaikille, työntekijä voi kuitenkin muokata vain niitä missä on itse osallisena.

Esimies:

Lisää/poistaa työntekijöitä mutta voi vain tarkastella näiden tietoja ja tunteja. Täydet CRUD oikeudet asiakaslaskutustunteihin. Saa tarkastella aggregoituja tuntitilastoja.

2.2.3 KÄYTTÖTAPAUSKUVAUKSET

Työntekijä ja Esimies yhteiset käyttötapaukset (myös esimies on työntekijä):

Merkkaa asiakaslaskutustunnit:

Merkataan HTML lomakkeella tunnit, jotka ovat asiakkaalle laskutettua työtä. Monesta moneen suhde käyttäjät – asiakastunnit, koska samaa asiakkaalle laskutettavaa projektia on saattanut olla tekemässä useampi työntekijä.

Merkkaa päivittäiset tunnit:

Tunnit merkataan HTML lomakkeella, lomakkeessa on erittelyjä eri kategorioille; yötyö, asiakkaalle laskutettava työ, normaalit työtunnit, jne.

Hyväksy kuukausiraportti päivittäisistä tunneista:

Palkka maksetaan aina kuukauden tuntien perusteella, eli käyttäjän pitää hyväksyä kuukauden työtunnit yhtenä kokonaisuutena; ohjelma osaa generoida kuukausiraportin hyväksymistä varten päivittäisistä tunneista.

Tarkastele omia työtunteja:

Työntekijä voi tarkastella omia merkattuja tuntejaan.

Oman tilin tietojen hallinta:

Omassa käyttäjätilissä on työntekijän palkanmaksuun tarvittavat tiedot, jotka voivat luonnollisesti muuttua esim. muuton tai pankin vaihtamisen takia.

Esimiehen käyttötapaukset (pl. edellä mainitut):

Lisää ja poista työntekijätilejä:

Create/Destroy operaatiot työntekijätileihin.

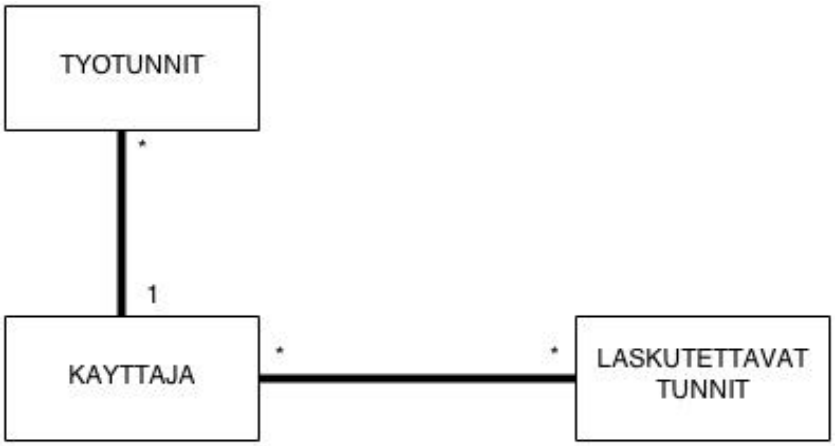
Tarkastele työntekijöiden tunteja:

Tarkastele yksittäisten työntekijöiden tuntimerkintöjä ja kuukausiraportteja. Työntekijäthän näkevät vain omat tuntinsa, mutta esimies näkee kaikkien työntekijöiden.

Tarkastelee aggregoituja tilastoja:

Kuukausiraporteista voidaan generoida aggregoituja raportteja työvoimakuluista, joita voidaan käyttää apuna esim. työn tuottavuuden mittaamisessa ja rekrytointitarpeiden selvittämisessä.

3. TIETOSISÄLTÖ



Tietokohde: Käyttäjä

Attribuutti	Arvojoukko	Kuvailu
Username	String, 16	Käyttäjänimi
Password	String, 32	Käyttäjän salasana
Firstname	String, 32	Etunimi
Lastname	String, 32	Sukunimi
Address	String, 128	Osoite, vapaamuotoinen... käyttäjän omalla vastuulla tulevatko palkkakuitit perille.
Email	String, 254	Sähköposti, spostin virallinen max pituus 254
Phone	String, 15	Puhelin
Rank	Boolean,	Työntekijä, Esimies

Tietokohde: Työtunnit

Attribuutti	Arvojoukko	Kuvailu
Id	Int	Uniikki id
Day	Date	Päivämäärä, minkä päivän tunnit
Hours	Int	Normaalit tunnit 6-22
Offhours	Int	Muut 22-6
Standbyhours	Int	Päivystystunnit
Username	String, 16	Käyttäjäviittaus.
Approved	Boolean	Hyväksytty / ei vielä hyväksytty

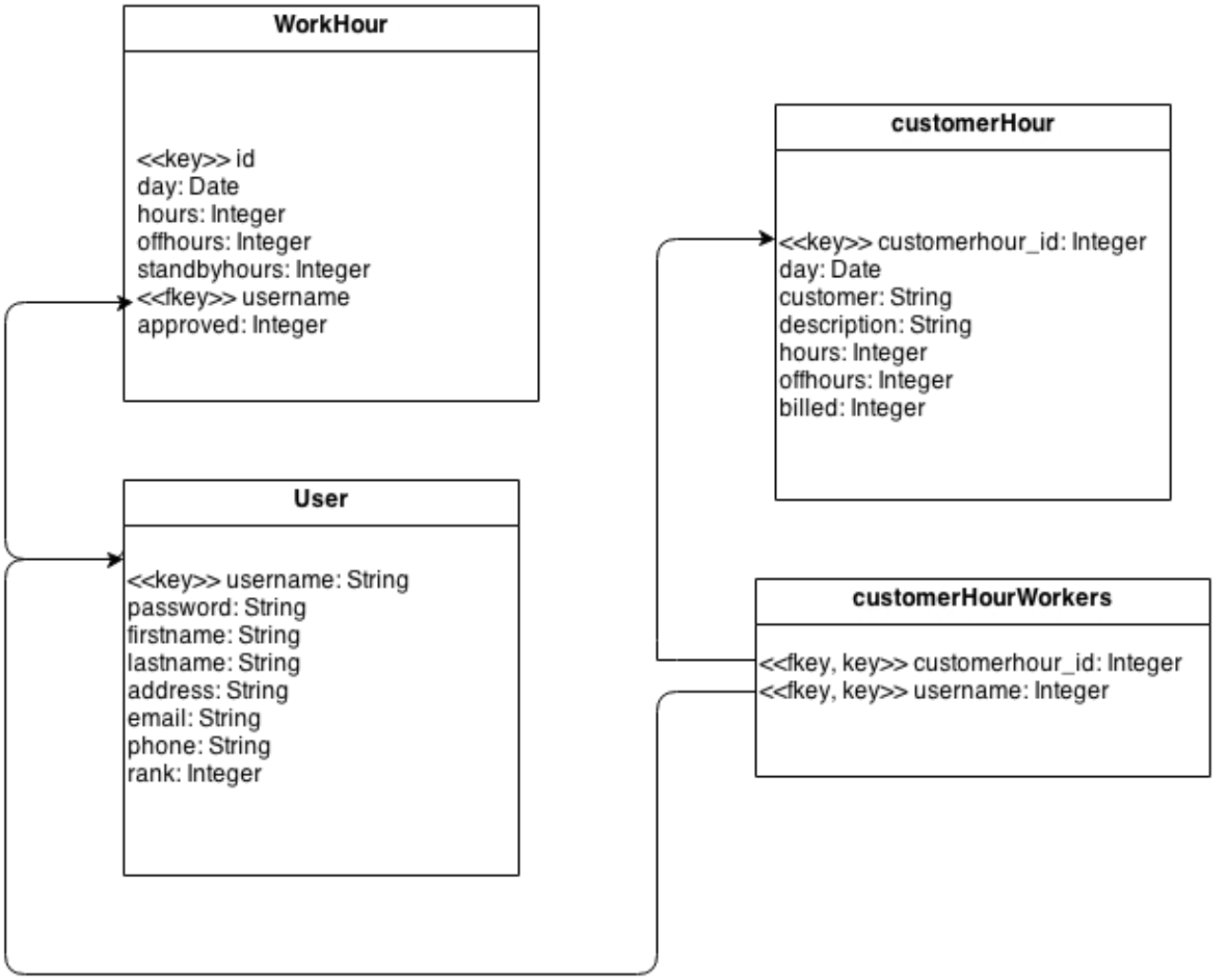
Tietokohde: Laskutettavat tunnit (asiakastunnit)

Attribuutti	Arvojoukko	Kuvailu
Id	Int	Uniikki id
Day	Päivämäärä Date	Minä päivänä tehtyjä tunteja?
Customer	String, max 20	Asiakkaan nimi (firman nimi)
Description	Text	Kuvaus, lisätiedot. Vapaamuotoinen.
Hours	Int	Tunnit 6-22
Offhours	Int	Tunnit 22-6
Billed	Boolean	Onko laskutettu

Monesta moneen yhteys asiakastuntien ja käyttäjien välillä toteutetaan välitaululla, jossa on vain viittaus kumpaankin - ks. Relaatiokaavio.

4. RELAATIOKAAVIO

Huom. MySQL Boolean on oikeasti Integer, missä 0=false ja 1=true. Tässä kaaviossa käytetään booleanin tilalla integeriä (vrt. Luvun 3 tietosisältö)



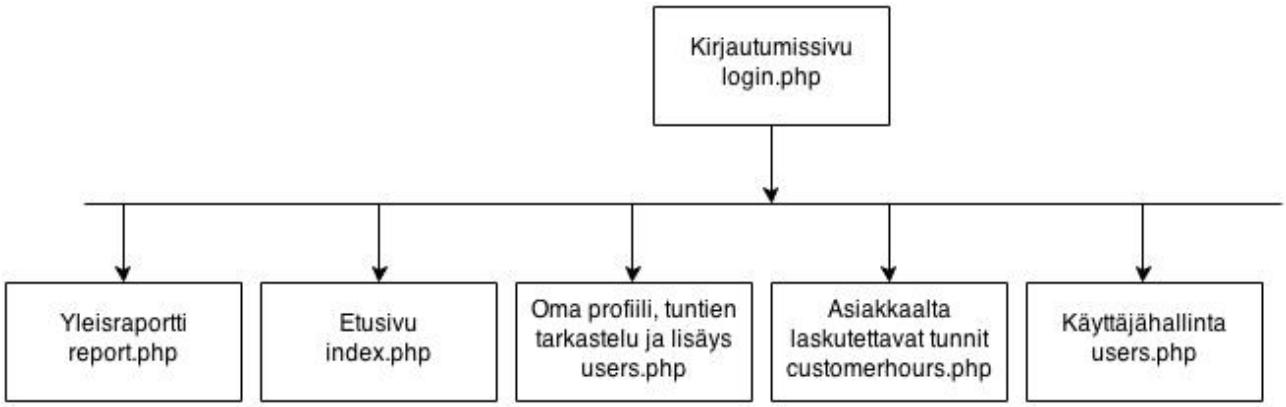
5. JÄRJESTELMÄN YLEISRAKENNE

Järjestelmä noudattaa MVC mallia, jossa tietokantayhteys ja joitain apufunktioita on asetettu /lib kansioon. Viewit on nimetty index,show jne. mukaisesti mutta ne ovat asetettu resurssiaan vastaaviin kansioihin. CSS tiedostot löytyvät assets kansiosta.

Järjestelmän kaikki pyynnöt ohjataan index.php tiedostoon, joka hoitaa autentikoinnin ja pyyntöjen reitityksen kontrollereille. Tämä oli mielestäni mielekkäämpää kuin antaa Apache hoitaa tämä, en halunnut hajauttaa ohjelman toiminnallisuutta liikaa eri järjestelmien hoidettavaksi.

Tietokanta pyörii omalla palvelimellaan, mutta tällä ei oikeastaan ole merkitystä; tietokantayhteys konfiguroidaan CONFIG.php tiedostoon.

6. KÄYTTÖLIITTYMÄ



Järjestelmään ei pääse ollenkaan sisälle ilman kirjautumista, tämän jälkeen eri osioihin pääsee navigointipalkista yllä esitetyllä tavalla. Pyrin lisäämään CRUD toiminallisuuden suoraan tuntien/asiakastuntien/käyttäjien jne. "index" näkymiin, ettei erillisiä muokkaus ja lisäyssivuja tarvitsisi olla – tämän on tarkoitus helpottaa järjestelmän käyttöä, sillä tarkoituksena on olla mahdollisimman helppokäyttöinen ja kevyt sovellus.

Normaalikäyttäjille ei näy yleisraportti tai muiden käyttäjien tiedot.

7. ASENNUSTIEDOT

Oletetaan, että asentajalla on LAMP stack toiminnassa. Tällöin riittää vetää sovelluksen GitHub repo ja laittaa CONFIG.php tiedostoon tietokannan tunnukset. Tietokannan saa pystyyn SQL lauseilla, jotka löytyvät kansiosta /sql/.

Huom. .htaccess on erittäin tärkeä sovelluksen toimivuudelle etenkin siltä osalta, että se ohjaa kaikki pyynnöt index.php tiedostolle reititettäväksi.

Tällä hetkellä sovelluksen tietokanta on eri palvelimella kuin itse sovellus ja tietokantayhteys tapahtuu julkiverkon yli (voin käyttää samaa kantaa useassa devausympäristössä) – oikeassa tilanteessa tietokantapalvelimeen yhdistettäisiin tietenkin lähiverkon yli (kun samassa pilvessä ovat) tai localhost kautta, jos kanta on samalla koneella.

8. KÄYTTÖOHJEET

Sovellusta on mahdoton käyttää ilman sisäänkirjausta, sillä .htaaccess ohjaa kaiken liikenteen index.php tiedostoon, joka taas ei päästä etenemään ellei ole kirjautunut sisään.

Sisäänkirjautumisen (testitunnus u: user1, p: user1) jälkeen sovelluksen käyttö on melko vaivatonta; listausnäkymistä voi suoraan tehdä CRUD operaatiot Bootstrap3 modalin kautta ('edit' avaa tämän).

HUOM! Tällä hetkellä CRUD operaatiot toteuttu vain "Asiakastunnit" osalta. Näissä myös monesta moneen suhde käyttäjiin, ja syötteeksi pitääkin antaa valideja käyttäjiä pilkulla eroteltuna.

9. TESTAUS, BUGIT

Ohjelmalla ei ole automatisoituja testejä, mutta ohjelmaa on käytetty käsin runsaasti.

9.1 PUUTTEET, BUGIT, JATKOKEHITYSIDEAT

Hyväksyttyjen työtuntien muokkaaminen ja poistaminen on edelleen mahdollista – ohjelma ei myöskään tarkista MITÄ tunteja käyttäjä on tarkalleen hyväksymässä (käyttäjää tosin muistutetaan, että hyväksymällä työtunnit hän vakuuttaa ne oikein merkatuiksi).

Aggregoitu data on melko suppeaa, esimerkiksi mitään palkkakustanuksiin tai palkanmaksuun liittyvää toiminnallisuutta ei ole. Ohjelmaan olisi voinut implementoida palkkaluokat ja näiden avulla laskea työntekijöiden palkat ja muodostaa dataa palkkakustannuksista.

10. OMAT KOKEMUKSET

PHP never again.

Mielestäni PHP-Apache duo on 90-luvun teknologiaa, jonka tunteminen ja osaaminen on hyödyllistä, jos haluaa parempaa kontekstia nykyisiin suosittuihin teknologioihin kuten Ruby on Rails, node.js, django, jne. Halusin Tsohassa oppia miten asiat toimivat ns. "under the hood", joten en ottanut käyttöön frameworkkeja ja rakensin myös LAMP stackin itse Ubuntu virtuaalipalvelimille.

Kielenä PHP on suhteellisen tehoton dynaamisesti tyypitetty kieli – aivan kuten Ruby, JavaScript ja Python, jotka kaikki ovat mielestäni paljon tätä mielekkäämpiä. Tyyppiturvallisuutta taas voi hakea esim. Scalasta, joka on noussut viimeaikoina web-backendeissa esim. Twitter ja LinkedIn toimesta. En siis oikeastaan nää syytä PHP:n käyttämiseen vuonna 2014 (vaikka Facebook tätä käyttääkin jossain muodossa, joka tuskin muistuttaa paljoa sitä PHP:ta mitä nyt itse koodasin).

Kiitos PHP ja LAMP.

Projekti oli työläs ja melko kylmää kyytiä verrattuna Rails/Node.js web-kehitykseen, mutta mielestäni olen oppinut paljon asioita, joita näiden frameworkkien automaatio piilottaa alleen. Hahmotan nyt tietokantoja käyttävät web-sovellukset paremmin ja ymmärrän myös nyt paremmin mitä asioita haluaa piilottaa automaation alle ja mitä ei. Olen myös aika varma, että jos nyt alottaisin uuden PHP/LAMP projektin, niin sujuisi se mukavammin.

11. LIITTEET

Kaikki kaaviot upotettu tähän dokumenttiin.

Projektin github: https://github.com/EliGit/Tsoha