Tietokantasovellus: muistilista.

Sisällysluettelo:

1. Johdanto

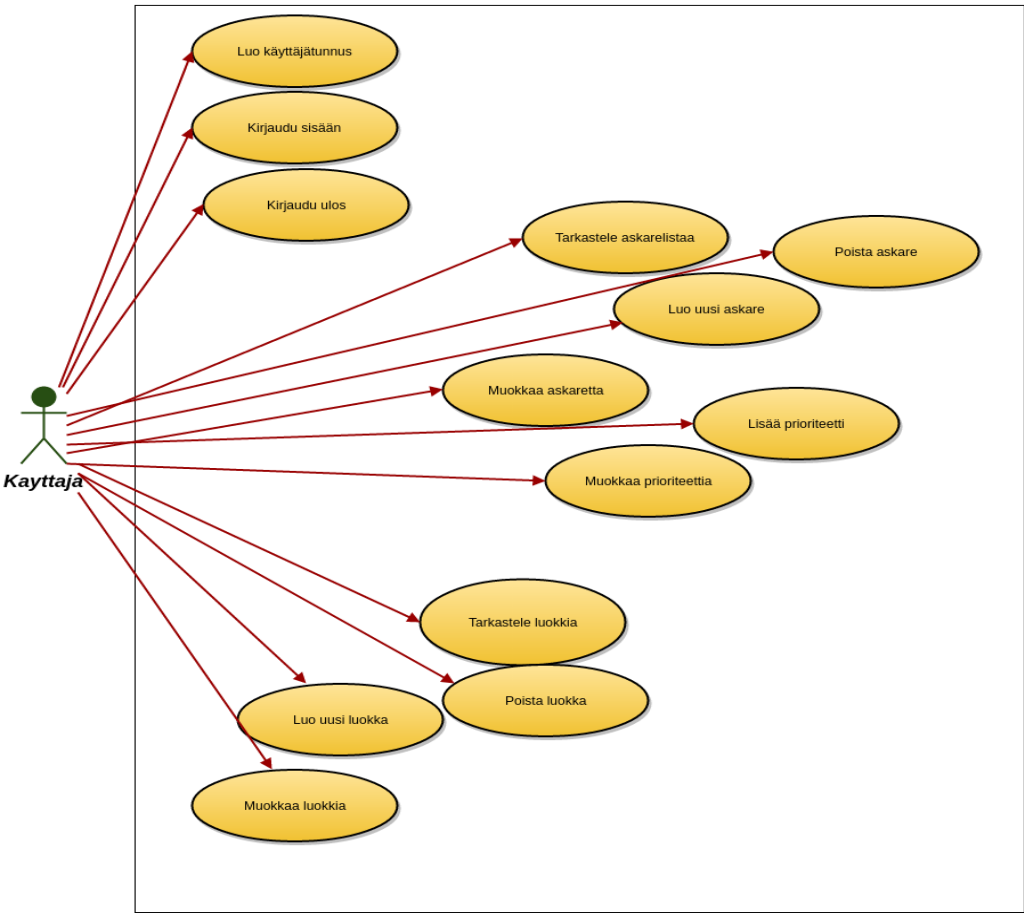
Työn tarkoitus oli luoda käyttäjäkohtainen web-sovellus, joka toimii muistilistana. Muistilistan avulla on mahdollista pitää kirjaa esimerkiksi kuukauden askareista niin, että ne ovat mielekkäästi luokiteltu ja priorisoitu.

Muistilistaan käyttäjä voi kirjata tehtäviä askareita, määrittää niille prioriteetin sekä luokitella askareet kategorioihin. Askareita ja luokkia on myös mahdollista muokata sekä poistaa. Käyttäjä pystyy tarkastelemaan askareita annetussa prioriteettijärjestyksessä, myös luokilla on oma näkymä. Koska askarelistat, niihin liittyvät prioriteetit ja luokat ovat käyttäjäkohtaisia, sovellusta voi käyttää vain kirjautuneena sisään. Käyttäjä voi myös luoda itselleen uuden käyttäjätilin.

Muistilista on web-sovellus, joka toteutetaan tietojenkäsittelytieteen laitoksen users-palvelimella. Sovelluksen alustan on tuettava PHP:ta. Sovellus kommunikoi SQL-tietokannan kanssa, ja tietokannan kielenä toimii PostgreSQL.

2. Yleiskuva järjestelmästä

Käyttötapauskaavio:



Käyttäjäryhmät:

Rekisteröitynyt *käyttäjä* on henkilö, joka on luonut itselleen käyttäjätunnukset sovellukseen.

Käyttötapauskuvaukset:

- Käyttäjä

1) Käyttäjätunnuksen luominen: käyttäjä voi antaa itselleen sopivan nimimerkin ja salasanan, jotka tallentuvat tietokantaan.

2) Kirjautuminen sisään: käyttäjä voi kirjata itsensä sisään antamalla käyttäjätunnuksen ja salasanan. Ilman kirjautumista sovelluksen muita käyttötapauksia ei voi toteuttaa.

3) Kirjautuminen ulos: käyttäjä voi kirjata itsensä ulos painamalla

uloskirjautumispainiketta.

4) Askarelistan tarkastelu: käyttäjä voi tarkastella askareitaan askarelistalla

5) Uuden askareen luonti: käyttäjä voi luoda uuden askareen “luo uusi askare”-painiketta painamalla ja syöttämällä askareen tiedot lomakkeeseen. Samalla käyttäjä voi luokitella askareen haluamiinsa luokkiin.

6) Askareen poisto: Käyttäjä voi poistaa askareen painamalla askareen kohdalla olevaa poistopainiketta joko askarelistassa tai tarkastelemalla yksittäistä askaretta askarenäkymässä.

7) Askareen muokkaus: käyttäjä voi muokata askaretta painamalla askarelistassa olevaa muokkauspainiketta askareen kohdalla, jolloin hän pääsee muokkaamaan askareen tietoja lomakkeessa.

8) Prioriteetin lisääminen: käyttäjä voi lisätä askareelleen prioriteetin samalla, kun hän luo askareen tai muokkaa askaretta. Askareella on oltava prioriteetti.

9) Prioriteetin muokkaus: käyttäjä voi muokata prioriteetiä samassa näkymässä, missä hän muokkaa askaretta.

10) Luokkien tarkastelu: käyttäjä voi tarkastella luokkia omassa näkymässään.

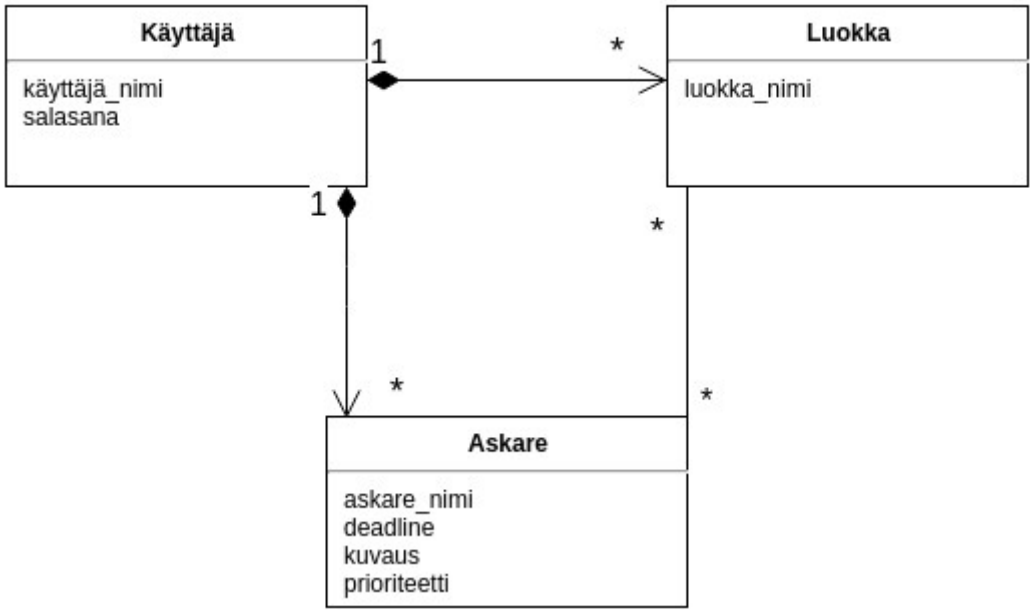
11) Uuden luokan luonti: käyttäjä voi luoda uuden luokan kirjoittamalla luontikenttään luokan nimen ja painamalla “luo uusi luokka”-painiketta.

12) Luokan poisto: käyttäjä voi poistaa luokan luokkanäkymässä painamalla halutun luokan kohdalla poistopainiketta.

13) Luokan muokkaaminen: käyttäjä voi muokata luokkaa painamalla luokkanäkymässä muokkauspainiketta, jolloin ilmestyvään tekstikenttään hän voi tehdä haluamansa muutokset.

3. Järjestelmän tietosisältö

Käsitekaavio



Tietokohde: *Käyttäjä*

Nimi	Arvojoukko	Kuvailu
kayttaja_nimi	Merkkijono	Käyttäjän nimi
salasana	Merkkijono	Käyttäjän salasana

Käyttäjä voi luoda monta luokkaa ja monta askaretta, ja hän voi myös luokitella askareen moneen luokkaan.

Tietokohde: *Luokka*

Nimi	Arvojoukko	Kuvailu
luokka_nimi	Merkkijono	Luokan nimi

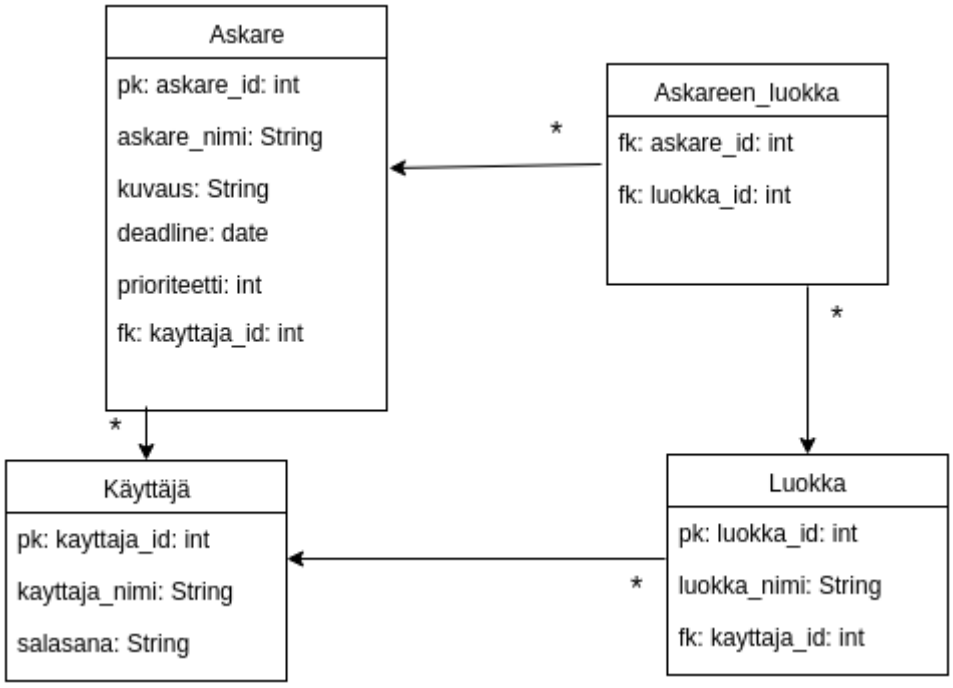
Luokkaa kohden voi olla vain yksi käyttäjä. Luokka voi liittyä moneen askareeseen, mutta sen ei ole pakko kuulua yhteenkään askareeseen.

Tietokohde: *Askare*

Nimi	Arvojoukko	Kuvailu
askare_nimi	Merkkijono	Askareen nimi
deadline	Päivämäärä	Askareen eräpäivä
kuvaus	Merkkijono	Askareen kuvaus
prioriteetti	Positiivinen kokonaisluku	Askareen prioriteetti

Askareella voi olla vain yksi käyttäjä. Askaretta ei voi olla ilman käyttäjää. Askaretta kohden voi olla monta luokkaa, mutta askareella ei tarvitse olla yhtäkään luokkaa.

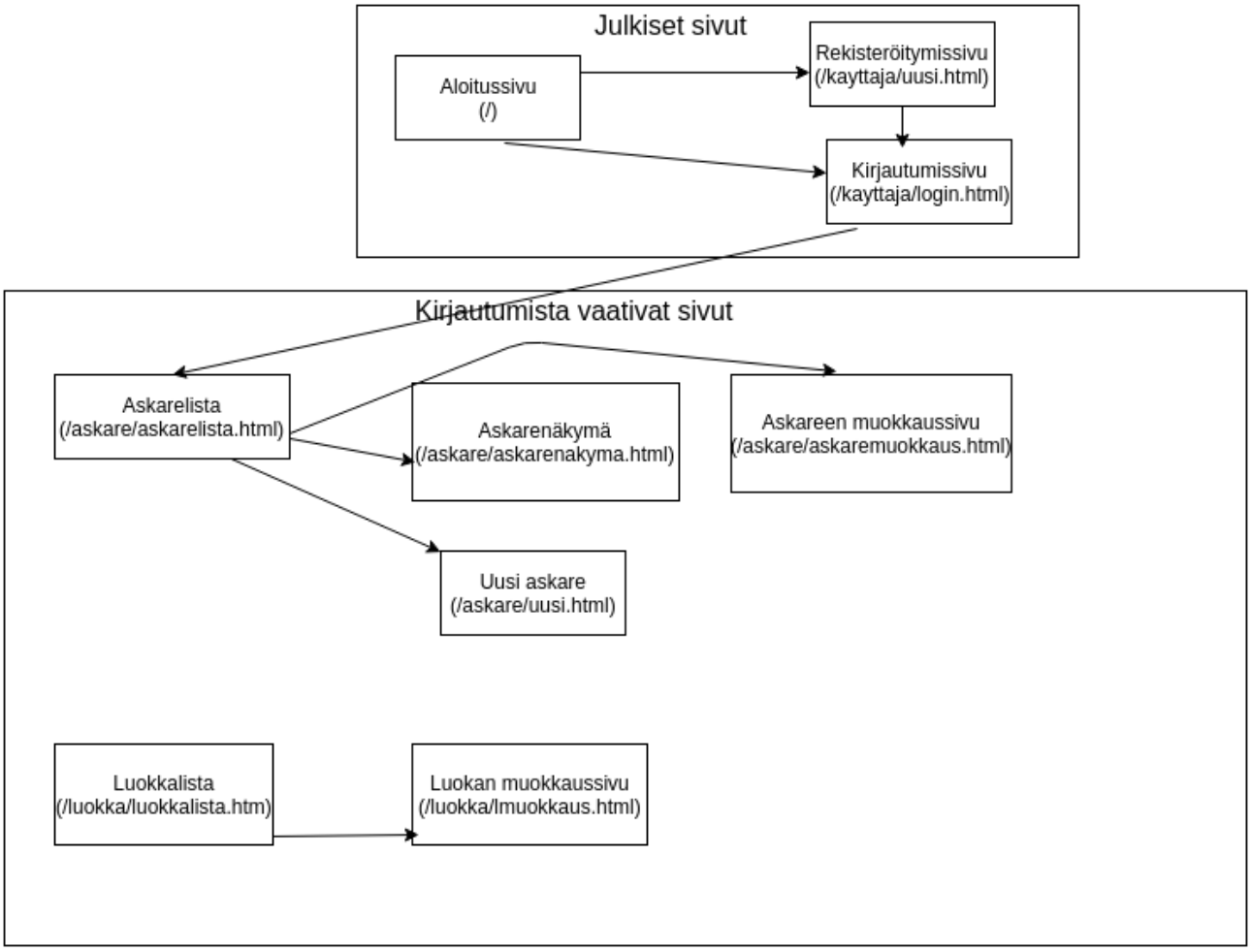
4. Relaatiotietokantakaavio



5. Järjestelmän yleisrakenne

Sovellus noudattaa MVC-mallia. Mallit, kontrollerit ja näkymät löytyvät Models, Views, ja Controllers-kansioista. Käytetyt apukirjastot löytyvät kansiosta lib.

6. Käyttöliittymä ja järjestelmän komponentit



7. Asennustiedot

Sovelluksen voi asentaa kopioimalla tiedostot palvelimen nettiin näkyvään kansioon, esimerkiksi laitoksen users-kansion htdocs-hakemistoon. Tämän jälkeen yhteystiedot korjataan oikeiksi tiedostossa libs/config.php.

8 . Käynnistysohje

Sovellus on asennettu osoitteeseen http://katukatu.users.cs.helsinki.fi/tsoha_tmkau. Sovellusta käytetään kirjautumalla sisään tai luomalla käyttäjätunnukset. Sovelluksessa on navigointipalkki, josta voi päästä tärkeimpiin näkymiin. Uuden käyttäjän kannattaa ensiksi lisätä jokin luokka muistilistaan, jotta hän voisi uutta askaretta luodessaan luokitella sen tähän luokkaan.

Sovelluksessa testaamiseen sopivat esimerkiksi seuraavat nimimerkki/salasanakombinaatiot:

a) tunnus: Heikki, salasana: Heikki123

b) tunnus: Kirka, salasana: Kirka123

c) tunnus: Hello, salasana: Hello123

9. Testaus, tunnetut bugit ja jatkokehitysideat

Sovellusta on testattu käyttämällä sitä mahdollisimman laajasti ja antamalla mahdollisimman erilaisia syötteitä. Yksi tunnettu bugi on, että luokkien muokkauspainiketta painaessa yksi painallus ei riitä, vaan pitää klikata nappia kaksi kertaa.

Seuraava askel sovellusta kehittäessä olisivat ehdottomasti erilaiset askareiden lajittelutoiminnot, jolloin askareita voisi tarkastella mielekkäästi esimerkiksi deadlinen tai luokan perusteella, tai että askareet voisi järjestää aakkosjärjestykseen. Olisi myös hyvä, jos käyttäjä pystyisi asettaa askareen deadlineksi muunlaisia aikaformaatteja kuin vvvv-kk-pp, esimerkiksi että deadlinessa olisi vain päiväys ja kellonaika. Ohjelman käytettävyys myös ehkä hieman paranisi, jos käyttäjä voisi saada valmiiksi muutaman “oletusluokan”, joihin hän voisi luokitella askareitaan, ja että hän voisi halutessaan lisätä omia luokkia ja poistaa oletusluokkia.

10. Omat kokemukset

Vaikka olen tehnyt tämän sovelluksen käyttämällä mm. php:ta, en silti ole oppinut vielä hyväksi php:n syntaksin käyttäjäksi, ja tämä varmasti vaikutti esimerkiksi koodin laatuun. MVC-mallin perusteet luulen nyt osaavani ja ymmärrän ainakin perustasolla, miten tietoa kuljetetaan eri komponenttien välillä. Parametrien kuljetus komponenttien välillä olikin yksi haastavimmista asioista ohjelmaa tehdessä. En kehittynyt kurssin aikana yhtään paremmaksi tietokantojen hallinnassa, mutta ymmärrän ehkä paremmin, miten kannattaa laatia tietokantataulu sovellustarkoituksia ajatellen. Jouduin ohjelmaa tehdessäni muokkaamaan alkuperäistä SQL-tietokantaani kohtalaisen paljon.