Tasausmaksujen hallintasovellus

Tekijä: Mikko Ruuskanen

Johdanto

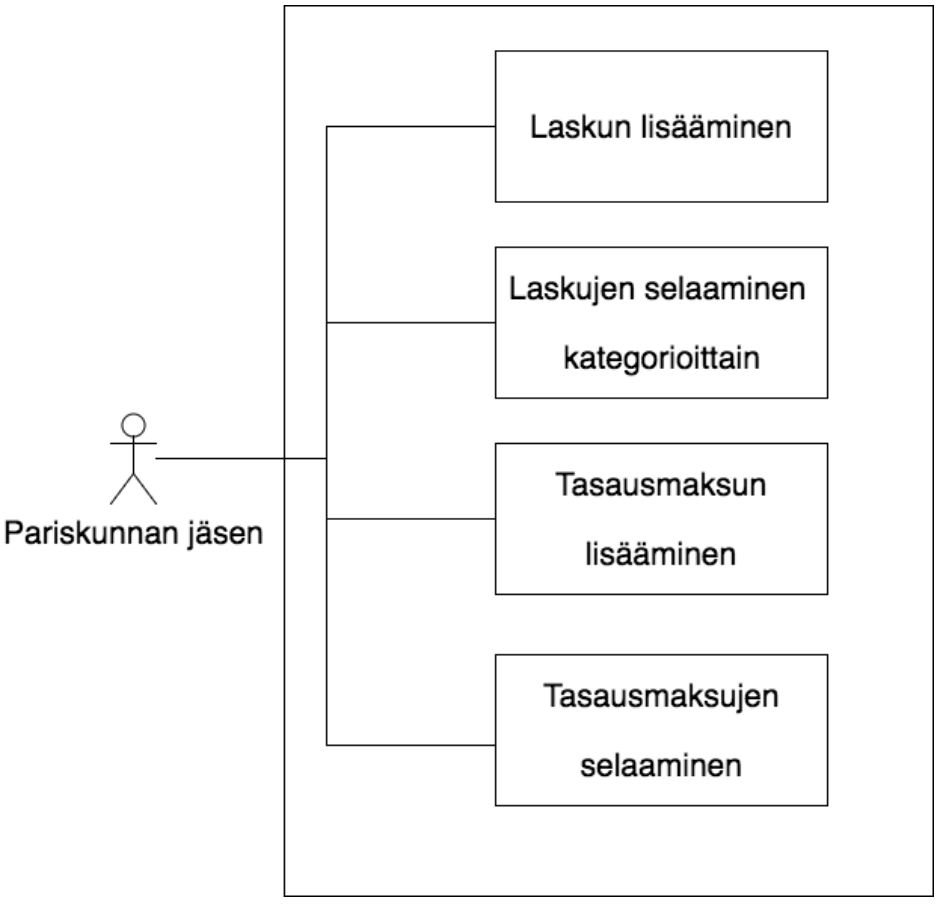
Sovellus on suunnattu pariskunnalle, joilla ei ole yhteistä tiliä, ja jotka haluavat jakaa maksamansa kulut tasan.
Käyttäjä voi kirjata sovelluksella maksamansa laskun ja antaa laskulle kategorian sekä päivämäärän seurantaa varten. Sovellus laskee jokaiselle kuukaudelle tasausmaksun jommallekummalle käyttäjistä. Käyttäjä voi kirjata maksamansa tasauksen sovelluksen avulla.
Käyttäjät voivat myös nähdä kuluraportin, jossa on eritelty jokaisen kategorian kulut vuosittain.

Sovellus toimii Helsingin yliopiston tietojenkäsittelytieteen laitoksen users-palvelimella. Alustajärjestelmänä toimii *node.js*. Sovelluksen käyttäjän selaimen täytyy tukea JavaScriptiä, sillä käyttöliittymä on toteutettu kokonaan Reactilla ja Reduxilla.

Sovellus toimii toistaiseksi vain sitä varten pystytetyllä PostgreSQL –tietokannalla. Tietokannan vaihtaminen vaatii muutoksia sovelluksen konfiguraatioon.

Yleiskuva järjestelmästä

Käyttötapaukset

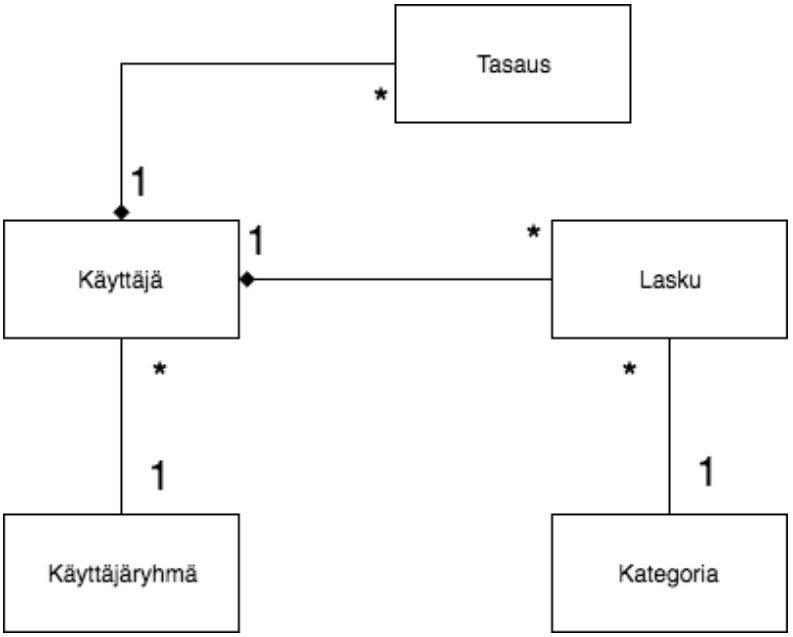


- Laskun lisääminen
    - Käyttäjä lisää sovellukseen maksamansa laskun ja antaa sille kategorian
- Laskujen selaaminen kategorioittain
    - Käyttäjä näkee vuosittain ja kategorioittain maksettujen laskujen euromäärät yhteensä
- Tasausmaksujen selaaminen
    - Käyttäjä näkee kenen käyttäjän tulee maksaa tasausmaksu tietylle kuukaudelle, kyseisen kuukauden laskujen perusteella
- Tasauksen lisääminen
    - Käyttäjä lisää maksamansa tasausmaksun sovellukseen

Muita käyttötapauksia: kirjautuminen, käyttäjäryhmään kirjautuminen, käyttäjäryhmän luonti, uuden käyttäjän luonti käyttäjäryhmään

Järjestelmän tietosisältö

Käsitekaavio



Tietokohde: Käyttäjä (=User)

Attribuutti	Arvojoukko	Kuvailu
username	Merkkijono	Käyttäjän nimi
password	Merkkijono	Salasana. Tallennetaan kryptattuna.

Attribuutti	Arvojoukko	Kuvailu
user_group_id	Kokonaisluku	Viittaus käyttäjän käyttäjäryhmään.

Tietokohde: Käyttäjäryhmä (=UserGroup)

Attribuutti	Arvojoukko	Kuvailu
name	Merkkijono	Käyttäjäryhmän nimi
password	Merkkijono	Salasana. Tallennetaan kryptattuna.

Tietokohde: Lasku (=Payment)

Attribuutti	Arvojoukko	Kuvailu
user_id	Kokonaisluku	Viittaus käyttäjään
category_id	Kokonaisluku	Viittaus laskun kategoriaan
amount	Desimaaliluku	Laskun summa
date	Päivämäärä	Laskun päivämäärä

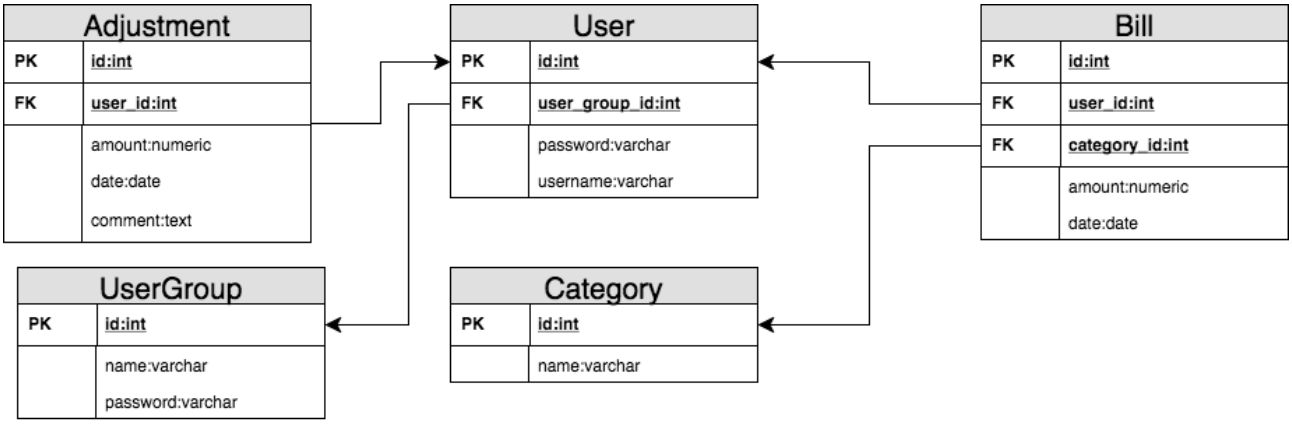
Tietokohde: Kategoria (=Category)

Attribuutti	Arvojoukko	Kuvailu
name	Merkkijono	Kategorian nimi

Tietokohde: Tasaus (=Adjustment)

Attribuutti	Arvojoukko	Kuvailu
user_id	Kokonaisluku	Viittaus käyttäjään
amount	Desimaaliluku	Tasausmaksun summa
date	Päivämäärä	Tasauksen maksupäivä
comment	Merkkijono	Tasauksen kommentti.

Relaatiotietokantakaavio



Järjestelmän yleisrakenne

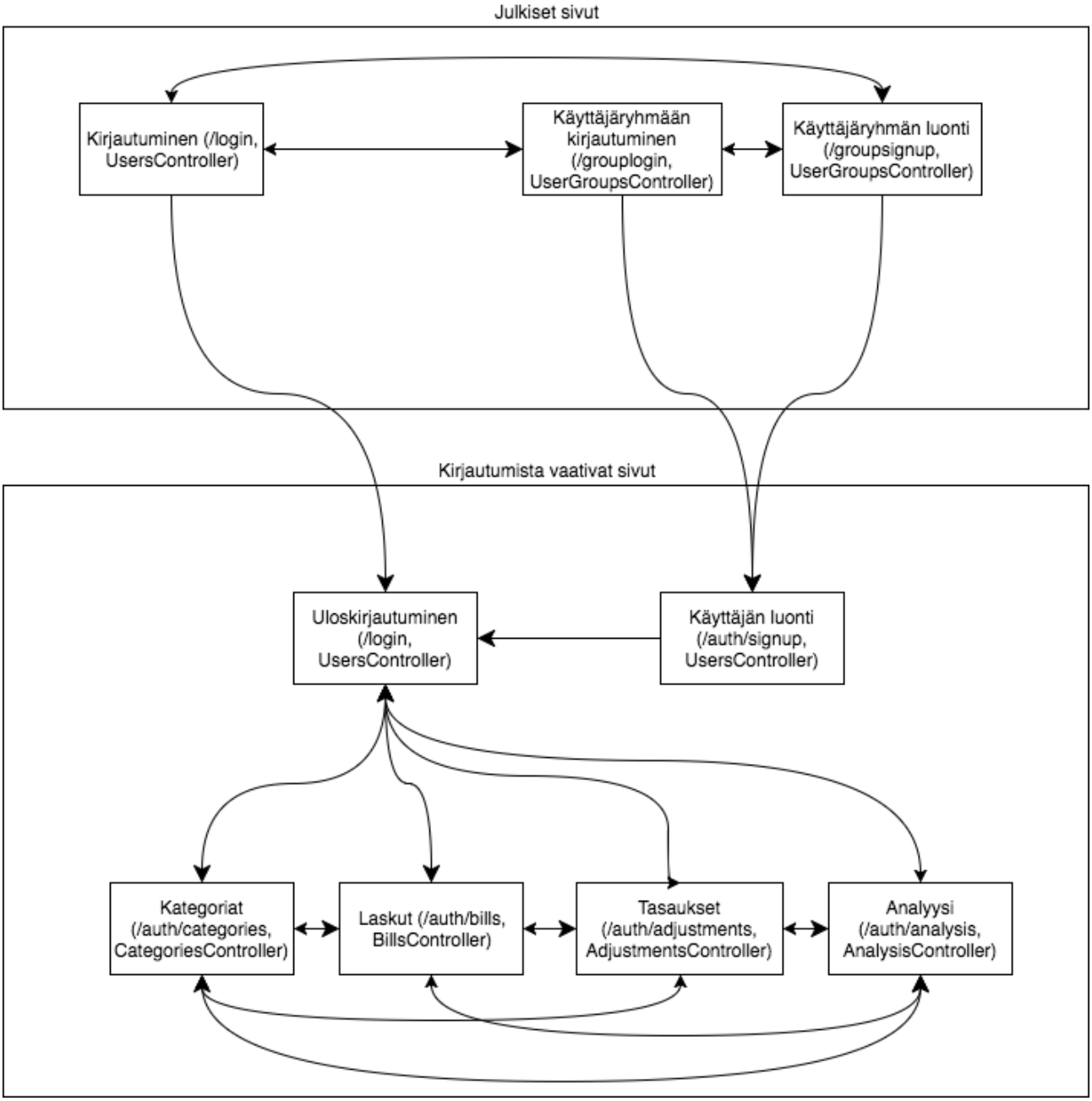
Sovelluksessa on käytetty MVC-mallia siten, että näkymät on toteutettu kokonaan omana asiakassovelluksenaan *React*:illa ja *Redux*:illa, ja kontrollerit ja mallit erillisenä palvelinsovelluksena *node.js:*llä.

Sovelluksen käyttöliittymän logiikka on *client/src* –kansion alla. *client/public/index.html* tiedostossa on määritetty näytettävä HTML-sivu, jossa on yksinkertaisesti määritetty *div*-elementti id:llä *root. client/src/index.js* on käyttöliittymän logiikassa ylimmällä hierarkiatasolla: siinä määritetään, mitä mainittuun *div*-elementtiin käytännössä tulee. Varsinainen käyttöliittymän sisältö määräytyy *client/src/routes.js* tiedoston reittien ja niitä vastaavien *Root* –komponenttien perusteella.

client/src –kansion alla on alikansio jokaiselle käyttöliittymän näkymälle. Näkymien tarvitsemat yhteiset komponentit ovat *shared* –kansion alla.

Palvelin käynnistetään tiedostossa *server.js*. Kyseisessä tiedostossa viitataan tiedostoon *backend/routes.js*. *routes.js* - tiedostossa määritetään, mitä endpointteja (esim. GET /adjustments) palvelin tukee, ja minkä kontrollerin mikäkin metodi on vastuussa ko. endpointin tietojen tuottamisesta. Kontrollerit ovat kansiossa *backend/controllers* ja mallit kansiossa *backend/models*.

Käyttöliittymän komponentit



Kaavioon on merkattu jokaisen käyttöliittymän sivun yhteyteen tieto siitä, mistä käyttöliittymän polusta on kyse (esim. /auth/categories), ja mikä palvelimen kontrolleri pääasiassa vastaa sivulla näytettävistä tiedoista (esim. CategoriesController).

Käynnistys/käyttöohje

Tuotannonomainen ympäristö on osoitteessa https://ruuskmik.users.cs.helsinki.fi. Kirjautumaan pääsee käyttäjätunnuksella ”eka” ja salasanalla ”testaa”.

Jos sovellukseen haluaa luoda uuden käyttäjän, tulee ensin luoda käyttäjäryhmä tai kirjautua sisään olemassa olevaan käyttäjäryhmään. Tämän jälkeen voi luoda käyttäjän ko. käyttäjäryhmään. Käyttäjäryhmien tarkoitus on rajata näytettävät tiedot vain ko. käyttäjäryhmän (=pariskunnan) käyttäjien tietoihin.

Asennustiedot

Ennen asennusta

* Apache-konfiguraatio on tehty, eli *.htaccess* tiedostossa oikea konfiguraatio.
* *nvm* on asennettu, ts. komento *nvm --version* toimii.
* *node* on asennettu, ts. komento *node --version* toimii.
* tietokanta sekä tietokantakäyttäjä on luotu.
* *pm2* on asennettu globaalisti, ts. komento *pm2 --version* toimii.

Ainakin *users* -palvelimella *nvm*:n ja *pm2*:n asentaminen vaati seuraavia asetuksia tiedostoon *~/.bashrc:*

```
export NVM_DIR="$HOME/.nvm"
[ -s "$NVM_DIR/nvm.sh" ] && \. "$NVM_DIR/nvm.sh"
```

Näiden avulla *nvm* -skripti on ajettavissa palvelimella, ja tätä kautta ilmeisesti myös *pm2*.

Asennus

1. luo build käyttöliittymäkoodista seuraavilla komennoilla juurihakemistosta alkaen:
* *cd client*
* *npm run build*
2. kopioi sovelluksen tiedostot palvelimen nettiin näkyvään kansioon.
3. luo sovelluksen juurikansioon *.env* tiedosto, johon lisäät seuraavat tiedot:

```
JWT_SECRET=[jokin vaikeasti arvattava merkkijono. käytetään JavaScript WebToken -avainta varten]
DB_NAME=[tietokannan nimi]
DB_USER=[tietokantakäyttäjän nimi]
DB_PASS=[tietokantakäyttäjän salasana]
```

4. aja komento *npm install* juurikansiossa. Tämä asentaa palvelinkoodin riippuvuudet.
5. käynnistä sovellus komennolla *pm2 start ecosystem.json --env production.* Käytännössä sovelluksen tiedot haetaan siis tiedostosta *ecosystem.json,* joka on osa repositoriota. Vastaavaan lopputulokseen pääsisi ajamalla *node server.js*, mutta tällöin ei käytettäisi *pm2*:n toiminnallisuuksia ollenkaan.

Jatkokehitysideat