Arttu Pennanen, Henri Vuento, Atte Räisänen, Perttu Vaarala

Kideappdataanalyysivehje toteutus-dokumentti

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tieto- ja viestintäteknologian tutkinto-ohjelma

Toteutusdokumentti

5.3.2023

Sisällys

1 Johdanto

Kideappdataanalyysivehje on Kide.app palvelun datan tarkkailuun luotu sovellus. Sovellus parsii Kide.app palvelun rajapinnasta dataa ja havaitsee siinä ilmaantuvia trendeja. Sovellus toimii apuvälineenä opiskelijaelämän aikataulutukseen sekä siivittää käyttäjä tekemään tietoon perustuvia arvauksia opiskelijakulttuurin tulevaisuudesta.

Sovelluksen toisena tavoitteena on auttaa käyttäjää pitämään kirjaa mielenkiintoisista tapahtumista ja auttaa muistuttaa niiden ajankohdista.

Oppimismielessä projekti kehittää projektin jäsenten ymmärrystä datan aggregoinnista. Sovellus on JavaFX -työpöytäsovellus.

2 Tuotteen vaatimukset

Tuote tarjoaa käyttäjille tavan analysoida ympärillä tapahtuvaa toimintaa. Sovellus tarjoaa paikan oppia kiinnostuksen lähteistä. Tuotteen omistajana toimii tuotteen kehittäjät.

2.1 Vaatimukset

Tuotteelle on asetettu vaatimuksia, jotta sen valmistuessa se tarjoaa käyttäjilleen arvokkaita kokemuksia.

Tuotteen on pystyttävä tallentamaan Kide.app palvelusta löytyvää dataa, jotta tuote pystyy muodostamaan historiaa tiedostoa, jota ei pidetä loputtomiin Kide.app palvelusta saatavilla.

Tuotteen on toimittava ketterästi, jotta käyttökokemus ei kärsi. Tuotteen on näytettävä edustavalta, ja pyrkiä olemaan tarpeeksi näyttävä ja asianmukainen.

Tuotteen on pystyttävä tallentamaan käyttäjän valintoja, jotta käyttäjä pystyy palaamaan tarkastelemaan käyttäjää kiinnostavien tapahtumien muutoksia.

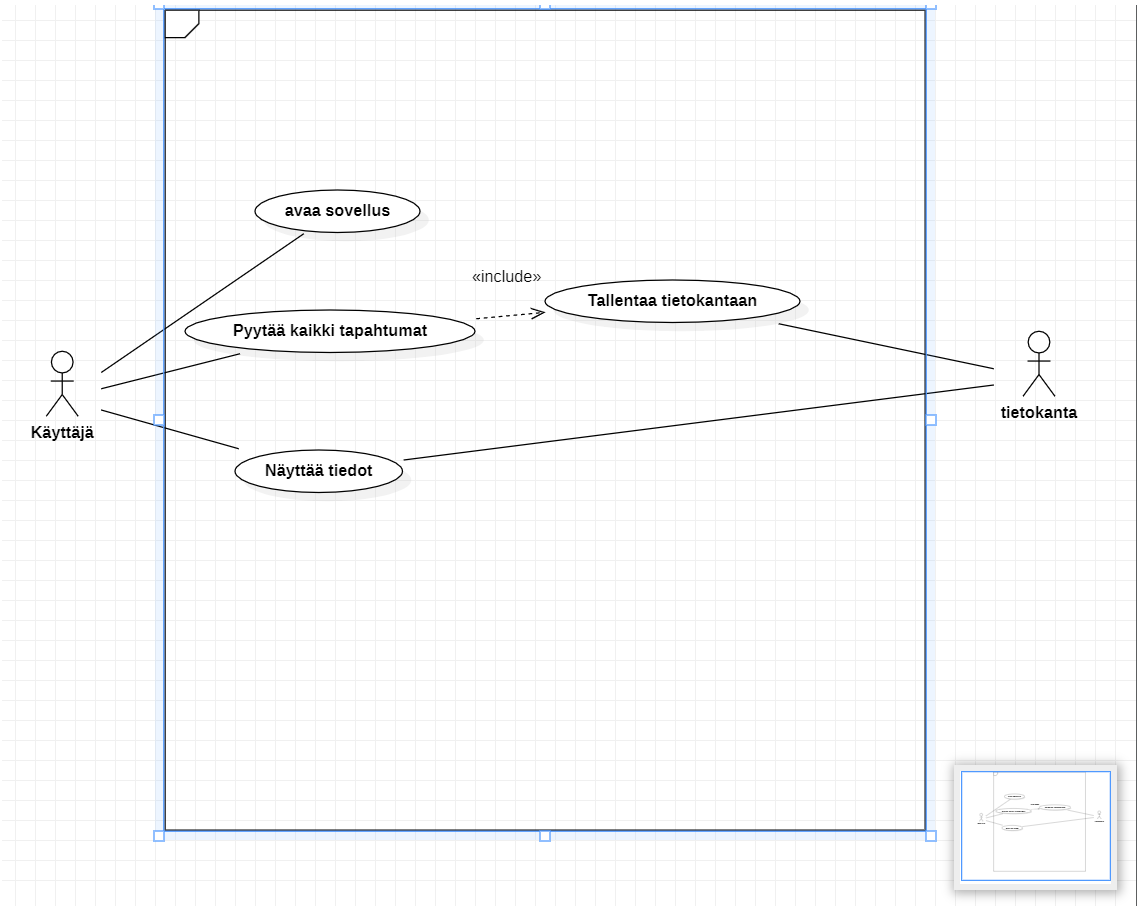
2.2 Käsitteet ja määritelmät

Tapahtuma
tapahtumalla kuvataan yksittäistä kide.app tapahtumaa tiettynä ajanhetkenä. Yksittäinen tapahtuma voi ilmaantua tuotteen kaavioissa ja tietueissa useaan otteeseen, sillä tuote tallentaa tapahtumista tilannevedoksia (en. snapshot).

Tapahtumadatapiste
tapahtumadatapisteellä tarkoitetaan jonkin ajanhetken kokonaistilannevedosta kide.app palvelun datasta. Yksittäinen datapiste sisältää kaikki sinä hetkenä kide.app palvelusta löytyvät erikseen määritettyihin suodattimiin vastaavat tapahtumat.

3 Käyttäjäroolit ja käyttötapaukset

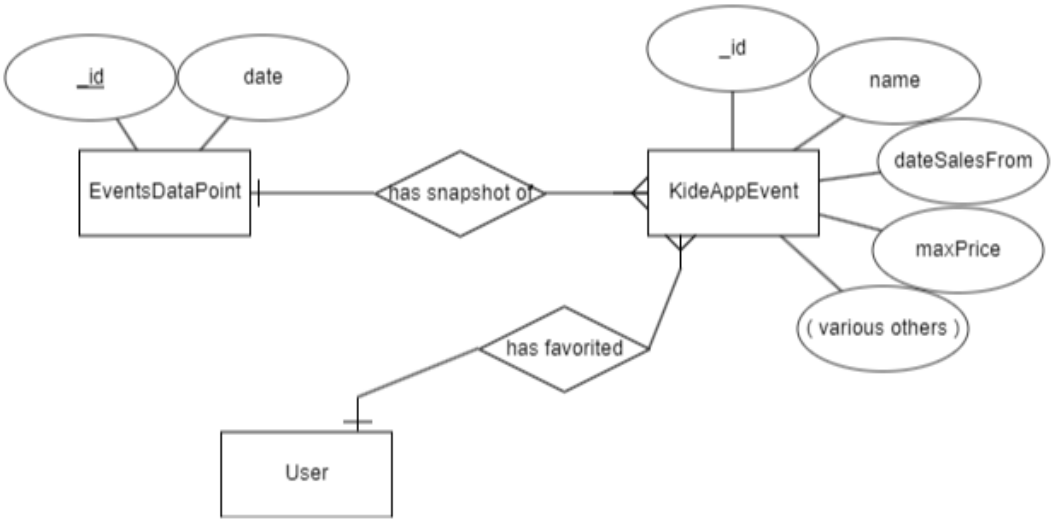
Sovelluksen käyttäjäroolit ovat kahdenlaiset käyttäjä joka on kuka tahansa kide app sovelluksen käyttäjä luultavasti opiskelija joka hakee tietoa tulevista tapahtumista ja lipuista koska haluaa itse osallistua kyseisiin tapahtumiin. Toisena roolina on tapatumien järjestäjät joita taas kiinnostaa enemmän sovelluksen tarjoama statistiikka sitä millon tapatumia on eniten ja miten suosittuja ne ovat jotta he voivat itse opitimoida omien tapahtumien järjestämisen



Kuva 1 Käyttätapauskaavio, jossa käyttäjä hakee kaikki tapahtumat Kide.app palvelusta. Tapahtumat tallennetaan omaan tietokantaan

4 Ohjelmiston tietomalli

Ohjelmiston pääosaisena tietokantana toimii dokumenttikanta MongoDB. Dokumenttikanta valittiin helpottamaan tietojen uudelleentallentamista.



Kuva 2 Korkean tason ER-kaavio

Käytännössä tietokannasta haetaan tapahtumadatapisteitä (EventsDataPoint), jotka sisältävät aiemmin Kide.app sovelluksen omilta palvelimilta saatua tietoa.

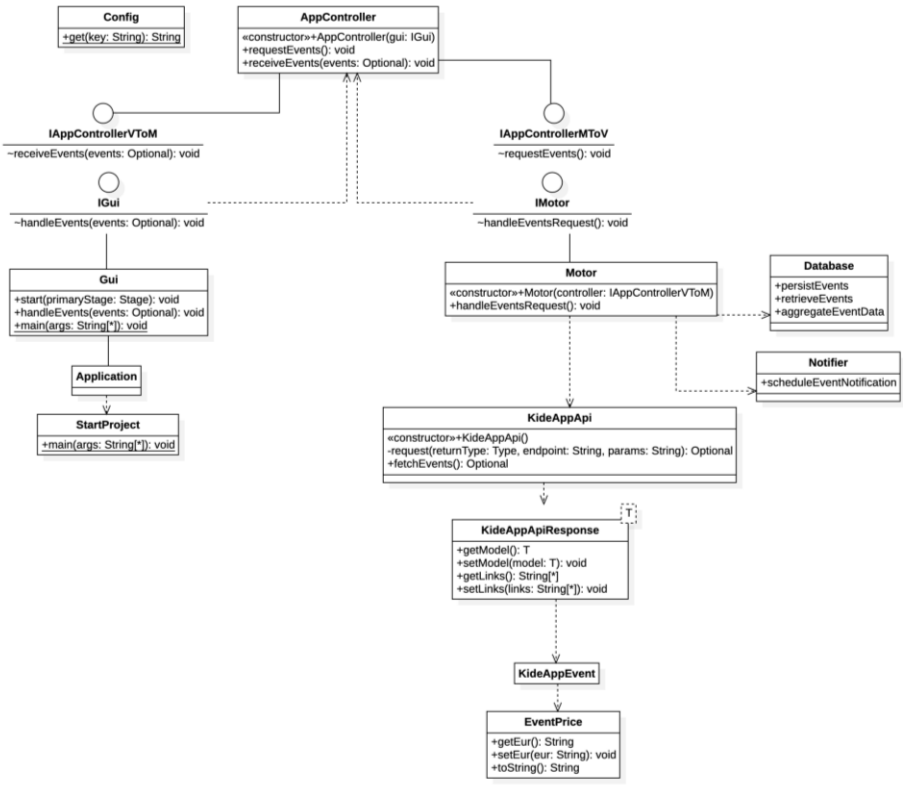
Sovelluksen käyttäjä voi tallentaa yksittäisiä tapahtumia omaan ”My Events” osioon.

Yksittäinen KideAppEvent sisältää monia kymmeniä eri kenttiä. ER-kaaviossa näitä ei ole mallinnettu (Kuva 2). KideAppEvent entiteetin id kenttä on uniikki, mutta yhdestä KideAppEventistä voi olla sovelluksessa useita eri versioita.

5 Ohjelmiston rakenne

Ohjelmisto on kehitetty käyttäen uudelleenmukailtua MVC-arkkitehtuuria (model, view, controller). Ohjelmistossa on kuitenkin otettu vapauksia MVC-arkkitehtuurin rajoihin liittyen sovelluskehityksen nopeuden kasvattamiseksi.

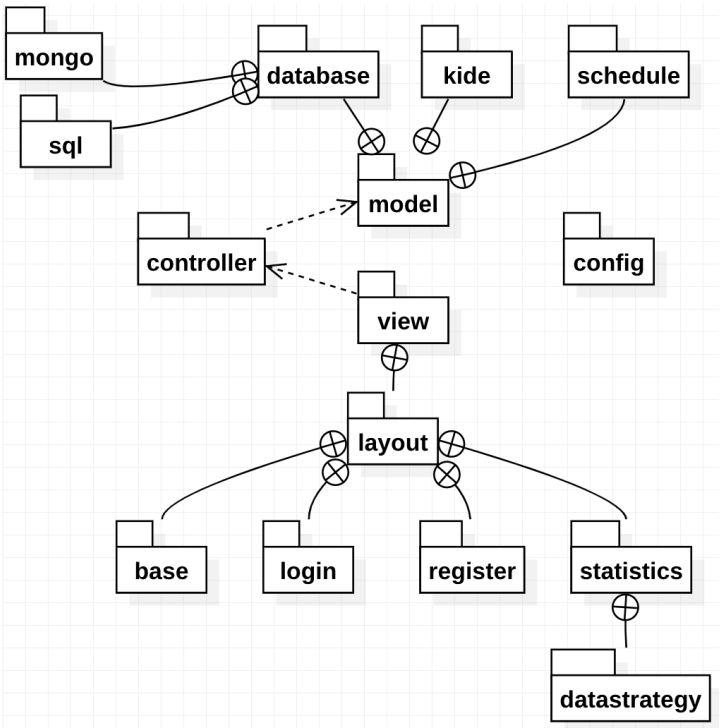
5.1 Luokkakaavio



Kuva 3 korkean tason luokkakaavio

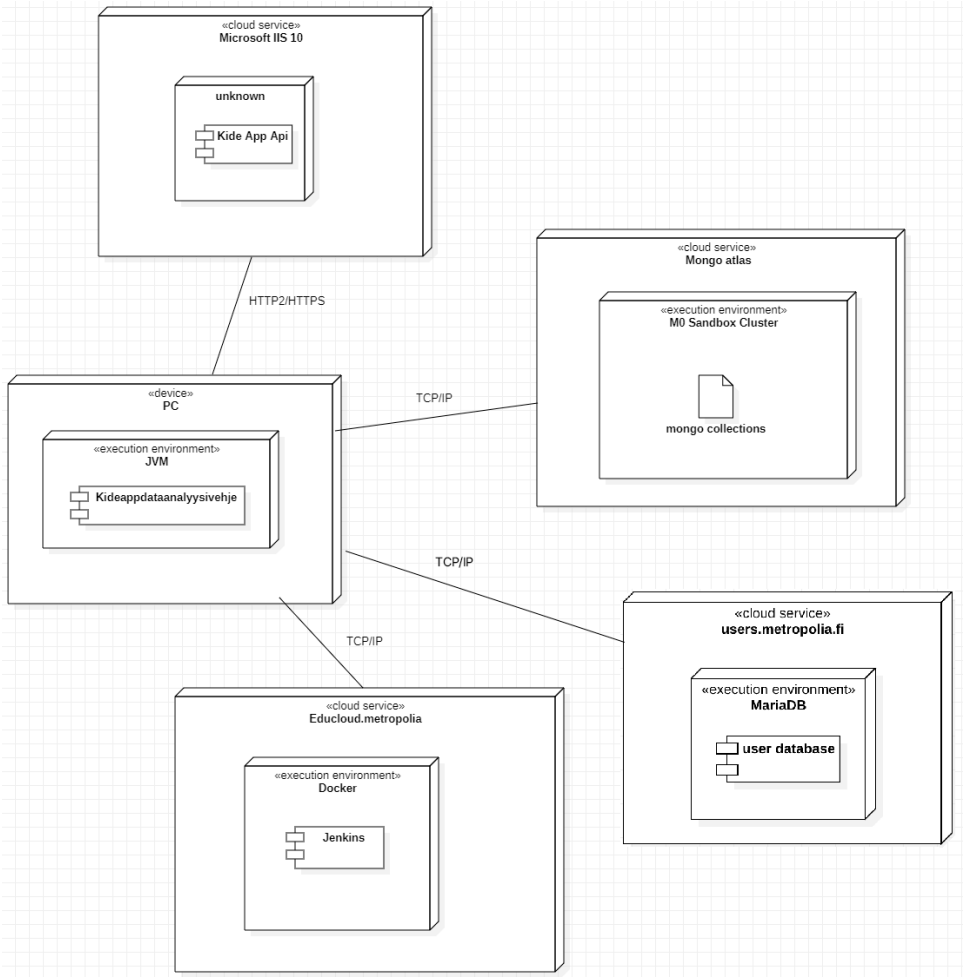
Tieto liikkuu käyttöliittymästä alaspäin. Käyttöliittymässä tehdään pyyntö, controlleri delegoi pyynnön ja lähettää käyttöliittymälle vastauksen. Paikoitellen käyttöliittymä ohittaa controllerin. Näin on tehty sovelluskehityksen nopeuttamiseksi. (Kuva 3.)

5.2 Pakkauskaavio



Kuva 4 Ohjelmiston pakkauskaavio

Ohjelmisto on jaettu MVC-arkkitehtuurin mukaisesti kolmeen pääpakkaukseen, Model, controller, ja view. Neljäntenä pakkauksena on myös config, jota mikä tahansa pakkaus voi tarvittaessaan käyttää. Config tarjoaa ympäristömuuttujia projektiin. Esimerkiksi tietokantojen osoitteet. (Kuva 4.)



Kuva 5 Tuotantosovelluksen sijoittelukaavio

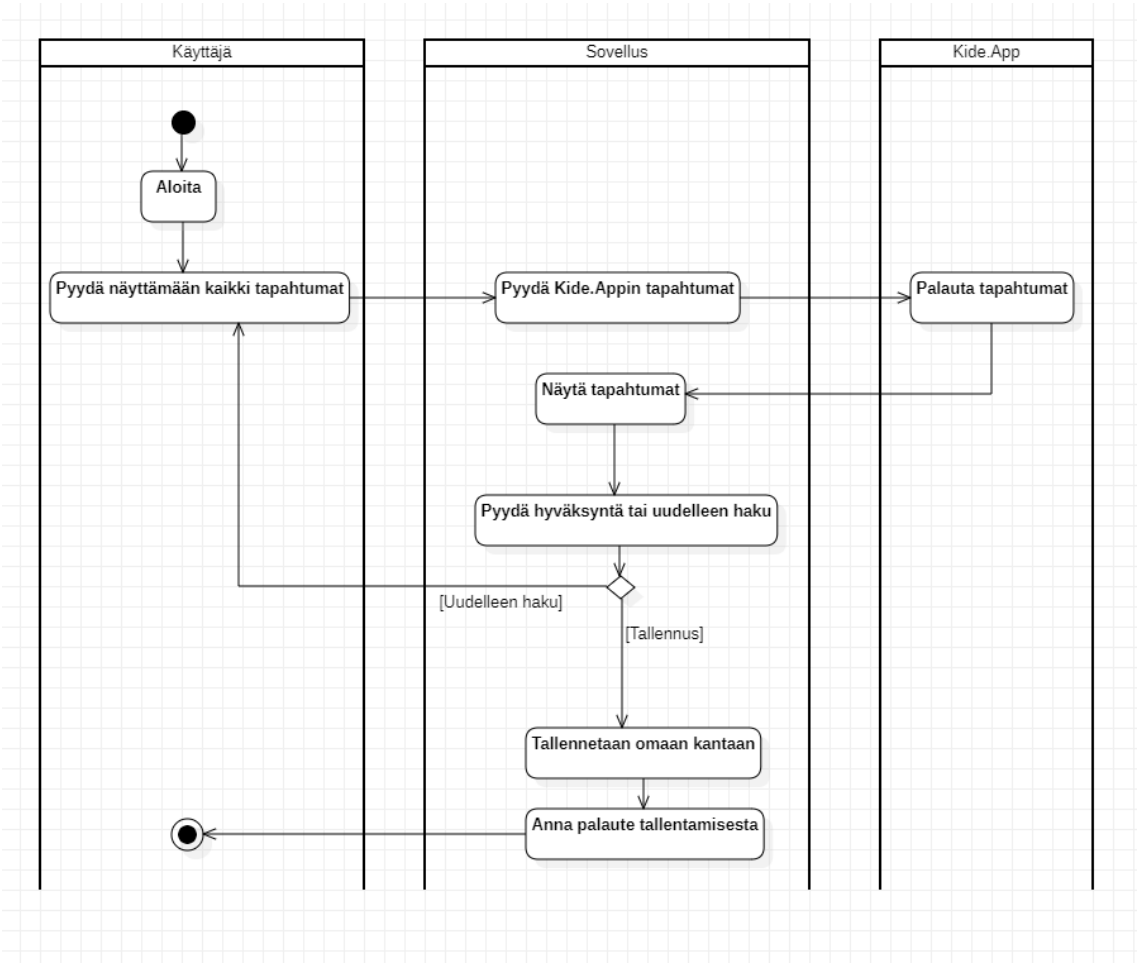
Sovellus ajetaan aina omalla päätelaitteella. Sovellus voidaan ajaa millä tahansa JVM ja JavaFX teknologioiota tukevaan ympäristöön.

Tuotantosovellus on käytön aikana yhteydessä kolmeen eri palvelimeen. Mongo dokumenttitietokantaan, SQL relaatiotietokantaan sekä Kide.app palvelun http rajapintaan (Kuva 5).

Tuotantosovelluksesta uusia versioita tarkastaessa otetaan myös yhteys Metropolian palvelimilla olevaan Jenkins instanssiin, joka tarkastaa, että sovelluksen testit menevät lävitse.

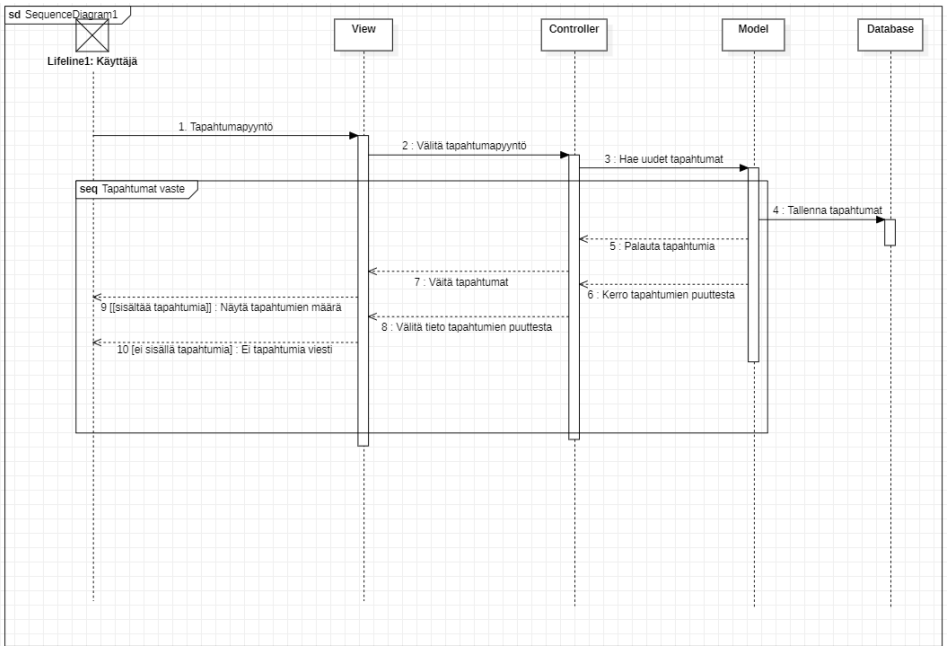
6 Ohjelmiston toiminta

Ohjelmiston toimintaa käsitellessä raporttiin on valittu yksi käyttötapaus, jolla saa yleisen kuvan siitä, miten tuote toimii, ja miten sitä käytetään.



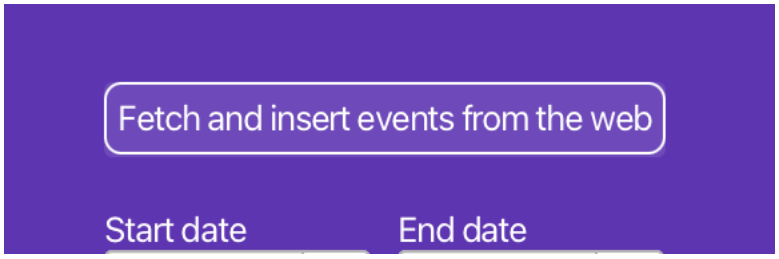
Kuva 6 Aktiviteettikaavio käyttötapaukselle, jossa käyttäjä tahtoo persistoida uusimmat tiedot tietokantaan

Tarkastelunkohteeksi on valittu käyttötapaus, jossa käyttäjä tahtoo tallentaa sillä hetkellä Kide.app palvelusta löytyvät tapahtumat tuotteen omaan tietokantaan (Kuva 6).



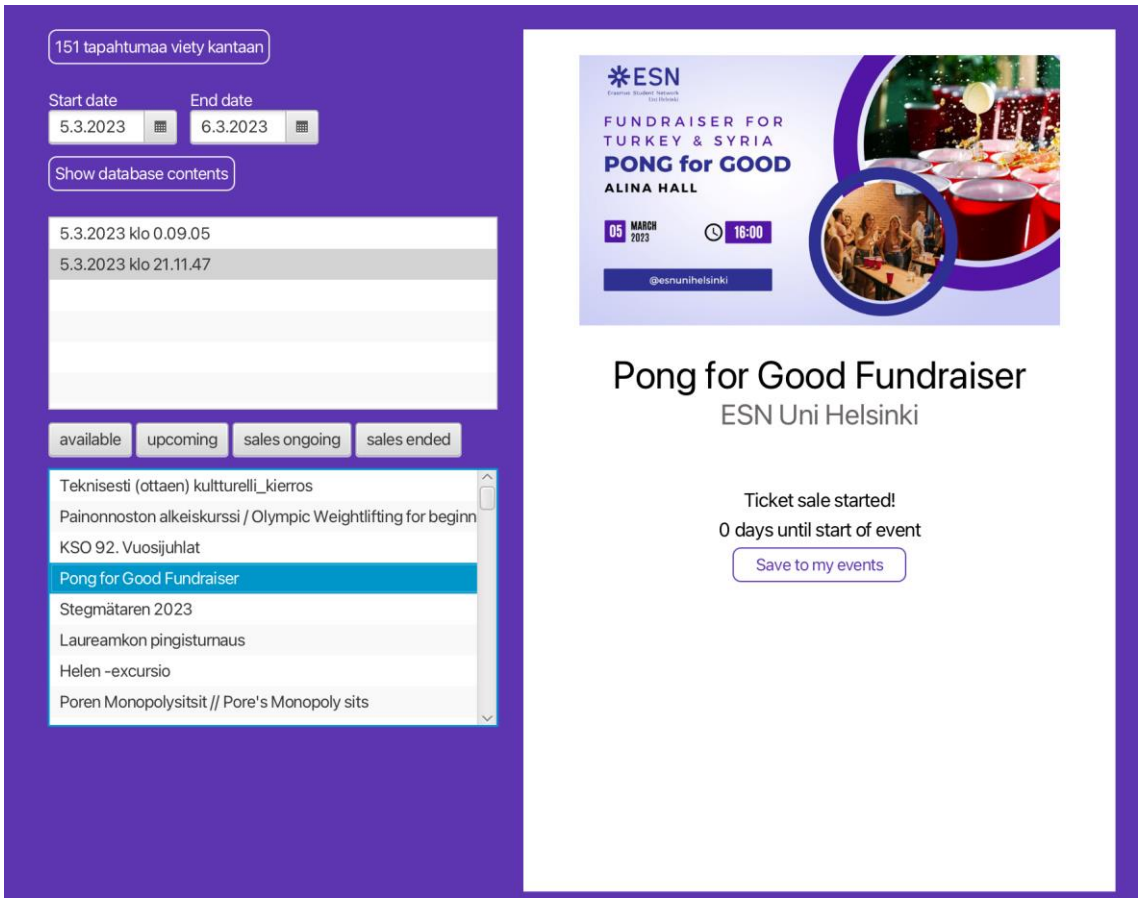
Kuva 7 Sekvenssikaavio samalle käyttötapaukselle kuin kuvassa 6

Yksi käyttötapaus johtaa pyyntöjä useaan eri palveluun. Siksipä on kehittämisen kannalta olennaista, että eri palvelut toimivat tuotteessa harmonisesti.



Kuva 8 Käyttötapauksen mahdollistava nappi käyttöliittymässä

Käyttötapauskokemus optimoitiin mahdollisimman pitkälle ohjelmiston kehityksen aikana. Yksi napinpainallus riittää (Kuva 8).



Kuva 9 uusien tapahtumien tallentamisen jälkeen tapahtumat ovat pysyvästi käytettävissä

Tapahtumat ovat käyttötapauksen mukaisesti saatavilla (Kuva 9).

7 Kehitysprosessi ja kehitysvaiheen tekniikat

Kuvatkaa ohjelmistotuotteen kehitysprosessi sekä käyttämänne kehitys- ja testausmenetelmät ja ympäristöt.

Projekti toteutettiin noudattamalla ketterää kehitys menetelmää neljän kahden viikon mittaisen sprintin aikana. Sprinttien monitorointiin käytimme Nektion palvelua ja joka sprintillä agiilin ohjelmistokehitykseen kuuluva ”scrum master”. Scrum master oli jokaisella sprintillä eri kehittäjä, jotta jokainen pääsi kokemaan

eri ohjelmistokehityksen rooleja. Versionhallinta-alustana toimi git, ja sen palveluna GitHub. Ohjelmointiympäristönä käytimme Eclipseä.

Ohjelmiston yksikkötestit on toteutettu JUnit kehystä käyttäen. Jatkuvaan testaukseen olemme käyttäneet Jenkins palvelinta, josta koontiversion tila on nähtävissä.

8 Yhteenveto

Tavoitteena oli tehdä sovellus, joka antaa käyttäjälle mahdollisuuden tutkia kide.app-sivustolta löytyvää dataa ja visualisoida sivuston trendejä. Tämän lisäksi sovellus antaisi käyttäjän tallentaa kiinnostavia tapahtumia ja pysyä ajantasalla tapahtumien ja lipunmyynnin alkamisesta.

Olemme onnistuneet saavuttamaan kaikki suunnitellut tavoitteet, mutta huomaamme, että käyttökokemuksen parantaminen on vielä suuressa määrin mahdollista. Lisäksi meille jäi paljon ideoita, joita emme ehtineet toteuttamaan projektin aikarajoitteiden sisällä.