**Android Sensor Logger**

Mikko Pakkanen (H8699)

TTOW0630  
December 2017

School of Technology, Communication and Transport

Degree Programme in Software Engineering

**Sisällysluettelo**

[1 Esittely 2](#_Toc500776667)

[2 Päämäärät 2](#_Toc500776668)

[3 Käyttötapauskaavio 3](#_Toc500776669)

[4 Käyttöliittymän luonnos 4](#_Toc500776670)

[5 Tietokannan suunnittelu 5](#_Toc500776671)

[5.1 Create 5](#_Toc500776672)

[5.2 Drop 5](#_Toc500776673)

[5.3 Tietokannan sarakkeet 5](#_Toc500776674)

[6 Aikataulu 6](#_Toc500776675)

[7 Suunnitelmassa määritellyt asiat verrattuna toteutukseen 6](#_Toc500776676)

[7.1 Datan lukeminen Android-laitteen sensoreista 6](#_Toc500776677)

[7.2 Datan tallennus SQL-relaatiotietokantaan 7](#_Toc500776678)

[7.3 Datan tallennus CSV-tiedostoon 8](#_Toc500776679)

[7.4 Datan visualisoiminen kuvaajien avulla 8](#_Toc500776680)

[7.5 Käyttäjälle mahdollisuus valita seurattavat sensorit 9](#_Toc500776681)

[8 Itsearviointi 10](#_Toc500776682)

**Kuviot**

[Kuva 1 - Käyttötapauskaavio 4](#_Toc500774424)

[Kuva 2 – Sovelluksen käyttöliittymä Android Application Development kurssin lopussa 5](#_Toc500774425)

[Kuva 3 - Tietokannan luontilause 6](#_Toc500774426)

[Kuva 4 - Tietokannan tuhoamislause 6](#_Toc500774427)

[Kuva 5 - Tietokannan sarakkeet 6](#_Toc500774428)

[Kuva 6 - Mobile Project-kurssin aikataulu 7](#_Toc500774429)

[Kuva 7 - Android Sensor Logger-sovelluksen päänäkymä 8](#_Toc500774430)

[Kuva 8 - Kiihtyvyysanturin visualisointi käyttäen MPCharts-kirjastoa 10](#_Toc500774431)

# Esittely

Otin Android Application Development TTOW0620-kurssille aiheeksi Android sensorit (Android Sensors) ja tutkin Android-laitteesta löytyviä sensoreita. Kurssin aikana muodostui pohja sovellukselle, joka lukee laitteesta löytyvien sensoreiden havaitsemaa dataa, tallentaa löydetyn datan SQL-relaatiotietokantaan ja piirtää datan perusteella erilaisia kuvaajia sovelluksen käyttäjän nähtäväksi.

Päätin jatkaa aiheen tutkimista Mobile Project TTOW0630 kurssilla ja parannella sovelluksen toimintaa ja ominaisuuksia.

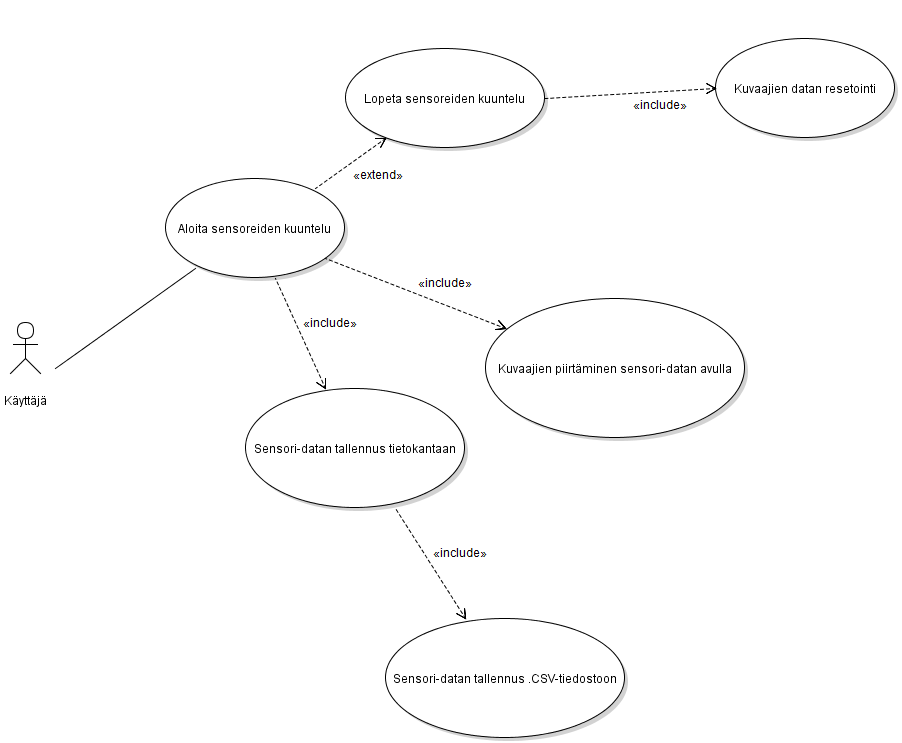
# Päämäärät

Pyrin toteuttamaan esittelyssä kuvatun applikaation, joka lukee dataa Android laitteen sensoreista esittelyssä kuvatulla tavalla ja visualisoi sitä käyttäjälle.

Sovelluksen tärkeimmät ominaisuudet / päämäärät:

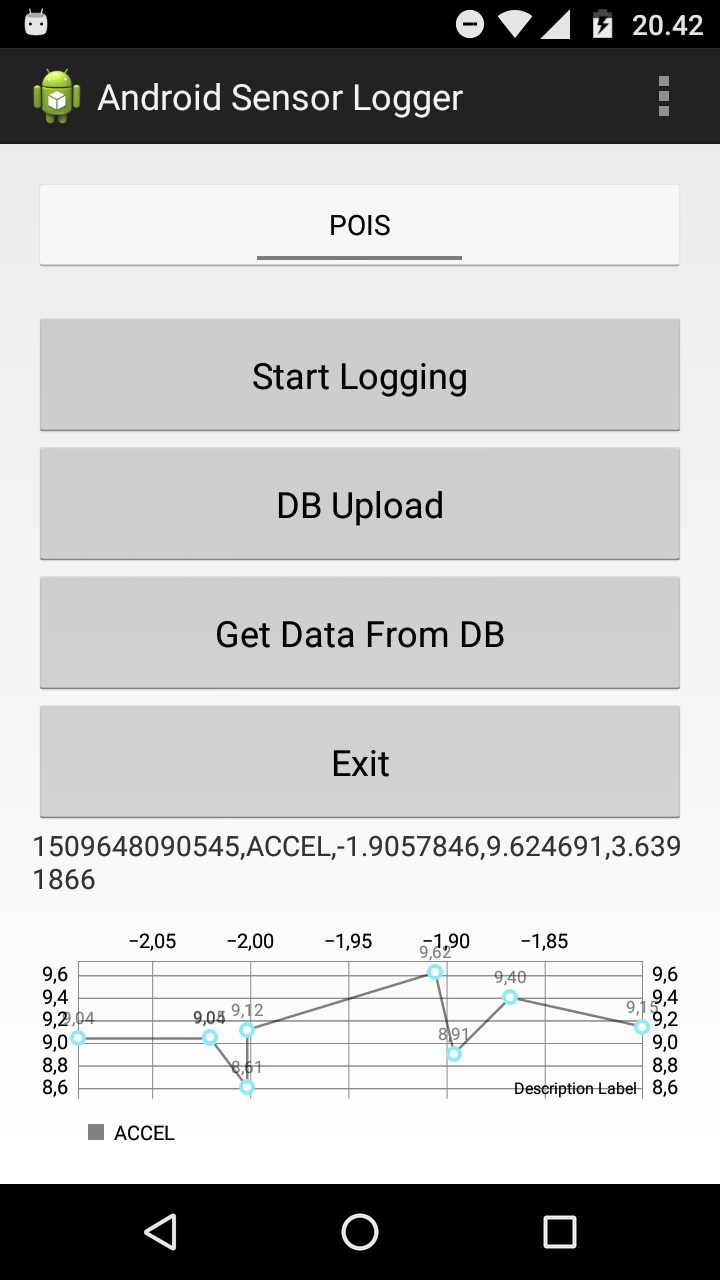
* Lukea dataa Android-laitteen sensoreista
* Datan tallennus SQL-relaatiotietokantaan
* Datan tallennus CSV-tiedostoon
* Datan visualisoiminen kuvaajien avulla
* Käyttäjälle mahdollisuus valita seurattavat sensorit

# Käyttötapauskaavio



Kuva 1 - Käyttötapauskaavio

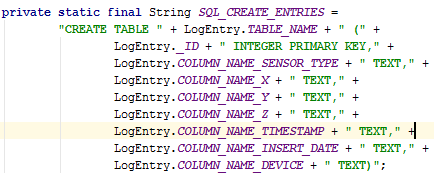
# Käyttöliittymän luonnos



Kuva 2 – Sovelluksen käyttöliittymä Android Application Development kurssin lopussa

# Tietokannan suunnittelu

## Create



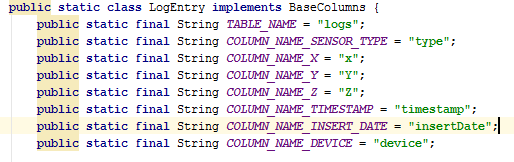
Kuva 3 - Tietokannan luontilause

## Drop



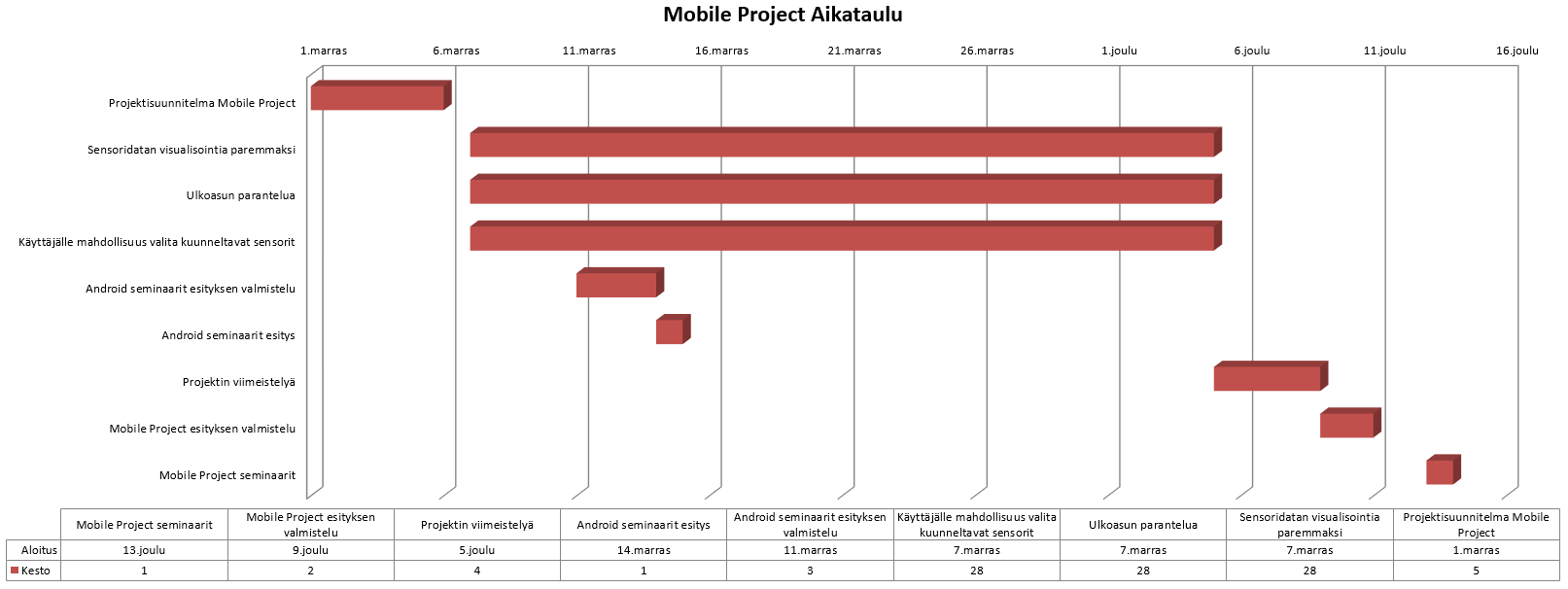
Kuva 4 - Tietokannan tuhoamislause

## Tietokannan sarakkeet



Kuva 5 - Tietokannan sarakkeet

# Aikataulu



Kuva 6 - Mobile Project-kurssin aikataulu

# Suunnitelmassa määritellyt asiat verrattuna toteutukseen

Sovelluksen tärkeimmät ominaisuudet oli määritelty suunnitelmassa seuraavasti:

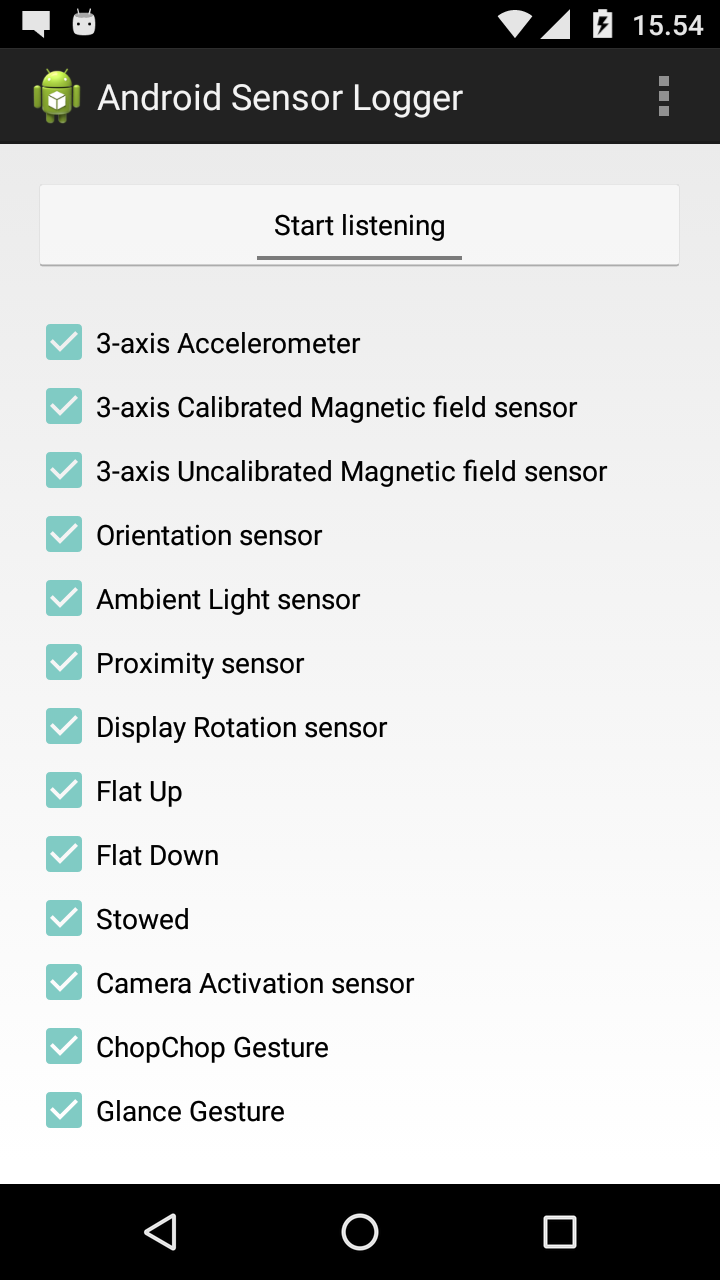
1. Lukea dataa Android-laitteen sensoreista
2. Datan tallennus SQL-relaatiotietokantaan
3. Datan tallennus CSV-tiedostoon
4. Datan visualisoiminen kuvaajien avulla
5. Käyttäjälle mahdollisuus valita seurattavat sensorit

Näistä ominaisuuksista toteutettiin ja paranneltiin kohdat **1, 2, 4** ja **5**. Kohta **3** tiputettiin pois turhana. Käydään yksitellen läpi jokainen kohta.

## Datan lukeminen Android-laitteen sensoreista

Sovellus lukee dataa käyttäjän valitsemista sensoreista. Sovelluksen käynnistyessä listataan näkyviin kaikki Android-laitteen tunnistamat sensorit ja ne liitetään *Checkbox-objekteihin*. Kun käyttäjä painaa *”Start listening”-nappia*, *MainActivity-luokka* kutsuu *GraphActivity-luokkaa*, joka käy läpi käyttäjän valitsemat sensorit ja rekisteröi tapahtumankäsittelijät *sensorManager-objektin* avulla.

*GraphActivity-luokassa* määritellyn *onSensorChanged()-metodin* kautta päästään käsiksi sensoridataan, kun jokin rekisteröidyistä sensoreista saa uuden arvon.



Kuva 7 - Android Sensor Logger-sovelluksen päänäkymä

## Datan tallennus SQL-relaatiotietokantaan

Datan tallennus SQLite-tietokantaan jätettiin sovellukseen, vaikkei sille varsinaista käyttöä löytynytkään; sovellus tallentaa käyttäjän valitsemien sensorien keräämän datan *X, Y ja Z-akseleilta* tietokantaan, lisää jokaiseen tapahtumaan aikaleiman sekä puhelimen mallin ja merkin. Tietokannan rakenne pysyi samana.

*DataLoggerContract-luokka* luotiin hoitamaan tietokantaan liittyviä toimintoja, kuten tietokannan luonti- ja tuhoamislauseet. *GraphActivity*-luokan metodeista *upload()* ja *readFromDB()* nähdään, kuinka tietojen lisääminen ja tietojen hakeminen onnistuu. *upload()-metodia* kutsutaan joka kerta, kun käyttäjän valitsemissa sensoreissa havaitaan muutos.

## Datan tallennus CSV-tiedostoon

Tämä ominaisuus oli sovelluksessa vielä Android Application Development-kurssin aikana, mutta se jätettiin Mobile Project-kurssin aikana pois turhana, sillä sovellus tallensi tiedot myös edellämainitulla tavalla SQLite-tietokantaan.

## Datan visualisoiminen kuvaajien avulla

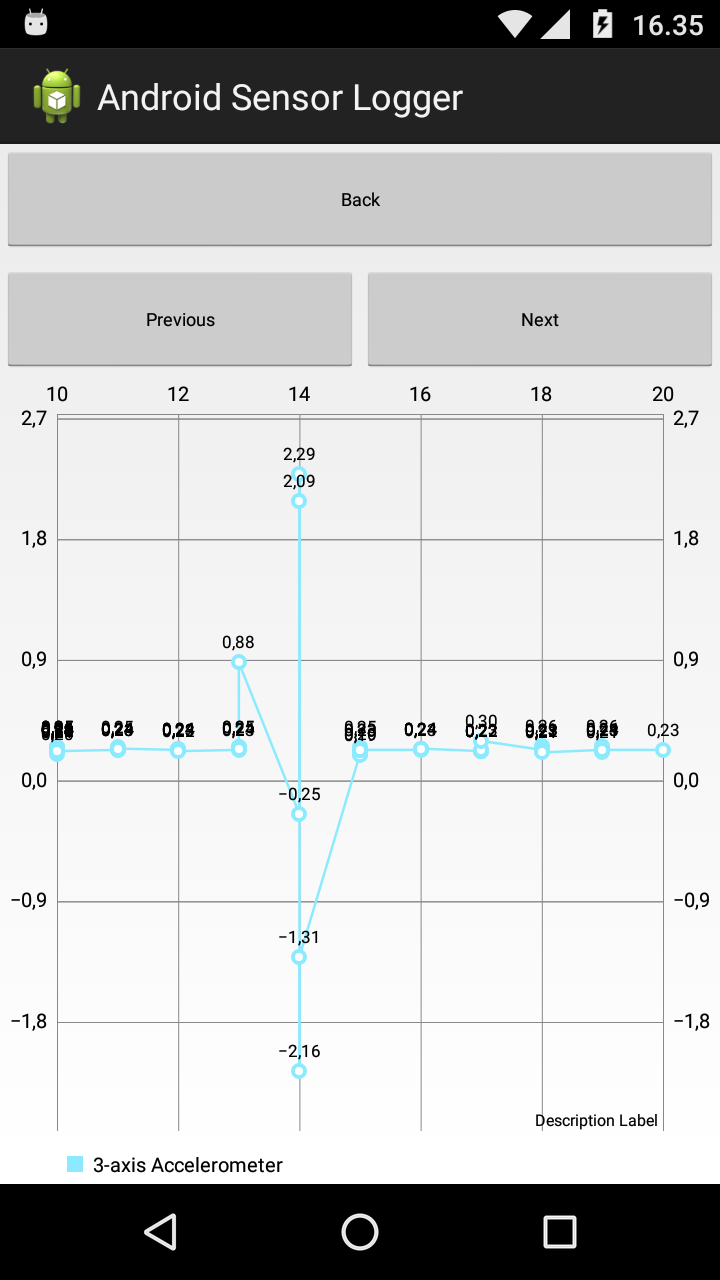
Sovelluksen tärkein ominaisuus, eli käyttäjän valitsemien sensorien datan visualisointi. Tässä käytettiin apuna *MPAndroidChart* nimistä kirjastoa.

*MPAndroidChart­-kirjasto* on suosittu, tehokas ja helppokäyttöinen työkalu kuvaajien luomiseen Android-sovelluksissa. Se on käytettävissä Android API 8:sta ylöspäin. Kirjasto mahdollistaa myös järjestelmäriippumattoman kehityksen Android- ja iOS-laitteiden välillä. Kirjaston versionhallinta löytyy osoitteesta <https://github.com/PhilJay/MPAndroidChart>.

Kaikki sovelluksessa käytetyt kuvaajat on toteutettu käyttäen *LineChart­-kuvaajia.* Kuvaajien *X-akselilla* näytetään aika sekunteina laskettuna siitä, kun käyttäjä painaa *”Start listening”-nappia*. *Y-akselilla* taas näytetään yksittäisen sensorin *X-akselin* data.

Kun käyttäjä painaa *”Start listening”-nappia,* jokainen käyttäjän valitsema sensori liitetään dynaamisesti luotuun *LineChart-kuvaajaan* eli jokaisen sensorin lukema data näytetään omassa kuvaajassaan. Kuvaajasta toiseen siirrytään käyttäen *”Previous”-* ja *”Next”*-painikkeita. Käyttäjen siirtyessä kuvaajasta toiseen, edellinen kuvaaja piilotetaan ja seuraava kuvaaja asetetaan näkyväksi.

Aloitusvalikkoon päästään *”Back”-painikkeella*. Aloitusvalikossa käyttäjällä on mahdollisuus lopettaa sensorien kuuntelu tai valita uudelleen kuunneltavat sensorit. Kun käyttäjä painaa *”Start listening”-nappia,* kuvaajat nollautuu ja datan visualisointi alkaa alusta.



Kuva 8 - Kiihtyvyysanturin visualisointi käyttäen MPCharts-kirjastoa

## Käyttäjälle mahdollisuus valita seurattavat sensorit

Kun verrataan kuvan 2. ja kuvan 7. näkymiä, huomataan, että sovelluksen käyttöliittymä on muuttunut Android Application Development- ja Mobile Project-kurssien välillä melko huomattavasti; sovellus etsii Android-laitteesta löytyvät sensorit ja antaa käyttäjälle mahdollisuuden valita kuunneltavat sensorit. Turhat napit, joita käytettiin lähinnä metodien testaukseen Android Application Development-kurssin aikana on jätetty pois ja käyttöliittymää yksinkertaistettu.

# Itsearviointi

Mietin pitkään millaista sovellusta lähtisin kurssille toteuttamaan ja sensorit kuulostivat kiinnostavalta aiheelta. En kuitenkaan keksinyt mitään mullistavan hienoa ideaa sovellukselle, joka hyödyntäisi sensoridataa, joten päätin tutustua aiheeseen yleisellä tasolla eli mitä erilaisia sensoreita Android-laitteesta löytyy sekä Android Sensor-rajapintaan.

Mielestäni Android-laitteista löytyvät sensorit eivät ole niin tarkkoja, että niistä saatava data olisi niin luotettavaa, jotta siitä olisi suurta hyötyä ja uskon, että ihmiset, jotka haluavat mitata suuretta X luotettavasti hankkivat juuri siihen tarkoitukseen kehitetyn laitteen, joka antaa mittaustuloksista varmasti luotettavaa dataa.

Kun en keksinyt mitään järkevää käyttöä sensoridatalle, päätin, että toteutan sovelluksen, joka visualisoi laitteesta löytyvien sensorien dataa kuvaajien muodossa. Tällä tavalla pääsisin tutustumaan erilaisiin sensoreihin sekä Android Sensor-rajapintaan mahdollisimman kattavasti.

Mielestäni projekti oli kokonaisuutena kiinnostava ja oli myös mielenkiintoista tutustua *MPCharts­-*kirjastoon. Sovelluksesta olisi mahdollisesti saanut paremman ja hienomman, mutta kurssin ohella tehdyt alan työt sekä opinnäytetyön tekeminen veivät suurimman osan käytettävissä olevasta ajasta sekä mielenkiinnosta. Aikataulu kuitenkin pysyi sellaisena kuin kuvassa 6. oli Mobile Project-kurssin alussa määritetty.

Arvosanaksi kurssilta antaisin 3, ottaen huomioon, että tuloksena oli toimiva sovellus ja että toteutin kaiken itse suunnittelun, ohjelmoinnin, dokumentoinnin ja esityksen osalta.