Weight and Diet helper

Tekijä Ville-Petteri Manninen

Kuvaus ohjelmasta

Ohjelma on rakennettu painonhallinnan ja kestävän dieetin rakentamisen ympärille tarjoten mahdollisuuden laskea käyttäjän pituuden ja painon perusteella BMI:n arvon ja esittään näiden muutokset havainnollisesti absoluuttisena arvona. (Vihreällä arvon laskiessa ja punaisella noustessa). Laskuriin on myös sisällytetty mahdollisuus valita “tavoite BMI”, jolloin laskuri ilmoittaa käyttäjälle tämän arvon vaatiman painon ja muutoksen nykyisestä saavuttaakseen valitun BMI:n.

Käyttäjän on mahdollista päivittää tämän päivän data-alkiota, joka syötetään tietokantaan automaattisesti päivän ensimmäisen kirjautumisen yhteydessä, mikäli edellinen data-alkio on olemassa. Eli sovellus olettaa käytettäessä, että henkilön tiedot eivät ole muuttuneet edellisen käyttöpäivän alkioon nähden ellei niitä käydä muuttamassa manuaalisesti. Sovellukseen on myös mahdollista syöttää nimi, mutta sille ei ole vielä rakennettu käyttöä.

Ilmastodieeetin (Swagger UI) tarjoaman julkisen API:n avulla käyttäjä voi myös laskea oman nykyisen dieettinsä hiilidioksidipäästöt arvioiden keskivertoihmisen kulutukseen verrattuna. Syötettynä arvona siis esimerkiksi: käyttäjä arvioi syövänsä 150% siitä liha määrästä, jonka keskiverto suomalainen syö vuodessa.

Sovellus on myös toteutettu tukemaan useampaa käyttäjää rekisteröinnin ja kirjautumisen avulla. Jokainen käyttäjä on paikallinen tietyllä laitteella, mutta yhdellä laitteella voi olla myös useampi eri käyttäjä. Jokaisella käyttäjällä on applikaatiossa oma data. Käyttäjältä on mahdollista vaihtaa salasana mikäli se on päässyt unohtumaan syötetyn syntymäpäivän avulla.

Tekijät

Sovelluksen tekijät:

- Ville-Petteri Manninen

Työnjako ja roolitus meni hyvin yksikäsitteisesti lievän resurssipulan takia. Päädyimme suunnitteluvaiheessa, että Ville-Petteri tekee sovelluksen kokonaan ja suorittaa dokumentaation kirjoittamisen ja testaamisen oman aikataulunsa mukaisesti.

Ohjelman toteutus

Ohjelma on toteutettu harjoitustyö vaatimusten pohjalta, eli sovelluksen minimi sdk versioksi on asetettu 23 ja sovellusta on testattu Android version 6.0 sisältävällä emulaattorilla ja version 8.0.0 sisältävällä mobiililaitteella (Galaxy S7). Molemmilla laitteilla sovellus on todettu toimivan vastaavasti, kuin käytetyimmässä ympäristössä (API 30, 1080 x 1920 resoluutio, android 11.0).

Sovellusta tehdessä on käytetty seuraavia kirjastoja:

- Volley 1.2.0
- lifecycle -viewmodel 2.3.0
- SQLite:n käytön vaatimat kirjastot (Cursor, helper, database) (3.9 - 3.19)
- JSON-kirjastot
- Androidin ja javan yleiset kirjastot

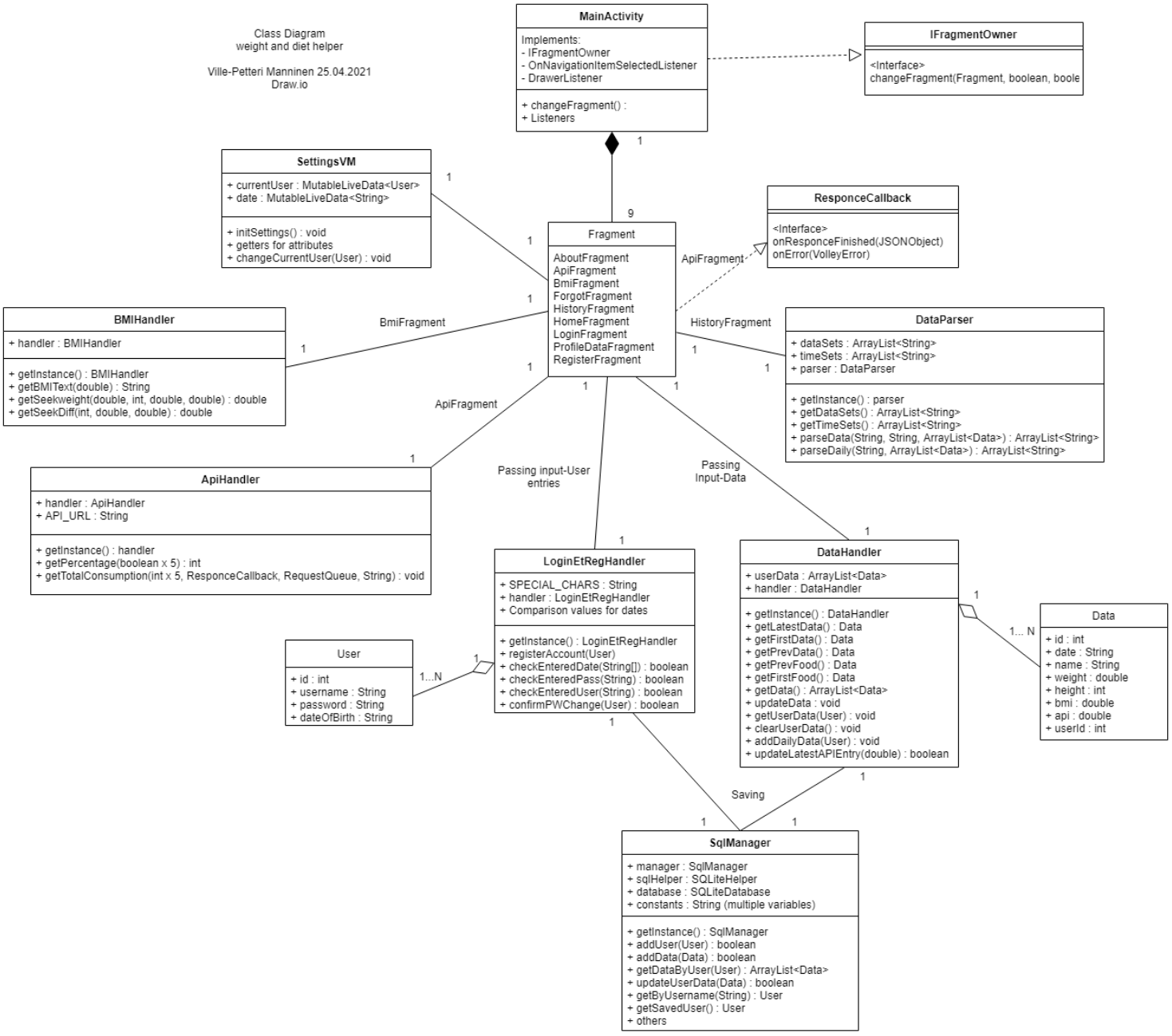
Muitakin kirjastoja voi tosiaan vielä olla käytettynä.

Projektia tehdessä työkaluina on hyödynnetty Android Studion lisäksi GitHubia projektityön jakamiseksi, Google-Docs -työkalua dokumentaation ja suunnitelman kirjoittamiseksi ja Draw.io / app.diagrams.net luokkakaavioiden tekemiseen.

Laitteistona on hyödynnetty jo valmiiksi omistamaa pöytätietokonetta ja mobiililaitetta, jonka tiedot mainittuna yllä.

Luokkakaavio

Luokkakaavio on myös tarjolla tiedostona, joko GitHubissa tai palautuksessa.



Toteutetut ominaisuudet

Ominaisuus	Perustelut	Pisteet
Olio-ohjelmoitu	Pakollinen ominaisuus	-
Luokat ja oliot	Pakollinen ominaisuus (väh. 5 kpl) Yhteensä 9 class-tiedostoa, joista 2 oliolle tiedon tallennusta varten ja 1 ViewModel. Niiden lisäksi MainActivity, fragmentit ja 2 rajapintaa.	-
Julkinen API	Pakollinen ominaisuus. Ilmastodieetti, FoodCalculator	-
Toiminnan tallennus	Pakollinen ominaisuus. SQLite-tietokanta, sisältää kirjautumisen muistamisen, tallennettavan datan ja käyttäjän tiedot.	-
Login tarkastelu	Pakollinen ominaisuus. Käyttäjän tallentamat tiedot syntymäpäivää ja salasanaa lukuunottamatta nähtävissä sovelluksen sisällä. Kaikki data tarkasteltavissa tietokantojen sisällä esim erillisellä sovelluksella (sqlitebrowser tms.)	Tämä + ylemmät = 13
UI-komponentit	Rakennettu Constraint layoutilla ja sovelluksen nappulat, syöttökentät ja valikot rakennettu käytettävyys mielessä. Esteettinen puoli jätetty vähemmälle yksin toimintaa rakentaessa. Sovellus on pakotettu portrait -tilaan.	4
Kirjautuminen	Sovelluksella paikallinen käyttäjän tallennus, joka vaatii rekisteröinnin ja sen jälkeen kirjautumisen.	3
Useampi käyttäjä	Rekisteröityminen sovellukseen, paikallinen tietokanta käyttäjien tallentamiseen. Käyttäjän vaihtaminen ja jokaisella oma data tallennettuna.	3
Salasanan säännöt	Salasana vaatii seuraavat: pituus >= 12, 1 numero, 1 erikoismerkki, 1 iso kirjain, 1 pieni kirjain.	2
Käyttäjänimien yksikäsitteisyys ja salasanan palautus	Käyttäjää luodessa käyttäjänimen olemassaolo tarkistetaan ja käyttäjältä pyydetään syntymäpäivä. Näiden tietojen avulla käyttäjä voi palauttaa salasanan (asettaa siis uuden) mikäli vanha on päässyt unohtumaan.	2
Perustiedot	Nimi, ikä (syntymäpäivä), paino ja pituus. BMI-laskuri hyödyntää käyttäjän pituutta ja painoa. Pääsivu esittää painon ja sen muutoksen kahteen ajankohtaan verraten.	2

Massan kehityksen data	Historia-näkymässä mahdollista tarkastella massan Arvoja päivän mukaan. Etusivulla massan muutosta edelliseen ja ensimmäiseen tarkastelu pisteeseen verraten.	2
Kirjautumisen muistaminen	Sovellukseen on mahdollista kirjautumisen yhteydessä asettaa sen muistaminen päälle, jolloin sovellus kirjautuu automaattisesti edelliselle käyttäjälle, kunnes ks. Käyttäjältä kirjaudutaan manuaalisesti ulos. Kuitenkin sovellus siirtyy aina home-fragmenttiin kirjautumisen jälkeen.	1
Asynkroniset HTTP-kutsut	Julkisen apin tiedonhaku hyödyntää Volley-kirjastoa, joka suorittaa HTTP-kutsut asynkronisesti. Tieto haetaan JSON-objekteina	2
Fragmenttien hyödyntäminen	Sovelluksessa MainActivity (Drawerlayout, custom actionbar and navigation menu), muuten sovellus hyödyntää fragmenttejä tarvittavien asioiden esittämiseen.	2
Responsiivinen käyttöliittymä	Käyttöliittymä toimi samalla tavalla 3 eri kokoisella näytöllä testatessa (24.04.2021) - Nexus 4 API 30 - Pixel 2 API 30 - Pixel XL API 30	2
SQLite tietokanta	Sovellus hyödyntää SQLite-tietokantaa tiedon tallennuksessa, joka koostuu kolmesta taulukosta: - Data alkiot käyttäjille - Käyttäjätiedot - Kirjautumisen tallennus Tietokannan käyttö ei vaadi käyttäjältä oikeuksia massamuistiin.	3
Sovelluksen muut toiminnot	Kts. kuvaus ohjelmasta	1
Summa		42

Työmäärät

Sovellus on tehty yksin, joten tekijä -kenttä on täydennetty vain ensimmäisen rivin osalta. Osa aika-arvioista voi heittää muutamalla tunnilla suuntaan tai toiseen.

Tekijä	Tehtävät	Tunnit
Ville-Petteri Manninen	Projektin alustus ja ideointi. Pohjana viikon 11 harjoitustehtävän rakenne.	2
-	Sovellukseen kirjautuminen ja rekisteröinti (käyttäjätiedon tallennus poislukien)	8
-	SQLite tietokannan rakentaminen ja käyttämisen opettelu.	7
-	Käyttäjätiedon tallennus ja salasanan palauttaminen	4
-	Datan tallennus ja kerääminen (käyttäjän perusdata)	6
-	Kerätyn datan hallinta ja Historia -näkymän login rakentaminen.	4
-	BMI-laskurin tekeminen (arvon laskeminen tehty jo aiemmin) Muut toiminnot tässä.	2
-	Julkisen apin toiminnot ja Volley-kirjaston käyttöönotto ja opettelu	5
-	About-fragmentti (applikaation informaatio)	0.5
-	Home-fragmentti	2
-	Kumulatiivinen käytetty aika sovellusta tehdessä testaamiseen, kommentointiin, näkymien hienosäätämiseen, virheiden korjaamiseen, uudelleen asetteluun ja kaikkeen ylimääräiseen säätämiseen.	12
-	Dokumentaatio + Luokkakaavio	4
Summa		56.5

Mitä opin harjoitustyöstä?

Ville-Petteri Manninen:

Harjoitustyöstä opin mielestäni paljon enemmän kuin viikko tehtävistä yhteensä, mutta niiden ansioista harjoitustyön tekeminen oli mahdollista. Yleisesti ottaen paransin osaamistani ohjelmoinnin ja etenkin olio-ohjelmoinnin osalta hyvin paljon, samalla kehittyen koodin lukemisessa, tutkimisessa ja virheiden etsimisessä.

Kuitenkin tarkentaen opin muun muassa seuraavia asioita:

- Fragmenttien toiminta ja niiden vaihtaminen rajapintojen avulla
- Tiedon tallennus ViewModelin ja Dataa hallitsevan luokan avulla
- SQLite tietokannan toiminta, alustaminen, ylläpito, muuntaminen, tarkastelu, tiedon tallennus, tiedon muuttaminen ja tiedon hakeminen.
- JSON-objektien käyttäminen tietoa haettaessa ja luettaessa.
- Volley:n avulla tiedon haku julkisesta API:sta

Kehitettävää vielä on opitun lisäksi paljon todella paljon, esimerkiksi olio-pohjaisen ohjelman rakenteen ja suunnittelun osalta, jonka voi huomata verrattaessa lopullista sovellusta ja dokumentaatiota suunnitelmaan ja ohjelman rakennetta tai luokkakaaviota tarkastellessa.

Palaute harjoitustyöstä (vapaaehtoinen)

- Mitkä ominaisuudet / toiminnot olivat helppoja / vaikeita toteuttaa?
    - Sovellukseen kirjautuminen ja rekisteröinti + salasanan palautus olivat mukavimmat toiminnot toteuttaa.
    - Vaikeimmaksi toiminnoksi sanoisin tiedon tallentamisen. SQLite tietokannan saaminen käyttövalmiiksi ei ole erityisen nopea tai mukava homma tehdä, mutta sen käyttämisen ollessa helppoa puolestaan nopeuttaa jatkoa huomattavasti.
- Oliko jokin asia aivan syvältä?
    - Valmiiksi annettujen sovellus ideoiden määrä olisi voinut olla suurempi (kuulemma aiemmin ~3).
    - Harjoitustyön tehtävänannon ollessa näin laaja ryhmäkoon voisi melkein pakottaa olevan 3 - 4, tai työstä voisi antaa hiukan “kompensaatiota” yksin tehneille.
- Oliko jokin asia todella hyvää tässä työssä?
    - Se määrä mitä työtä tehdessä on mahdollista oppia ohjelmoinnista on aivan loistava asia. On käytännössä pakko lähteä tekemään kaikkea mitä koko kurssin aikana on harjoiteltu ja kokeiltu, samalla uusia asioita opetellen.
- Mitä toivoisit ensi vuoden harjoitustyöhön?
    - Kuten yllä mainitsin useampi “tehtävänanto” olisi hyvä ja ryhmä kokoon nähden jonkinlaista tasapainotusta kannattaisi suunnitella.