LUT School of Engineering Science

CT60A2411 Olio-ohjelmointi

Carbonize

23.4.2021

Jan Tapper

Joona Saaresto

Aki Laine

1 Kuvaus ohjelmasta

Tämä harjoitustyö on osa Lappeenrannan-Lahden Teknillisen Yliopiston LUTin kurssia Olio-ohjelmointi. Tehtävän tarkoituksena on kehittää Android sovellus, jonka tarkoitus on osoittaa osaamisemme Java-pohjaisten sovellusten kehittämisessä. Harjoitustyö on Android-sovellus, joka pohjautuu tulevaisuuden maailmaan, jossa kiristynyt EU-lainsäädäntö vaatii asuntojen vuokrausliiketoimintaa harjoittavia yrityksiä ilmoittamaan liiketoiminnasta aiheutuneet hiilidioksidipäästöt yhteiseen tietokantaan esim. haittaverojen perintää varten. Sovellus käyttää pääasiassa Ilmastodieetin API-rajapintaa, joka laskee asumisen hiilidioksidipäästöjä, rajapinnan tietoja voi tarkastella Swagger UI-palvelussa osoitteessa: https://ilmastodieetti.ymparisto.fi/ilmastodieetti/swagger/ui/index#!/HousingCalculator/HousingCalculator_GetPurchaseEstimate. Ohjelma käyttää kirjautumisen hallintaan Googlen Firebase-palvelua, joka mahdollistaa nopean ja turvallisen kirjautumisen Googlen hallinoidessa kirjautumista ja käyttäjätietokantaa.

Sovellus käyttää Googlen Cloud Firestore -palvelua NoSql -tietokantana, johon käyttäjän hallinnassa olevien asuntojen tiedot tallennetaan ja josta niitä haetaan tarpeen mukaan sovelluksesta käsin.
Ohjelmassa oletetaan vuokranantajan käyttävän (harjoitustyön ulkopuolista) taustajärjestelmää, johon halukkaat vuokralaiset voivat rekisteröityä etukäteen. Tämä sitä varten, että taustajärjestelmään rekisteröityessään vuokralaiset antavat omat henkilötietonsa ja vuokranantaja voi tehdä esim. luottotietotarkistukset etukäteen ennen vuokrasopimuksen tekemistä. Tätä ratkaisua ei ole kuitenkaan tarkistettu GDPR-näkökulmasta, joten sen tekninen toteutus todellisessa järjestelmässä saattaa poiketa tästä toteutusmallista (vuokralaisten nimet näkyvät muille sovelluksen käyttäjille asukasvalikossa).

Sovelluksessa käytetään kuvina Picsum.photos -palvelusta löytyviä kuvia teemoittain, esim. asuntojen kohdalla käytetään satunnaisia rakennusten kuvia ja profiilikuvan kohdalla käytetään itse luodulla pseudorandom -menetelmällä henkilökuvia.

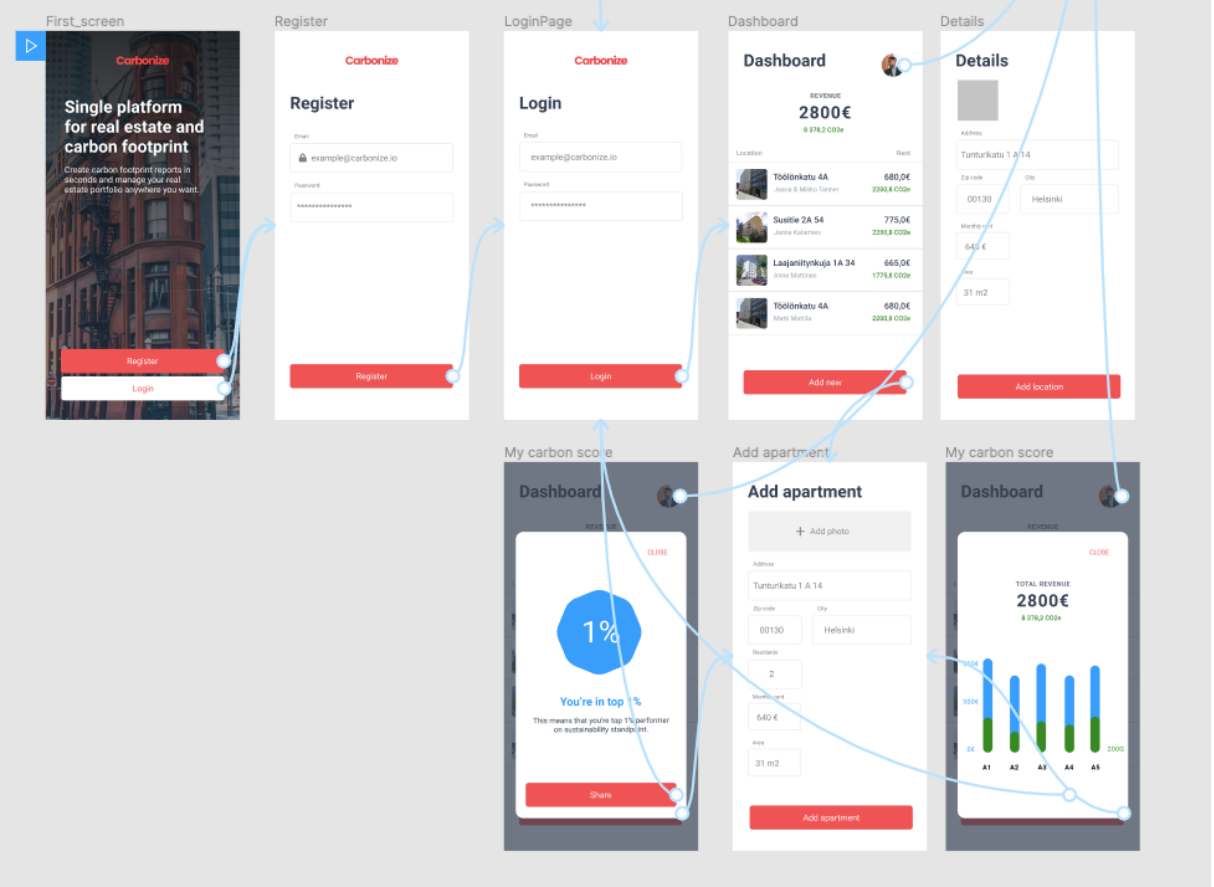
2. Ohjelman toteuttajat ja työmäärä

Taulukko 1. Ohjelman toteuttajat ja vastuualueet

Tekijä	Vastuualueet sovelluksessa	Arvioitu työmäärä (h)
Jan Tapper	Suunnittelu, Dashboard näkymä, ilmastodieetti-apin implementointi, Luokkien täydennys, Rekisteröinti- ja login fragmenttien toiminnallisuudet, bugikorjaukset	60
Aki Laine	Suunnittelu, UI-luokkakaavio, profiilinäkymä, statistiikan visualisointi, käyttäjän antamien tietojen kirjaus .csv tiedostoon, kokonaistuoton ja päästöjen laskenta, testaus	60
Joona Saaresto	UI Suunnittelu, Login, Rekisteröinti, Password reset, Apartmentin lisäys, API-perustoiminnallisuus, Firebase konfigurointi	60

3 Suunnittelu ja toteutus

Suunnittelu aloitettiin alkuperäisen idean hahmottelun jälkeen Figma-prototypointityökalussa. Aloimme tekemään näkymiä Figmaan, ja loimme interaktiivisen prototyypin sieltä käsin.

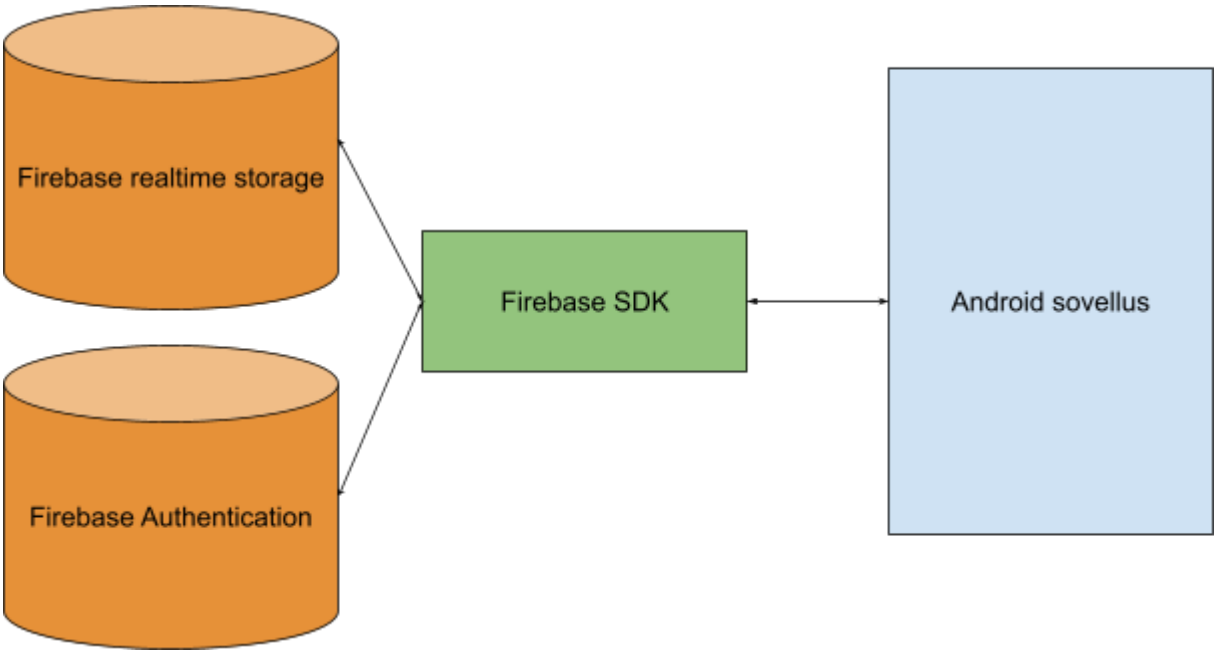


Kuva 1: Figma prototyyppi

Suunnittelun jälkeen jaoimme kullekin jäsenelle vastuualueet, joita jokainen lähti itsenäisesti toteuttamaan. Työmäärällisesti kaikilla oli tässä vaiheessa yhtä paljon tekemistä.

3.1 Arkkitehtuuri

Harjoitustyö koostuu kolmesta eri pääkomponentista. Tietokannasta, backendistä sekä Android-sovelluksesta.



Kuva 2: Arkkitehtuuri

3.1.1 Firebase

Firebase on Googlen ylläpitämä alusta, joka tarjoaa sovelluskehittäjille tarpeellisia kokonaisuuksia kuten backend-infrastruktuurin, sovelluksen monitoroinnin ja kattavia työkaluja muun muassa testaukseen ja analytiikkaan. Firebasen SDK mahdollistaa alustan käytön monipuolisesti eri tekniikoille, kuten Androidille, Javalle, JavaScriptille tai C++:lle. https://firebase.google.com/

3.2 Alusta ja kirjastot

Ohjelma toimii Android 11 (R) tai uudemmalla versiolla. Ulkoisia kirjastoja on hyödynnetty joihinkin toiminnallisuuksiin, mitkä näkyvät taulukosta 2.

Taulukko 2. Käytetyt kirjastot

Kirjasto	**Toiminnallisuus**
MpAndroidChart (v3.1.0)	Kuvaajan piirtäminen datasta
picasso(2.71828)	Kuvien lisääminen recyclerviewiin Dashboardissa sekä profiilikuvan ohjelmalliseen muuttamiseen.
circleimageview(3.1.0)	Kuvien pyöristäminen, esimerkiksi profiilinäkymässä

3.3 Käytetyt työkalut

Projektin toteuttamiseen hyödynnettiin monia eri työkaluja niin kommunikoinnissa, kehittämisessä, testauksessa, suunnittelussa ja dokumentoinnissa. Käytetyt työkalut näkyvät taulukosta 3.

Taulukko 3. Käytetyt työkalut

Työkalu	Käyttökohde
Figma	Käyttöliittymäsuunnittelu, mockup-kuvat
Android Studio	Sovelluskehitys
Android Emulator	Testaus
Google Docs	Dokumentaation kirjoitus
Draw.io	Luokkakaavioiden piirtäminen
Google Firebase	Kirjautuminen, tietojen tallennus
Discord	Kommunikaatio palavereissa
Whatsapp	Kommunikaatio viesteillä
Github	Versionhallinta
Picsum.photos	Ulkopuolinen kuvalähde

4 Ominaisuudet

Alla olevassa taulukossa 4 on esitetty ohjelman toteutetut ominaisuudet. Näiden toiminta kuvataan tarkemmin demovideossa.

Taulukko 4. Ohjelman ominaisuudet

Ominaisuus	Pisteet	Carbonize
Olio-ohjelmoitu	Pakollinen	Kyllä
Vähintään viisi erilaista luokkaa & oliota (käyttöliittymäluokkia ei lasketa)	Pakollinen	Kyllä
Vähintään yhden APIn käyttö, esim. Ilmastodieetti:	Pakollinen	Kyllä

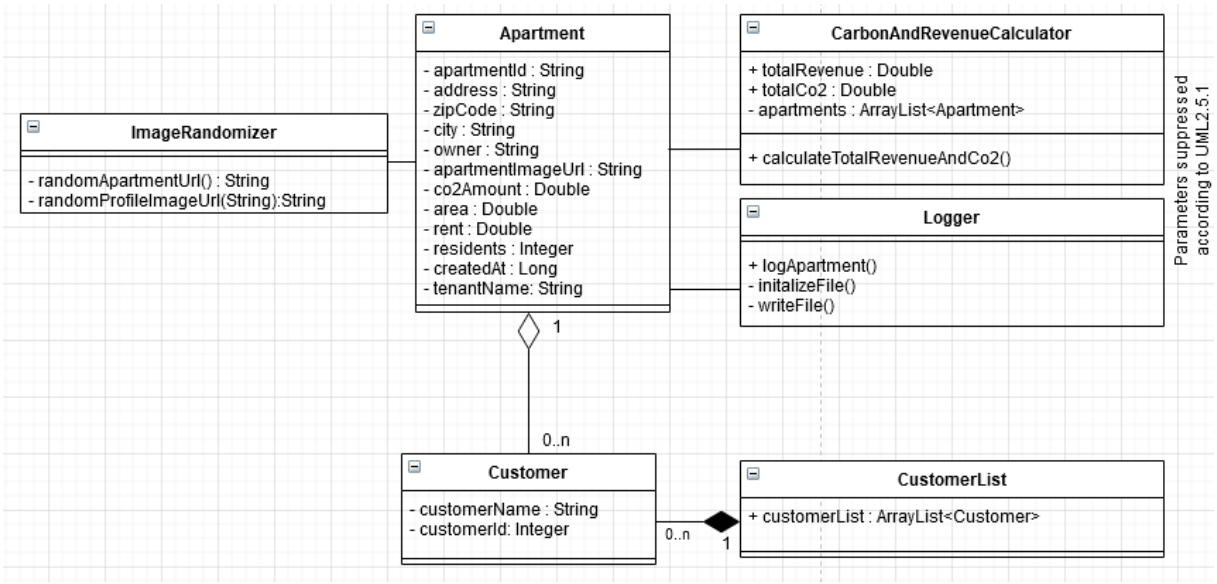
https://ilmastodieetti.ymparisto.fi/ilmastodieetti/swagger/ui/index		
Sovellus tallentaa käyttäjän toiminnan (käyttäjän syöttämät arvot / tulokset) logiin (JSON, XML jne.)	Pakollinen	Kyllä
Logia on mahdollista tarkastella (puhtaana tekstinä, graafisilla käppyröillä jne.), eli voidaan tutkia arvojen (esim. oma massa) kehitystä kirjausten edetessä	Pakollinen	Kyllä
Ohjelma on rakennettu hyvin suunnitelluista UI-komponenteista	1 – 5 pistettä	Kyllä (Kustomoitu Material UI)
Kirjautuminen applikaatioon	3 pistettä	Kyllä (Firebase)
Sovelluksella voi olla useampi käyttäjä (ja niiden luominen), tietojen tallennus järkevästi jonnekin	3 pistettä	Kyllä (Firebase)
Kirjautumisen salasana noudattaa hyvän salasanan sääntöjä (sisältää vähintään yhden numeron, erikoismerkin, ison ja pienen kirjaimen, on vähintään 12 merkkiä pitkä)	2 pistettä	Kyllä
Salasanan tallennus käyttää jonkinlaista hash-menetelmää ja suolausta (esim SHA-512 + salt)	2 pistettä	Kyllä (Firebase tekee hashayksen)
Jokin toinen datalähde fiksusti implementoituna (näitä löytää esim. https://www.avoindata.fi/, https://www.europeandataportal.eu/fi tai https://thl.fi/fi/tilastot-ja-data/aineistot-ja-palvelut/avoin-data/avoimet-rajapinnat) eli sovellus käyttää siis	2 – 5 pistettä	Firebase Cloud Firestore Picsum.photos

useampaa datalähdettä kerralla		
Ohjelmaan on mahdollista syöttää perustiedot (esim. pituus, paino, ikä(/syntymävuosi), kuva, asuinkunta) käyttäjästä ja näitä arvoja käytetään jossakin	2 pistettä	Kyllä (Apartmentin lisäys)
Ohjelma kerää käyttäjän massan kehityksestä dataa ja näyttää muutokset graafisesti havainnollistaen ruudulla	3 pistettä	Kyllä (Vuokratulot ja niiden hiilijalanjäljen kehitys)
Ohjelma näyttää graafisesti ilmastodieetin tarjoamien arvojen muutokset käppyröillä (esim. kuinka lihan kulutus ja hiilijalanjälki on muuttunut aikojen saatossa)	3 pistettä	Kyllä (Vuokratulot ja niiden hiilijalanjäljen kehitys)
Ohjelma kertoo asuinkunnan, ikäluokan yms. riskitekijät (esim. kunnassa X tupakoitsijoita on Y%) pohjautuen THL:n dataan omien syötteiden lisäksi	2 pistettä	
Ohjelma peilaa eri datalähteidä ristiin ja muodostaa uutta tietoa (THL:n dataa höystettynä tilastokeskuksen datalla ja saldona saadaan ulos kaavio Z)	3 pistettä	
Ohjelma muistaa käynnistämisen / kirjautumisen jälkeen missä näkymässä käyttäjä oli ennen ohjelman sulkemista	2 pistettä	Kyllä (Etsii instanssin Firebase tokenista)
Asynkronisten HTTP-kutsujen käyttö dataa haettaessa	2 pistettä	Kyllä (Firebasesta Apartment tietojen haku)
Fragmenttien hyödyntäminen aktiviteettien sijasta	2 pistettä	Kyllä (Melkein kaikissa näkymissä)

käyttöliittymiä rakennettaessa		
Scoped storagen käyttäminen tiedon tallennuksessa (ei vaadi käyttäjän myöntämiä oikeuksia laitteen massamuistiin, vaan toimii omassa “hiekkalaatikossaan”)	2 pistettä	Kyllä
Responsiivinen käyttöliittymä (toimii siis erikokoisilla ruuduilla sulavasti)	2 pistettä	Kyllä
Puutteellinen dokumentaatio	-1 – -5 pistettä	
Ohjelmassa on ongelmia, jotka haittaavat sen käyttöä	-1 – -5 pistettä	
Ohjelma ei noudata kunnollista oliomallia	-1 – -10 pistettä	
Ohjelmakoodia on kirjoitettu/kommentoitu suomeksi (huomaa, että käyttöliittymätekstit voivat tietysti olla suomeksi)	-3 pistettä	
Ohjelmassa ja/tai dokumentaatiosta on rasismia, vihapuhetta tai muuta epäsoveliasta	-20 pistettä	
Jokin oma hieno ominaisuus tai toiminto (tai useampi)	Max 5 pistettä per ominaisuus	Pöhinäkelpoinen UI Picsum.photos -kuvien hyödyntäminen
Yhteensä (mikäli kaikki toteutetut ominaisuudet tuottavat enemmän kuin 40 voidaan lisäominaisuuksilla korvata huonompia, mutta yli 40 pistettä ei voi harjoitustyöstä saada)	Max 40 pistettä.	40-46 pistettä

4 Luokkakaavio

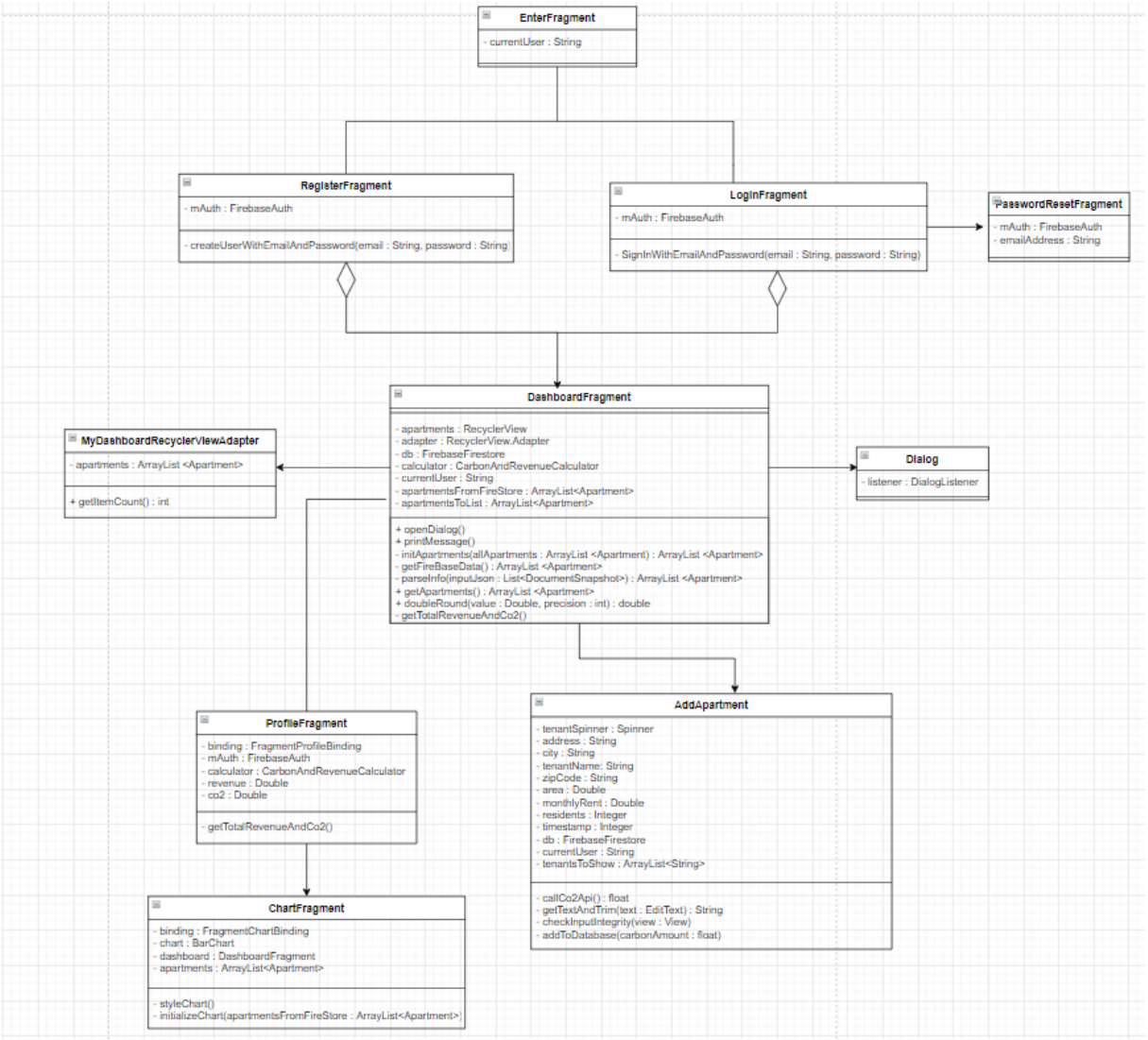
Linkki luokkakaavioon:
https://drive.google.com/file/d/1rypaWRfDRoxEhQkiSbKcT5fSXg8bfrKc/view?usp=sharing



Kuva 3: Luokkakaavio

Luokkakaavioon ei ole merkitty yksinkertaisia get- ja set-metodeja attribuuttien käsittelyyn, eikä UML 2.5.1 mukaisesti kaikkia parametrisarjoja ole lueteltu.

Linkki UI-luokkakaavioon:
https://drive.google.com/file/d/1abM_o68WSrfbOGIDUKU4bfEx2UYR_ZIa/view?usp=sharing



Kuva 4: UI-luokkakaavio

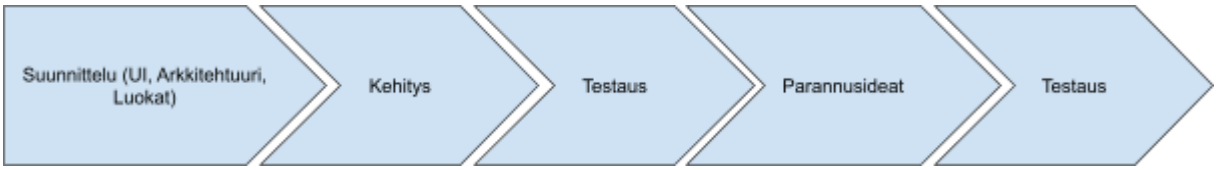
Mitä opin harjoitustyöstä?

Jan Tapper: Harjoitustyössä opin ohjelmoimaan laajempaa projektia yhdessä muiden kanssa, sekä käyttämään versionhallintajärjestelmää (github) monen käyttäjän projektissa. Tämän projektin myötä opin myös paljon käyttöliittymäsuunnittelusta ja toteutuksesta ryhmässä mukana olleen Joonan otteita seuraamalla, sekä kantapään kautta siitä mikä kommentoinnin/dokumentoinnin merkitys on toisen kirjoittaman koodin debuggaamisessa. Harjoitustyössä tuli myös ilmi, miten helposti itse arvioi osakokonaisuuksiin kuluvan ajan liian pieneksi, joka näkyi alunperin arvioimani ajankäytön ylittymisenä. Hyödyllisin uusi oppimani asia jatkoa ajatellen on kuitenkin luultavasti Googlen Firestoren ja Firebasen hyödyntäminen, joka helpottaa varmasti työtä myös tulevissa ohjelmointitehtävissäni.

Joona Saaresto: Tämä harjoitustyö oli aika erilainen kun muut kurssit mitä olen tehnyt. Kauppatieteiden kursseilla tuntuu olevan niin, että harkkatöitä pystyy tekemään aika helposti työmäärällisesti, tietotekniikan kursseilla on kyllä päässyt oikeasti tekemään asioita, ja tuntuu, että tähän kurssiin käytin enemmän aikaa kun muihin neljään (kauppatieteiden) kurssiin tällä jaksolla yhteensä.

Java myös yllätti sen verran, että vaikka työskentelen päivittäin Javascriptin, Reactin, NodeJS:n parissa, niin ensimmäisiä kertoja Javaa tehdessä tuntui, etten osaa ohjelmoida lainkaan. Kieltämättä tällä kurssilla mentiin syvään päätyyn ja kyllä sieltä pinnalle päästiin. Ylivoimaisesti mullistavin asia, joka vei harjoitustyömme seuraavalle tasolle oli Firebase ja siihen liittyvät hyödylliset toiminnot, kuten autentikointi ja dokumenttitietokanta, näiden toteuttamiseen olisi kyllä joutunut hakata päätä seinään. Harjoitustyön alkuvaiheessa olin jo rakentamassa Java-backendiä, mutta yhden viikonlopun päänhakkaamisen jälkeen löysin Firebasen ja tajusin, että tämähän on hieno keksintö. Tämän kurssin aikana kyllä sai hyvän peruskäsityksen mitä Android sovelluksen kehittäminen vaatii.

Toimintatapamme tätä projektia tehdessä muistutti aika hyvin oikeaa pienen mittakaavan softaprojektia. Prosessimme oli jotakuinkin seuraavanlainen:



Kommunikaatiomme oli WhatsAppissa, joten juttelimme aika matalalla kynnyksellä kaikenlaisista projektiin liittyvistä asioista ja reagoimme niihin nopeasti.

Aki Laine: Opin ohjelmoimaan laajempaa projektia Javalla, ja siten luomaan ratkaisuja erilaisten luokkapohjaisten olioiden yhteistoiminnan muodossa. Opin myös paljon Android-sovelluksen kehittämisestä, ja yllätyin siitä miten sujuvaa se on, sitten kun perustaidot ovat hallussa. Projektin aikana myös ensimmäistä kertaa ryhmässä ohjelmointi ja versionhallintatyökalu GitHub tuli kunnolla tutuksi minulle. Mikä parasta, opin kokeneilta ryhmätyökavereiltani paljon uusia hyödyllisiä asioita, ja he toivat työhön erinomaista tietotaitoa kokemuksen kautta. Päällimmäinen esimerkki tästä se, miten Joona päätyi Googlen Firebasen käyttämiseen, joka tuli sitten minullekin tutuksi. Firebase ja Firestore tarjoavat laajan kirjon mahdollisuuksia ja ominaisuuksia, jotka ovat helposti käytettävissä, kunhan niihin osaa tutustua. Yksi merkittävimmistä asioista mitä opin oli ulkoisten kirjastojen ja APIen hyödyntäminen omassa projektissa. Projektin aikana tajusin, että niitä hyödyntämällä voi tehdä ohjelmaan laajoja ja ammattimaisia toteutuksia pienellä vaivalla.

Palaute harjoitustyöstä (vapaaehtoinen)

- Mitkä ominaisuudet / toiminnot olivat helppoja / vaikeita toteuttaa?
    - Scoped storage -ominaisuus tuli ihan lahjana ensin tietämättä mitä se merkitsee
- Oliko jokin asia aivan syvältä?
    - Moni asia oli Javalla Android Studiossa vaikeampi toteuttaa, kuin esim. Xamarinililla tai Flutterilla.
    - Moni vaatimus oli sellainen, mistä ei oltu edes mainittu kurssin opetusmateriaalissa, ja se oli turhauttavaa.
- Oliko jokin asia todella hyvää tässä työssä?
    - Oli avartavaa huomata haasteet ja hyödyt monen henkilön kehittäessä samaan aikaan samaa ohjelmaa.

- Mitä toivoisit ensi vuoden harjoitustyöhön?
    - Harjoitustyössä joutuu (ainakin suurempia pistemääriä tavoitellessa) opettelemaan merkittävän osan itse, eikä kyseisiä aihealueita käsitellä opetusmateriaalissa lainkaan. Toivoisimme opetusmateriaalin vastaavan paremmin harjoitustyön vaatimuksia, tai toisinpäin.