On Track

Tekijät Mette Suvanto, Margeritta El-Khoury ja Helmi Toropainen

Kuvaus ohjelmasta

Terveyssovellus, johon useampi käyttäjä voi rekisteröityä ja käyttäjät voivat muokata ja tarkastella tietojaan asetuksissa sekä pop up –ikunnassa Calorie/SleepActivityssä.

Sovellus pitää kirjaa käyttäjän syöttämistä uni/kalori tiedoista ja tallentaa ne logiin. Käyttäjä voi muokata päivän aikana syöttämiään tietoja ja muutokset päivittyvät logiin.

Sovellus piirtää login perusteella kuvaajia kysesisen käyttäjän unen ja kalorien kehityksestä ja käyttäjä voi tarkastella näitä muutoksia.

Aloitussivulla on myös ohjeet sovelluksen käyttämiseen ja fakta-boxi, jota klikkaamalla sovellus arpoo käyttäjälle satunnaisen faktan.

Tekijät

- Kuka teki ja mitä?
- Kuvaus työnjaosta ja rooleista

Helmi Toropainen:

Tein pohjan käyttöliittymästä eli loin kaikki fragmentit ja aktiviteetit (paitsi RegisterActvity (Mette)) sekä muokkasin niiden ulkoasut ja niiden välillä liikkumisen sovellusta käyttäessä. Keksin myös sovelluksen nimen.

Loin kaikki CalorieActivityn toiminnallisuudet: etsin datalähteet (Fineli API (https://fineli.fi/fineli/api/v1/foods), jonka avulla lasken käyttäjän valitseman ruoan ja syöttämän massan perusteella saadut kalorit sekä csv-tiedosto, jonka avulla lasken käyttäjän valitseman urheilulajin ja syöttämän keston perusteella kalorinkulutuksen, (löytyy täältä: https://www.kaggle.com/aadhavvignesh/calories-burned-during-exercise-and-activities, olisin halunnut löytää toisen APIn, mutta sellaista ei tullut vastaan, joten päädyin tiedoston lukemiseen, en mene takuuseen tietojen oikeellisuudesta.) sekä loin niiden pohjalta Spinner-valikon. Loin RecyclerViewAdapter luokan, joka yhdessä CalorieActivityn kanssa hoitaa RecyclerViewn päivittämistä käyttäjän muutosten mukaisesti (lisää dataa, poista painamalla pitkään, tyhjennä lista). Aktiviteetti pitää myös listaa kalorien summasta (syödyt – kulutetut) Loin myös CaloireActivityyn pop up -ikkunan, jossa käyttäjä voi muokata omaa kaloritavoitettaan. Tätä pohjaa hyödynnettiin myös SleepActivityssä (Margeritta).

Loin luokat UserEntry, CalorieEntry, FoodEntry ja SportEntry sekä SportData ja FoodData. CalorieEntry periytyy UserEntrystä ja CalorieEntrystä periytyy FoodEntry ja SportEntry. EntryManager hallitsee Entryjä ja CalorieEntryn loin, jotta RecyclerViewAdapter voi käsitellä sekä Sport- että FoodEntryjä. APIn lukeva readJSON funktio, joka luo APIn pohjalta FoodData olioita, jotka se laittaa ArrayListiin, jonka pohjalta luodaan spinnervalikko olioista. Olion tietojen pohjalta sitten asetetaan FoodEntryn nimi, massa ja kalorimäärä. Samalla tavalla readCSV luo tiedoston pohjalta SportData olioita ArrayListiin, joka laitetaan spinneriin ja jota klikkaamalla saadaan saadaan nimi, kesto ja kalorit.

Etsin myös netistä faktat strings-kansioon (lähteet: https://www.thegoodbody.com/health-facts/, https://spectrumhealthcare.com/resources/do-you-know-these-surprising-health-facts/) ja loin

sovellukseen toiminnon, joka arpoo käyttäjälle faktan, ja joka voidaan arpoa uudestaan tekstiä klikkaamalla.

Loin EntryManager luokan (Singleton periaate), johon lähetetään data käyttäjän muutosten perusteella, ja kun käyttäjä on onnistuneesti syöttänyt sekä kalori- että unientryn EntryManager kirjoittaa tiedot logiin (UsetEntryLog luokka (Mette)). Jos kirjautuneella käyttäjällä on jo logi-rivi kyseiseltä päivältä, sen päälle kirjoitetaan päivitetyt tiedot. Logissa on siis vain yksi rivi / käyttäjä / päivä. Logi-tiedosto luodaan MainActvityssä, jos sitä ei ole jo olemassa. Logi on siis kaikille käyttäjille yhteinen ja se luodaan vain, kun sovellusta käytetään ensimmäistä kertaa tai vanha logi on tuhoutunut.

Toteutin entryjen tiedonsiirron sovelluksessa. Sleep/CalorieActivity lähettää aktiviteetista poistuttaessa käyttäjän muutokset MainActivityyn, joka päivittää tiedot HomeFragmentin tektikenttiin. Myös käyttäjän muokkaukset goal-arvoihin päivittyvät pop up -ikkuna suljettaessa. Kaikki käyttäjän syötteet/muokkaukset (CalorieActivityssä RecyclerView-listat (sisältävät olioita), SleepActivityssä käyttäjän syötteestä lasketut tunnit ja minuutit sekä näistä laskettu readiness-prosentti ja MainActivityssä näkyvä kalorien/unen summa sekä Entry-oliot) tallennetaan SharedPrefrences avulla, ja nämä tiedot ladataan, kun käyttäjä palaa näkymään. Eli siis sovellus muistaa käyttäjän syötteet (joiden summat ja oliot ovat myös logissa tallessa) ja etsii nämä käyttäjänimen perusteella, kun käyttäjä palaa sovellukseen tai aktiviteettiin/fragmenttiin.

Sovelluksessa myös toiminto, että MainActivitystä ei voi poistua puhelimen back-näppäimellä. Eli siis käyttäjän tulee kirjautua ulos asetuksissa Log out -napilla (Mette) tai sulkea sovellus (kirjaa käyttäjän ulos). Kun käyttäjä kirjautuu takaisin, hän pätyy MainActivityn HomeFragmenttiin ja näkee aiemmin syöttämiensä tietojen summan. Ajateltiin, että tämä on käyttäjäystävällinen ja toteutettavissa oleva tapa.

Margeritta El-Khoury:

Loin AnalyticsFragmenttiin kuvaajat käyttäjän syöttämien tietojen perusteella viideltä viimeisimmältä päivältä. Analyses-luokassa luetaan csv-muotoinen logi, josta poimitaan käyttäjänimen perusteella kyseisen käyttäjän tiedot ja verrataan poimitun rivin päivämäärää nykyiseen. Näin valituilta riveiltä luetaan käyttäjän syömät ja kuluttamat kalorit sekä nukutut tunnit. Tiedot tallennetaan omiin nollilla alustettuihin ArrayListoihin siltä varalta, ettei käyttäjä ole tallentanut tietojaan jokaisena päivänä. Luodaan myös ArrayLista kalorien erotukselle ja käyttäjän omille tavoitteille. Itse kuvaajien piirtämiseen liittyvät komennot löytyvät AnalyticsFragment-luokasta, jossa kuvaajat luodaan GraphView-master –kirjastoa käyttäen. Kalorikuvaajaan tulevat arvot piirretään viivadiagrammeina ja unikuvaajan arvot pylväsdiagrammeina. Käyttäjän arvot näkyvät kunkin pisteen kohdalla ja kuvan alalaitaan on lisätty selitteet.

Tein sovellukselle kuvakkeen, jonka asetin näkymään laitteen kotisivuilla (ja muilla sivuilla, joista sovelluksen voisi avata). Kuvake on luotu Paint 3D –ohjelmalla sovelluksen värimaailmaan mukautuen.

Tein osan unen seurantaan liittyvistä toiminnoista. Loin SleepEntry –luokkaan ja tein siihen laskurin, joka ottaa vastaan käyttäjän syöttämän nukkumaanmeno- ja heräämisajan ja laskee niiden erotuksen (Mette). Luokka käsittelee puutteelliset ja virheelliset syötteet esimerkiksi ilmoittamalla käyttäjälle virheestä. Jos minuutteja ei syötetä, niiden arvoksi oletetaan 0. Näytölle tulostuu laskun tulos muotoiltuna tuntien ja minuuttien määrän mukaan: tulos tulostetaan oletuksena tunteina ja

minuutteina, mutta jos jommankumman arvo on 0, sitä ei tulosteta. Kun unimäärä on laskettu, lasketaan tämän perusteella vireystila, eli kuinka suuren osuuden tavoitteestaan käyttäjä on nukkunut. Vireystilan tason mukaan näytölle päivitetään kannustava tai neuvova kommentti. Loin myös SleepPopUp-ikkunan vastaamaan CaloriePopUp-ikkunan ulkonäköä, mutta omilla teksteillään ja syötekentillään.

Mette Suvanto:

- Kirjautuminen
    - LogInActivity (UI-komponentit: Helmi)
        - Kirjautuminen sovellukseen käyttäjätunnuksen ja salasanan avulla.
        - Käyttäjän antaman käyttäjänimen perusteella etsitään käyttäjän salasana ja verrataan sitä syötettyyn salasanaan.
- Rekisteröityminen
    - RegisterActivity (UI-komponentit: Helmi)
        - Käyttäjä voi rekisteröityä syöttämällä perustiedot itsestään. Tietoja käytetään sovelluksen tarjoamiin ominaisuuksiin, esim. Painoa ja pituutta bmi:n laskemiseen, ikää unisuositusten määrittämiseen.
        - Bmi lähde: https://www.cancer.org/cancer/cancer-causes/diet-physical-activity/body-weight-and-cancer-risk/adult-bmi.html
        - Unisuositus lähde: https://www.mayoclinic.org/healthy-lifestyle/adult-health/expert-answers/how-many-hours-of-sleep-are-enough/faq-20057898
        - Salasana tulee syöttää kaksi kertaa, jotta voidaan varmistaa, että käyttäjä on syöttänyt salasanansa oikein. Salasanasta tarkistetaan, että se täyttää hyvän salasanan piirteet.
        - Salasanan tallentamisessa käytetään SHA-512 menetelmää ja suolausta.
- Käyttäjätietojen tallentaminen
    - User
        - Yhden käyttäjän tiedot olioon
    - UserLocalStore
        - User luokasta luotu olio tallennetaan SharedPreferences avulla xml tiedostoon. Avainsanoihin on yhdistetty käyttäjänimi, jotta halutun käyttäjän tiedot saadaan noudettua tiedostosta.
- Käyttäjätietojen muokkaus asetuksissa
    - SettingsFragment (UI-komponentit: Helmi)
        - Käyttäjätiedot päivittyvät asetuksiin ja niitä voidaan muokata sieltä.
    - PasswordFragment (UI-komponentit: Helmi)
        - Salasanan vaihtaminen. Salasanalle samat ominaisuudet kuin rekisteröitymisessä.
- Log
    - UserEntryLog
        - Login luominen (csv-tiedosto) ja metodi, jolla voidaan lisätä entryjä tiedoston loppuun.
- Testidata

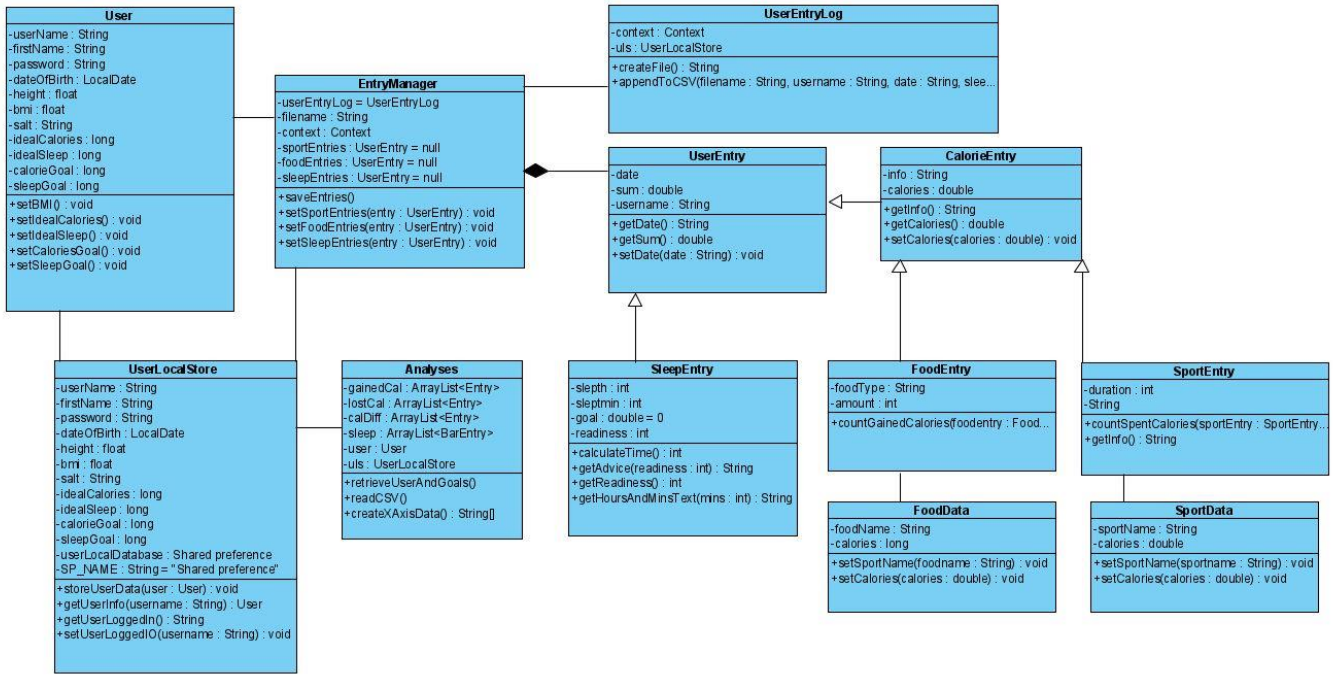
    - Loin testidataa, jonka avulla voitiin testata sovelluksen toimivuutta, sillä logiin kirjoitetaan yksi entry/päivä ja entryjen päivämäärät toimivat sovelluksessa reaaliaikaisesti (Helmi).

Lisäksi kaikki ryhmäläiset testasivat sovellusta, korjasivat mahdollisia bugeja ja muokkasivat käyttöliittymää selkeämmäksi, jos on ollut tarvetta.

Ohjelman toteutus

- Millaisella teknisellä alustalla ohjelma toimii?
    - Android 8.0 – 11.0 (requires min. SDK/API 26, target 30)
    - Virtual device näyttö: Pixel 2
- Mitä kirjastoja on käytetty?
    - Helmi: Käytin Googlen Gson-kirjastoa apuna, kun tallensin olioita ja olioita sisältävän ArrayListin SharedPrefrences:iin muuttamalla ne ensin json-muotoon.
    - Margeritta: Käytin kuvaajien piirtämiseen GraphView-master –kirjaston komentoja ja UI-elementtejä.
- Mitä työkaluja on käytetty?
    - Ryhmätyökalut
        - versionhallinta GitHub, Snapchat-ryhmä, Zoom projektin ja ajankäytön suunnittelussa
    - Ohjelmistokehitystyökalut
        - AndroidStudio, kuvake Paint 3D
    - Testaustyökalut
        - AndroidStudio, Virtual device/emulator
    - Dokumentoitityökalut
        - Word dokumentaatiolle, Visual Paradigm luokkakaaviolle, tietokoneen screen capture –toiminto

Luokkakaavio



Luokkakaavio löytyy myös tästä linkistä.

Toteutetut ominaisuudet

Ominaisuus	Perustelut	Pisteet
Olio-ohjelmoitu	Olemme käyttäneet oliota hyväksi muun muassa tiedonsiirrossa ja käyttäjien luomisessa.	Pakollinen
Vähintään viisi erilaista luokkaa & oliota (käyttöliittymäluokkia ei lasketa)	Luokkia on yhteensä 12. Jokaisen osa-alueen ominaisuudet vaativat oman luokkansa, esim. CalorieEntry käsittelee kaloreita ja EntryManager käsittelee tiedon siirtoa muista luokista logiin	Pakollinen
Vähintään yhden APIn käyttö: https://fineli.fi/fineli/api/v1/foods	Käytimme APIa erilaisten ruokalajien kalorisisällön saamiseksi. Tietoa käytetään käyttäjän syömien kaloreiden laskemiseksi.	Pakollinen

Sovellus tallentaa käyttäjän toiminnan (käyttäjän syöttämät arvot / tulokset) logiin (JSON, XML jne.)	Sovellus tallentaa käyttäjän syöttämät kalori- ja uniarvot csv-muotoiseen logiin myöhempää käyttöä varten. Tunnisteina on käyttäjänimi ja päivämäärä.	Pakollinen
Logia on mahdollista tarkastella (puhtaana tekstinä, graafisilla käppyröillä jne.), eli voidaan tutkia arvojen (esim. oma massa) kehitystä kirjausten edetessä	Logia on mahdollista tarkastella Analytics fragmentin kuvaajista, jotta käyttäjä voisi tarkastella edistystään.	Pakollinen
Ohjelma on rakennettu hyvin suunnitelluista UI-komponenteista	Sovelluksen käyttöliittymä on selkeän ja tyyliltään yhtenäinen. Fontti ja värit lisäävät sovelluksen viihtyvyyttä ja tekstit ohjaavat ja neuvovat käyttäjää sovelluksen käytössä. Siirtyminen fragmenttien ja aktiviteettien välillä on sujuvaa.	1 – 5 pistettä
Kirjautuminen applikaatioon	Sovelluksessa voidaan luoda käyttäjä, jotta käyttäjä voisi syöttää ja tarkastella omia tietojaan, sekä luoda uuden käyttäjän halutessaan.	3 pistettä
Sovelluksella voi olla useampi käyttäjä (ja niiden luominen), tietojen tallennus järkevästi jonnekin	Omat tiedot tallennetaan omaan käyttäjään, jolloin käyttäjän on mahdollista tarkastella vain omia tietojaan.	3 pistettä
Kirjautumisen salasana noudattaa hyvän salasanan sääntöjä (sisältää vähintään yhden numeron, erikoismerkin, ison ja pienen kirjaimen, on vähintään 12 merkkiä pitkä)	Tietoturvan takia salasanan on noudatettava mainittuja hyvän salasanan piirteitä.	2 pistettä
Salasanan tallennus käyttää jonkinlaista hash-menetelmää ja suolausta (esim SHA-512 + salt)	Salasanan tallentaminen tietoturvallisempaa.	2 pistettä
Jokin toinen datalähde fiksusti implementoituna eli sovellus käyttää siis useampaa datalähdettä kerralla. Csv-tiedosto tallennettu assets kansioon, lähde löytyi osoitteesta: kaggle.com	Sovellus saa tiedostosta urheilulajin nimen ja kalorinkulutuksen. Kalorien määrä lasketaant tietoja sekä käyttäjän painoa ja suorituksen kestoa hyödyntäen.	2 – 5 pistettä
Ohjelmaan on mahdollista syöttää perustiedot (nimi, pituus, paino, syntymäpäivä, sukupuoli (halutessaan)) käyttäjästä ja näitä arvoja käytetään jossakin	Käyttäjän luomisessa sovellus pyytää käyttäjältä mainitut tiedot. Tietoja käytetään BMI:n sekä kalori- ja unisuositusten laskentaan	2 pistettä

Ohjelma kerää käyttäjän nukkumasta unesta dataa (massan sijasta) ja näyttää muutokset graafisesti havainnollistaen ruudulla pylväsdiagrammeina.	Pylväsdiagrammin avulla käyttäjä voi helposti seurata nukkumistottumustensa muutosta viimeisen viiden päivän aikana. Näin hän voi myös kätevästi katsoa, toteutuuko hänen asettamansa tavoite.	3 pistettä
Ohjelma näyttää graafisesti käyttäjän syömät ja kulutetut kalorit päivän aikana viivadiagrammeina viimeisimpien päivien aikana. Kaloriseuranta korvaa ilmastodieetin seurannan.	Päällekkäisten viivadiagrammien avulla käyttäjä voi helposti seurata syömiensä ja kuluttamiensa kalorien muutosta viimeisen viiden päivän aikana. Näin hän voi myös helposti verrata arvojaan asettamaansa tavoitteeseen.	3 pistettä
Ohjelma peilaa eri datalähteitä ristiin ja muodostaa uutta tietoa: saa +/- –kalorit kahdesta eri lähteestä ja luo niiden pohjalta kuvaajat ja summakuvaajan	Ohjelman on saatava ruokien sisältämien ja liikunnan polttamienkalorien määrä ulkoisista lähteistä, joina olemme käyttäneet APIa ja csv-tiedostoa. Näistä tiedoista piirretään kuvaaja ja havainnollistetaan näiden erotustakin kuvaajalla käyttäjän tarkasteltavaksi.	3 pistettä
Ohjelma muistaa käynnistämisen / kirjautumisen jälkeen mitä arvoja käyttäjä on syöttänyt aikaisemmin.	UI:n “tila” tallennetaan SharedPrefrences avulla ja käyttjän kirjautuessa takaisin näkymä on sama, kun se oli lähtiessä. Kirjautuessa päätyy kuitenkin aina MainActivityn HomeFragmenttiin. (sovelluksesta pääsee pois vain kirjautumalla tai sulkemalla koko sovelluksen).	2 pistettä?
Fragmenttien hyödyntäminen aktiviteettien sijasta käyttöliittymiä rakennettaessa	Fragmentteja hyödynnetään MainActivityssä DownBar:ssa, jotta käyttäminen olisi sujuvaa ja nopeaa. Fragmenttia hyödynnetään myös uuden salasanan asettamisessa asetuksissa. Uuden Entryn lisääminen sekä popup-ikkuna avaa uuden aktiviteetin ja entry tallentuu aktiviteetista poistuttaessa. Myös kirjautuminen ja rekisteröityminen on toteutettu aktiviteeteilla.	2 pistettä
(Oma ominaisuus) Käyttäjätietojen muokkaus asetuksissa. Tavoiteunen ja kalorien saannin tavoitteen muokkaaminen omissa popup-ikkunoissaan	Käyttäjä on saattanut syöttää jotakin väärin, hänen tietonsa ovat saattaneet muuttua tai hän saattaa haluta muuttaa tietojaan muuten vaan.	0 – 5 pistettä

(Oma ominaisuus) Random fact –laatikko home näkymässä. Päivittyy satunnaisesti erilaisia faktoja terveyteen liittyen, kun laatikkoa napauttaa.	Ominaisuus on hauska ja käyttäjä varmasti pitää faktoja mielenkiintoisina. Ne voivat myös auttaa käyttäjää parantamaa elintapojaan.	0 – 5 pistettä
(Oma ominaisuus) Käyttäjän syöttämät kalori-entryt päivittyvät RecyclerView:n, mistä niitä voi tarkastella ja poistaa.	Käyttäjää voisi kiinnostaa nähdä aikaisemmat entrynsä arvioidessaan kyseisen päivän syömisiään tai liikkumisiaan. Hän saattaa myös haluta poistaa tiedot ja asettaa niille uudet arvot.	0 – 5 pistettä
(Oma ominaisuus) Laskee yöunien pituuden ja tavoiteunen perusteella käyttäjän vireystason ja antaa ohjeita jaksamiseen.	Näin sovellus voi ohjata käyttäjää parantamaan nukkumistottumuksiaan jaksaakseen paremmin.	0 – 5 pistettä
Summa		40 pistettä (41-68)

Työmäärät

Tekijä	**Tehtävät**	**Tunnit**
Helmi Toropainen	UI:n pohja, CalorieActivity ja datalähteet, entryjen tiedonsiirto ja kirjaus logiin, näkymien tallennus SharedPrefrences (, repositoryn omistaja)	Noin 80 h
Margeritta El-Khoury	Graafisten esitysten laatiminen, SleepActivity, kuvake, luokkakaavio	Noin 45h
Mette Suvanto	Rekisteröityminen, käyttäjätietojen tallennus, kirjautuminen, käyttäjätietojen pävittyminen asetuksiin ja mahdollisuus niiden muokkaamiseen, login tekeminen, testidata, luokkakaavio	Noin 65 h
Summa		Noin 190 h

Mitä opin harjoitustyöstä?

Helmi Toropainen:

- Tiedonsiirtoa fragmenttien/aktviteettien välillä syvälliesemmin
- Funktioiden kutsuminen eri luokkien/aktiviteettien/fragmenttien välillä
- SharedPrefrences käyttö syvällisemmin (miten sinne saa ArrayListin<olio> tai olion json avulla)
- RecyclerView syvällisemmin
- JSON-muotoisen APIn lukeminen
- Ryhmätyötaitoja
- GitHub ja git

Margeritta El-Khoury:

- Aktiviteettien ja fragmenttien käyttöä luontevammin
- Syvällisempi ymmärrys olioiden tärkeydestä ja käytöstä tiedonsiirrossa
- Syvempi ymmärrys string[] ja ArrayListojen käytöstä
- GraphView-kirjaston käyttö ja sen ominaisuudet
- GitHubin käyttö ja versionhallinta yleensä ja sovelluksen rakentamiseen kuuluvat vaiheet

Mette Suvanto:

- Aktiviteettien ja fragmenttien käyttämistä
- UI-komponenttien käyttöä syvällisemmin
- SharedPreferences käyttäminen
- Tiedoston luominen ja sen tallentamisen emulaattorin muistiin
- Salasanan tallentamisen yhteydessä tehtävä suolaus ja hash
- Ryhmätyötaitoja
- Github ja git

Palaute harjoitustyöstä (vapaaehtoinen)

- Mitkä ominaisuudet / toiminnot olivat helppoja / vaikeita toteuttaa?
    - Välillä oli vähän hankaluuksia, mutta meillä oli sellainen vaikeuksien kautta voittoon –asenne, muilta ja googleta kysyttiin apua, jos ei meinannut onnistua. Välillä keksittiin uusi tapa toteuttaa joku asia, jos alkuperäinen suunnitelma ei tahtonut onnistua.
    - Vaikeinta taisi lopulta olla ajankäyttö ja sen suunnittelu – aloitettiin vähän liian myöhään. Myös GitHub tuotti vähän ongelmia, mutta niistä selvittiin.
    - Meillä oli selkeä työnjako, joten jokaisen oli helppo kehittää omia alueitaan, ja koodien yhdistäminen sujui hyvin. Myös ryhmätyöskentely oli helppoa ja onnistunutta.
- Oliko jokin asia aivan syvältä?

    - GitHub.. Vähän
- Oliko jokin asia todella hyvää tässä työssä?
    - Se, että oli aika vapaat kädet suunnitteluun ja toteutukseen.
- Mitä toivoisit ensi vuoden harjoitustyöhön?
    - Opiskelijoita voisi varoitella enemmän siitä, kuinka paljon työ vie aikaa, me ei ainakaan fukseina osattu varautua tähän ja palautus jäi aika viime tippaan ja jouduttiin karsimaan toimintoja ja silti tuli aika myöhään valvottuja öitä ja pitkiä päiviä.
    - Lisäksi voisi vahvasti suositella käyttämään versionhallintaa jo viikkotehtävien aikana, sen (GitHub ja git) opettelu vasta harkkatyötä tehdessä tuotti välillä hiukan päänvaivaa... se vei paljon aikaa (ja motivaatiota), jota olisi voinut käyttää harjoitustyön työstämiseen