Tomi Manninen

Oppimispäiväkirja

Informaatioteknologian ja viestinnän tiedekunta

Mobiiliohjelmointi -kurssin oppimispäiväkirja

Helmikuu 202

# Sisällys

[1 Mitä olen tehnyt kurssin eteen 1](#_Toc534589074)

[2 Käyttämäni AI -työkalut 1](#_Toc534589075)

[3 Ajankäytön dokumentointi 2](#_Toc534589076)

[4 Harjoitustehtävän dokumentointi 3](#_Toc534589077)

[5 Käyttämäni verkkoaineistot 5](#_Toc534589078)

[6 Harjoitustyön dokumentointi 7](#_Toc534589079)

[7 Lähdeluettelo 10](#_Toc534589085)

# Mitä olen tehnyt kurssin eteen

* 31.1.2024 Katsoin luennot 0 ja 1, tein GIT -etärepositorion johon lisäsin oppimispäiväkirja pohjan ja tein harjoituksen 1. Asensin Android Studion. Aikaa meni noin 4 tuntia.
* 1.2.2024 Tein harjoituksen 3. Aloitin etsimään hyviä resursseja Kotlinin ja Android kehityksen oppimiseen. Tähän mennessä olen päätynyt Philipp Lacknerin YouTube kanavaan (https://www.youtube.com/@PhilippLackner) ja Kotlinin viralliseen dokumentaatioon (<https://kotlinlang.org/docs/home.html>). Tutustuin Kotlin kieleen. Aikaa meni noin 3 tuntia.
* 2.2.2024 Tutustuin Kotlin kieleen aiemmin mainitsemieni YouTube -kanavan ja Kotlinin dokumentaation avulla. Harjoittelin myös itse koodin kirjoittamista. Aikaa meni noin 4 tuntia.
* 3.2.2024 Viimeistelin Kotlinin perusteisiin tutustumisen aiemmin mainitsemani YouTube kanavan avulla ja tutustuin hieman aiheeseen *Concurrency*. Aikaa meni noin 2 tuntia.
* 5.2.2024 Aloin suorittamaan *Android Basics with Compose* -kurssia (<https://developer.android.com/courses/android-basics-compose/course>) Aikaa meni noin 4 tuntia.
* 6.2.2024 Suoritin edellä mainitsemaani Android -kurssia. Aikaa meni noin 3 tuntia
* 7.2.2024 Suoritin Android -kurssia ja aloitin harjoituksen 4. Aikaa meni noin 5 tuntia.
* 8-11.2.2024 Jatkoin Android -kurssin suorittamista. Aikaa meni noin 3 tuntia.
* 12.2.2024 Jatkoin Android -kurssin suorittamista ja aloitin tehtävät 4-5 uudestaan. Aikaa meni noin 4 tuntia.
* 13.2.2024 Viimeistelin tehtävät 4-5 ja suoritin Android -kurssia. Aikaa meni noin 4 tuntia.
* 14.2.2024 Suoritin Android -kurssia. Aikaa meni noin 3 tuntia.
* 15.2.2024 Suoritin Android -kurssia. Aikaa meni noin 3 tuntia.
* 17.2.2024 Tein Android -kurssiin liittyvää harjoitusprojektia. Aikaa meni noin 3 tuntia.
* 18-22.2.2024 Tein Android -kurssia. Aikaa meni noin 10 tuntia.
* 23.2.2024 Aloitin tekemään harjoituksia 6-8. Aikaa meni noin 2 tuntia.
* 24-27.2.2024 Sain päätökseen harjoitukset 6-8. Aikaa meni noin 5 tuntia.
* 28.2-5.3.2024 Suoritin harjoitukset 9-10. Aikaa meni noin 8 tuntia
* 6.3.2024-26.3.2024 Suoritin harjoitukset 11-16. Aikaa meni noin 16 tuntia.
* 27-31.3 Suoritin harjoitukset 17-20. Aikaa meni noin 6 tuntia.
* 1.4-5.5.2024 Työstin kurssin loppuprojektia. Aikaa meni noin 60 tuntia.

# Käyttämäni AI -työkalut

Olen käyttänyt ChatGPT AI -työkalua tähän mennessä selittämään erilaisia mobiiliohjelmointiin liittyviä konsepteja. Käytän sitä mahdollisimman vähän itse koodaamiseen, jotta opin toteuttamaan asioita itse ja ymmärtämään koodin toimintaa. Olen käyttänyt sitä myös debugaamiseen ja ideointiin.

# Ajankäytön dokumentointi

|  |  |
| --- | --- |
| Tehtävä | Tunnit |
| 1-5 | 14 |
| 6-10 | 15 |
| 11-15 | 14 |
| 16-20 | 8 |
| Muu harjoittelu | 42 |
| Projektityö | 60 |
| Yhteensä | 153 |

# Harjoitustehtävien dokumentointi

**Harjoitus 1**

Tämän tehtävän tavoitteena on selvittää jonkin laitteen ohjelmoinitiin liittyviä asioita.

* Valitsin laitteeksi oman puhelimeni.
* Valmistaja on Google ja malli on Pixel 6a.
* Android versio on 13. Ominaisuuksia esitteessä: <https://store.google.com/in/product/pixel_6a?hl=en-GB&pli=1>
* Ohjelmointikielet ovat pääasiassa Java ja Kotlin
* Ohjelmointiin tarvittava työkalu on Android Studio, joka sisältää lähes kaiken tarvittavan.
* Googlen Pixel -puhelimet sisältävät Googlen sovelluksia, mutta ei paljokaan muita valmistajakohtaisia sovelluksia tai ominaisuuksia, vaan ne pyrkivät tarjoamaan mahdollisimman ”puhtaan Androidin”, joka tarkoittaa alkuperäistä Android käyttöjärjestelmää hyvin lähellä olevan kokemusta. (https://www.androidauthority.com/what-is-stock-android-845627/)
* Puhelimen sovelluskauppana toimii Google Play Store. Siellä voi julkaista itse kehitettyjä sovelluksia.
* Puhelin sisältää useita ominaisuuksia, kuten GPS ja erilaisia sensoreita, joihin voi tutustua aikaisemmin tarjoamastani linkistä. Näitä ominaisuuksia voidaan käyttää Java ja Kotlin -kielillä Androidin rajapintojen avulla.

**Harjoitus 3**

Tämän tehtävän tavoitteena on asentaa Android työkalut koneelle, testata ympäristön toimiminen HelloWorld sovelluksella ja kokeilla lisätä se Git-repoon.

Heti aluksi ongelmaksi muodostui Android Studiolla ohjelman käynnistäminen, koska Android Emolator Hypervisor Driver asennus puuttui. Sen asentaminen ei myöskään onnistunut AMD prosessorilla helposti. Löysin YouTube videosta (<https://www.youtube.com/watch?v=Y1WhS2yuF8I>) onneksi ratkaisun tähän ongelmaan, joka oli BIOS asetuksen, SVM Moden kytekeminen päälle.

Alun pienten vaikeuksien jälkeen itse HelloWorld sovelluksen laatiminen, testaus ja Git-repoon lisäys oli melko suoraviivaista.

**Harjoitus 4 - 5**

Tämän tehtävän tavoitteena on tehdä yksinkertainen laskinsovellus, jolla voi laskea yhteen, erotus, kerto ja jakolaskuja.

Harjoituksessa ei tullut juurikaan vastaan ongelmia, muuta kuin tiedostoon kirjoittamisessa ja lukemisessa siten, että tiedosto, johon kirjoitetaan ja luetaan, ei menisi käyttäjän laitteen perustiedostoihin, vaan itse sovelluksen tiedostoihin. Tässä ongelmassa käytin apuna ChatGPT:tä, koska en löytänyt pienellä googlettamisella kunnollisia ohjeistusmateriaaleja.

**Harjoitus 6 - 8**

Tämän tehtävän tavoitteena on tehdä tietokantaa käyttävä Android sovellus. Vaatimuksena oli lisäys, poisto ja järjestäminen. Toteutin näiden lisäksi muokkauksen. Tein toiminnoille omat näkymät, jotta samalle näkymälle ei tulisi liikaa tavaraa.

Toteutin sovelluksen rakenteen hyödyntäen ”Repository Pattern” lähestymistapaa, jossa muu sovellus ei kommunikoi suoraan datalähteen kanssa (tässä tapauksessa tietokanta), vaan tekee sen repository -kerroksen kautta. Näin datan hakemiseen ja muokkaukseen liittyvät yksityiskohdat abstrahoituvat repositorion tarjoaman käyttöliittymän taakse. UI staten ja logiikan asetin kunkin näkymän omaan ViewModeliin.

StateFlow:n kanssa oli hieman hankaluuksia saada muutokset päivittymään käyttöliittymään, mutta Android developer guide auttoi tässä. Käytin hyödyksi Android -kurssilla oppimiani asioita tässä tehtävässä.

**Harjoitus 9 – 10**

Tämän tehtävän tavoitteena oli tehdä Firebase:n autentikaatiota ja firestore-tietokantaa hyödyntävä Android sovellus. Vaatimukset olivat samat, kuin harjoituksessa 6-8, mutta nyt oli lisäksi autentikointi. Toteutin myös uloskirjautumisen ja salasananvaihdon. Oli jälleen hieman vaikeuksia saada muutokset päivittymään käyttöliittymään itemin poiston tai lisäyksen jälkeen, mutta tuntien pohtimisen jälkeen päädyin siihen, että kun navigoin takaisin lista näkymään, niin pakotan sen ”Composoitumaan” uudestaan. Käytin avuksi Firebasen tutoriaaleja, blogia ja hieman ChatGPT:tä. Kopioin edellisen tehtävän koodin ja muutin sen käyttämään firebasea, siksi pakkausnimet ovat väärin.

**Harjoitus 11**

Tämän tehtävän tavoitteena oli saada sensoridata näkyviin laitteesta, erityisesti asento, etäisyys ja valo sensorien data. Tehtävän teko oli melko suoraviivaista. Käytin apuna YouTube-tutoriaalia ja Androidin omaa dokumentaatiota. (https://www.youtube.com/watch?v=IU-EAtITRRM)

**Harjoitus 12**

Tämän tehtävän tavoitteena oli toteuttaa oikeaa kameraa eikä mobiililaitteen kamerasovellusta käyttävä appi ja kuvan talletus. En löytänyt modernia Jetpack Composea käyttävää tutoriaalia CameraX:n Androidin dokumentaatiosta, joten käytin YouTube tutoriaalia avuksi. (<https://www.youtube.com/watch?v=12_iKwGIP64>)

**Harjoitus 13-14**

Tämän tehtävän tavoitteena oli toteuttaa TYPE\_STEP\_COUNTER, TYPE\_STEP\_DETECTOR tai kiihtyvyysanturilla toimiva askelmittari sovellus. Itse toteutin sovelluksen siten, että käytettävä sensori riippuu käyttäjän puhelimessa saatavilla olevista ominaisuuksista. TYPE\_STEP\_COUNTER ei kuulemma toimi esimerkiksi joissain Xiaomi puhelimissa. Tallennuksessa käytin Datastorea. Lähteinä käytin: <https://medium.com/@huuphuoc1396/android-step-detector-using-sensors-9f1a265a6069>, <https://developer.android.com/courses/pathways/android-basics-compose-unit-6-pathway-3>, <https://github.com/huuphuoc1396/AndroidStepDetector> sekä hieman Chat GPT. Tehtävä oli kohtalaisen helppo suorittaa näiden lähteiden avulla, eikä suurempia ongelmia tullut vastaan.

**Harjoitus 15-16**

Tämän tehtävän tavoitteena oli toteuttaa laitteen sijainnin paikannus gps -pisteenä eli longitudi ja latitudi ja mahdollisuus tallentaa ne tiedostoon. Sen lisäksi tallennetut pisteet ja laitteen nykyinen sijainti näytetään kartalla. Käyttäjältä pitää myös kysyä tarvittavat luvat, jotta hänen laitteensa sijainti voidaan selvittää. Tehtävässä käytin Maps Composea: <https://github.com/googlemaps/android-maps-compose> ja sijainnin selvitykseen käytin apuna Androidin tutoriaalia: https://developer.android.com/develop/sensors-and-location/location/retrieve-current. Tehtävän teossa ei tullut mainitsemisen arvoisia ongelmia. Kuva laitteen sijainnista ja 10 talletetusta paikasta:

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, Fontti

Kuvaus luotu automaattisesti

**Harjoitus 17**

Tämän tehtävän tavoitteena oli tehdä valuuttalaskin sovellus, joka laskee eurojen vaihdon kolmella eri valuutalla ja näyttää tulokset. Valuuttakurssit haettiin verkossa olevasta rajapinnasta. Käyttämäni API oli: <https://www.exchangerate-api.com/>. HTTP – pyynnöt tein Retrofit kirjastolla. Serialisoinnin ja deserialisoinnin hoidin Gson kirjastolla. Tehtävän teko oli melko suoraviivaista, sillä olen tehnyt saman tyyppisiä asioita aikaisemmin tekemässäni android sovelluksessa, ja käytin sitä apuna, kun en muistanut jotain.

**Harjoitus 18**

Tämän tehtävän tavoitteena oli tarkoitus toteuttaa sovellus, joka antaa käyttäjän hakea sijaintia ja antaa sijainnin perusteella viisi viimeisintä lämpötila-arvoa ilmatieteenlaitoksen palvelusta mittausajan kanssa. En ole ennen käsitellyt XML dataa ja sen kanssa oli hieman haasteita. Käytin avuksi artikkelia: <https://etemnogrudova.medium.com/parsing-xml-using-retrofit2-in-kotlin-8e0ff6420bd7> ja ChatGPT data luokkien luomiseen.

**Harjoitus 19**

Tämän tehtävän tarkoitus oli toteuttaa Widget mobiililaitteen ruudulle, jossa hyödynnetään jommankumman edellisen tehtävän dataa. Itse tein valuuttalaskin sovellukselle. Näytin Widgetissä muuntokurssit kolmelle eri valuutalle. Toteutin tehtävän Jetpack Glancella ja tällä ohjeistuksella: <https://developer.android.com/develop/ui/views/appwidgets/overview>. Toteuttamisessa ei tullut vastaan ongelmia.

**Harjoitus 20**

Etäopetus toteutettiin ihan hyvin. Katsoin aika vähän luentoja, joten niistä ei ole juurikaan sanottavaa. Moodle alustalta löytyy riittävät tiedot kurssin suorittamiseen. Coursegitlab toimii hyvänä tehtävien palautus alustana.

Kurssilla oli hyvää sisältöä android kehityksestä, mutta esimerkiksi layoutista, teemoista ja semmoisista olisi voinut olla enemmän. Jetpack composesta olisi voinut myös olla opetusmateriaalia, koska se taitaa olla se teknologia millä modernit android sovellukset tehdään.

Itse luen mieluummin koodia ja tekstimuodossa olevaa opetusmateriaalia, kuin katson luentoja koska se sopii paremmin oppimistyyliini.

Käytin tarjottua oheismateriaalia, mutta etsin myös itse netistä erilaisia materiaaleja. Tarjotut materiaalit olivat hyödyllisiä, mutta Jetpack Compose materiaaleja olisi voinut olla enemmän.

Käytin apuna satunnaisesti ChatGPT:tä. Käytin sitä koodin debugaamiseen, ideointiin ja konseptien selittämiseen. Koin tekoälyn käytön hyödylliseksi. Mielestäni tekoälyn käyttöä tulisi ohjeistaa siten, että suositellaan opiskelijoita itse keksimään ratkaisuja ongelmiin, sillä tämän kaltainen lähestymistapa palvelee heitä enemmän työelämää ajatellen, kuin se, että kopioidaan tekoälyn tuottama koodi ilman, että mietitään miten se toimii.

Mielestäni kaikki tehtävät olivat hyödyllisiä, sillä niissä opeteltiin asioita, joita esiintyy useimmissa Android sovelluksissa. En kokenut mitään tehtävää liian vaikeaksi tai helpoksi.

Kurssi tuntui laajuudeltaan sopivalta. Laajuuden sopivuus tosin riippuu siitä, miten tehtävien tekoa lähestyy. Voisin kuvitella, että kurssi tuntuu laajuudeltaan aika pieneltä jos antaa tekoälyn hoitaa suurimman osan.

# Harjoitustyön dokumentointi

Kurssin harjoitustyöksi päätin tehdä simppelin frisbeegolfsovelluksen, jolla voi pitää kirjaa pelatuista kierroksista yhdelle henkilölle, tarkastella aikaisempia kierroksia, etsiä ratoja kartalta tai listalta ja tarkastella tilastoja pelatuista kierroksista eri aikaväleillä. Sovellus on toteutettu käytettäväksi vain suomessa. Sovellus on ottanut inspiraatiota UDisc sovelluksesta.

Kierrosten kirjapito-ominaisuuden toteutuksessa tuli vastaan hieman ongelmia. PDGA:n frisbeegolfrata dataa kattavasti tarjoava API ei ollut saatavilla enää kehittäjille, joten jouduin hakemaan datan Disc Golf Metrix nimisen palvelun API:sta. Sieltä saatava data ei ollut kovin täydellistä, koska joistain radoista puuttuu esimerkiksi kokonaan tiedot koreista. Ratkaisuksi tähän ongelmaan päätin luoda ominaisuuden, jolla käyttäjä voi itse syöttää tiedot korien määrästä ja niiden par lukemista, jos ne puuttuvat. Lisäsin myös mahdollisuuden luoda mukautetun radan, jolloin käyttäjä voi valita itse radan nimen, korimäärän ja par lukemat. Tämä mahdollistaa sen, että käyttäjä voi seurata tuloksia radalta, jota ei löydy valmiiksi sovelluksesta. Kun käyttäjä valitsee rataa, jolla pelata, hän voi valita radan joko lähellä (25 km säteellä) olevan radan, tai jonkun viimeksi pelatuista radoista.

Ratojen tarkastelun voi tehdä joko listalta tai kartalta, ja niiden välillä voi vaihtaa helposti yhdellä painikkeella. Kun hakukenttään alkaa kirjoittamaan radan nimeä, niin ruudulle suodattuvat käyttäjän kirjoittamien kirjainten perusteella täsmäävät radan nimet. Kartalta ratoja voi etsiä tutkimalla ympäriinsä ja rata markkeria klikkaamalla näkee radan nimen, sijainnin ja korimäärän, jos se on saatavilla. Toteutin karttaan zoomaus tasosta riippuvaisen markkerien klusteroinnin, joka parantaa kartan suorituskykyä moninkertaisesti. Kartta on rajoitettu siten, että kameraa ei pysty liikuttamaan ulos suurin piirtein suomen alueelta. Kartassa on myös käyttäjän sijainnin näyttävä ikoni, joka näyttää myös mihin suuntaan käyttäjän laite osoittaa. Kartassa on mukautettu tumma teema, ja ylimääräiset markkerit on poistettu(kaupat, nähtävyydet jne.).

Sovelluksesta voi tarkastella myös statistiikka dataa. Sivu näyttää käyttäjän yhteismäärän käyttäjän pelatuista kierroksista ja radoista, väylistä, heitoista ja pelatuimman radan. Lisäksi sovelluksessa voi tarkastella par tuloksia eri aikaväleiltä, jossa on vaihtoehdot: viimeiset 10 kierrosta, viimeiset 20 kierrosta, tulokset valitulta vuodelta ja viimeiseksi tulokset kaikelta ajalta. Tulokset näytetään horisontaalisesti asetetussa pylväsdiagrammissa, josta pylvästä klikkaamalla voi nähdä tuloksen.

Sovelluksen käyttöliittymä on toteutettu Jetpack Composella. Tiedon tallennukseen käytin Room tietokanta kirjastoa ja datastorea. Toteutin myös WorkManagerin, joka hakee tiedot APIsta ensimmäisellä käynnistyskerralla ja sen jälkeen viikon välein sen varalta, jos dataan tulee muutoksia. Käytin API datan hakuun Retrofit kirjastoa. Sensorit, jotka käytin ovat kiihtyvyys ja magnetometer sensorit. Käyttäjän sijainnin hakuun käytin FusedLocationProviderClient. Kartan totetin android-maps-compose kirjaston avulla. Par tulos diagramming toteutin YCharts diagrammi kirjaston avulla.

Käyttöohje:

Kuva, joka sisältää kohteen teksti, kuvakaappaus, ohjelmisto, numero

Kuvaus luotu automaattisesti

1. Tarkastele pelatun kierroksen tietoja, klikkaa korttia nähdäksesi yksityiskohtia

2. Aloita kierroksen valmistelu

3. Navigoi eri ruutuihin

4. Valitse rata lähellä olevista, viimeksi pelatuista tai luo oma

5. Klikkaa rataa aloittaaksesi kierros

6. Anna omalle radalle nimi ja aloita kierros

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, ohjelmisto

Kuvaus luotu automaattisesti

7. Siirry väylien välillä swaippaamalla

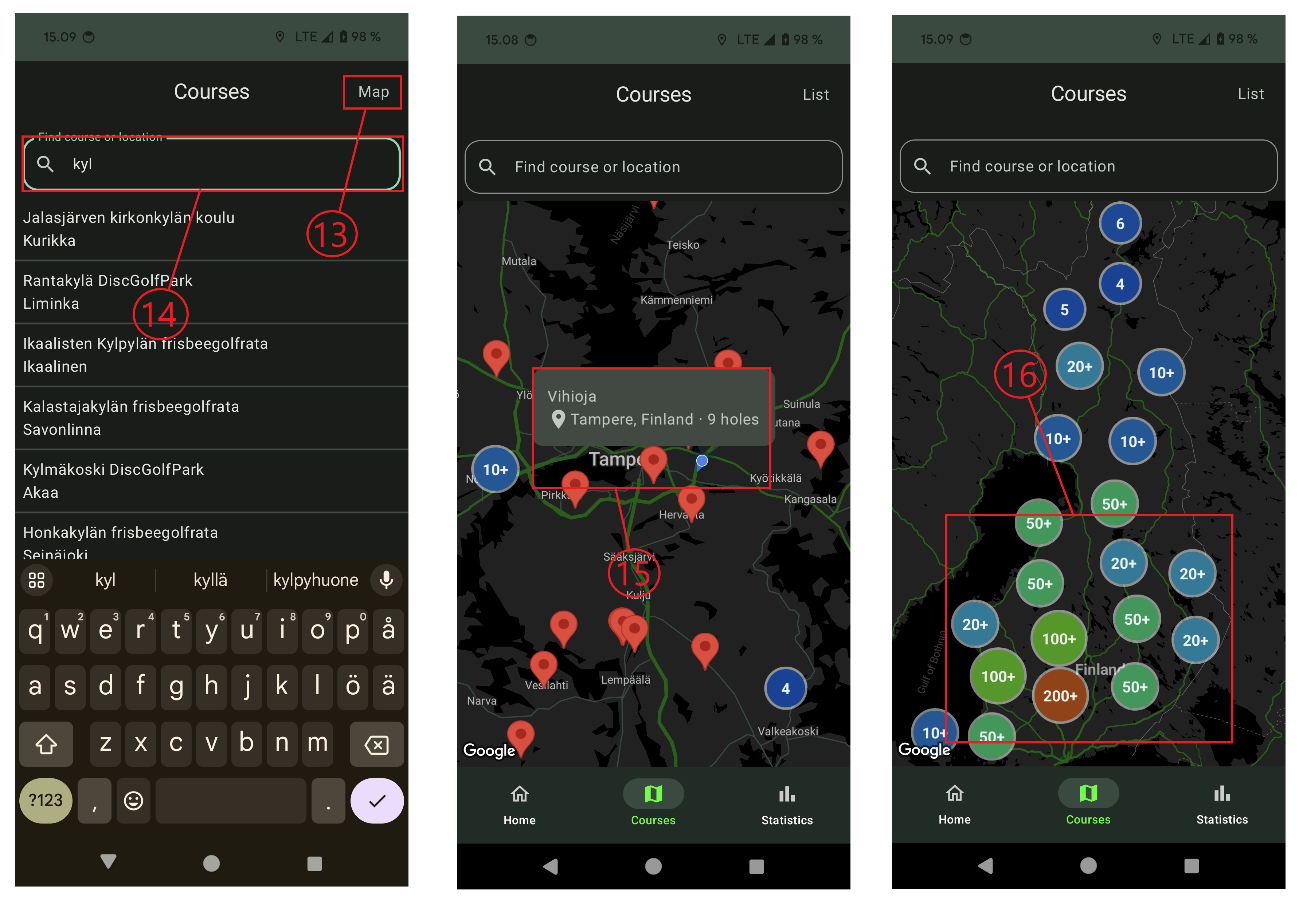
8. Muokkaa väylän par lukemaa

9. Muokkaa väylän tulosta

10. Siirry väylien välillä klikkaamalla väylän numeroa

11. Täytä väylien määrä, jos ei saatavilla API:sta

12. Tarkastele kierroksen tuloksia ja lopeta kierros

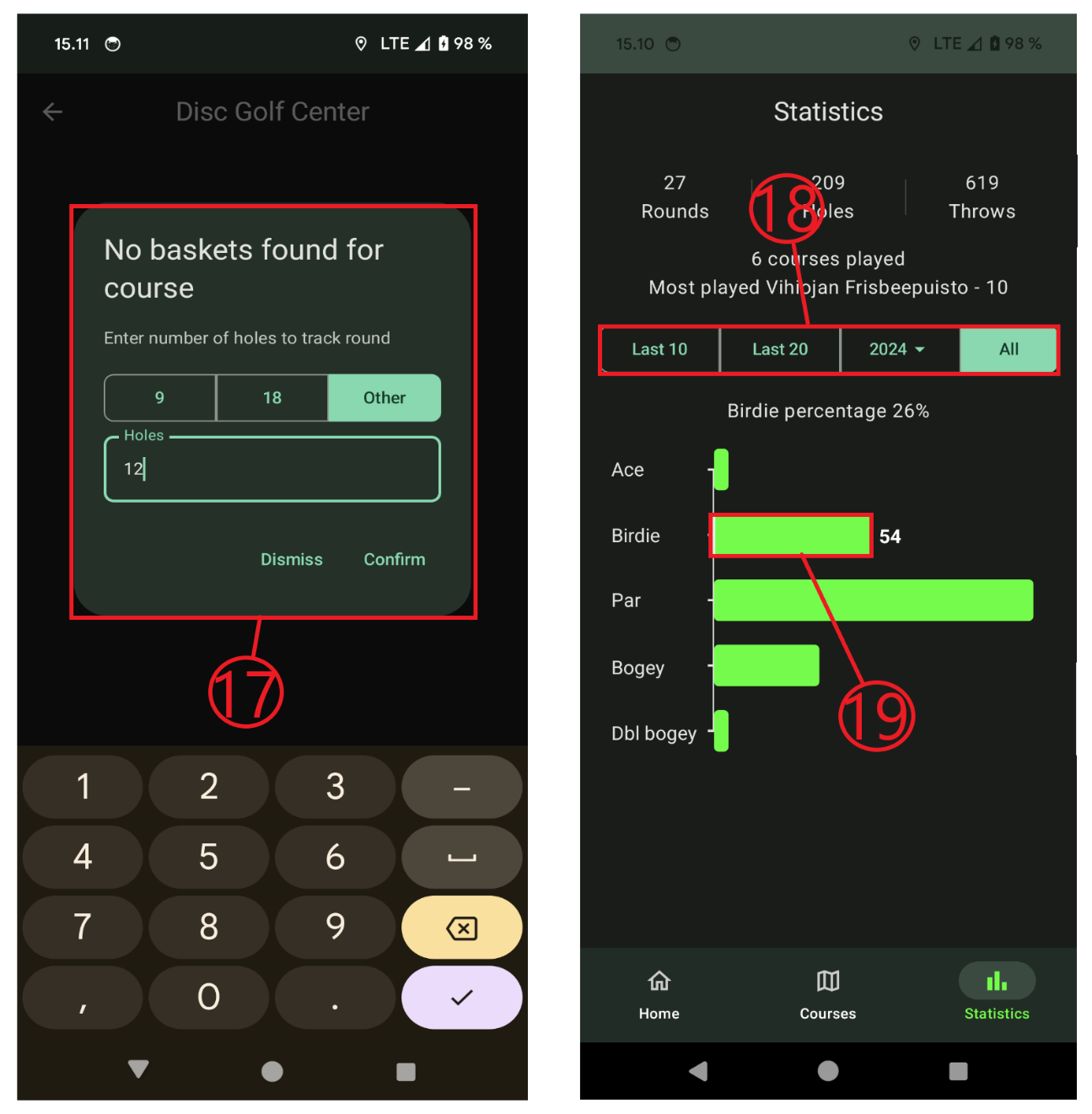


13. Siirry kartta ja lista näkymän välillä

14. Etsi ratoja nimeltä

15. Klikkaamalla markkeria näkee radan tietoja

16. Zoomaamalla ulos näkee radat klusteroituna määrillä

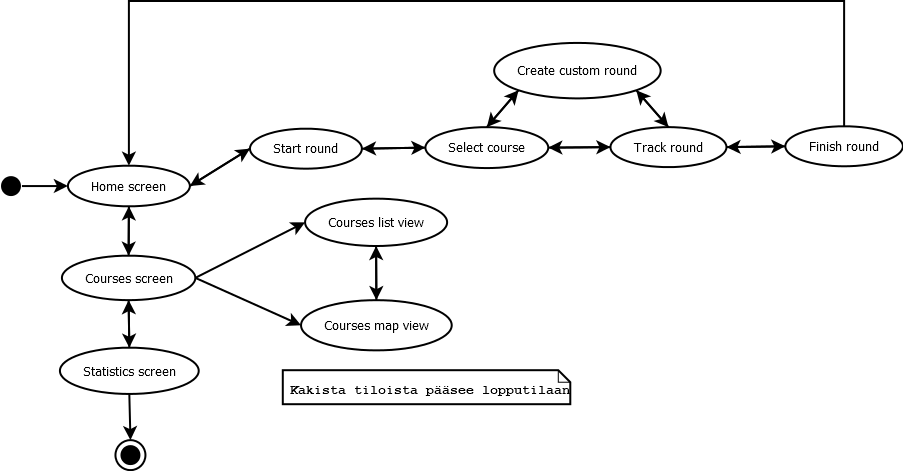


17. Täytä väylien määrä, jos ei saatavilla API:sta

18. Valitse miltä aikaväliltä näytetään par tuloksia

19. Klikkaa pylvästä jotta näet tuloksen lukumäärän

Activity diagram:



# Lähdeluettelo

Verkkoaineistot, joita olen käyttänyt (olen käynyt kaikkia niistä kattavasti läpi):

Android developer documentation, <https://developer.android.com/>

Android developer course, <https://developer.android.com/courses>

Philipp Lackend youtube kanava, https://www.youtube.com/@PhilippLackner

Android maps compose, <https://github.com/googlemaps/android-maps-compose/tree/main>

YCharts library <https://github.com/codeandtheory/YCharts>

ChatGPT, <https://chatgpt.com/>

Metrix course data API, <https://discgolfmetrix.com/?u=rule&ID=37>

Material Design https://m3.material.io/