

# PGR208 Android Programming

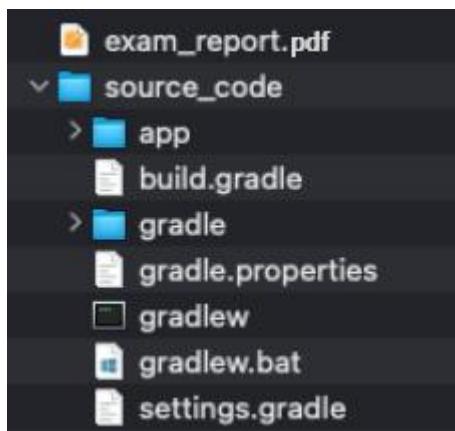
## Eksamens Høst 2025

---

Løses i gruppe på 2 eller gruppe på 3. Det forventes at en gruppe på 3 har mer omfang og kompleksitet enn en gruppe på 2.

Instruksjoner for opplasting av besvarelse til WISEFlow:

1. Kjør Build -> Clean Project for å minske prosjektstørrelse
2. Slett local.properties-filen. Dette er for å sikre kandidatens anonymitet da local.properties-filen kan inneholde brukernavn. Se bilde under for eksempel på hvilke filer/mapper (øverste nivå) som skal være med i leveransen. Andre filer enn disse er ikke nødvendige for leveransen og skal ikke legges ved.
3. Legg alle deler av eksamenen i en felles mappe som zippes før opplasting til WISEFlow.
4. Merk at ren kopi paste fra hvor som helst, uavhengig av om det er nettsteder eller medstudenter eller KI, ikke er lov og vil kunne rapporteres til eksamensavdelingen som plagiering. Det er lov å finne ressurser og tilpasse og innlemme dem i sin løsning for deler av koden og funksjonalitet. Alt som er gitt av underviser i emnet er vel å merke lov å bruke direkte.



Underviseren kan legge ut avklaringer og presiseringer i emnesiden i Canvas under eksamensperioden.

En sensorveiledning er inkludert på siste side av denne eksamenen.

## **Hoveddelene av eksamenen**

I denne eksamenen skal dere gjøre bruk av teknikker dere har lært i emnet PGR208 Android Programmering og på den måten vise hva dere behersker. Eksamenen består av to hoveddeler:

1. App (75%)
2. Dokumentasjon til app (25%)

## **Generelle krav og retningslinjer**

- Dere skal gjøre bruk av hovedteknikkene som er gjennomgått i emnet. Det å velge andre hovedteknikker istedenfor vil kunne lede til ingen eller lavere poeng. Hovedteknikker refererer blant annet til Jetpack Compose, Retrofit, Compose Navigation, Room database og MVVM.
- Kotlin er programmeringsspråket og Android Studio din IDE
- Dere forventes å implementere god prosjektstruktur, koderyddighet, og god bruk av teknikker som vist i emnet.
- Dere blir ikke vurdert direkte på vakkert design, men dere blir vurdert på kode for å lage design med tanke på kompleksitet og omfang. Tenk også over hvordan en app bør utformes for å være brukervennlig – tenk over hvordan den vil oppleves av en bruker. I den sammenheng kan det være greit å tenke på å implementere feedback til bruker der det gir mening; eksempelvis etter handlinger bruker gjør osv.
- Bruk av ressurser som KI, nettsteder, guides/tutorials, andre studenter, osv. som er brukt i løsningen må henvises til i dokumentasjonen (kildereféranser). Merk: Det er lov å benytte seg av ressurser som dette, så lenge det ikke er ren plagiering (kopi av andres kode uten videre formuleringer, forklaringer og føringer).

# Android-appen (75%): JIKAN Anime App

Hovedendepunkter:

- Hente alle: <https://api.jikan.moe/v4/anime>
- Hente etter id: <https://api.jikan.moe/v4/anime/{id}>

Appen skal bestå av følgende 4 skjermer:

## Skjerm #1: Vise alle animer fra JIKAN anime

Bruker får vist x antall animeer fra API'et på skjermen.

## Skjerm #2: Søke etter en anime fra JIKAN anime

Bruker kan søke etter en anime etter id.

## Skjerm #3: Brukerens egne animeideer

Brukeren har mulighet til å lagre ideer til nye animeer og så videre redigere, slette og vise dem.

## Skjerm #4: Egendefinert skjerm

Her har dere mulighet til å selv bestemme hvilken funksjonalitet skal være med. Kravene for denne skjermen er at det er koblet til temaet. Denne skjermen **må** inneholde noe slags funksjonalitet og interaksjon, og kan derfor ikke være kun en statisk side av appen.

## Tilleggskrav:

Dere skal legge til valgfri funksjonalitet i tillegg til funksjonalitet beskrevet over. Dette kan for eksempel være en/flere ekstra egendefinert(e) skjerm(er), filtrering-/søke-funksjon i skjermer der dette gir mening, ytterligere bruk av endepunkter som APIet tilbyr for å tilgjengeliggjøre mer data/funksjonalitet i appen, osv.:

- For en gruppe på 3 vil kravet være innfridd ved at hver skjerm inneholder ekstra funksjonalitet/brukerinteraksjon som gir verdi til sluttbruker, utover det som er spesifisert over.
- For en gruppe på 2 vil kravet være noe mindre, eksempelvis kan dette være ekstra funksjonalitet på halvparten av appens skjermer utover det som er spesifisert.

# **App-dokumentasjon (25%)**

## **Del 1. Redegjørelse av app per skjerm**

For hver skjerm skal dere ha separat underkapittel i dokumentet på 250-300 ord som inkluderer følgende punkter:

- 1-3 skjermdump for å vise innholdet, før og etter bruker har gjort noe, tilbakemelding til bruker om feil/OK osv.
- Kommentarer til skjermdumpene om hvilke teknikker og evt. ressurser som er brukt og hvorfor. Det skal ikke bare være oppramsing, men refleksjon og redegjørelse av tankene rundt hvorfor det er blitt gjort slik og hvilke valg som er tatt underveis.

## **Del 2. Redegjørelse av planlegging og utvikling**

I tillegg skal dere ha et underkapittel på 250-300 ord som har med hvordan dere har arbeidet med utviklingen:

- Hvordan har dere planlagt løsningen?
- Hvordan har dere lagt opp arbeidet og fordelingen av arbeidet?
- Hvordan kommet frem til avgjørelser?
- Hvordan gikk samarbeidet?

# Assessment guide for students and assessors

## The app: 75%

The screens will have weight according to the complexity in them, as some may be simpler than others.

Make sure you think about writing good, scalable code, and using the principles learnt in the course. Also make sure to use a variety of the most important coding techniques and methods which you have learnt.

The coding of the UI also counts, not in the sense of being “beautiful” in itself, but rather that UI elements have been styled in a complex/amount-wise good enough manner. Feedback to the user after important actions or about situations is also important.

Some main elements/parts:

- **Screen #1-4**
  - Diverse use of Jetpack Compose, Kotlin programming to implement functionality and components.
- **HTTP / API with Retrofit**
  - General implementation and usage of HTTP to retrieve data from the provided third-party API
  - Coil
- **Local storage with Room**
  - General implementation and usage of storing data locally
- **UI/UX**
  - Overall UI and user experience implementation (i.e. error handling, and messages to user)
- **Navigation between screens**
  - Compose Navigation
  - Navigation and passing of data between screens where/if necessary
- **Architecture and MVVM**
  - ViewModel per screen
  - Repository

## Report: 25%

Show awareness of the techniques, code, and functionality created. Be able to have an overview of techniques. Show understanding of the selected techniques. Be able to reflect on code quality and structure.