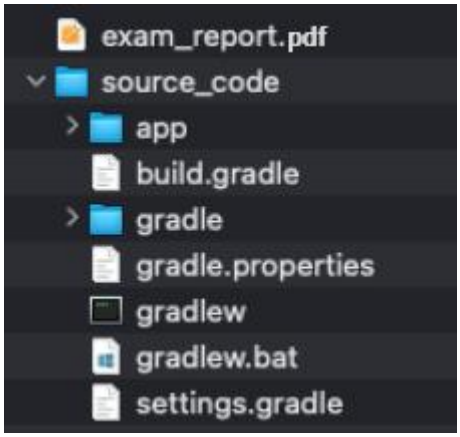
PGR208 Android Programming
Eksamen Høst 2024

Kan løses alene eller i gruppe på opptil 3. Det forventes at en gruppe på 3 har mer omfang og kompleksitet enn en gruppe på 1 eller 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.



Underviser(ne) kan legge ut avklaringer og presiseringer i emnesiden i Canvas.

En sensor/sensurveiledning er inkludert på siste side av denne eksamenen.

Hoveddelene av eksamenen
I denne eksamenen skal du gjøre bruk av teknikker du har lært i emnet PGR208 Android Programmering og på den måten vise hva du behersker. Eksamenen består av to hoveddeler:
1. App (80%)
2. Rapport (20%)

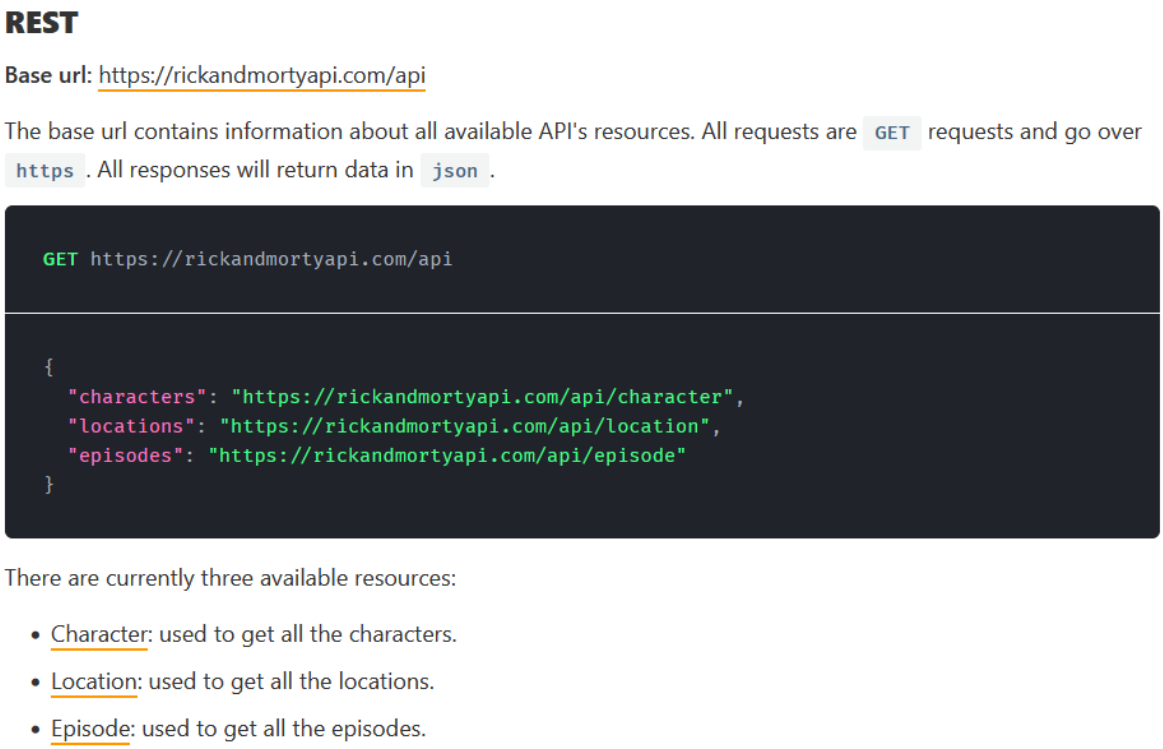
Appen (80%): RickAndMortyApp

Du skal lage en app for fans av Rick and Morty som blant annet tillater brukerne å finne opp nye karakterer og hente informasjon fra Rick and Morty Apiet:
https://rickandmortyapi.com/
Beskrivelsen nedenfor viser hva som minimum skal være av funksjonalitet per skjerm.

Appen består av følgende 4 skjermer:

Skjerm #1: Vise Rick and Morty-karakterer fra rickandmortyapi.com

Skjerm som viser liste med Rick and Morty-karakterer hentet fra apiet – ref. dokumentasjonen om endepunkt for «characters» i nettsiden
https://rickandmortyapi.com/documentation



REST

Base url: https://rickandmortyapi.com/api

The base url contains information about all available API's resources. All requests are `GET` requests and go over `https`. All responses will return data in `json`.

```
GET https://rickandmortyapi.com/api
```

```
{
  "characters": "https://rickandmortyapi.com/api/character",
  "locations": "https://rickandmortyapi.com/api/location",
  "episodes": "https://rickandmortyapi.com/api/episode"
}
```

There are currently three available resources:

- Character: used to get all the characters.
- Location: used to get all the locations.
- Episode: used to get all the episodes.

Skjerm #2: Vise brukerens Rick and Morty-karakterer

Her viser du liste med de karakterene som brukeren har lagret i lokal database ut ifra logikk implementert i skjerm #3 under.

Skjerm #3: Skap din egen Rick and Morty-karakter

På denne skjermen skal brukeren kunne opprette nye Rick and Morty-karakterer. Karakterene skal lagres i lokal database på enheten.

Skjerm #4: Egendefinert skjerm

Her har du/dere mulighet til å selv bestemme hvilken funksjonalitet skal være med. Kravene for denne skjermen er at det er koblet til temaet Rick and Morty, og ideelt at det er koblet til enten Rick and Morty-APIet og/eller data som er lagret på lokal database (se skjerm #3 og #4). Dette betyr også at denne skjermen **må** inneholde noe slags funksjonalitet og interaksjon, og kan derfor ikke være kun en statisk side av appen.

Ekstra krav til grupper på 2-3 studenter:

Det forventes at leveranser gjort av grupper på 2-3 studenter leverer ekstra, valgfri funksjonalitet i tillegg til funksjonalitet beskrevet for hver skjerm 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 APIet for å tilgjengeliggjøre mer data/funksjonalitet i appen, osv.
For en gruppe på 3 vil kravet være innfridd om f.eks. 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.
Dere velger selv hva denne ekstra funksjonaliteten skal være.

Generelle krav og retningslinjer

- Du skal gjøre bruk av teknikkene som er gjennomgått i emnet
- Kotlin er programmeringsspråket og Android Studio din IDE
- Du forventes å implementere god prosjektstruktur, koderyddighet, og god bruk av teknikker som vist i emnet.
- Du blir ikke vurdert direkte på vakkert design, men du 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.

Rapporten (20%)

En viktig del av denne eksamenen er å vise at du er bevisst teknologien du har lært i emnet og reflektere over teknikkene du bruker, og vise din kunnskap om teknikkene.

Skriv en rapport på 500-600 ord (hvis du jobber individuelt) 900-1000 ord (hvis gruppe på 2-3) som utreder og dokumenterer følgende fire ting:

1. **Oversikt over funksjonalitet du har laget:** Lag en oversikt over hvilken funksjonalitet du har laget. Dette kan eksempelvis lages med en tabell med to kolonner; navn på funksjonalitet, beskrivelse av funksjonalitet.
Merk at for grupper på 2-3 studenter skal det også dokumenteres hva dere har valgt som ekstra funksjonalitet i appen utover spesifikasjonen for hver enkelt skjerm, og hvordan/hvorfor dere har valgt å implementere dette (se "ekstra krav til grupper på 2-3 studenter" i oppgaveteksten).
2. **Skjermdump av samtlige skjermer i appen:** Skjermbilder av alle skjermer med en kommentar om grensesnittet og hva som skjer der/hva man kan gjøre der.
3. **Beskrivelser og skjermbilder av hovedteknikker brukt**: Beskriv hovedteknikkene du har brukt i appen. Med beskrivelse menes minimum navn på teknikk og hva teknikken er for, og der hvor det passer seg, også hvorfor man bruker teknikken. Inkluder skjermbilder av kode og eventuelt modeller/diagrammer for å eksemplifisere teknikkene du omtaler.
4. **Kvalitet og struktur**: Skriv hva du har gjort for å sørge for at prosjektstrukturen er god, hvordan og hvorfor du har navngitt variabler, funksjoner, klasser osv. slik du har gjort, og eventuelt andre ting som angår struktur og kodekvalitet.

Det forventes også at grupper på 2-3 studenter i rapporten utreder noe om hvordan de har samarbeidet: Fortell om hvordan dere har fordelt arbeidet, hvordan dere rent praktisk har delt/samarbeidet på koden, samt litt om hvordan dere føler samarbeidet fungerte og evt. hva som fungerte bra/dårlig, hva dere ville gjort annerledes, osv.

Rapporten skal leveres som en PDF-fil.

Assessment guide for students and assessors

The app: 80%

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**
- **HTTP / API**
  - General implementation and usage of HTTP to retrieve data from the provided third-party API
- **Local storage**
  - 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**
  - Navigation and passing of data between screens where/if necessary

Report: 20%

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.