PGR208 - Rapport

Implementasjon

Til vår løsning har vi brukt Jetpack Compose som UI-rammeverk for Android-programmering med Kotlin. Android Compose er en samling av komponentbibliotek og verktøy.

Vår løsning henter data fra Fakestoreapi.com ved hjelp av retrofit og okhttp3, samt bygger en database for å lagre/hente ut data ved hjelp av room. Vi oppretter en httpclient med logging interceptor fra okhttp3, som vi bruker fremfor default klienten i retrofit for å få mer detaljert loggdata. Videre bruker vi retrofit til å opprette en implementasjon av ProductService interfacet vårt, som vi benytter til å sende http-spørringer til API'et. Ved hjelp av coil og async image kan vi asynkront laste inn bilde fra URL uten å måtte lagre bildefiler lokalt i databasen. Vi bruker også en navcontroller for å navigere mellom appens sider, gjenspeilet i løsningens filsystem, der alle “screens” har sin egen mappe som inneholder kotlin klassefil og tilhørende viewmodel.

Når du kjører appen sender den automatisk en forespørsel til API’et, deretter lagres all informasjon i databasen. Herfra foregår all kommunikasjon videre til databasen, som ligger lokalt på device. Derfor må løsningen først kjøres gjennom med nett for å laste ned fra API’et, for at brukeren senere skal kunne bruke appen i offline modus.

Vi har implementert arkitektur mønsteret MVVM, Model-View-ViewModel, for å strukturere løsningen. Model benyttes til databehandling, og View for å definere UI og hvordan grensesnittet skal se ut. Eksempler på dette er våre data klasser (Model) hvor vi definerer og håndterer løsningens data. I

tillegg setter vi vårt design gjennom våre composable funksjoner (View). I ViewModel forenes Model og View, hvor data håndteres samt eksponeres for det gjeldende grensesnittet. Dette er gjennomgående for hele vår løsning. Vi valgte også å lage egne filer for ProductItem og OrderItem for å printe ut enheter med data og gjøre det ryddigere i filsystemet. ProductItem blir også brukt flere steder.

Av UI elementer valgte vi å implementere et søkefelt på forsiden, samt anledning til å filtrere etter produktkategori. Dette var elementer vi anså som relevant og viktig for å forbedre en eventuell brukeropplevelse. Når man trykker på et produkt får man mer informasjon om gjeldende vare. Her valgte vi å implementere UI elementer som muligheten til å sette en vare som favoritt, samt anledning til å legge igjen en rating til et produkt. Vi har også implementert ikoner/IconButtons for enkel navigering.

Refleksjoner

Noe vi ser at kunne vært forbedret eller gjort annerledes er håndteringen av å legge til flere antall av et produkt. Per nå så trykker man på “Add to cart” x antall ganger for ønsket antall. Et alternativ vi kunne utforsket videre ville vært å opprette en egen fil for “shoppingCartItem” som tar inn og displayer quantity. Denne kunne vært benyttet istedenfor “productItem” for å display innholdet til handlekurv. Vi kunne deretter implementert to knapper, increment og decrement quantity, samt funksjoner for å håndtere dette. Enten Buttons, IconButtons eller lete videre etter eventuelle andre Compose UI løsninger som kunne vært mer passende til dette.

Alt i alt er vi fornøyd med eget arbeid og sluttresultat med tanke på hvilket utgangspunkt vi har arbeidet ut ifra. All kunnskap vi har tilegnet oss fra dette faget har vært gjennom eget arbeid og fire lærerike streamet

forelesninger fra Bergen. Vi har fått implementert alle elementer vi så for oss til denne løsningen, og har forsøkt å gripe oppgaven ved godt mot tross utfordringer. At vi fikk lagt til et system for rating av produkt er noe vi er spesielt fornøyd med, for å trekke frem noe konkret fra løsningen.