**Rapport - Eksamen PGR208 - Android Programmering**

I denne eksamenen har jeg blitt spurt om å utvide funksjonaliteten til en allerede eksisterende plattform. På grunn av økende bruk av mobil på deres plattform, vil de nå introdusere en mobilapp. Jeg har laget en prototype og tatt i bruk FakeStore API, som etterligner en online butikk. Mitt mål var å forhåpentligvis tilby en sømløs opplevelse for brukeren når de browser, legger til ordre, bestiller og ser på gamle bestillinger.

Jeg startet med å skissere en løs handlingsplan for meg selv. Deretter gjorde jeg klar mappestrukturen, og implementerte de forskjellige dependencies og plugins og lignende som jeg visste jeg ville trenge. Etter det forsøkte jeg å dele koden inn i logiske komponenter basert på deres ansvar. Jeg har brukt Jetpack Compose og Navigation Component for å lage et moderne brukergrensesnitt og navigasjonssystem.

Appen bruker Retrofit for å håndtere nettverksforespørsler og Room for Local Storage. Jeg valgte å ikke opprette flere domenemodeller fordi de ville ha inneholdt nøyaktig det samme i dette tilfellet, som ville skapt unødvendig duplisering av kode. “ShoppingRepository” inneholder metoder for å hente produktene og for å administrere handlekurven, ordrehistorikken og favorittene.

Noen av UI elementene og komponentene jeg har brukt er:

* Compose Funksjoner som for eksempel “@Composable”, Scaffold“, “LazyColumn” og “Row”
* Material Components som “TopAppBar“, “IconButton”, “”Button”, “Icon” og “Text”
* ViewModel-klasser
* Coroutines (viewModelScope.launch) for å håndtere asynkrone oppgaver.
* Navigering
* Database og Room for å håndtere Local Storage

Med mer.

Appen har flere destinasjoner som ShoppingListScreen, ShoppingDetailsScreen, BagListScreen, FavoriteListScreen og OrderHistoryScreen, som er håndtert av “NavController”.

ShoppingListScreen viser en fullstendig liste over alle produktene i nettbutikken. Man kan søke etter produkt og kategori, og trykke på hver av produktene for å se flere detaljer i ShoppingDetailsScreen. Inne i ShoppingDetailsScreen kan man i tillegg til å se detaljer som produktbeskrivelse, pris og rating, både legge til som favoritt, legge til i handlekurven og legge igjen en rating. Ratingen lagres dessverre foreløpig ikke.

Videre kan man navigere til favoritter (FavoriteListScreen), handlekurv (BagListScreen) og ordrehistorikk (OrderHistoryScreen).

I FavoriteListScreen ser man enten en tom liste, eller en liste over produkter brukeren har likt. I BagListScreen ser man enten en tom handlekurv, eller produkter brukeren har lagt til i handlekurven. Man kan fjerne produkter fra handlekurven, eller gjennomføre ordren.

Tilbake på forsiden man navigere seg til OrderHistoryScreen. Her ser man alle tidligere ordrer, samt detaljer om dem, som klokkeslett og dato ordren ble gjort, hvilke produkter det gjelder og antall, og sammenlagt pris.

En betydelig utfordring oppstod under forsøket på å oppdatere antall varer i handleposen. I ettertid ser jeg at det ville vært fordelaktig å dedikere mer tid til grundig testing av denne funksjonen. For å forbedre brukeropplevelsen og sikre riktig funksjonalitet, ønsker jeg ved ytterligere utvikling av appen å legge til funksjonaliteter som gjør det mulig å justere kvantitet på hvert enkelt produkt som er lagt til. Jeg endte med å ta bort koden for denne funksjonen, da den ikke fungerte slik jeg planla, og heller opplevdes forstyrrende.

Å vise detaljer om ordrene så også ut til å bli mer krevende enn først antatt, men det er kun fordi jeg misforsto hva som ble bedt om. Dessverre brukte jeg mye tid på denne misforståelsen. Jeg innså at det som ble bedt om, altså å vise dato, pris og alle produktene i en ordre, var noe jeg allerede hadde implementert fra da jeg startet med Order History-siden. Det lå likevel lærdom i dette også, det er fort gjort å se seg blind på egne fremskritt og egen kode.

Sett vekk fra det, er jeg generelt fornøyd med prosjektet, selv om jeg ser potensial for struktur-forbedringer. Jeg kan bli bedre på å organisere koden innenfor modulene og mellom forskjellige komponenter. Det vil bidra til en mer skalerbar kodebase. Jeg kommer også til å se nærmere på arkitekturmønstre, for å forbedre strukturen ytterligere.

Jeg har lært mye, og overvunnet flere utfordringer når det kom til datahåndtering, UI-sammensetning og ViewModel-arkitektur. Jeg skulle selvfølgelig ønske jeg fikk gjort lagt til flere funksjonaliteter, som å endre kvantiteten av produkter i handlekurven og å lagre rating gjort av brukerne, men det får være et mål. Jeg har identifisert områder der feil kan oppstå, med tanke på for eksempel nettverksforespørsler og databaseoperasjoner. Et mål er å implementere grundig error handling, slik at brukerne får klare tilbakemeldinger i tilfelle feil, som for eksempel uventede databasetilstander, nettverksproblemer og andre potensielle feil.