Laboratorijska vežba 3
Perzistencija – Repository Pattern i Room Database

Opis projekta

Projekat sadrži implementaciju Room baze koja skladišti narudžbine (orders). Sadrži tri ekrana: listu svih objekata u bazi, detaljan pregled informacija o jednom objektu i ekran za dodavanje novog objekta (OrderListScreen, ViewOrderScreen, AddOrderScreen).

Opis postojeće strukture

Implementirani su sledeći ekrani:

1. **OrderListScreen** – prikazuje listu svih objekata u bazi čitajući Flow promenljivu iz OrderViewModel-a.
    - Klik na View u okviru svakog objekta vodi na ekran za detalje o objektu.
    - Klik na Floating Action Button (plus u donjem desnom uglu) vodi na ekran za dodavanje objekta.
2. **ViewOrderScreen** – prikazuje detalje o objektu.
    - Dugme Delete treba da briše objekat iz baze.
3. **AddOrderScreen** – implementira formu za dodavanje objekta.
    - Objekat ima ime, cenu, tip i listu topinga.
    - Polje za unos cene prihvata samo cifre, a polje za izbor tipa objekta je dropdown.
    - Polje za unos topinga funkcioniše tako što se ukuca naziv topinga i na tastaturi uređaja izabere Enter (Done), nakon čega se ispod polja prikazuje Chip Composable sa imenom topinga. Klikom na Chip uklanja se toping.

Implementirani su sledeći ViewModel-i:

1. **OrderViewModel** – komunicira sa repozitorijumom, sadrži Flow promenljivu koja modeluje sve podatke iz baze, i selectedOrder koja sadrži objekat koji je selektovan za detaljan prikaz.
    - Potrebno je da ovaj ViewModel poziva metode repozitorijuma.
2. **EditViewModel** – koristi se za implementaciju funkcionalnosti AddOrderScreen, sadrži vrednosti editable polja sa ekrana.

Room Database komponente:

1. **Order Entity** – sadrži ID, name, price, type (koji uzima jednu od vrednosti enumeracije OrderType) i toppings (string vrednosti razdvojenih zarezom)
2. **OrderDao** – interfejs koji sadrži definicije read metoda (getOrder i getOrders), insert metodu i deleteAll metodu.
3. **OrderRepository** – repository klasa koja prihvata orderDao kao Dependency Injection i implementira metode koje pozivaju DAO metode.

- Metode su suspendable i izvršavaju se na WorkerThread-u.
- Property orders je tipa Flow i dobija se pozivom getOrders() metode, čime uvek dobija ažurne podatke iz baze.

4. **OrderDatabase** – klasa koja modeluje bazu podataka, sadrži callback kojim se vrši inicijalizacija baze početnim podacima.

Pored ovih komponenti, implementirana je i **OrderApplication** klasa, koja služi za prosleđivanje konteksta korutinama. Sadrži reference na bazu podataka i repository.

U okviru **MainActivity** klase, instanciran je OrderViewModel korišćenjem delegata koji poziva **OrderViewModelFactory**, koji prihvata kao parametar referencu na repozitorijum preko objekta aplikacije. Razlog za ovo je Dependency Injection u okviru ovog View Model-a, koji zahteva da se isti instancira pomoću Factory objekta.

ZADATAK:

1. Dovršiti implementaciju **OrderDao**:
   - implementirati metodu delete, koja je suspendable i vrši brisanje prosleđenog Order objekta.
2. Dovršiti implementaciju **OrderRepository**:
   - implementirati metodu delete, koja radi na radnoj niti i poziva metodu delete iz OrderDao, prosleđujući joj Order objekat za brisanje.
3. Dovršiti implementaciju **OrderViewModel**-a:
   - dovršiti metodu addOrder, koja treba da pozove insert metodu repository objekta
   - dovršiti metodu deleteOrder, koja treba da pozove delete metodu repository objekta