



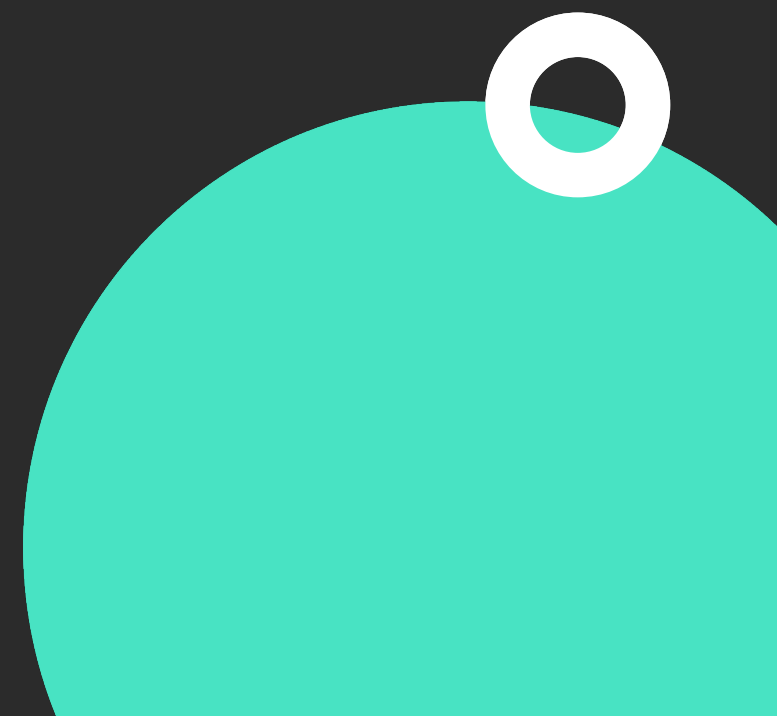
Repository & UI Layer



Belal Khan

 @probelalkhan

 in/probelalkhan



Section Overview

- Add repository layer. Learn to create repository classes to facilitate various CRUD operations defined in their respective DAO classes
- Learn to add ViewModels which will act as a bridge for communication between the UI layer and the repository layer
- Learn to use Android Studio's Database Inspector tool to inspect app database



Code Challenge Recap

- Creating Entities and DAOs.



Repository Layer

- BusinessRepository
- CustomerRepository
- InvoiceRepository
- TaxRepository



BusinessRepository

- Repository will have the functions that will be consumed by ViewModel.

```
interface BusinessRepository {  
    suspend fun addUpdateBusiness(business: Business)  
    fun getBusinesses(): Flow<List<Business>>  
    suspend fun deleteMyBusiness(id: Int?)  
}
```



BusinessRepository

- Repository will have the functions that will be consumed by ViewModel.

```
class BusinessRepositoryImpl(
    private val dao: BusinessDao
) : BusinessRepository {

    override suspend fun addUpdateBusiness(business: Business) {
        dao.addUpdateBusiness(business)
    }

    override fun getBusinesses(): Flow<List<Business>> {
        return dao.getBusinesses()
    }

    override suspend fun deleteMyBusiness(id: Int?) {
        dao.deleteBusiness(
            Business("", "", "", "").also {
                it.id = id
            }
        )
    }
}
```



Code Challenge

- CustomerRepository
- TaxRepository



Understanding the UI Layer

- HomeNavHost
 - businessNav
 - MyBusinesses()
 - ManageMyBusinesses()
 - customersNav
 - Customers()
 - ManageCustomer()
 - taxNav
 - Taxes()
 - ManageTaxes()
 - **invoiceNav**



Understanding the UI Layer

- InvoiceNav
 - PickBusinessScreen()
 - PickCustomerScreen()
 - PickTaxScreen()
 - AddInvoiceItem()



BaseViewModel

- A base viewmodel class to define all other viewmodels.



MyBusinessViewModel

- ViewModel for MyBusiness Screen.



Code Challenge

- Finish the following Screens.
 - Customers
 - Taxes



Summary

- Understood Repository Layer of our Application
- Built a Repository for every Entity of the project
- Understood the UI Layer of our Project
- Learned building ViewModel
- Stitched all components together and saw the application running





Defining Relationship Between Objects

Up Next

