Win-Scan Android App

Team: Rohan Chhokra 2016080
Ankur Sharma 2016225
Vaibhav Goel 2016111
Manan Gupta 2017372
Guide : Saransh Gupta | Org: Embereon

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Project Goals

- Implement- **Payments** feature and **Awards** Feature

- Payment
  - This feature enables the user to use multiple UPI applications available on their phone
  - Payments can be done through QR code scans or filling in the fields such as phone number, UPI ID, bank account number etc.
  - Functionality similar to an existing application called - ScanPay

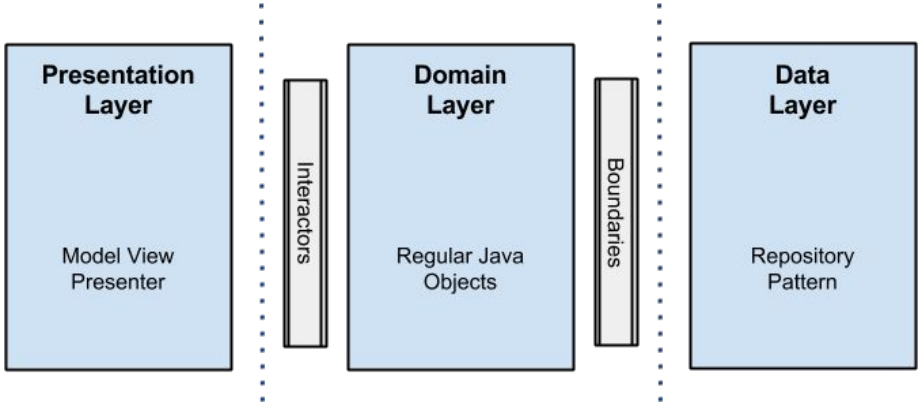
- Awards
  - Implement a reward feature where different applications like Zomato,Swiggy,MakeMyTrip can be endorsed through rewards
  - Implement a ‘claim’ mechanism
  - Once claimed, user can use this award on the respective application for which the award was endorsed
  - Usage of award will be through an in-app view of the application where responsive web design of applications will be leverage

Architecture

Our project will be based on Clean Architecture with a structural design pattern known as Model-View-ViewModel (MVVM). We will be strictly following this architecture to support Test Driven Development.

4 Layers of Architecture

- **UI Layer**- Responsible for displaying visual elements and controls on the screen.
- **Presentation Layer**- This layer manages the state of the application and implements UI logic containing handling of user inputs, etc.
- **Domain Layer**- This layer constitutes a set of all use cases of the application and therefore contains models and business rules of the application. Example use-cases: money transfer request, get available apps for money transfer, reward users with cashback, etc
- **Data Layer**- This layer provides abstract definitions for accessing local and remote data sources.



Tools and Technologies

- **RxAndroid**- We will be using reactive programming to achieve the architectural approach as shown in Fig. 1.
- **Dagger 2.0**- It is a fast and compile-time dependency injector for Android. It will help us keep the code clean. Moreover, It is a key concept to get testable code. Using Dagger, it's easy to replace an object with a mock, to change and verify the behavior of a system.
- **Live Data**- An advanced observable data holder class, works best with android as it takes care of activity lifecycle.

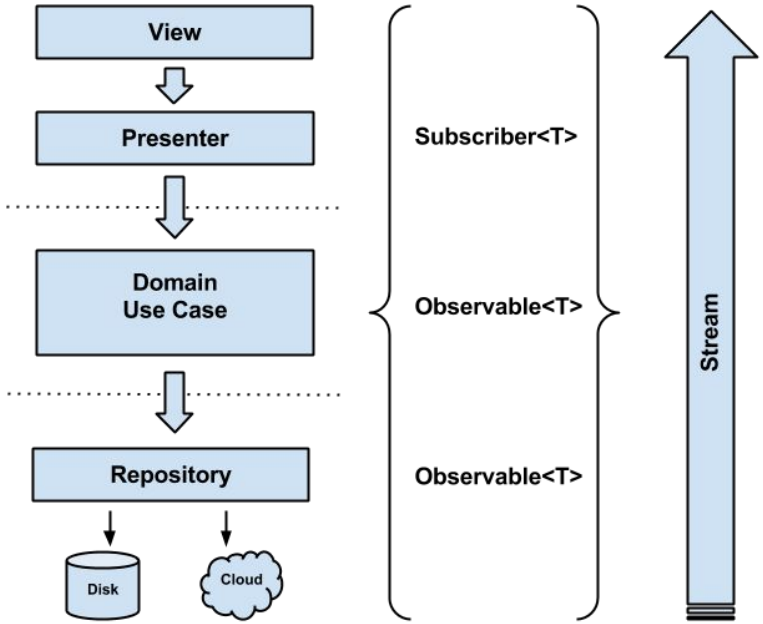
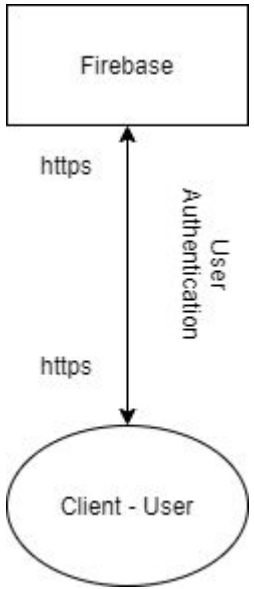


Fig. 1

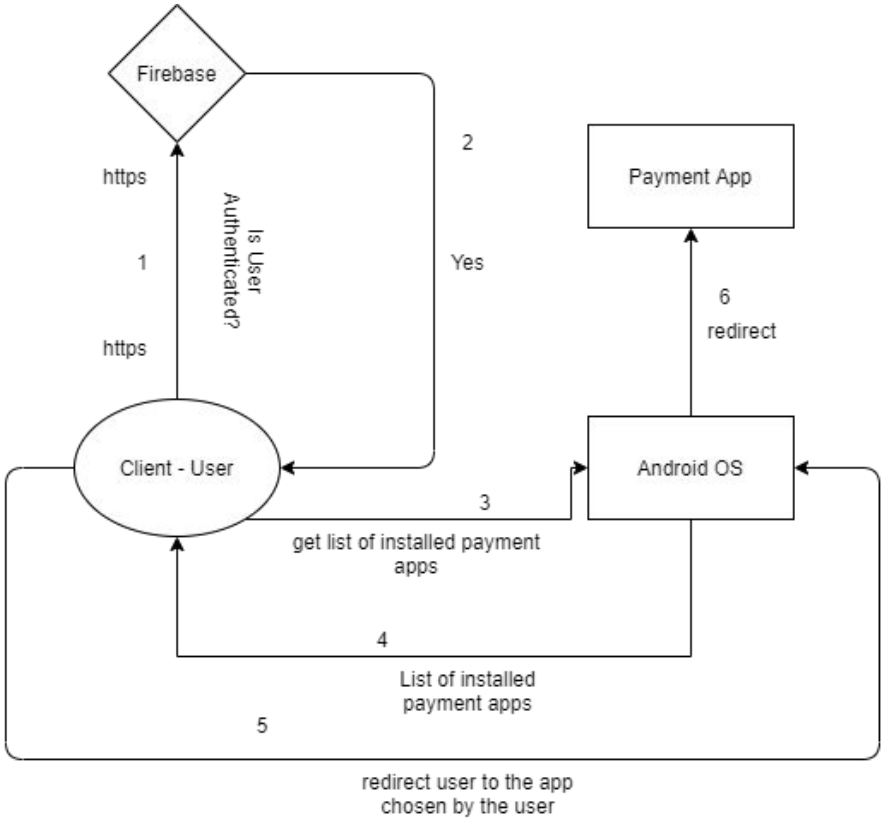
Architectural Attributes

- Availability and Reliability- Computation extensive tasks will run in the background thread with priority queue to make user experience flawless.
- Performance- Depends on device configuration. Architecture will optimally use device's resources.
- Configurability - Eg.: switching from dark theme to light theme.
- Portability - Same architecture can be used in other platforms: iOS. Code needs to be written separately as iOS can not compile JAVA to machine level.
- Localization/internationalization- Android platform supports multi-languages with ltr (left-to-right) as well as rtl (right-to-left) feature.
- Extensibility- Separate domain layer takes care of all use-cases. Easy to extend.
- Authentication & Authorization- Firebase SDK takes care of these attributes.
- Scalability

Component and Connector View



User Authentication



Payment Initiation Use-case