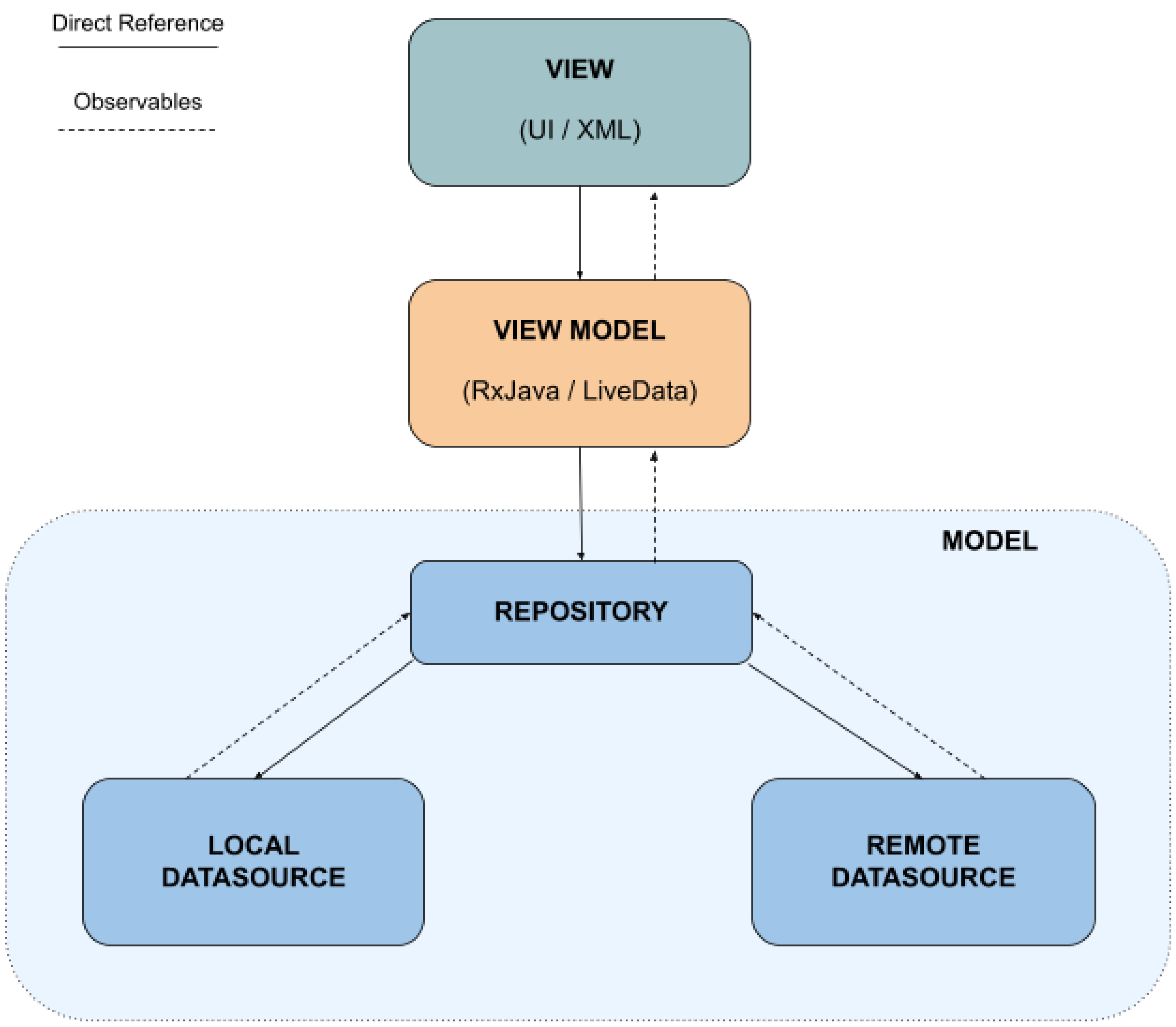
Software Architecture

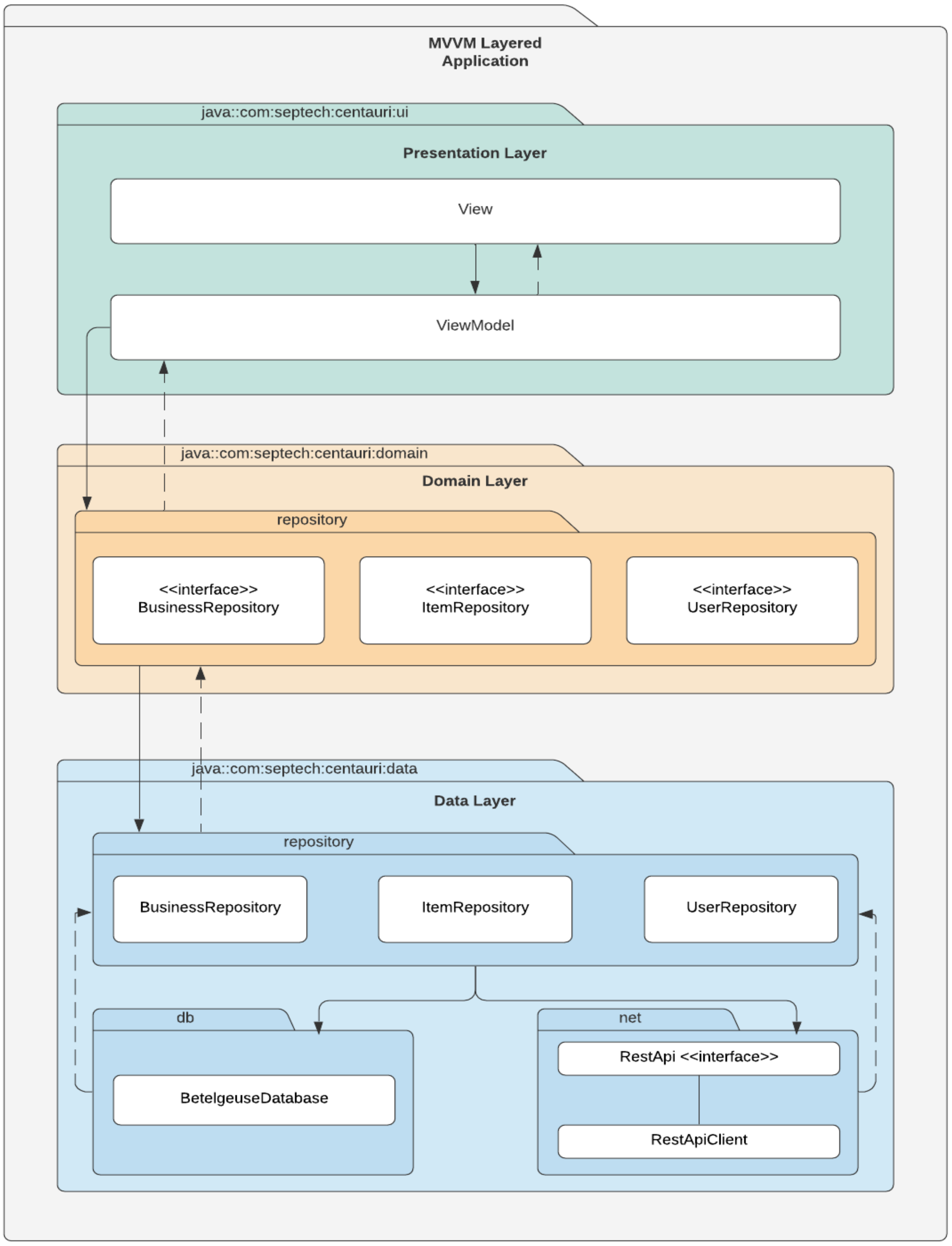
Centauri

Diagrams

MVVM Overview Diagram



Centauri UML Package Diagram



Description

The architecture chosen for the Centauri application was the Model-View-ViewModel (MVVM) architecture. The following describes the components:

Model
This is the data access layer, which contains the components that handle the data for our application.

View
This contains what the user will see on their mobile device: structure, layout, and appearance. It receives user interaction and forwards handling to the View Model.

View Model
This is the component that handles the state of the data in the model. The View Model is not bothered by UI components, makes calls to other components to load the data, and forward srequests to modify data.

Our reasoning for choosing this architecture over others is due to the fact that we are creating an Android app. Unlike desktop applications, which typically have a single entry point and runs as a monolithic process, Android apps tend to have many components (such as activities, services, broadcast receivers, content providers) and should be prepared to handle app-hopping on a mobile device.

Due to this environment, it’s advised that we don’t store app data or states in our components. MVVM is ideal because, in order to avoid storing app data and states, it implements the design principle separation of concerns.

Another design principle utilized in MVVM is driving UI from a model. Models are independent of View and are therefore unaffected by the app’s lifecycle.