Application architecture

Architecture was a main concern when the application was designed, as it represents the foundation for an app that is well maintainable, scalable and reliable. The aspects taken into account when deciding on an architectural pattern where the separation of concerns, the code reusability, testability and independence. Even though the term “good architecture” may sound slightly abstract, having these concerns in mind, the MVVM architecture was chosen for this project. It maintains a clear separation between application logic and the UI, therefore addressing numerous development issues and making the application easier to test and maintain. The emphasis is put on dividing the responsibilities, so the UI components are separated from the business logic and the business logic is separated from the data access.

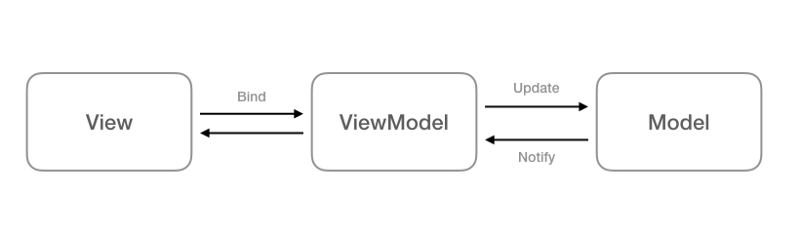
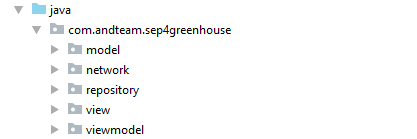
There are three core components in the MVVM patter: the Model, the View and the ViewModel, each serving a distinct purpose. The figure below illustrates the relation between the components:

Figure Data flow in MVVM

The ViewModel isolates the View from the Model, so it receives its data from the Model and exposes the data and command objects that the View requests. The Model is unaware of the ViewModel and the ViewModel is unaware of the View.

In the figure below, how the MVVM pattern was related to the application can be observed:



**The Model** represents the actual data that is dealt with in the application but it does not hold behaviors or services that manipulate the information. It does not have anything to do with the UI or with fetching any data. Business logic is kept separate from the Model as it belongs to other classes that act on the model.

**The View** is responsible for the structure and appearance of what the user sees on the screen. Each view is defined in an XML file containing code without any business logic. The View retrieves its data from the ViewModel thorough the use of binding.

**The ViewModel** is the component that connects the View to the Model by accessing the methods and properties of the Model that are then made available to the View.