SCRUMTIOUS

Sprint 3

Swetha Maramganty
Muhammad Waleed
Shabaz Badshah
Usaim Bhayat
Nensi Deliana

Contents

CRC Cards

Class name: User	
Responsibilities: • Has an identifier(UUID) • Knows Email, password • Know the projects they created • Know the projects they are in • Create projects • Delete projects they created • Add other users to projects you created	**Collaborators:** Project

Class name: Project	
Responsibilities: • Has a PID • Knows the owners email • Knows the owners UUID • Knows the title and description	**Collaborators:** User

Class name: User Invites	
Responsibilities: • Knows the PID of the project • Knows the UUID of the person that is the inviter • Knows the UUID of the person that is invited • Knows the email of the person that is the inviter • Knows the email of the person that is the inviter	**Collaborators:** Project User

System Architecture Description

- OS: Android Lollipop
- Database: Firebase
- Network connectivity is required for now to use the app

System Decomposition

Model

Model is used by the Presenter. All the plain old java files (POJOs) are in the Model and they represent parts of our database schema in Firebase. Our schema is stored as JSON objects, somewhat like a cloud-hosted JSON tree. Since there are no tables or records, when you add an object, it becomes a node in the tree. We have keys as identifiers, for example, projects have PID's and users have UUID's in the database.

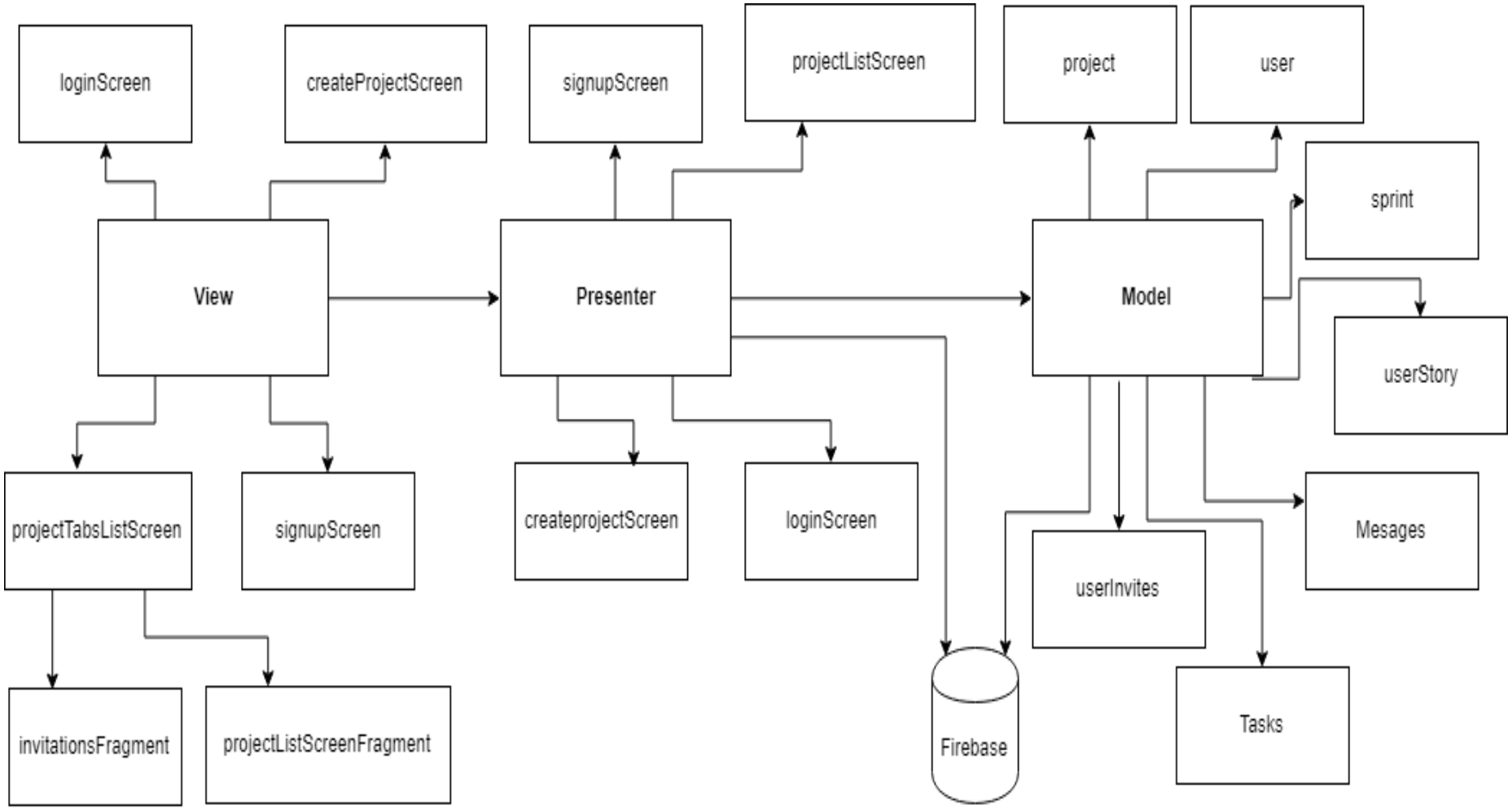
View

The view, usually implemented by the activity/fragment will contain a reference to the presenter. The view calls a method every time there is a user interface action, for example, clicking a button. It consists of an XML files and activities.

Presenter

Presenter is used for the backend operations in Firebase by using the Model. The presenter is responsible for connecting the database and view. It gets the data from the database and presents and formats it to the view for the user.

UML Diagram for MVP



The architecture did not change overall because everything worked as intended, everything was organized in a specific way that made it easy to bug fix and add new classes. We made changes to the model because we added properties to the database that require the model(POJO) to be updated. For this sprint, we added messages and tasks to the model.