Android app that provides club updates and tournament information and stats for the Annual Cupertino Judo Tournament.

GitHub Username: fhirata

Cupertino Judo Android App

Android app that provides club updates and tournament information and stats for the Annual Cupertino Judo Tournaments.

This Android application will provide general club information:

- General club and contact information (map of club, practice hours)
- Tournament information

For tournament information, we will provide app Notifications of up-to-date status of pooling information on the day of the tournament. We will also provide the brackets of judokas fighting for current and previous tournaments.

This application gives users the peace of mind to know exactly when the polling of players start and when they complete on the day of the tournament.

Intended User

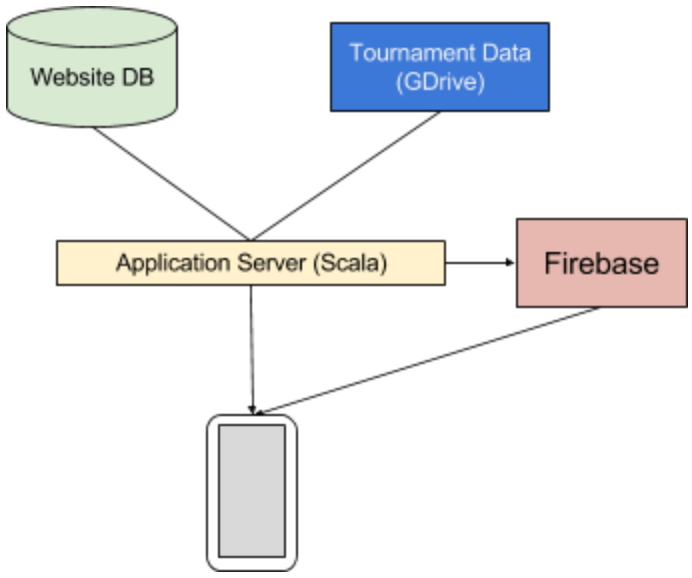
This app is intended for families and club members of the Cupertino Judo Club as well as general people participating in the tournament that want up-to-date information on the tournament day. Throughout the event, the app will provide app notifications for different categories.

Features

Below are the main features of the app:

- Club tab: Provides general club information (practice day/time, club location, instructor bio)
- Tournament tab: Stats of previous tournaments (interesting graph trivia like number of participants for each category)
- Tournament tab: Provides tournament information (Registration opens/closes, Pooling has finished, and when pools are posted on the website)
    - Categories: Junior Male/Female, Senior Male/Female
    - Search for Player
    - View pooling brackets and player information
- Notifications tab: Shows previous notifications from the last 7 days. This is important for people that just downloaded the app (e.g. on the day of the tournament) to see which notifications have taken place.

High-level architecture



Application Server

Application server provides a federation layer and fetches data from multiple sources. JSON feeds should contain:

- Previous year's Player information (/2016/tournament/pools.json)
- Previous year's Tournament information (/2016/tournament/general.json)
- Current year's tournament information (/current/tournament/general.json)
- Current year's General Tournament information (/current/tournament/general.json)

Application server will also hold the information for pushing notifications to Firebase. These will be Topic base.

Website DB

Holds information that the Android app will display that is common content between the Android application and the Judo website. These are information like: Club location, Practice times, any club images, assets, Tournament registration information. This information is mostly available today from http://cupertinojudoclub.org but will, most likely, require some refactor to expose data in shared format between the mobile application.

Tournament Data

Tournament data contains the spreadsheets that are used for calculating the tournament pools and final brackets. Once the spreadsheets are determined per category, I shall publish them via the GDrive Publish feature and expose the CSV feeds in the format below:

```
curl
"https://docs.google.com/spreadsheets/u/1/d/1G4ihpdpi-o1VQuBs5IJh-UU3iCt4divzVkVSWujFXPk/pub?gid=1458298243&single=true&output=csv"
```

The Application Server will be updated to fetch this data, parse them and serve them as JSON data to the mobile application.

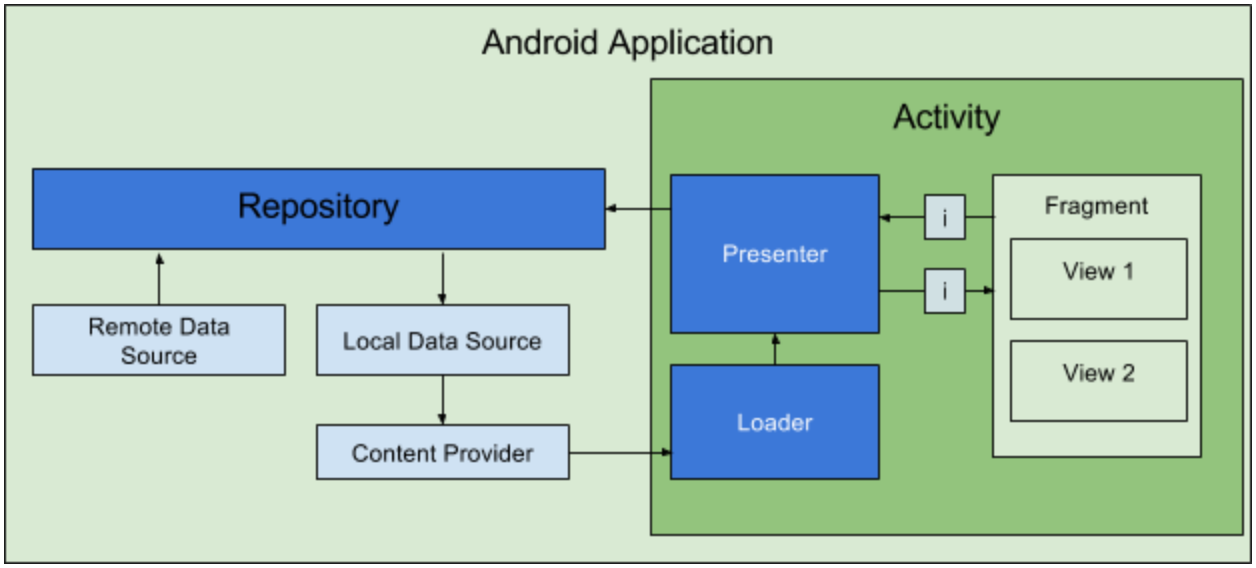
Firebase

I plan to have app notifications for users that download the app for up-to-date notifications on the tournament day. Below are the types of notifications that will go out to app users. To make it simpler on the server implementation, we will rely on topic base notifications only with a single topic for the entire application.

Notification name	When to notify	Notification Action
“Cupertino Judo Date set!”	When the public announcement of the tournament is made available	Open the app on the tournament Tab showing tournament schedule
“Players Registration is now Open!”	One day before the tournament informing users that registration has started	Open the app on the tournament Tab showing tournament schedule
“Registration for {Junior Males|Females, Intermediates Males|Females, Seniors Males|Females} is now closed. Pooling has started.”	On the day of the tournament as different categories start pooling	Open the app on the tournament Tab showing tournament schedule
Pooling is now available for {Juniors Males|Females, Intermediates Males|Females, Seniors Males|Females}	On the day of the tournament as different poolings become available.	Open the app on the pooling fragment.

Client Architecture

The project shall leverage the MVP content-provider[1] pattern for managing access to the local storage and remote data sources. In the diagram below, Presenters are responsible for interacting and issuing commands to the View (mostly Fragments) and will be issue calls to the repository to get data.



Activities/Fragments

I plan to have just two Activities for this app: the launcher activity that will have the dashboard with various tabs:

Activity 1: Dashboard

- Tournament
- Club
- Notifications

Activity 2: Tournament Pools

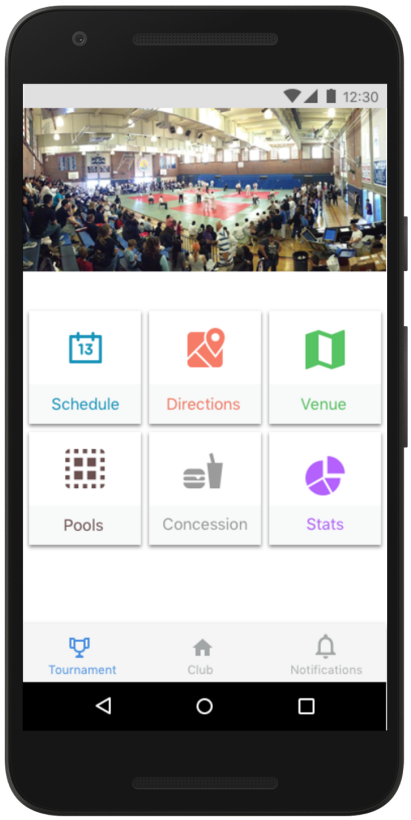
- Entry point from Tournament app notifications
- Displays pools for the year (previous or current year)

Settings Page

- Turn on/off app notifications
- View app version

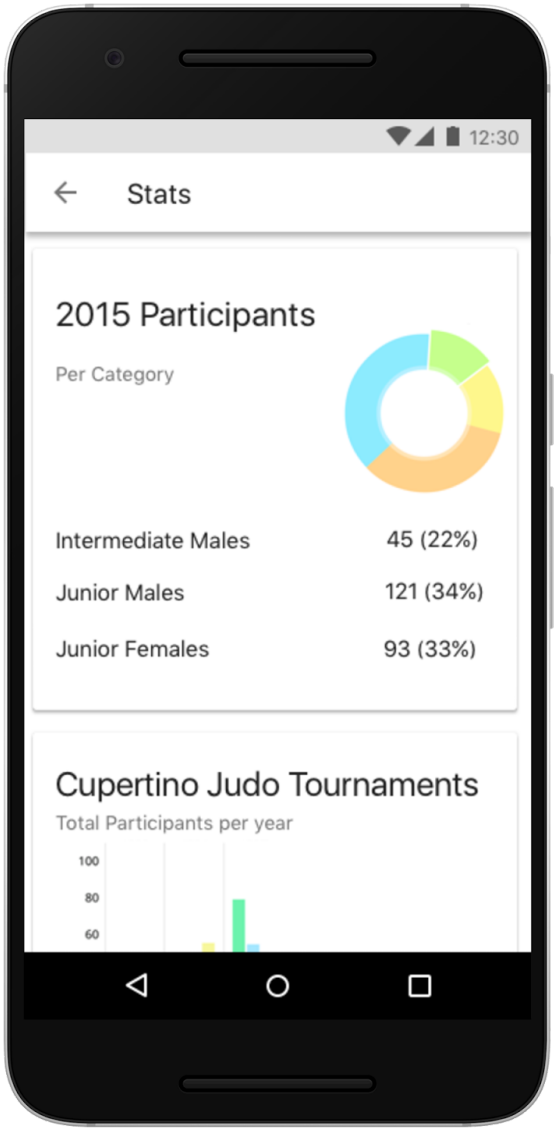
User Interface Mocks

Screen 1 - Tournament Tab



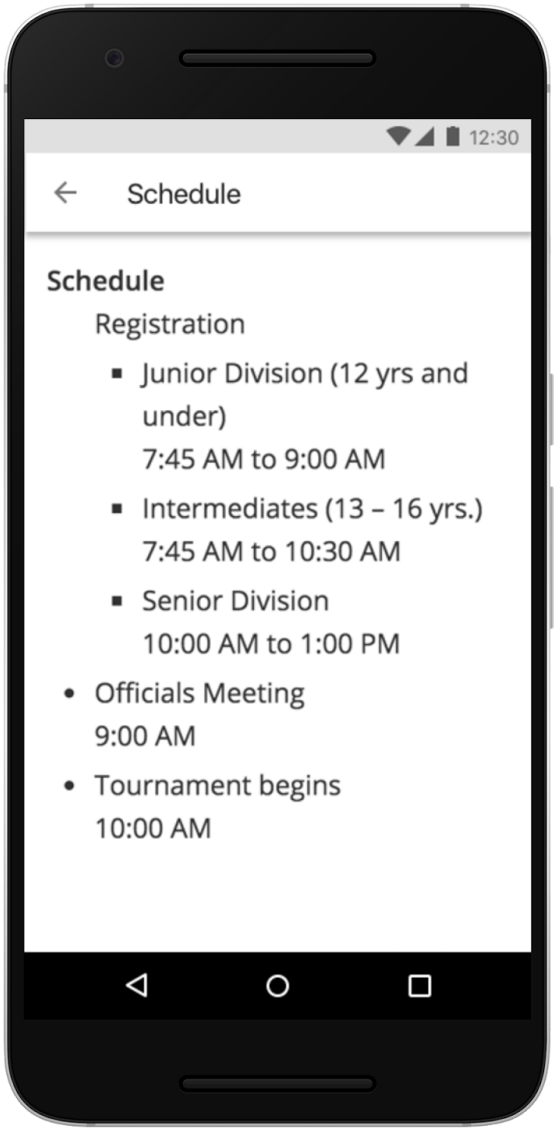
Tournament Tab that contains tiles for sub-views. I have not mocked every single tile for this document. Only the ones that I believe will be very unique and should be explained further.

Screen 2 - Tournament Tab -> Stats Sub-View Example



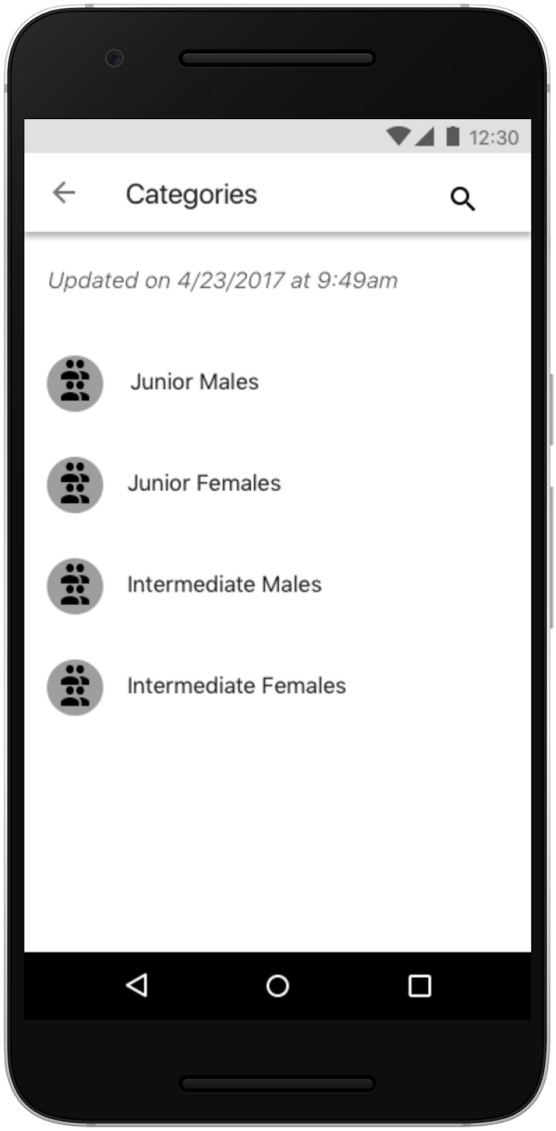
Cards for showing previous tournaments stats. This is something that people often ask, “How many participants did we get last year?” Using the cards layout, I plan to pull and render the stats using the MPAndroidChart library.

Screen 3 - Tournament Tab -> Schedule Sub-view

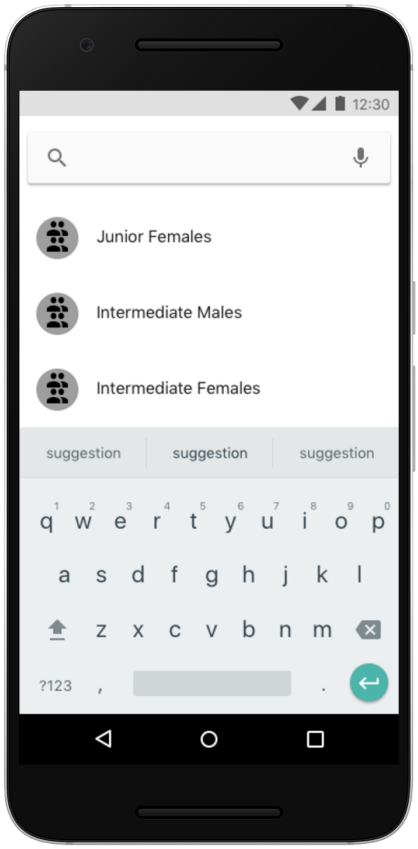


For simple views like the Schedule, it may make sense to have a Webview for rendering the same content from the Website. I will experiment with that to see if there are any issues.

Screen 4 - Tournament Tab -> Pools Sub-view

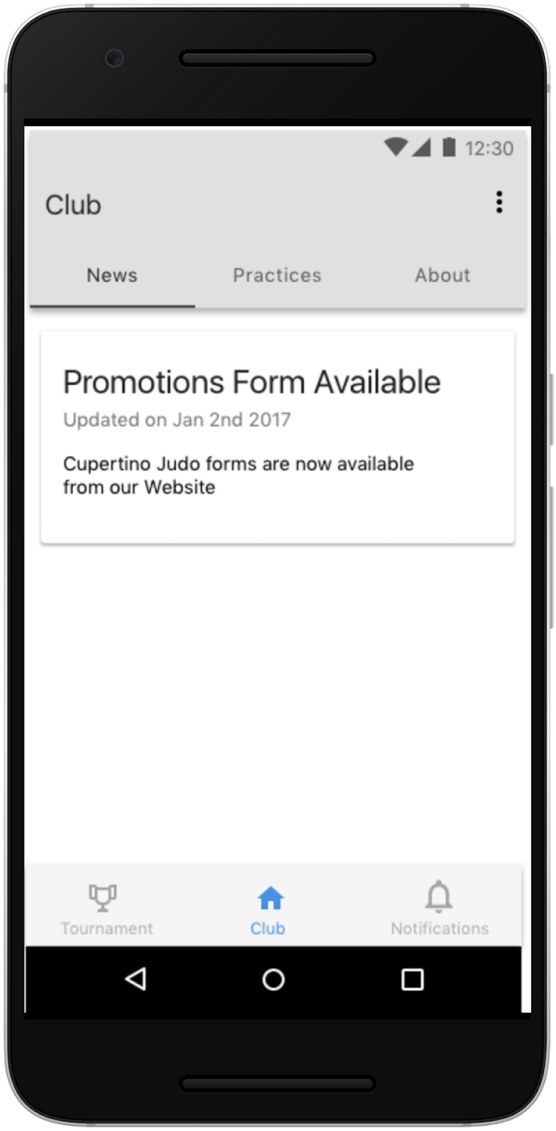


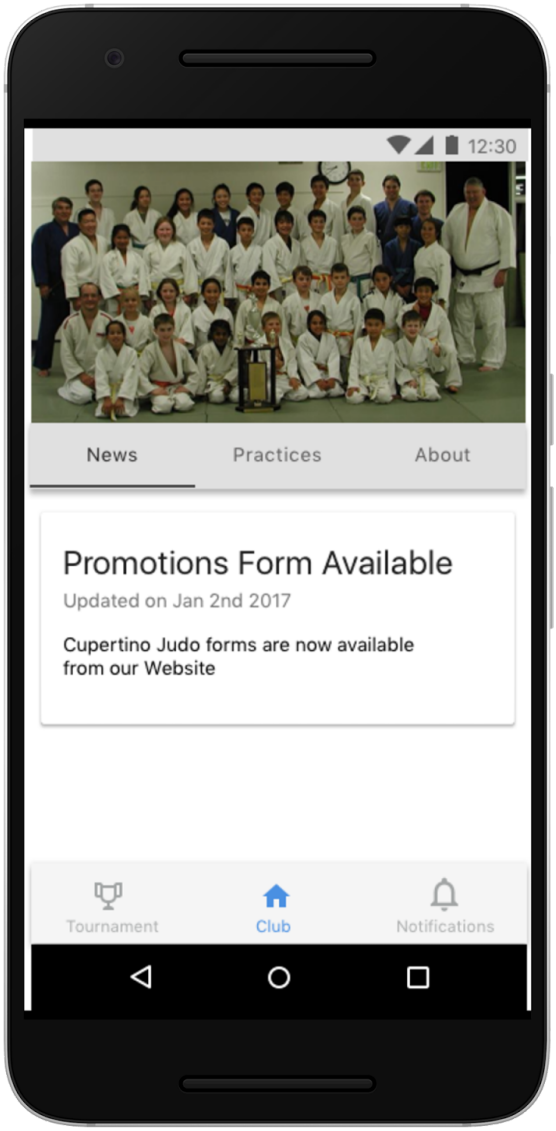
Pulling the tournament data from current and previous years, I plan to render them as simple lists, for different categories and further drilling of each category and pool. This will be different SQLIte queries for each sub-category list.



Search feature will allow for quick look up of player names and which pool he or she belongs to.

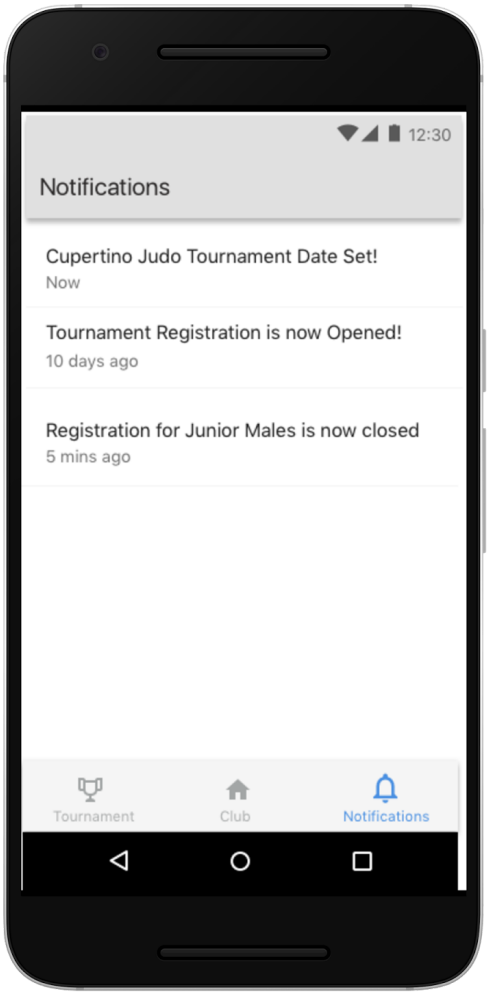
Screen 5 - Club Tab - Tab ViewPager for General club information


12:30
Club
News
Practices
About
Promotions Form Available
Updated on Jan 2nd 2017
Cupertino Judo forms are now available from our Website
Tournament
Club
Notifications



For simple general Website content, I will explore Webview to display the information. Otherwise, I will convert the data parsing the JSON and build individual cards for each section of the tabs. The plan is to have only News, Practices, and About information.

Screen 6 - Notification Tab - Sub-view



Notification actions will either link to the News tab of the Club bottom navigation tab, or to the Tournament section for the poolings.

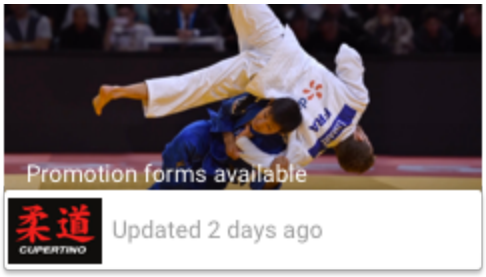
Screen 7 - Settings Page (No Mock available)

The Settings page will contains general version information and preference sections for disabling notifications for the app.

Screen 8 - Widget App

A widget will be available that will display the latest news from the News tab. Because the club does not provide a lot of updates, it doesn't make sense to have more than one. I will use a

SyncAdapter to make sure the latest news is updated on the widget app, fetching data from the content provider.



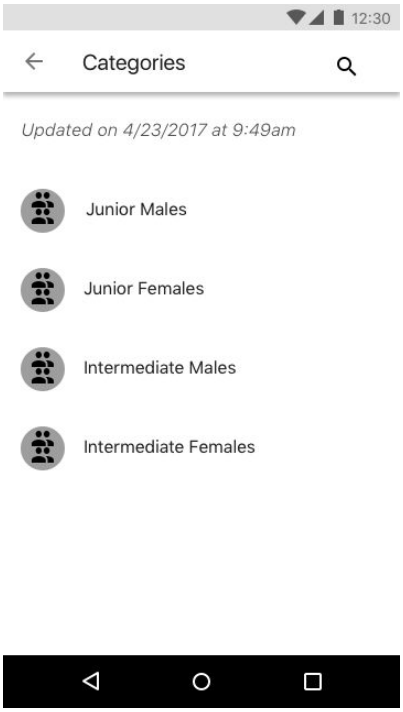
Key Considerations

How will your app handle data persistence?

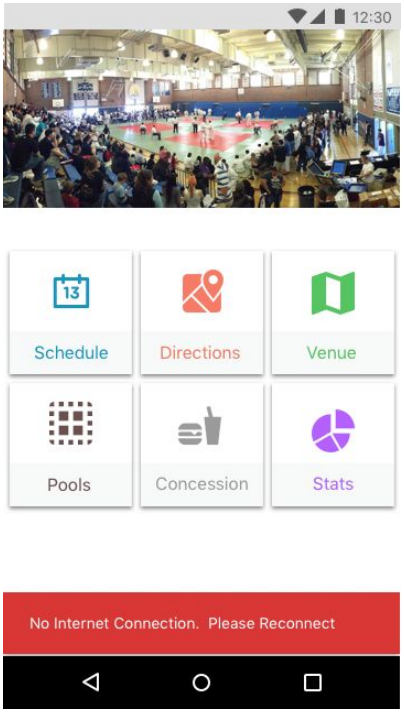
For data persistence, I shall use a Content Provider (SQLite) which will be populated once the initial data is fetched from network. Data refresh shall always happen when the local storate is queried and subsequent changes shall update the UI thread.

Describe any corner cases in the UX.

Data can be stale on the client if they did not sync with the server after a long period of time and their device was not connected to the internet. To handle this edge case, I can provide a “Last Updated” string displaying on critical screen to make sure the last update time is shown.



No internet connection shall be displayed every time users have the app opened and they attempt to fetch data from the network. I'm planning to show that only once per app load so that it is not annoying to users that want to browse the local data offline.



Describe any libraries you'll be using and share your reasoning for including them.

Glide: for handling image loading and caching
MPAndroidChart: for rendering pie charts
Realm or **SQLite**: for data persistence on device
Dagger2: for Dependency Injection
Retrofit2: for HTTP client layer

Describe how you will implement Google Play Services.

Firebase: For topic based notifications on the Judo Tournament event day
Firebase Analytics: for capturing basic client-side instrumentation
Firebase Crash: for capturing crash information

Next Steps: Required Tasks

Task 1: Project setup - 1 day

- Setup Git
- Setup gradle files with initial Android project and specify minimum version
- Initial Unit/Functional test running
- Setup Travis CI

Task 2: Implement UI for Each Activity and Fragment - 3 days

- Build Dashboard View with different bottom navigation tabs
- Build Club fragment
- Build Settings fragment
- Build Notifications fragment
- Build Tournament fragment

Task 3: Hook up application flow - 1 day

- Hook up the app with all the fragments to make sure back buttons work properly

Task 4: Implement Repository Layer - 3 days

- Setup Mock end-point in Git
- Build Network layer pulling JSON mock and implementing Retrofit
- Setup client-side Models for JSON parsed data
- Setup Content Provider for Local storage layer

No caching layer will be done for this initial version of the project.

Task 5: Implement Application Server Layer - 3 days

- Setup Git project for Judo club in Scala
- Setup Heroku end-point for serving JSON
- Setup production/development end-points
- Return mocks end-points through application layer
- Setup Travis CI

Website will continue to be served from existing http://cupertinojudoclub.org domain and will be migrated at a later stage. Also, proper club API CNAMEs will not be setup for this project, but I am planning to have them served from https://api.cupertinojudoclub.org/

I think switching CNAMEs might be time consuming at this time and so I will not bother for this project to have them working. It will require a full website migration to another hosting service, which is outside the scope of this project.

Task 6: UI Polishing - No more than 3 days

- Setup No Network Snackbar
- “Error occurred, Please try again” Snackbar in case the network returns an unknown error
- Transition animations between Activities and fragments
- To keep this project small, I will lock the number of polish days to 3 :)