How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: lgiulian

OpenGotha’n’Droid

Description

OpenGotha’n’Droid is an Android application aiming to manage Go Tournaments and is based on open source OpenGotha project. It will solve the following problems:

- The tournament can be fully managed on the phone/tablet, no need for a desktop PC or laptop
- Results will be immediately available for all subscribers
- Simplifying the process of player registration in tournament by letting participants to sign up for themselves
- Will make it simple and effective to inform all participants about last-minute changes by sending them notifications

Intended User

Users will be both Go tournament organisers and players joining tournaments.

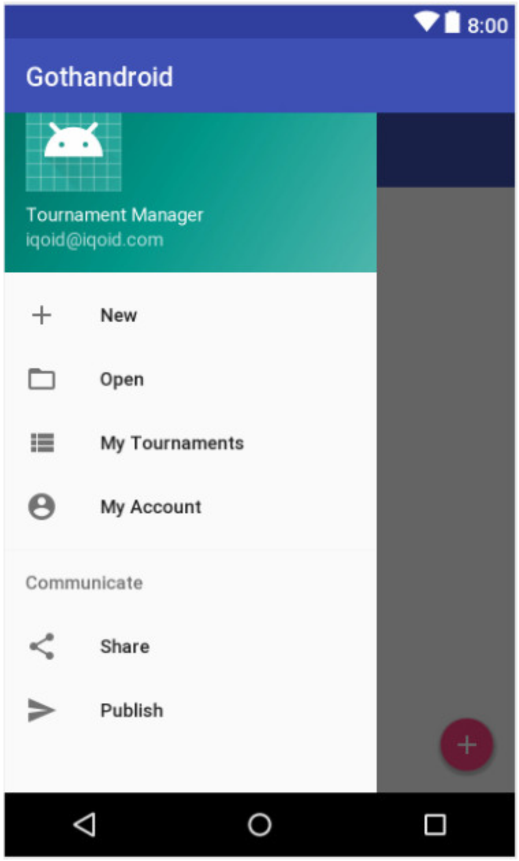
Features

- Create, save, export and import tournaments
- Download and manage players worldwide (from EGF, FFG, AGA)
- Helps organiser to easily enroll participants
- Enable the participants to join the tournaments
- Generate rounds and display results
- Provide different type of tournaments: swiss, round-robin, macmahon

- The newly defined tournaments can be announced to all users having the app installed (also messages related to a certain tournament can be sent to all registered participants)
- The tournament organizer can send private messages to one or more participants
- Publish results on internet
- Taking pictures of a Go player and immediately update the official european federation site

User Interface Mocks

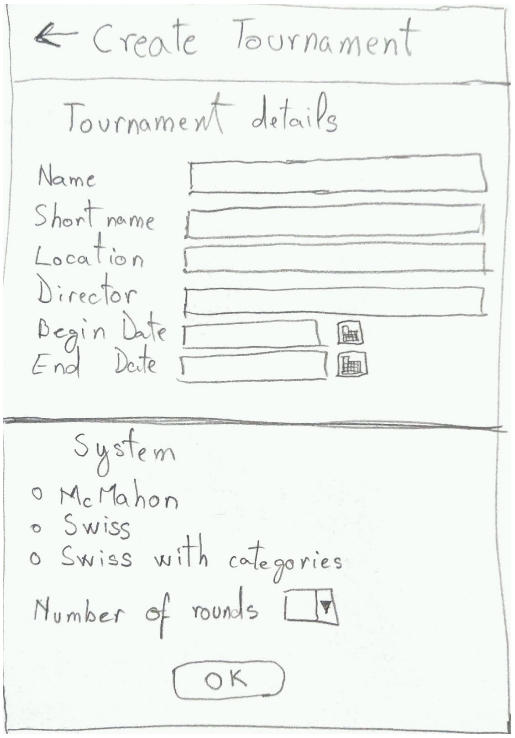
Home screen



Home Screen implements a navigation drawer with the most important actions available in this app:

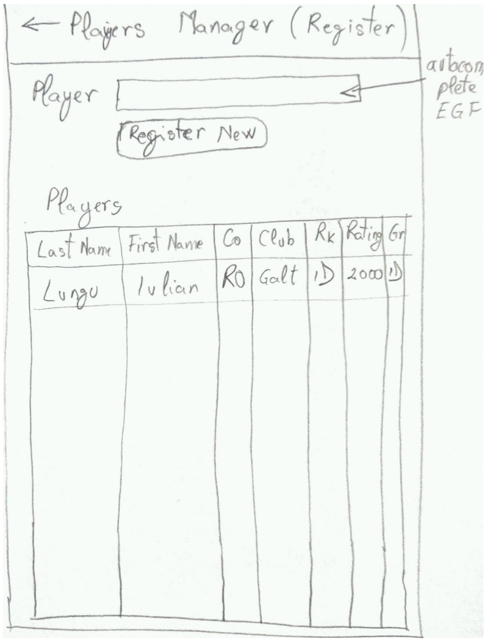
- create new tournament
- open previously saved tournaments
- display personal tournaments
- edit user account
- share tournament via email
- publish tournament on the web

Create Tournament



This screen requires some mandatory general informations about the tournament: name, location, begin/end date.
After that, the user chooses the pairing algorithm used and the number of rounds.
Fields are text, begin / end date can be filled in manually or by accessing a DatePicker.
Choosing the pairing system is done by selecting the desired radio button and "Number of rounds" is a drop-down list with values between 2-20, type integer.
Pressing the "OK" button saves the tournament locally and moves to the next stage of enrolling participants (Go players).
.

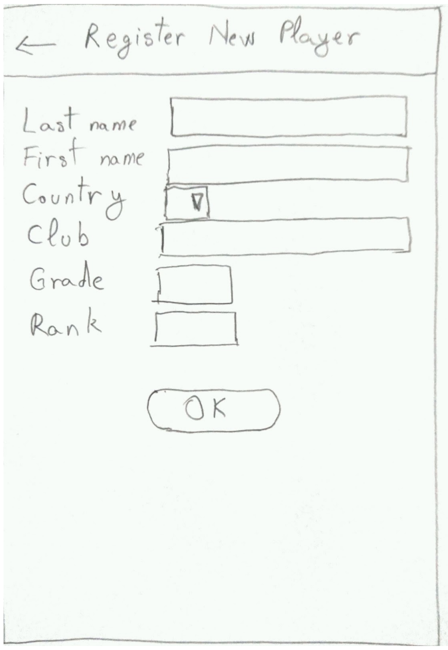
Players Manager



Last Name	First Name	Co	Club	Rk	Rating	Gr
Lungu	Iulian	RO	Galt	1D	2000	1D

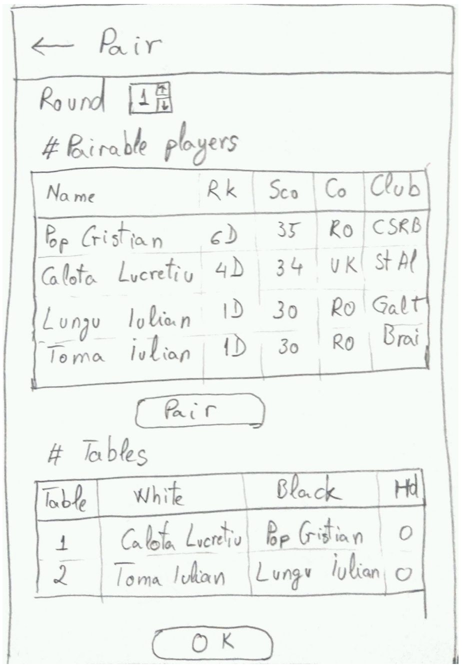
To add players to the tournament this screen allows searching them in the EGD, AGA or FFG databases. To register a new player, press "Register New" button.
In the list at the bottom of the screen are displayed the registered players in the tournament.

Register new player



For new players, fill in the data in this screen. All fields are required. Country and Club contains distinct values already existing in the registered players database. Country and Club fields allow you to enter new values beyond the predefined ones.

Pair



Pairable players

Name	Rk	Sco	Co	Club
Pop Cristian	6D	35	Ro	CSRB
Calota Lucretiu	4D	34	UK	St Al
Lungu Iulian	1D	30	RO	Galt
Toma iulian	1D	30	RO	Brai

Tables

Table	White	Black	Hd
1	Calota Lucretiu	Pop Gristian	0
2	Toma Iulian	Lungu iulian	0

In the Pair screen is happening the pairing for all the rounds in tournament. Here are all players registered in tournament. Players already paired are in the "Tables" list. Unpaired players are listed in the "Pairable players" table. Here you can pair either manually by selecting 2 players from the list and clicking the "Pair" button, or leaving the pairing algorithm to find the best combination. The Pairing process will move the players from the "Pairable players" list to the "Tables" list.

By swiping a row in "Tables" list, players paired on that row are deleted from the list and moved back to "Pairable players".

Results

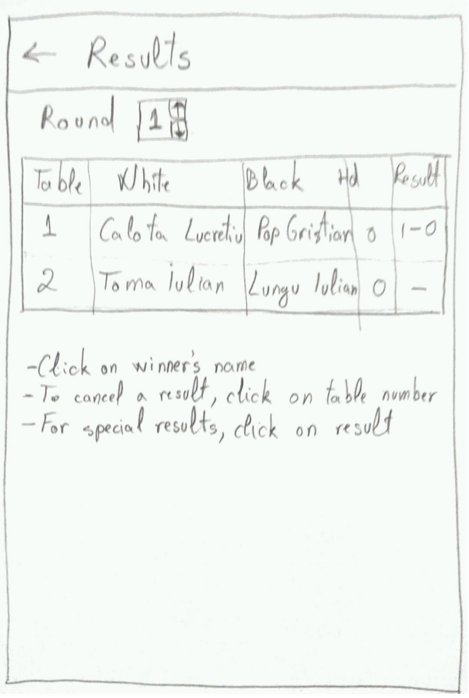
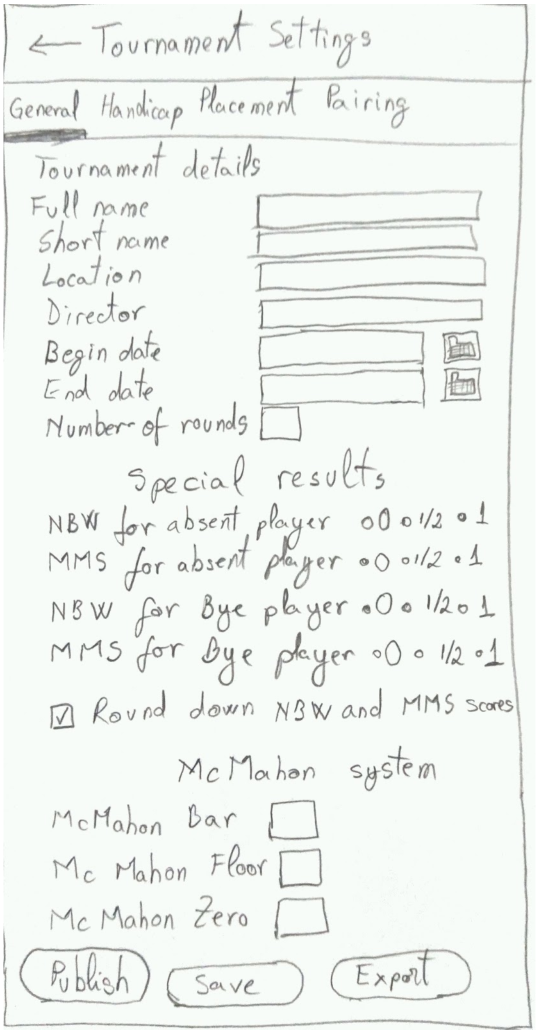


Table	White	Black	Hd	Result
1	Calota Lucretiu	Pop Gristian	0	1-0
2	Toma Iulian	Lungu Iulian	0	–

-Click on winner's name
- To cancel a result, click on table number
- For special results, click on result

To enter the results of the current round this screen provides an easy way of doing it by clicking in the table on the player that won the game. The winner will be colored red and in blue the other.

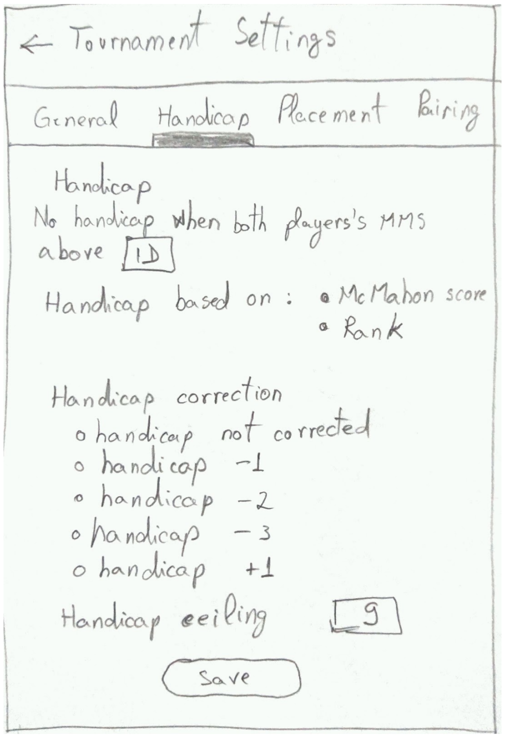
Tournament Settings - General



In this screen there will be 4 tabs that group different tournament settings.
The "General" tab defines the main tournament information: name, location, director, start/end dates, number of rounds (maximum 20) and some other basic tournament settings.

The "Special Results" section is to set the default result for Absent / Bye players.
The "Save" button will be disabled if no changes are made.
"Publish" will publish the current status of the tournament on the web.
"Export" will save the tournament to local storage.

Tournament Settings - Handicap



Handicap

No handicap when both players's MMS above 1D

Handicap based on : ● McMahon score ● Rank

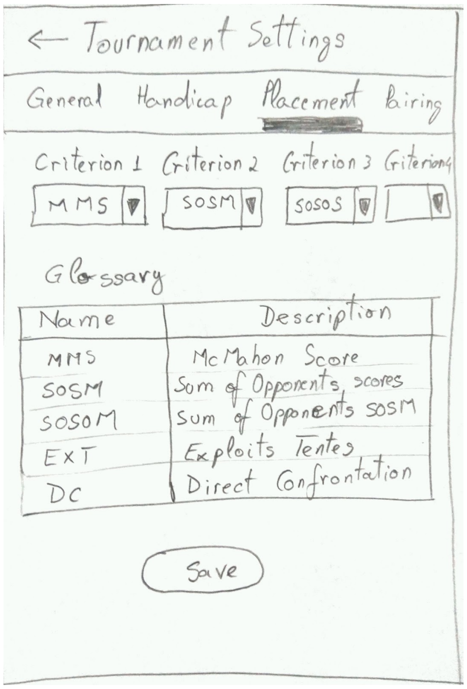
Handicap correction

○ handicap not corrected
○ handicap -1
○ handicap -2
○ handicap -3
○ handicap +1

Handicap ceiling 9

The "Handicap" Tab defines the specific settings for handicapped tournaments (where there is a huge difference in players' levels).

Tournament Settings - Placement

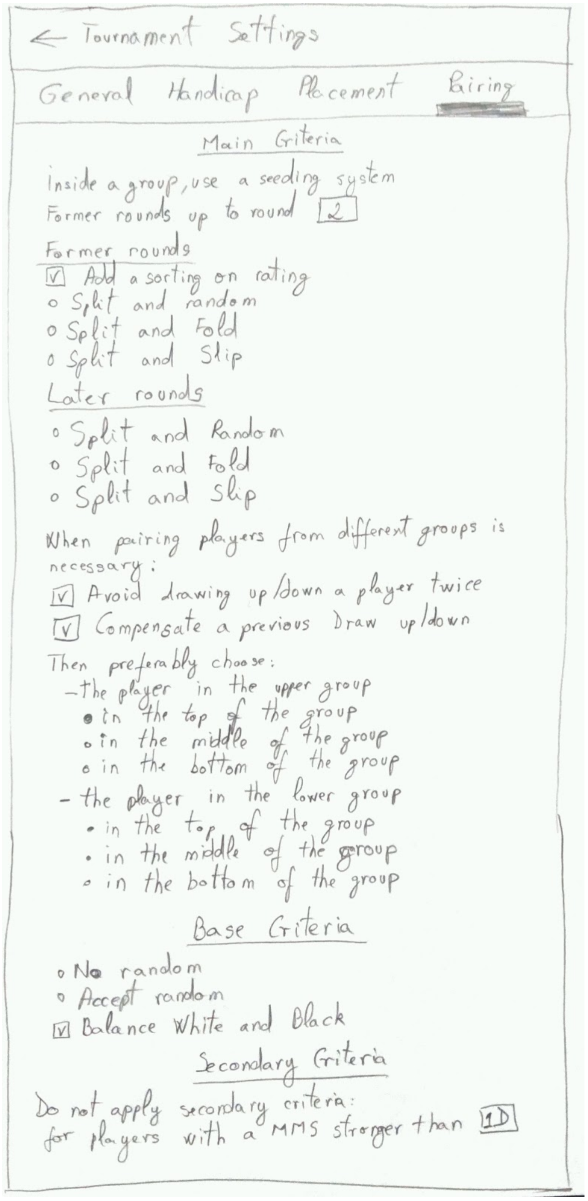


Criterion 1: MMS
Criterion 2: SOSM
Criterion 3: SOSOS
Criterion4:

Name	Description
MMS	McMahon Score
SOSM	Som of Opponents scores
SOSOM	Sum of Opponents SOSM
EXT	Exploits Tentes
DC	Direct Confrontation

In this tab from the "Tournament Settings" screen you can set the sorting criteria to determine the place occupied by each participant in the tournament.

Tournament Settings - Pairing



Main Criteria

inside a group, use a seeding system

Former rounds up to round 2

Former rounds

☑ Add a sorting on rating

○ Split and random

○ Split and Fold

○ Split and Slip

Later rounds

○ Split and Random

○ Split and Fold

○ Split and Slip

When pairing players from different groups is necessary:

☑ Avoid drawing up/down a player twice

☑ Compensate a previous Draw up/down

Then preferably choose:

- the player in the upper group
  - ● in the top of the group
  - ○ in the middle of the group
  - ○ in the bottom of the group
- the player in the lower group
  - ○ in the top of the group
  - ○ in the middle of the group
  - ○ in the bottom of the group

Base Criteria

○ No random

○ Accept random

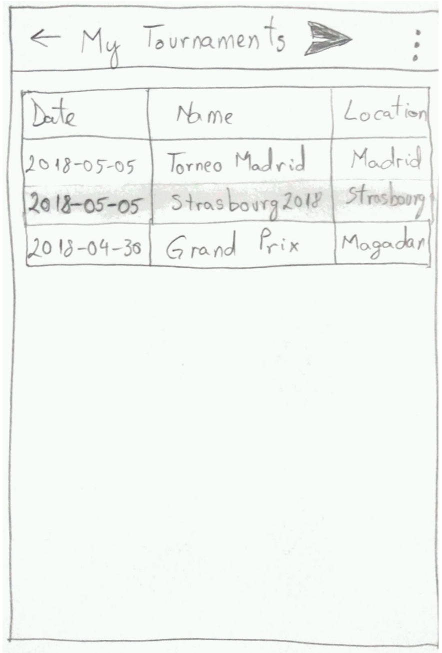
☑ Balance White and Black

Secondary Criteria

Do not apply secondary criteria:
for players with a MMS stronger than 1D

The "Pairing" tab contains specific settings that will be used by the pairing algorithm. The settings are divided into three main categories: "Main Criteria", "Base Criteria" and "Secondary Criteria" to better understand how the pairing algorithm works and can be fine tuned.

My Tournaments



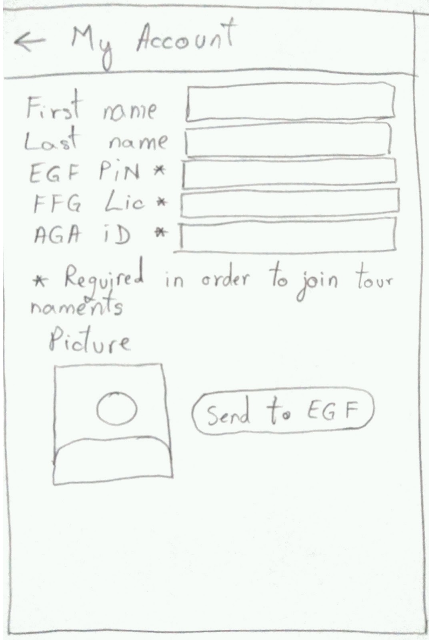
Date	Name	Location
2018-05-05	Torneo Madrid	Madrid
2018-05-05	Strasbourg2018	Strasbourg
2018-04-30	Grand Prix	Magadan

"My Tournaments" screen displays the tournaments created by the app user or the tournaments to which the user has registered for participation.

This screen defines the following possible actions after a tournament is selected in the list:

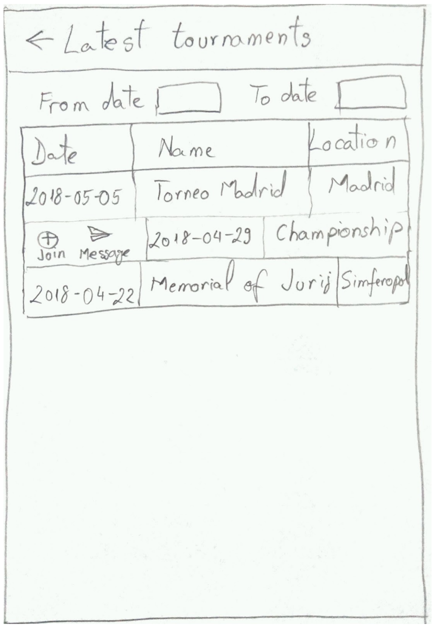
- Publish (available if the user is the creator of the tournament)
- Send Message. If the user is the tournament organizer, they will send a message to all the tournament participants. If he participates in the tournament, he can send a message to the organizer. The message will be entered via a "Dialog" screen with an input field to enter the message.
- Unsubscribe if the user is a participant in the tournament.
- Export. This option saves the tournament file to local storage.
- Edit. Available if the user is the one who created the tournament. This option will display the "Tournament Settings" screen

My Account



In order for the user of the application to create tournaments or to be able to register in a tournament defined by someone else, it is necessary to identify the user in players database. Because there are 3 different databases of players, the user will need to be identified by at least one of the 3 available IDs: EGF, FFG or AGA.
In addition, the user has the possibility to upload the profile photo on the EGF accessing the camera by pressing the picture below "Picture" label and taking a selfie.

Latest Tournaments

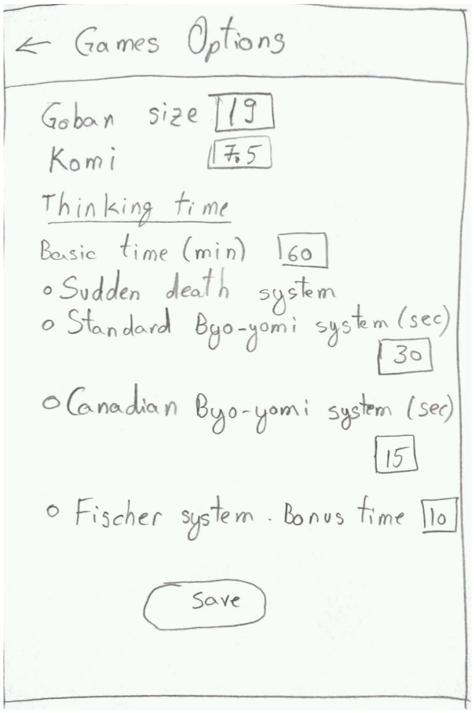


Date	Name	Location
2018-05-05	Torneo Madrid	Madrid
Join Message	2018-04-29	Championship
2018-04-22	Memorial of Jurij	Simferopol

“Latest Tournaments” is displayed on app’s home screen. Here the users can see what tournament was published recently and can search for other tournaments uploaded in a specified interval at the top of the screen. By pressing on a row from the list, a detailed tournament info will be presented. Those tournaments are taken from http://www.europeangodatabase.eu/EGD/

and will be fetched through an API published on firebase platform.

Games options



← Games Options

Goban size 19

Komi 7.5

Thinking time

Basic time (min) 60

○ Sudden death system

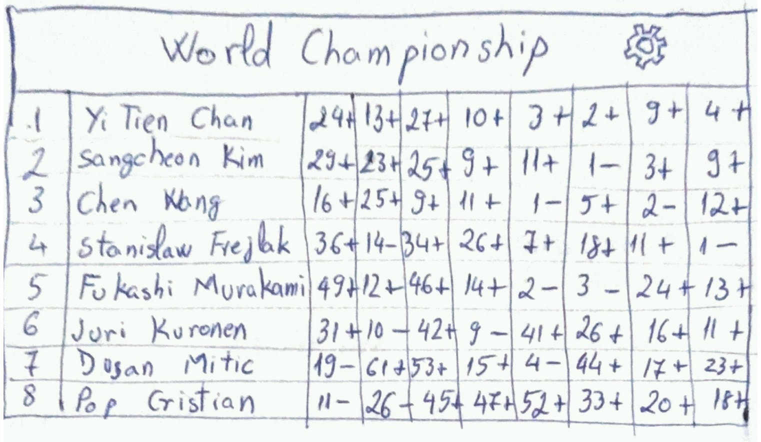
○ Standard Byo-yomi system (sec) 30

○ Canadian Byo-yomi system (sec) 15

○ Fischer system . Bonus time 10

This screen is defining how games from a tournament will be played (official rules set by the tournament director).

Widget



World Championship

1	Yi Tien Chan	24+	13+	27+	10+	3+	2+	9+	4+
2	Sangcheon Kim	29+	23+	25+	9+	11+	1-	3+	9+
3	Chen Kang	16+	25+	9+	11+	1-	5+	2-	12+
4	Stanislaw Frejlak	36+	14-	34+	26+	7+	18+	11+	1-
5	Fukashi Murakami	49+	12+	46+	14+	2-	3-	24+	13+
6	Juri Kuronen	31+	10-	42+	9-	41+	26+	16+	11+
7	Dusan Mitic	19-	61+	53+	15+	4-	44+	17+	23+
8	Pop Gristian	11-	26-	45+	47+	52+	33+	20+	18+

This widget will display the wall list including results for a tournament selected by the user. By pressing the settings button from the top-right corner, the user can select another tournament or the frequency for retrieving updates.

Key Considerations

App is written solely in the Java Programming Language. I will use Android Studio 3.1.2, gradle android plugin 3.1.2 and gradle wrapper 4.4.

How will your app handle data persistence?

Data persistence will be provided using content provider for database operations.

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

I will use OkHttp or Retrofit to get data over the internet, because it's handling many aspects, such as making connections, caching and retrying failed requests. This application will define a backend which will be deployed on firebase and will expose an API to get the latest tournaments published on EuropeanGoDatabase.

Describe how you will implement Google Play Services or other external services.

Firebase Mobile Ads - I will display interstitial ads
Google Analytics - to understand the app usage and to identify problems I will activate screen tracker and send events on the most important screens and actions available.
Firebase Authentication - In order to manage the tournaments an identification is needed. In this matter I will accomplish this by using FirebaseUI.
Firebase Cloud Messaging - for push notifications like: new player checked in, new tournament announcement, new results published, messages of general interest for the participants in a tournament.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Install Firebase SDK
- Create OpenGotha'n'Droid android project
- Configure libraries by adding dependencies in build.gradle

```
implementation 'com.google.firebase:firebase-ads:15.0.0'

implementation 'com.google.firebase:firebase-core:15.0.0'

implementation 'com.google.firebase:firebase-auth:15.1.0'

implementation 'com.google.firebase:firebase-messaging:15.0.2'

implementation 'com.squareup.okhttp3:okhttp:3.10.0'
```

- Sign up for an AdMob account and link the app to Firebase project
- Add this app to Firebase project in Firebase console
- Get the project server's key for Firebase authentication
- For Firebase Cloud Messaging add in manifest file the services that extends FirebaseMessagingService and FirebaseInstanceIdService to handle the registration tokens.
- Create a Java Module to provide the core functionality of this Tournaments Manager. Include opengotha.jar in lib directory.
- Create a Backend Module to define Google Cloud Endpoints that will provide the latest published tournaments on EuropeanGoDatabase.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity (Home screen)
- Build UI for activity Create Tournament
- Build UI for two fragments in Players Manager (one for selecting the player to register in tournament and one for listing the already registered players in tournaments.
- Build UI for registering a new player
- Build UI for fragment which display the pairings

- Build UI for fragment which display the results in the current round
- Build UI for Tournament Settings Activity
- Build UI for tab fragments in Tournament Settings: General, Handicap, Placement, Pairing
- Build UI for fragment that displays the user's tournaments
- Build UI for activity MyAccount
- Build UI for fragment displaying the latest published tournaments on EGD (European Go Database) provided by the backend module.
- Build UI for activity Games Options.

Task 3: Authenticate with Firebase

- Add dependencies to build.gradle

```
dependencies {
    // ...
    implementation 'com.firebaseui:firebase-ui-auth:3.2.2'
    // Required only if Facebook login support is required
    compile('com.facebook.android:facebook-android-sdk:4.27.0')
    // Required only if Twitter login support is required
    compile("com.twitter.sdk.android:twitter-core:3.0.0@aar") { transitive = true }
}
```

- Follow this documentation for the rest of implementation:

https://firebase.google.com/docs/auth/android/firebaseui

Task 4: Create, save and export a tournament

- Creating a tournament consists in entering the required data in the create screen. The data will be saved on a local database using content provider.
- In order to export the tournament as a file on local storage, the OpenGotha java library will be used.

Task 5: Download and manage world players

I will download the players using OkHttp (or retrofit) from this link:
http://www.europeangodatabase.eu/EGD/EGD_2_0/downloads/allworld_lp.html
The players will be saved in the database to be available offline. This will be implemented through an IntentService.

Task 6: Enable participants to join to the tournament

Using Firebase, the participants will subscribe to tournaments defined as topics.

Task 7: Generate rounds and display results

This is a straightforward operation described above in Pair and Results screens.

Task 8: Provide different types of tournaments

I rely on the pairing algorithm from the open source project OpenGotha, included in java module, to take care of this.

Task 9: The newly defined tournaments can be announced to all the users having the app installed (also messages related to a certain tournament can be sent to all registered participants)

I will use firebase cloud messages to send those announces to all the users of this app. In case the organizer needs to notify the participants of a certain tournament about something important to know, I will send cloud messages to all users that subscribed previously to that topic (aka tournament)

Task 10: The tournament organizer can send private message to a certain participant

Like above, the difference is that the organizer is selecting a player from the participants list. After selecting the player, a DialogFragment will be shown to enter the message to be sent. Pressing a send button, the message will be sent to the participant using Firebase cloud messaging service using the associated registration ID that will be provided when the participant joined to the tournament.

Task 11: Publish the results on the web

Here I will send the tournament file to backend. The backend will take the file and save it to a database. All the saved tournaments are displayed on the web to show the published tournaments along with the partial / final results recorded in each round.
Publishing tournaments will automatically trigger the notification of participants about the fact that new information has been published.

Task 12: Take pictures of the user and can update the official european federation site

This will simply start the Camera to make a selfie. The photo will be resized and sent to European Go Database to update the Profile photo of the user. This will consist in sending an email to aldo.podavini@europeangodatabase.eu (there is no other alternative for the moment)

Task 13: Get the latest published tournaments on EuropeanGoDatabase

These will be regularly requested from the API provided by backend, the app updating the data in its database at regular intervals using a JobDispatcher.

Task 14: Create Build Variant

Define free and paid flavors. The difference consists only in implementing different layouts and classes to show interstitial ads after a tournament is created or published to the web.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"