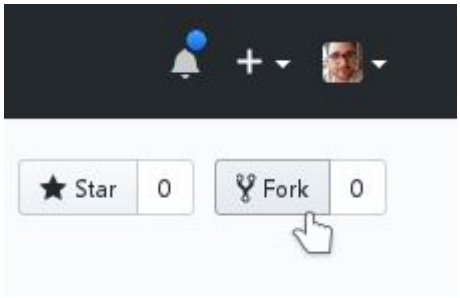
Miniprojekt – Android

In den nächsten Wochen werden Sie im Rahmen des Miniprojekts verschiedene Erweiterungen an der *Beer Pro* App vornehmen. Das Google-Doc mit aktualisierten Hinweisen finden Sie unter https://goo.gl/1EiG6N.

Die Teams sind hier einzutragen:
https://docs.google.com/spreadsheets/d/1elvtQfkUvOrIl8bsM-_Ut4EkPsTDIRi8Q9wVpqfZNRY/

Projekt forken

Erstellen Sie auf GitHub einen Fork des Vorlagen-Repositories https://github.com/HSR-MGE/beerpro.
Wichtig: die Abgabe erfolgt am Ende des Projekts per Pull-Request. Das ist am einfachsten, wenn Sie gleich zu Beginn schon einen Fork erstellen:



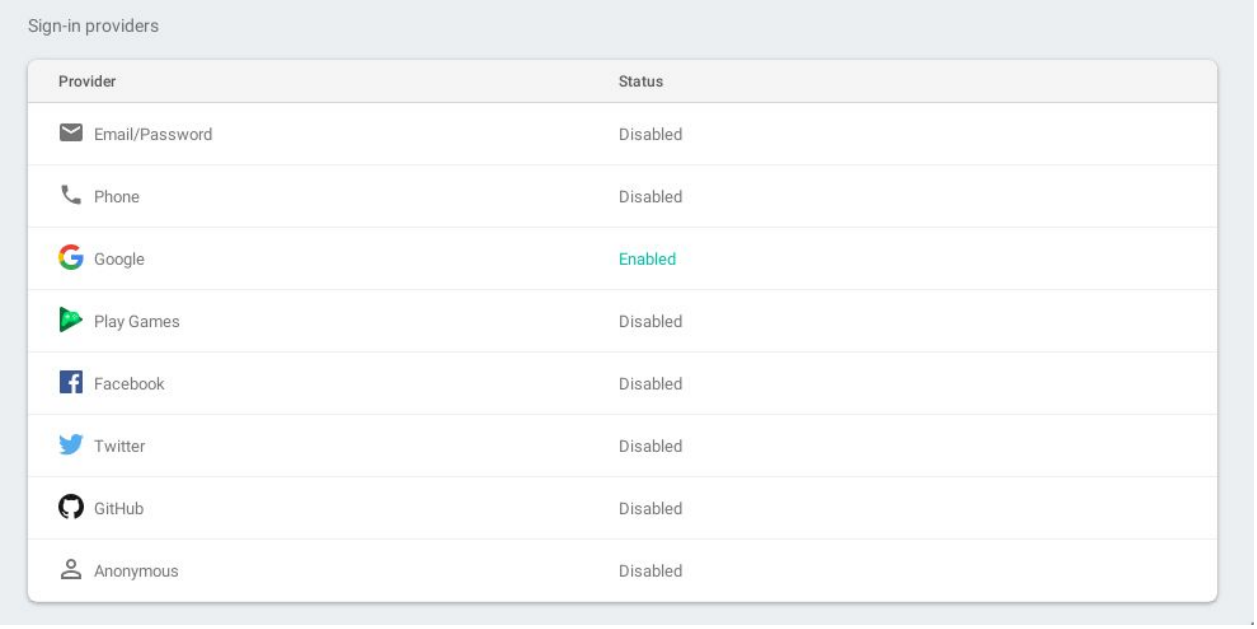
Falls Sie das Projekt nicht forken möchten (da Sie z.B. in einem privaten Repository arbeiten möchten), können Sie es auch klonen und in ein privates Repository committen. Wichtig ist, dass die bestehende History nicht verloren geht damit ich am Ende ein Diff machen kann. Um mir den Zugriff zu ermöglichen können Sie den GitHub-Benutzer *misto* als Contributor hinzufügen.

Firebase Projekt erstellen

Als nächstes benötigen Sie pro Team ein Firebase-Projekt. Erstellen Sie unter https://firebase.google.com/ ein neues Projekt. Sie können auch mit anderen Teams zusammen dasselbe Projekt verwenden, bei Änderungen an der Struktur der Datenbank sollten Sie sich aber mit den anderen Teams absprechen. Die App benötigt die Firebase **Authentication**, **Firestore** und **Functions**. Klicken Sie sich bei allen drei Services durch den Einrichtungsprozess:

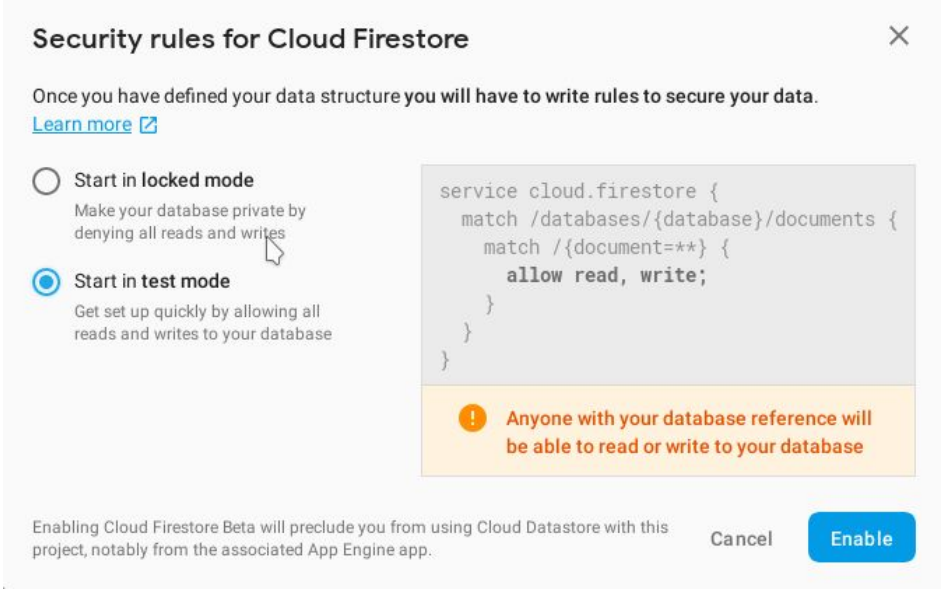
Authentication

Bei den Sign-in Providern können Sie Google aber auch beliebige andere Provider aktivieren, aber mindestens “Google”.



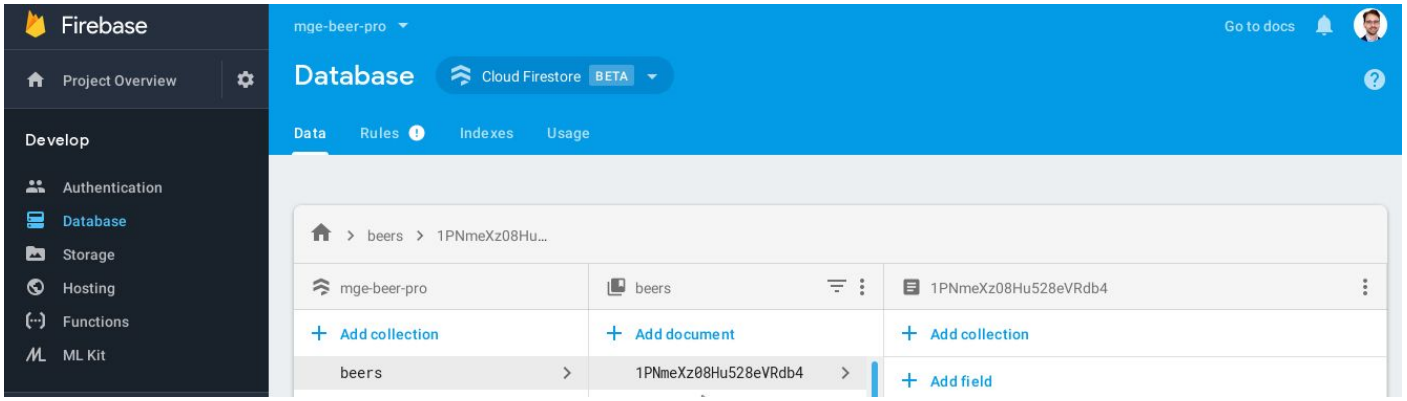
Firestore

Bei der Erstellung der Datenbank werden Sie nach den Security Rules gefragt, verwenden Sie dort für den Anfang den *test mode*:



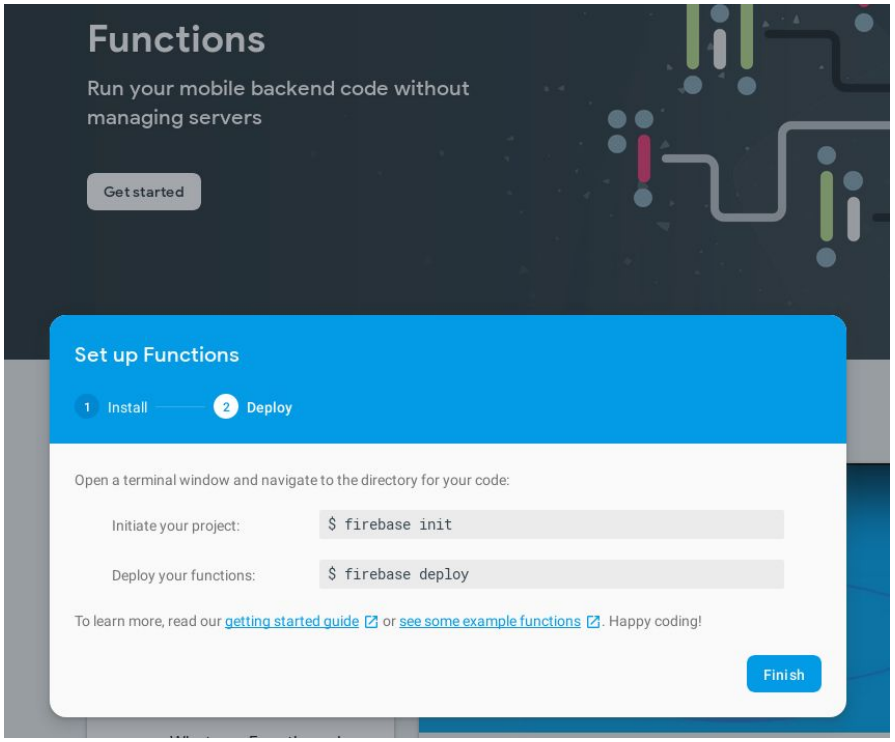
Initiale Daten importieren

Als nächstes können Sie die initialen Daten importieren. Dazu finden Sie im *scripts* Verzeichnis eine weitere **Anleitung**. Danach sollten Sie in der Weboberfläche eine Collection *beers* sehen:



Functions

Neben den Daten benötigen wir auch noch serverseitige *Functions*, Code der aufgerufen wird wenn sich Daten (in unserem Fall Ratings) ändern. Sobald ein neues Rating erstellt wird, berechnet diese Funktion die neue Durchschnittsbewertung. Folgen Sie auch hier der Firebase-Anleitung Schritt 1 (`npm install -g firebase-tools) und Schritt 2 (im Rootverzeichnis des geklonten Projekts ausführen).



Bei Schritt zwei müssen Sie auf der Commandline nochmals Functions auswählen. Als nächstes wählen Sie das eben erstellte Projekt aus. Die restlichen Eingabeaufforderungen können Sie wie folgt beantworten:

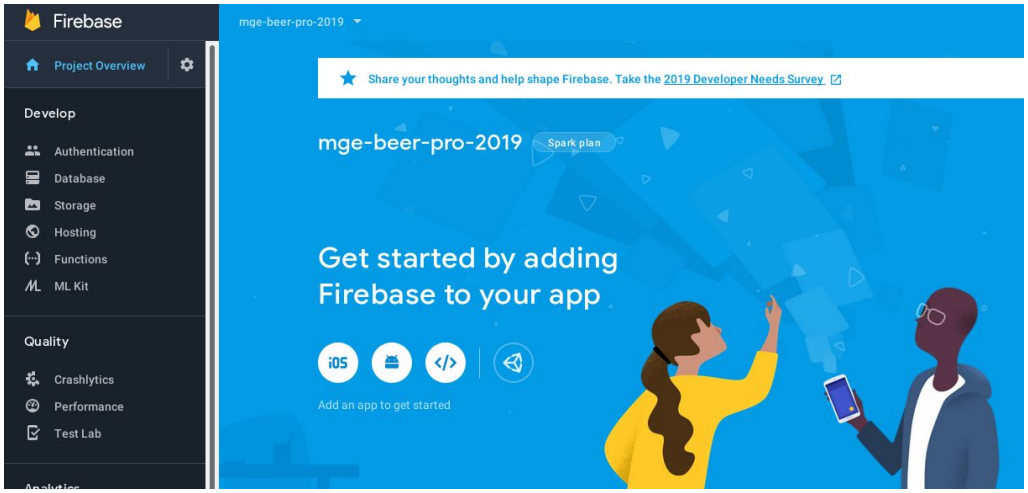
```
$ firebase init

     ######## #### ########  ######## ########     ###    ######  ########
     ##     ##  ##  ##        ## ##        ##     ## ##   ##    ## ##
     ######  ##  ########  ######   ########  ######### ######  ######
     ##     ##  ## ##  ##  ##       ##     ## ##  ##     ## ##
     ##     #### ##      ## ######## ########  ##   ##  ######  ########
```

```
You're about to initialize a Firebase project in this directory:

  /home/misto/repos/beerpro

? Which Firebase CLI features do you want to set up for this folder? Press Space to select
features, then Enter to confirm your choices. Functions: Configure a
nd deploy Cloud Functions

=== Project Setup

First, let's associate this project directory with a Firebase project.
You can create multiple project aliases by running firebase use --add,
but for now we'll just set up a default project.

? Select a default Firebase project for this directory: mge-beer-pro-2019 (mge-beer-pro-2019)
i  Using project mge-beer-pro-2019 (mge-beer-pro-2019)

=== Functions Setup

A functions directory will be created in your project with a Node.js
package pre-configured. Functions can be deployed with firebase deploy.

? What language would you like to use to write Cloud Functions? JavaScript
? Do you want to use ESLint to catch probable bugs and enforce style? No
? File functions/package.json already exists. Overwrite? No
i  Skipping write of functions/package.json
? File functions/index.js already exists. Overwrite? No
i  Skipping write of functions/index.js
? File functions/.gitignore already exists. Overwrite? No
i  Skipping write of functions/.gitignore
? Do you want to install dependencies with npm now? Yes
```

Um die Functions zu deployen reicht dann ein `firebase deploy`.

Firebase Android

Nun können wir endlich das Projekt mit Android-Studio öffnen. Es ist aber noch ein weiterer Schritt mit Firebase nötig, und zwar müssen wir auch noch die Android-Vorlage mit Firebase verbinden. Wählen Sie dazu hier das Android Icon:



Als Android package name verwenden Sie `ch.beerpro`, zudem müssen Sie unbedingt den SHA Zertifikatsfingerprint angeben (die Fingerprints der weiteren Teammitglieder können Sie in einem zweiten Schritt hinzufügen – Achtung, laden Sie dann jeweils wieder die Konfigurationsdatei neu herunter). Danach laden Sie die Konfigurationsdatei herunter und sind bereit für den Start.

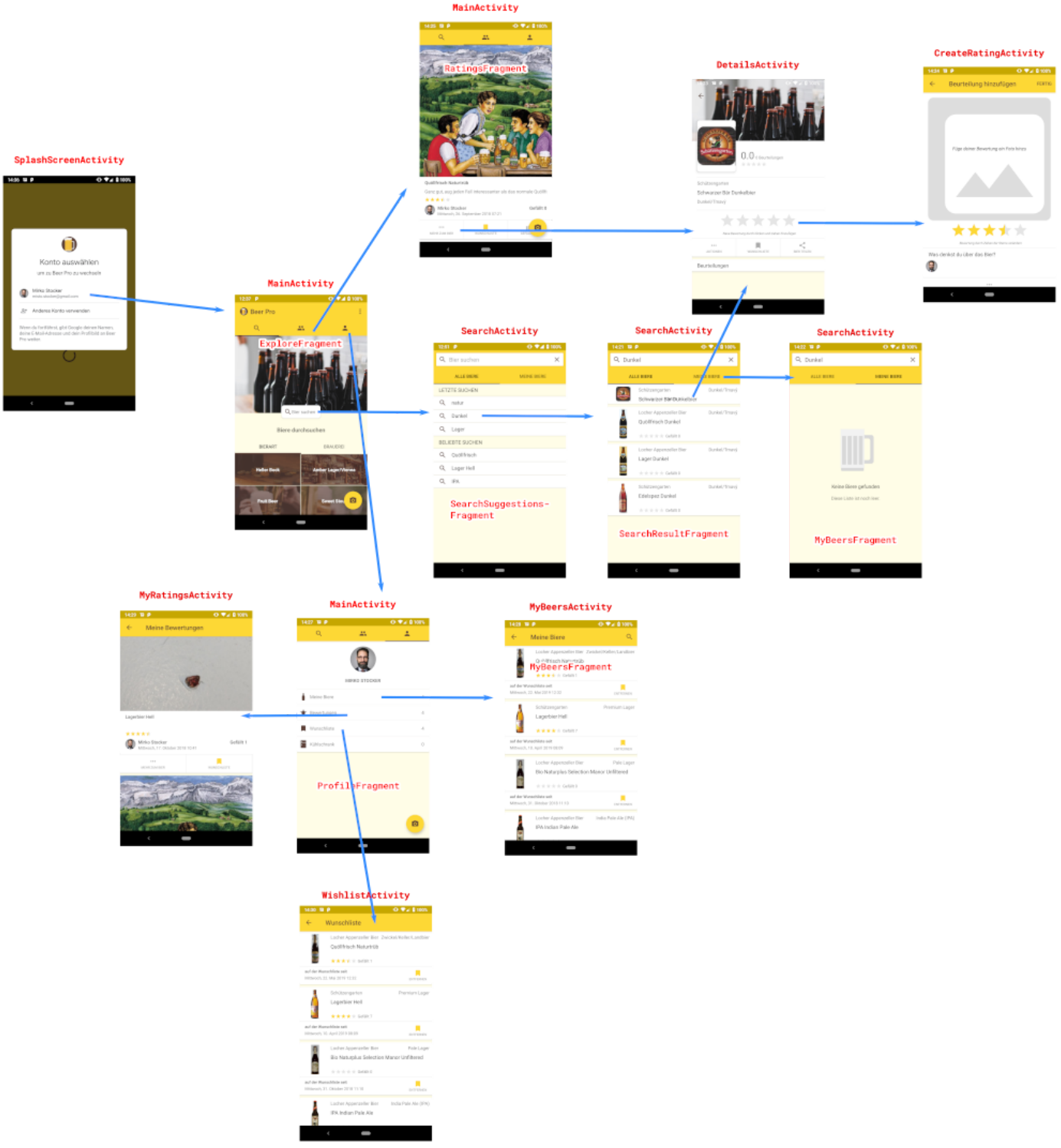
Projekt starten

Wenn alles geklappt hat sollten Sie jetzt die App auf Ihrem Smartphone (empfohlen) oder dem Emulator starten können.

Tipp: Falls Sie nicht über den Login-Dialog hinwegkommen könnte es gut sein, dass etwas mit dem Zertifikatsfingerprint nicht stimmt.

Übersicht über das Projekt

Die Vorlage ist zugegebenermassen ziemlich umfangreich. Es ist aber nicht nötig und auch gar nicht das Ziel, jede Zeile des Codes gelesen oder verstanden zu haben. In einigen zentralen Klassen finden Sie auch JavaDoc Kommentare die weiterhelfen. Um den Einstieg zu erleichtern hier noch eine Übersicht über die einzelnen Screens mit den jeweiligen Activities und Fragments:



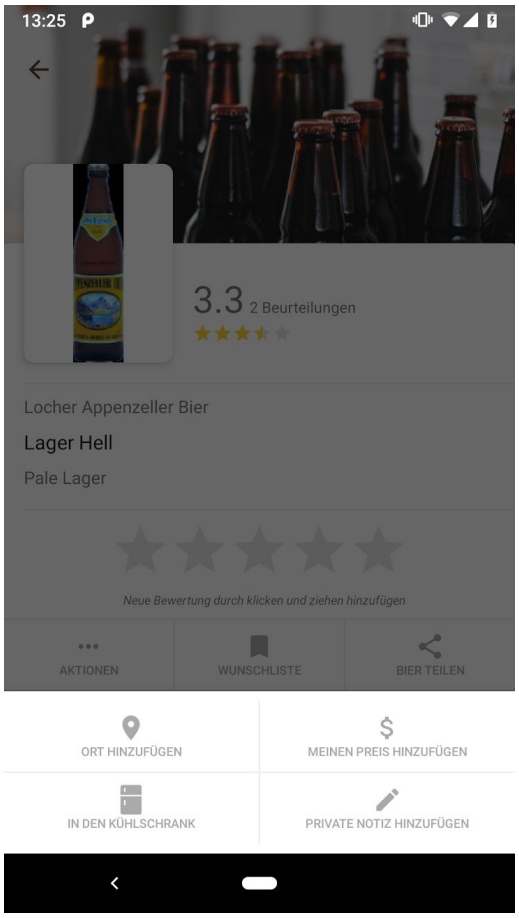
Aufgabenstellungen

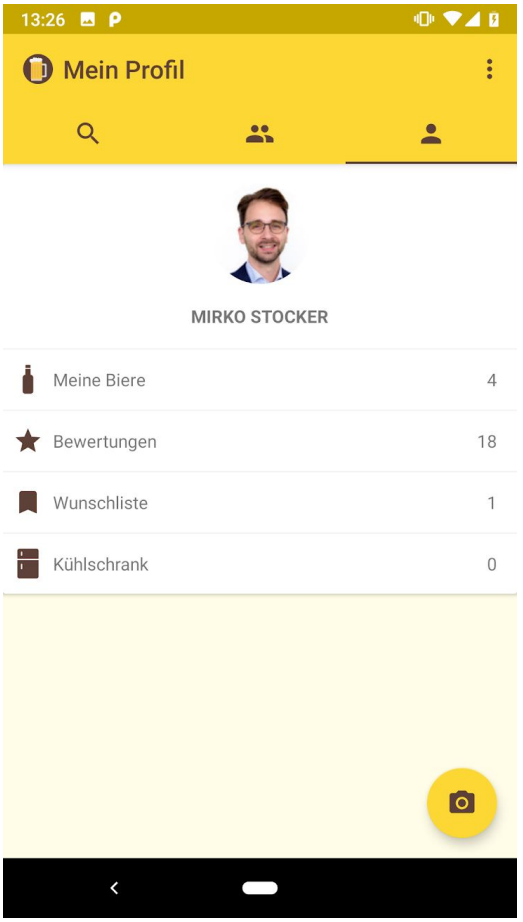
Die Beer Pro App ist bereits funktionstüchtig, allerdings werden Sie merken, dass einige Features noch nicht eingebaut sind. Nachfolgend finden Sie einige Themenvorschläge die Sie umsetzen können. Jede Aufgabe gibt je nach Schwierigkeit eine Menge an Punkten, für die **Testaterreichung** benötigen Sie mindestens **10 Punkte** (2er Team, für ein 3er Team entsprechend 15 Punkte, und so weiter). Sie können sich auch mit

anderen Teams absprechen und zusammen arbeiten um am Ende eine möglichst vollständige App zu haben.
Zur Abgabe gehört auch eine kleine Beschreibung / ein Readme der Features die Sie implementiert haben (zum Beispiel: User Story, Mockup falls vorhanden oder Screenshots).

Kühlschrank (4P)

Analog zur Wunschliste sollte man eine Liste führen können mit den Bieren die sich gerade im Kühlschrank (oder Keller) befinden. Dazu muss es möglich sein, ein Bier zum Kühlschrank hinzuzufügen sowie die Liste aller Biere im Kühlschrank anzuzeigen:

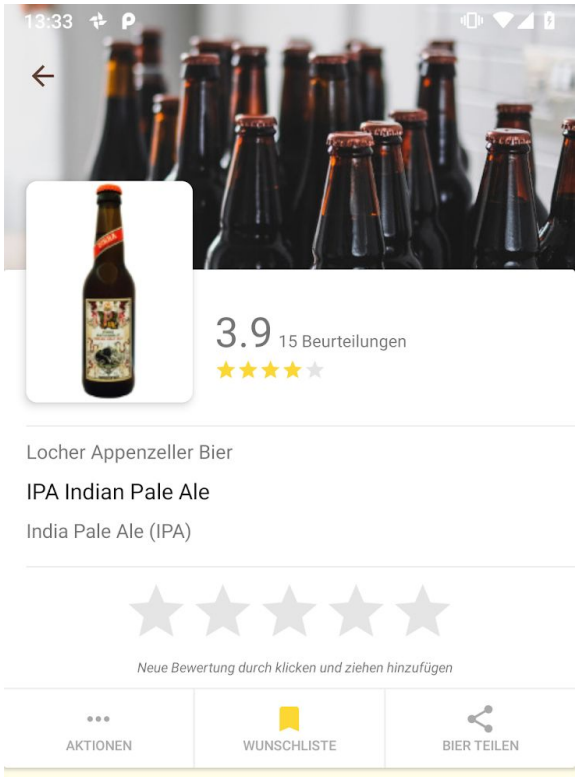




Sie benötigen dazu ein neues Repository sowie eine neue Model-Klasse um die beerId sowie die Menge abzulegen. In der Liste “Kühlschrank” wäre es zudem nützlich, wenn man die Menge einstellen könnte. Zudem soll bei der “Meine Biere” Ansicht auch der Inhalt des Kühlschranks angezeigt werden, Einträge aus der Wunschliste oder meine Biere soll man einfach in den Kühlschrank “verschieben” können.

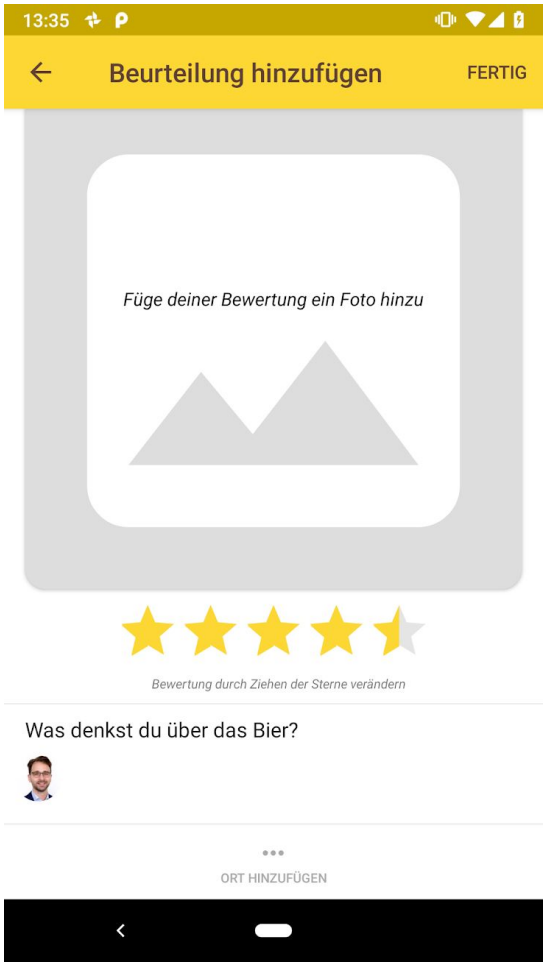
Eigene Bewertung (2P)

Bei der DetailsActivity (siehe unten) soll die eigene Bewertung (read-only) anstelle der Sterne angezeigt werden, falls eine Bewertung vorhanden ist. Um trotzdem noch neue eigene Bewertungen machen zu können soll dafür eine Menupunkt bei den *Aktionen* hinzukommen.



Beurteilung mit Zusatzinformationen (5P)

Im Moment kann bei einer Beurteilung bloss ein Foto sowie ein Text hinzugefügt werden. In der Vorlage bereits vorgesehen ist, dass auch ein Ort angegeben werden kann (Ort Hinzufügen). Dies liesse sich mit dem Place Picker API von Google Maps (https://developers.google.com/places/android-sdk/placepicker) realisieren. Der ausgewählte Ort soll natürlich auch bei jeder Bewertung angezeigt werden. Neben dem Ort wäre es für Profis nützlich, noch weitere qualitative Merkmale zu bewerten (z.B. Aromen als Aufzählungsliste () oder Bitterkeit als Skala von 1 bis N). Alle diese zusätzlichen Informationen können Sie in der Rating Klasse hinterlegen.



Übersicht nach Orten (5P)

Basierend auf der *Beurteilung mit Zusatzinformationen* liesse sich auf dem MainActivity-Screen ein neuer Tab hinzufügen der die Bewertungen auf einer Karte angezeigt werden. Die Orte (wahrscheinlich primär Restaurants und Bars) sollten auch durchsuchbar sein und als Liste angezeigt werden. (Tipp zur Implementierung: Erstellen Sie eine neue Collection *places* um die Orte abzulegen. Wird zum ersten Mal eine Bewertung erstellt wird der Ort in der Collection erfasst, andernfalls einfach referenziert).

Freunde (10P)

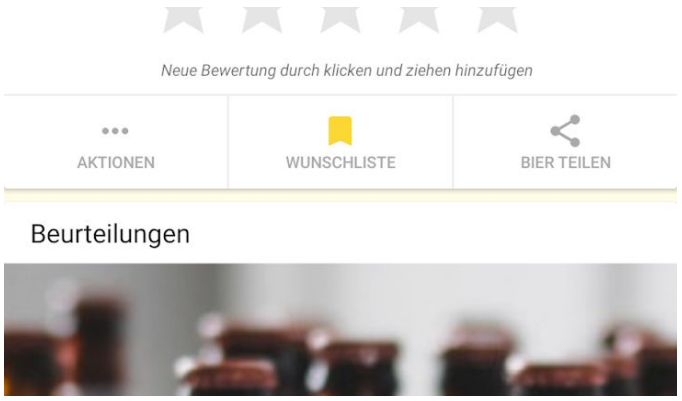
Im Moment werden die Beurteilungen aller Benutzer angezeigt. Besser wäre es, wenn man sich mit seinen Freunden vernetzen könnte und dann (wahlweise) nur deren Beurteilungen sieht. In der DetailActivity könnten z.B. Tabs für die Beurteilungen erstellt werden (Beste Beurteilungen, Freunde, Eigene).

Dark Theme (3P)

Erstellen Sie ein Dark Theme für die ganze App, inkl. Setting um den Modus zu wechseln. Achtung, auch wenn sich diese Aufgabe vielleicht einfach anhört, sie ist es definitiv nicht.

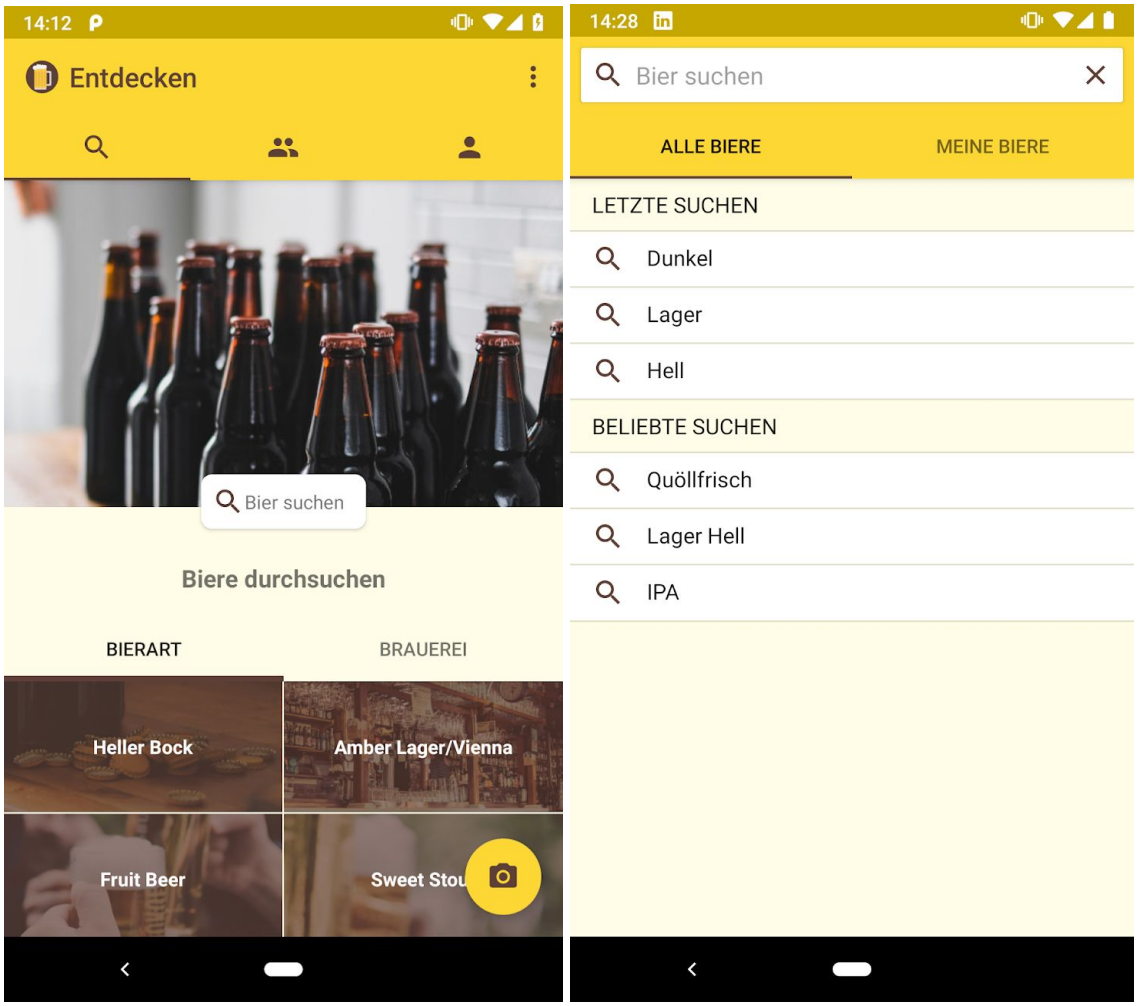
Testing (nP)

Falls Sie sich fürs Testing interessieren, besteht die Option, automatisierte Tests für **Ihre Erweiterung** der App zu schreiben. Pro 10% Test-Coverage die Sie erreichen erhalten Sie 1 Punkt für das Testat. Dazu müssen Sie natürlich noch was zusätzliches implementieren, es gilt nicht 0 Zeilen eigenen Code zu schreiben und dann mit einem Test 100% davon abzudecken und 10 Punkte abzuholen :-) .



Entdecken nach Kategorien (4P)

Die beiden *Biere durchsuchen* Tabs Bierart und Brauerei sind zwar bereits mit Kategorien und Herstellern versehen, allerdings kann man damit noch nicht viel anstellen. Wünschenswert ist eine Übersicht mit allen Bieren der jeweiligen Kategorie. Als Variante wäre es auch möglich, diesen zusätzlichen Filter in der SearchActivity (rechts) einzubinden. Zum Beispiel mit Filter Chips oder analog zum Filter der Friendly Eats App.



Stammdaten editieren (2P)

Die Daten zu einem Bier sind in der Firestore Collection *beers* abgelegt, können jedoch nicht über die App verändert werden. Eine Funktion in der Anwender falsche Daten (oder ein besseres Bild) melden können wäre nützlich.

Lokales Update der Bewertung (1P)

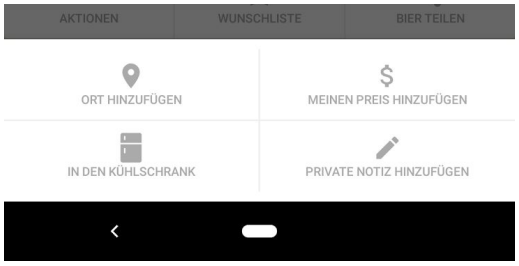
Wird eine neue Bewertung hinzugefügt ändert sich das Total an Bewertungen sowie wahrscheinlich auch die Durchschnittsbewertung. Im Moment wird der neue Durchschnitt Serverseitig berechnet und die Anzeige in der App wird erst mit einigen Sekunden bis Minuten Verzögerung angezeigt. Eine bessere Lösung wäre, wenn die CreateRatingActivity das Rating-Objekt an die aufrufende Activity zurückgibt und diese für die lokale Darstellung einen neuen Durchschnitt berechnet.

Bier Teilen (1P)

Untersuchen Sie die Sharing-Funktion von Android und ermöglichen Sie es dem User, ein Bier zu sharen. (Hat ein anderer User die App auch installiert, kann dieser mittels Deep Link direkt auf die Detail-Ansicht in der App gelangen. Die Implementierung mittels des verlinkten Deep Links gibt 3 Punkte!)

Eigenen Preis hinzufügen (3P)

Anwender sollen bei einem Bier eigene Preise hinterlegen können. Analog zum Rating lässt sich damit auch ein Durchschnittspreis berechnen.



Sobald bei einem Bier ein eigener Preis hinzugefügt worden ist wird dieses auch in *Meine Biere* angezeigt.

Private Notiz hinzufügen (2P)

Bewertungen sind immer öffentlich. Als Alternative dazu soll der Anwender private Notizen (siehe Screenshot oben) zu einem Bier hinzufügen und editieren können. Auch hier gilt, sobald eine private Notiz hinzugefügt wurde soll das Bier in *Meine Biere* gespeichert werden.

Hintergrundbilder (1P)

Diverse Hintergrundbilder in der App (z.B. DetailsActivity) sind im Moment statisch hinterlegt. Stattdessen soll z.B. bei der DetailsActivity das Bild des Herstellers (ein beliebiges Hintergrundbild das fest in der Datenbank hinterlegt werden kann aber für jeden Hersteller unterschiedlich sein soll). Dasselbe Bild soll dann auch bei der Brauerei-Kachel in der MainActivity angezeigt werden.

Bug Bounty (nP)

Haben Sie einen Bug im bestehenden Code der Applikation gefunden? Je nach Schwierigkeit erhalten Sie hier variable Punkte. Sprechen Sie mit Ihrem Übungsbetreuer!

Ihre eigene Idee (nP)

Haben Sie selbst noch eine Idee? Fragen Sie mich nach den Punkten die Sie dafür bekommen. Falls Sie mit Bier nicht viel anfangen können dürfen Sie die App auch in eine andere Domäne verpflanzen. Wie wäre es mit Gin? Oder Cordon Bleu?

Hinweise zur Implementierung

Nachfolgend ein paar Hinweise zur Umsetzung des Miniprojekts. Sie können das Dokument auch selbst um nützliche Tipps und Links ergänzen.

Datenmodellierung mit Firestore

Firestore ist eine sogenannte NoSQL-Datenbank und verfügt über einige Einschränkungen. Falls Sie neue Collections hinzufügen, lohnt es sich die Artikel zu Firestore NoSQL Relational Data Modeling und Advanced Data Modeling With Firestore by Example anzuschauen.