

Dokumentation

CaretakerApp

von Fabian Berkowitz und Christoph Kaiser

in Zusammenarbeit mit

ucura GmbH

Gewerbestraße 3

8500 Frauenfeld, CH

Einleitung

Im Rahmen der Vorlesung „Mobile Computing“ musste eine App entwickelt werden. Welchen Use Case man abdecken möchte wurde den Studenten frei überlassen. Die ursprüngliche App Idee wurde von Armando Statti geliefert. Dieser begleitet, im Rahmen der Start-Up Gründung, die ucura GmbH bei der Fertigstellung einer IOS App. Um das Portfolio zu erweitern befassten wir uns mit der Android basierten Entwicklung und somit zum Gegenstück der IOS Version. Die zu entwickelnde App sollte somit dieselbe Grundfunktionalität zur Verfügung stellen. Auf die Umsetzung aller Features wurde aufgrund des kurzen Projektzeitraums verzichtet. Zudem wurde die App in zwei Teile aufgespalten. Eine Pfleger - wird in dieser Dokumentation behandelt - und eine Patientenseite.

Idee

Die App ist zur Vernetzung von Patienten und Pflegern gedacht. Doch wie gesagt wird im Folgenden nur auf die Pfleger Seite eingegangen. Nach einer initialen Registrierung wird der Pfleger durch den sogenannten Onboardingprozess geführt. Hierbei werden persönliche Daten (Name, Geburtstag, Adresse, ...) sowie Informationen zum geplanten Arbeitsverhalten (Stunden pro Woche, Interessen, Gehalt/h und Arbeitszeiten). Nachdem der Onboardingprozess abgeschlossen ist wird der Pfleger zu seinem „Dashboard“ weitergeleitet. Das Dashboard ist das zentrale Element der App. Es enthält Kontaktanfragen von Patienten und ist somit der Startpunkt um eine Geschäftsbeziehung mit einem Patienten aufzubauen.

Architektur

Bei der Architektur haben wir uns für MVVM entschieden. Diese Architektur ermöglicht es uns eine modulare App aufzubauen. Durch die verschiedenen Schichten Modelview, View und Model ist eine klare Abgrenzung möglich. D.h. durch die quasi non-existente Beziehung zwischen Modelview und View ist es ohne Probleme möglich die einzelnen Komponenten zu Testen. Die Kommunikation zwischen View und Modelview findet über LiveData statt. Dies ist eine von Google entwickelte Datenstruktur zum Datenaustausch, bei dem sich die beteiligten Komponenten gegenseitig nicht kennen müssen.

Unsere genutzte Modelschicht enthält die zugrunde liegenden Datenstrukturen um Pfleger sowohl lokal als auch online abzuspeichern. Für diese Funktionalitäten haben wir uns zur lokalen Speicherung (Datenbank) für Room und zur online Speicherung für Retrofit entschieden. Diese beiden Bibliotheken bieten eine Abstraktion zur wirklichen Umsetzung und ermöglichen somit eine einfachere Implementierung als „händisch“ den Datenverkehr zu behandeln. Die Datenbank wurde über das Singleton Pattern implementiert und liefert somit immer nur eine Instanz des Datenbankobjekts was eine ressourcenschonendere und vor allem eine korrekte Implementierung ermöglicht. Um die Synchronisierung des Online- und Lokalbestands sicherzustellen wurde eine Repositoryklasse erstellt. Diese wird ebenfalls durch das Singleton Pattern auf eine Instanz begrenzt.

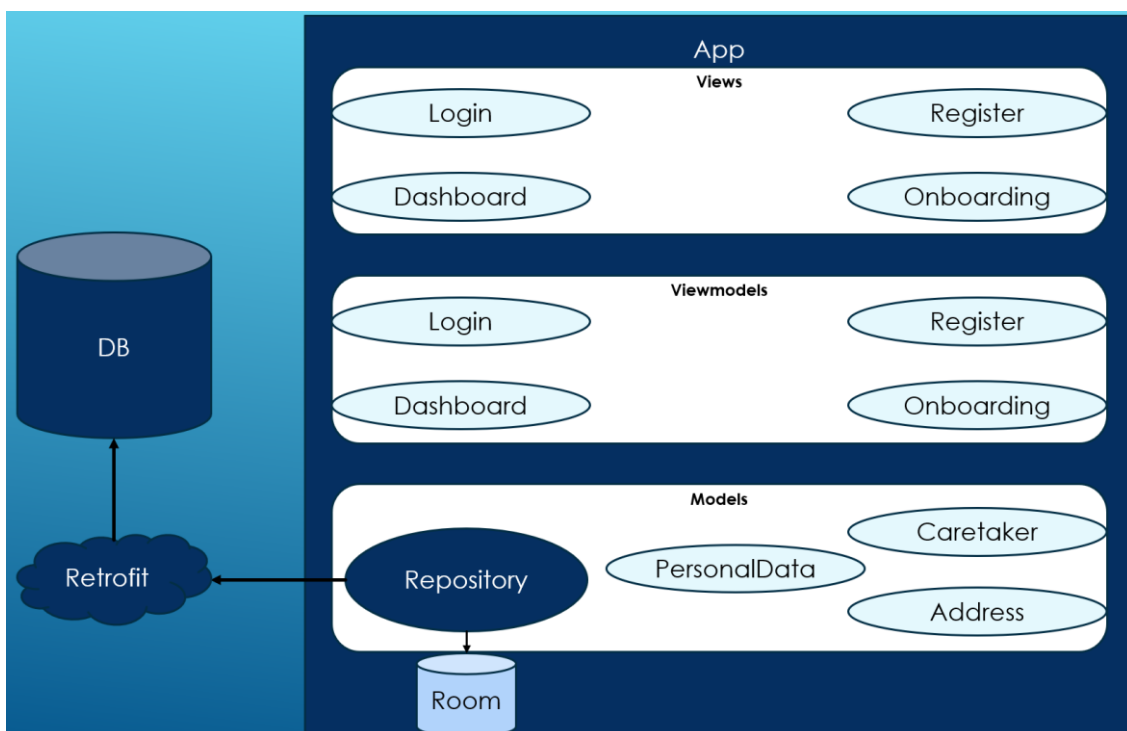
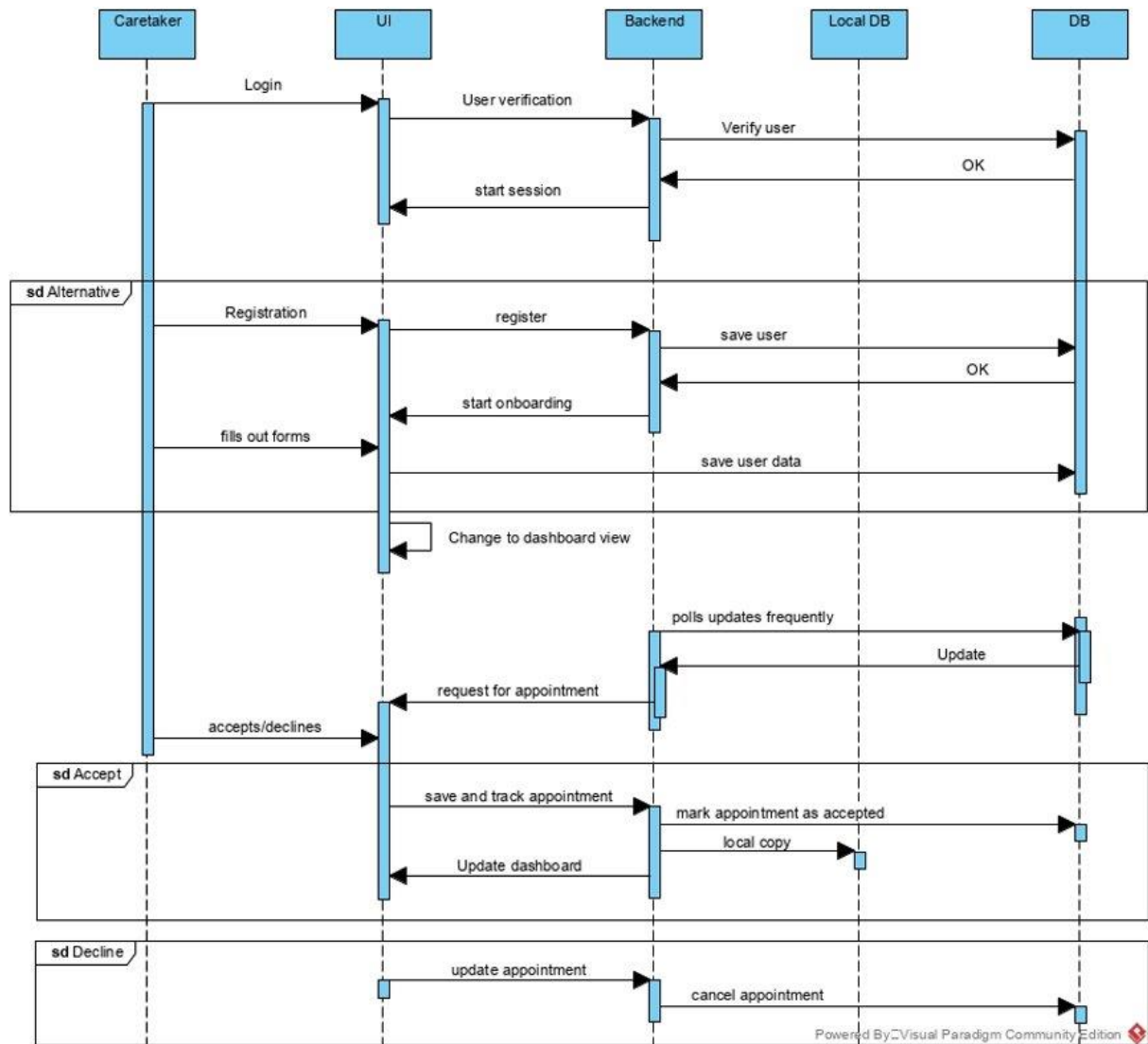


Abbildung 1 Grobe Architektur

Sequenzdiagramm



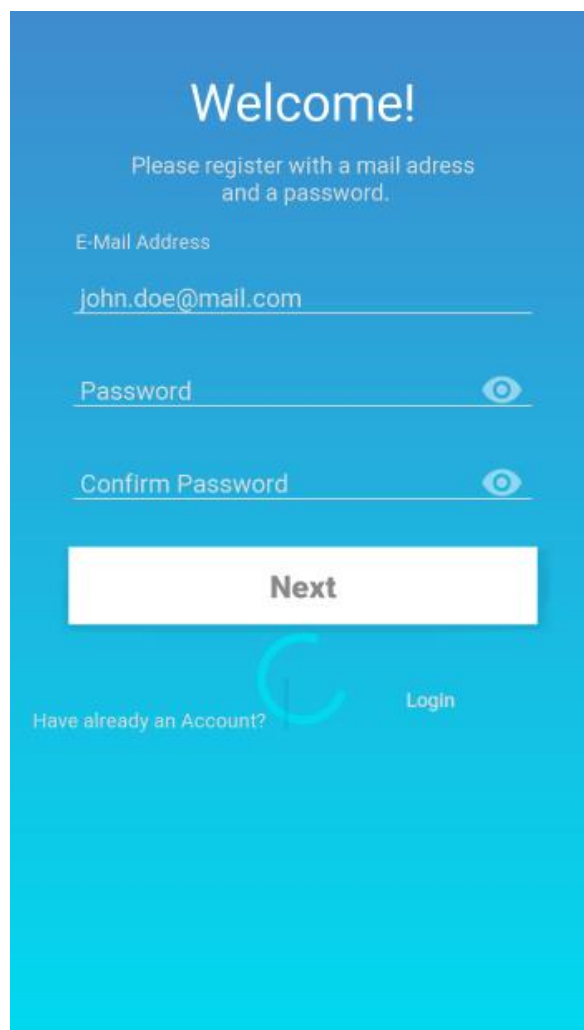
Terminanfragen konnten leider zeitlich nicht mehr implementiert werden. Deshalb wurde es durch eine einfache Kontaktaufnahme ersetzt, die vom Caretaker akzeptiert werden kann. Ein Patient bekommt nun eine Übersicht aller Caretaker angezeigt.

Projektplanung

Die Projektplanung war vorwiegend rudimentär vorhanden. Grund hierfür war die Zusammenarbeit mit der ucura GmbH. Da die Schnittstellen zum Großteil nicht passend für unsere Umsetzung beziehungsweise teilweise noch fehlerhaft waren gestaltete es sich schwer einen konkreten Zeitplan zu erstellen. Doch durch den direkten Kontakt mit diversen Mitarbeitern konnte dies bewerkstelligt werden. Durch diese Abhängigkeiten wurde allerdings kein konkreter Projektplan erstellt werden.

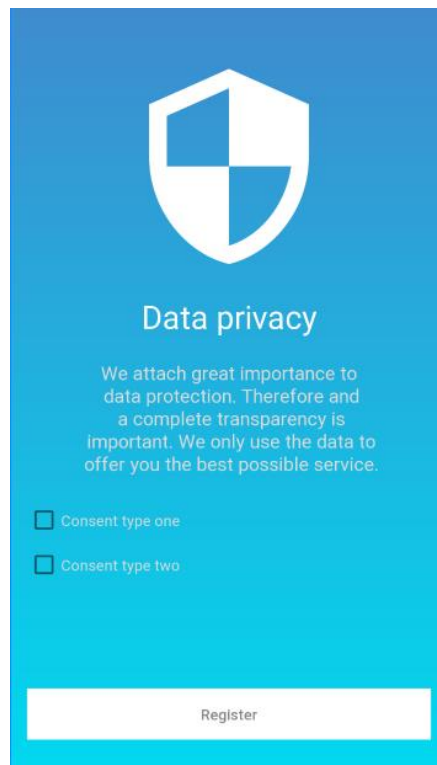
Design

In den folgenden Bildern wird der Workflow der App dargestellt:



The image shows a registration screen for an application. The background is a solid blue color. At the top, the word "Welcome!" is written in a large, white, sans-serif font. Below it, in a smaller white font, is the instruction "Please register with a mail address and a password." There are three input fields: "E-Mail Address" with the text "john.doe@mail.com", "Password", and "Confirm Password". Each input field has a white underline and a small white eye icon to its right. Below the input fields is a white rectangular button with the word "Next" in a bold, black, sans-serif font. At the bottom of the screen, there is a white circular loading spinner. To the left of the spinner is the text "Have already an Account?" and to the right is the text "Login".

Nach dem erstmaligen Starten der App wird der Registrierungsbildschirm gestartet. Hier kann sich der Pfleger mit E-Mail und Passwort einen Account erstellen.

A registration screen with a blue gradient background. At the top is a white shield icon with a blue cross. Below it, the text "Data privacy" is centered. A paragraph follows: "We attach great importance to data protection. Therefore and a complete transparency is important. We only use the data to offer you the best possible service." Below this are two unchecked checkboxes labeled "Consent type one" and "Consent type two". At the bottom is a white button with the text "Register".

Data privacy

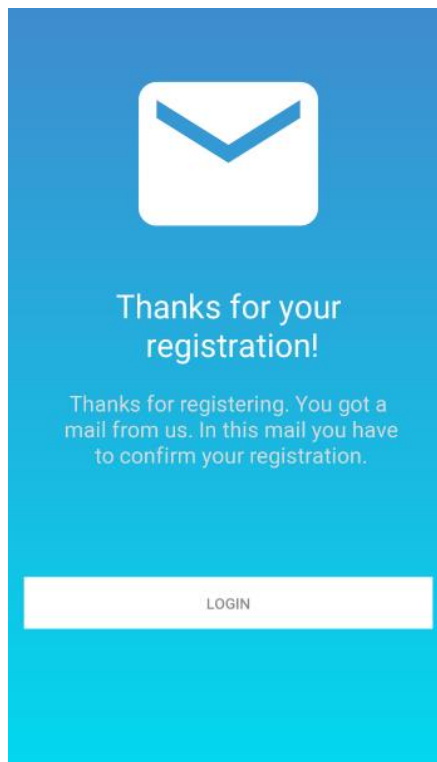
We attach great importance to data protection. Therefore and a complete transparency is important. We only use the data to offer you the best possible service.

☐ Consent type one

☐ Consent type two

Register

Um den Vorgang der Accounterstellung abzuschließen müssen noch zwei Einverständnisse gegeben werden.

A screen with a blue gradient background. At the top is a white envelope icon. Below it, the text "Thanks for your registration!" is centered. A paragraph follows: "Thanks for registering. You got a mail from us. In this mail you have to confirm your registration." At the bottom is a white button with the text "LOGIN".

Thanks for your registration!

Thanks for registering. You got a mail from us. In this mail you have to confirm your registration.

LOGIN


Nachdem man den Link in der Email bestätigt hat kann sich der Pfleger einloggen.

Welcome!

Login with your mail and password.

E-Mail Address

john.doe@mail.com

Password 


Login

Register

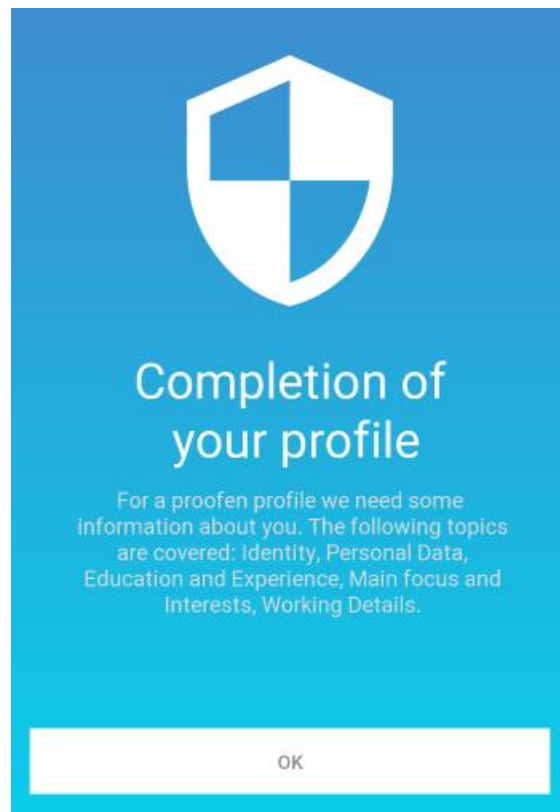
Durch diesen Login wird ein Accesstoken generiert welches für die folgenden Netzwerkaufrufe wie das Speichern der Adresse benötigt wird.

Thanks for your confirmation!

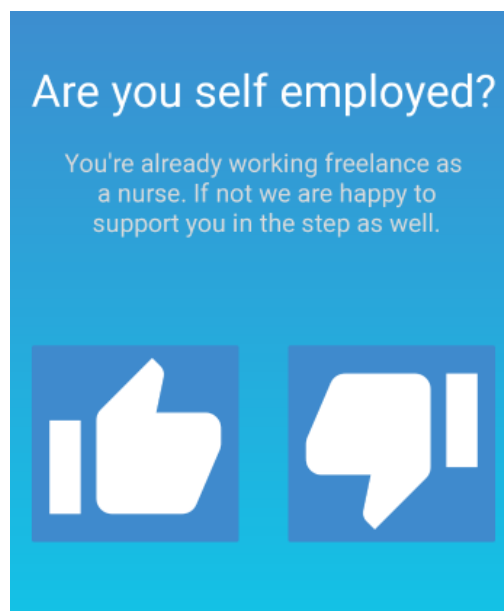
You're almost there. Now fill in the rest of your information so that you can start as a nurse.



COMPLETE YOUR PROFILE



Wenn die beiden oberen Screens durchlaufen wurden beginnt das wirkliche Onboarding.





Der Pfleger muss angeben, ob er bereits als Freelancer im Bereich der Pflege tätig ist.

Personal Data

Where are you from and where are you going to work?

Gender

☐  ☐ 

Name

Birthday YYYY-MM-DD

Phone number

Address

Street No.

Zip City

Country

Next

Danach werden die persönlichen Daten des Pflegers gespeichert.

Your Goals

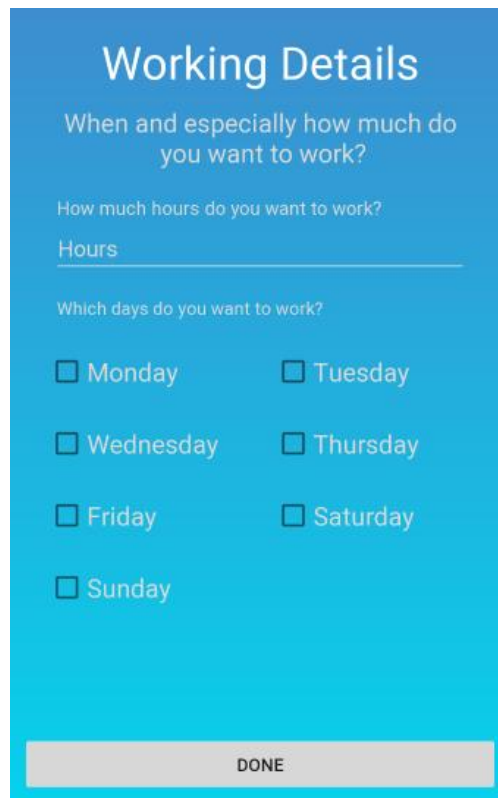
Fill out your goals so we can find the best patient matches for you.

How much do you want to earn (netto)?

What are your interests?

Next

Um im Anschluss seine persönlichen Ziele abzufragen.



Working Details

When and especially how much do you want to work?

How much hours do you want to work?

Hours

Which days do you want to work?

☐ Monday ☐ Tuesday

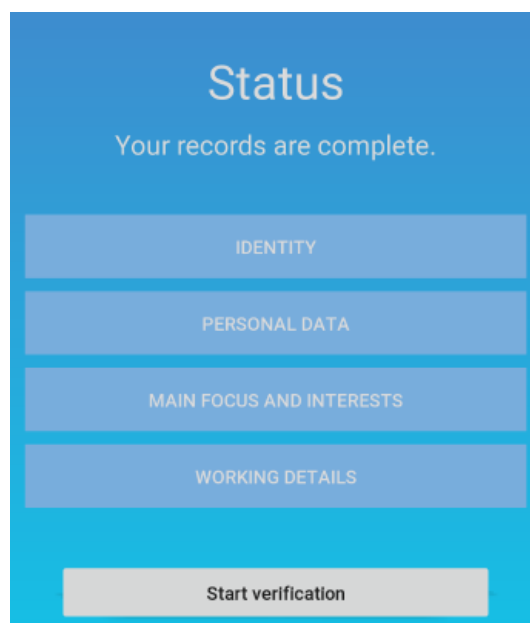
☐ Wednesday ☐ Thursday

☐ Friday ☐ Saturday

☐ Sunday

DONE

Zudem muss der Pfleger Angaben zu seinen Arbeitszeiten geben.



Status

Your records are complete.

IDENTITY

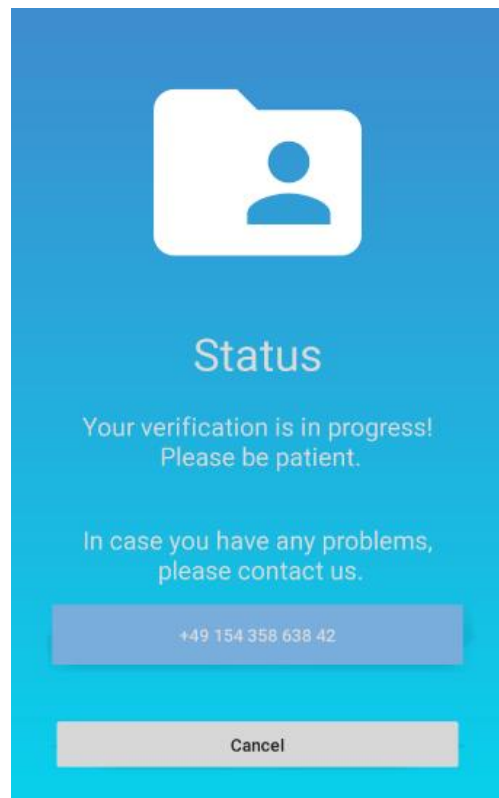
PERSONAL DATA

MAIN FOCUS AND INTERESTS

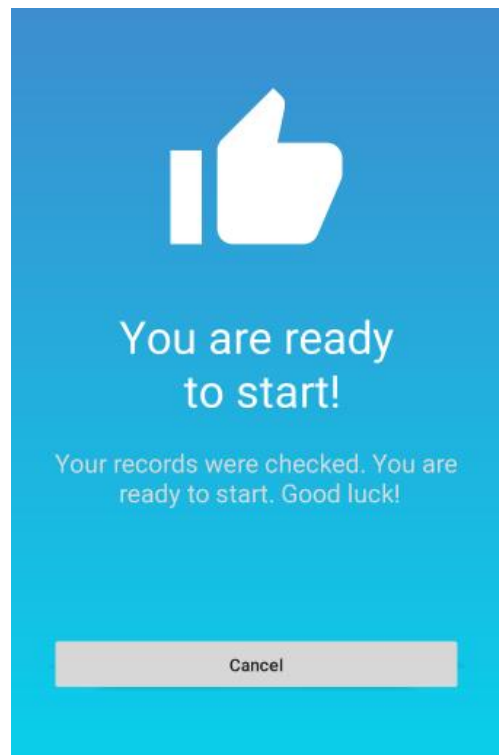
WORKING DETAILS

Start verification

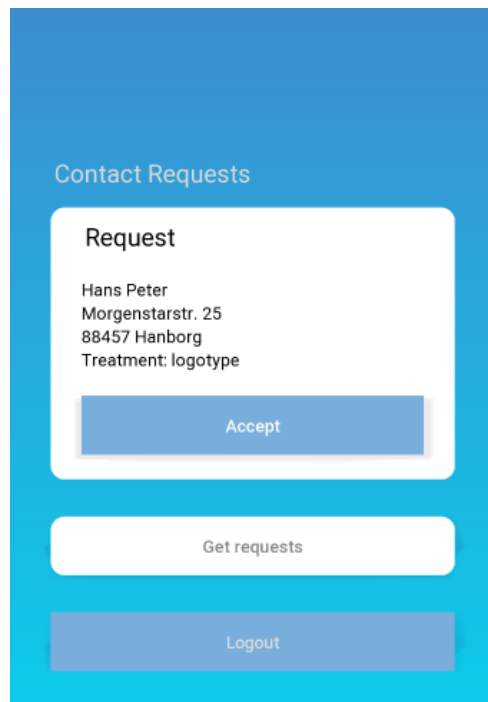
Nun wird dem Pfleger noch einmal eine Zusammenfassung der durchlaufenen Prozesse angezeigt und er kann die Verifikation starten.



Nach dem Starten der Verifikation wird der Pfleger aufgefordert die Verifikation abzuwarten.



Wurde die Verifikation erfolgreich abgeschlossen wird der Kunde darüber informiert und kann zum Dashboard gelangen.



Im Dashboard hat der Pfleger die Möglichkeit neue Anfragen abzurufen, diese zu akzeptieren und sich auszuloggen. Im Nachhinein kann hier auch noch das Termintracking implementiert werden.

Probleme

Bei der Umsetzung des Projekts sind einige Fallstricke und Probleme entstanden, auf welche im Folgenden eingegangen wird.

Prinzipiell verlief die Entwicklung anfangs sehr schleppend, da erst eine Menge Dokumentation zur Androidentwicklung gelesen werden musste, um die nötigen Datenklassen und ihre Abhängigkeiten zu verstehen.

Durch einen Guide, der Android Developer Seite, entwickelten wir ein sogenanntes NetworkFragment. Dieses sollte unsere REST API-Aufrufe behandeln. Im Laufe der Zeit hat sich auf Pfleger Seite allerdings ein Problem bezüglich des Handlings erwiesen. Um über die Repositoryklasse eine Synchronisierung zwischen Room und der Onlinedatenbank zu erhalten war es schlicht unmöglich mit dem NetworkFragment zu arbeiten. Das Debugging erwies sich ebenso als nicht hilfreich, da sämtlicher Netzwerkverkehr sowie die lokalen Datenbankaufrufe in einem asynchronen Thread stattfinden müssen, um den Mainthread nicht zu blockieren. Somit haben wir nach einiger Zeit Retrofit als geeignete Alternative gefunden.

Ebenfalls als sehr umständlich hat sich das Design der Views gestaltet. Die Dokumentation ist nicht sehr aufschlussreich und viele Funktionalitäten

verstecken sich hinter nichtssagenden Attributen. Somit war selbst das Einbinden eines einfachen Buttons im Textfeld, um die Sichtbarkeit des Passworts zu regeln, eine Herausforderung.

Als Fazit lässt sich sagen, dass die Entwicklung für die Androidplattform um einiges komplexer ist als beispielsweise die Standardentwicklung für PC-Systeme. Jedoch war es eine interessante Erfahrung eine neue Plattform kennenzulernen und somit neue Erkenntnisse zu gewinnen und umzusetzen. Jedoch wird es höchstwahrscheinlich unser letztes Androidprojekt gewesen sein.