4.12.2019

RaviAnand Mohabir

BBBaden

IPERKA M183

Projektdokumentation zum Modul 183 «Mobile-Applikation realisieren»

Inhalt

[1 Changelog 3](#_Toc29706982)

[2 Einleitung 4](#_Toc29706983)

[2.1 Generelle Anforderungen 4](#_Toc29706984)

[3 Informieren 5](#_Toc29706985)

[3.1 Anforderungsanalyse 5](#_Toc29706986)

[3.2 Plattformen 5](#_Toc29706987)

[3.2.1 Android 5](#_Toc29706988)

[3.2.2 iOS 5](#_Toc29706989)

[3.3 App-Frameworks 6](#_Toc29706990)

[3.3.1 Codename One 6](#_Toc29706991)

[3.3.2 Xamarin 6](#_Toc29706992)

[3.3.3 Nativescript 6](#_Toc29706993)

[3.3.4 Android Java/Kotlin 6](#_Toc29706994)

[3.4 Web APIs 7](#_Toc29706995)

[3.4.1 Google Firebase 7](#_Toc29706996)

[3.4.2 Myjson 7](#_Toc29706997)

[3.4.3 Schweizer Transport API 8](#_Toc29706998)

[3.5 Datenspeicherung 8](#_Toc29706999)

[3.5.1 Offline 8](#_Toc29707000)

[3.5.2 Online 8](#_Toc29707001)

[3.6 Sensoren 8](#_Toc29707002)

[3.6.1 Kamera 8](#_Toc29707003)

[3.6.2 GPS 8](#_Toc29707004)

[3.6.3 Gyroskop 8](#_Toc29707005)

[4 Planen 9](#_Toc29707006)

[4.1 Projektpool 9](#_Toc29707007)

[4.2 Grundkonzept 9](#_Toc29707008)

[4.2.1 Use-Case Diagramm 10](#_Toc29707009)

[4.3 Storyboard 11](#_Toc29707010)

[4.4 Anpassung an verschiedene Bildschirmformate 11](#_Toc29707011)

[4.5 Aufgabenteilung 11](#_Toc29707012)

[4.6 Zeitplan 0](#_Toc29707013)

[5 Entscheiden 0](#_Toc29707014)

[5.1 App Framework 0](#_Toc29707015)

[5.2 Datenspeicherung 0](#_Toc29707016)

[5.3 Web-Schnittstelle 0](#_Toc29707017)

[5.4 Navigation 0](#_Toc29707018)

[6 Realisieren 1](#_Toc29707019)

[7 Kontrollieren 2](#_Toc29707020)

[7.1 Testen 2](#_Toc29707021)

[7.1.1 QR-Codes 2](#_Toc29707022)

[8 Auswerten 3](#_Toc29707023)

[8.1 Reflexion 3](#_Toc29707024)

[9 Quellen 4](#_Toc29707025)

[9.1 Offiziell 4](#_Toc29707026)

[9.2 Artikel & Foren 4](#_Toc29707027)

[9.2.1 Android UI 4](#_Toc29707028)

[9.2.2 Google Firebase 5](#_Toc29707029)

[10 Anhang 6](#_Toc29707030)

# Changelog

|  |  |  |
| --- | --- | --- |
| Datum | Was | Wer |
| 01.12.2019 | Dokument mit Grundgerüst erstellt | RaviAnand Mohabir |
| 04.12.2019 | Informierungsphase der Dokumentation angefangen, Grundkonzept erstellt | RaviAnand Mohabir |
| 05.12.2019 | Informierungsphase der Dokumentation fortgesetzt | RaviAnand Mohabir |
| 07.12.2019 | Informierungsphase weitergeführt | RaviAnand Mohabir |
| 11.12.2019 | Planungsphase fortgesetzt, Zeitplan und Aufgabenteilung gemacht | RaviAnand Mohabir |
| 20.12.2019 | Quellen hinzugefügt | RaviAnand Mohabir |
| 28.12.2019 | Quellen hinzugefügt | RaviAnand Mohabir |
| 01.01.2020 | Quellen hinzugefügt | RaviAnand Mohabir |
| 08.01.2020 | Quellen hinzugefügt | RaviAnand Mohabir |

# Einleitung

Für die Leistungsbeurteilung zum Modul 183 muss eine Mobile-Applikation erstellt werden, welche aus einem eigenen Konzept stammt oder vom Projektpool gewählt wurde. Es sind einige generelle Anforderungen gegeben, welche in jedem Fall gedeckt werden sollten und das Projekt, wenn es nicht aus dem Projektpool stammt muss vom Vorgesetzten besprochen und freigegeben werden.

Das Projektpool kann im Anhang gefunden werden.

## Generelle Anforderungen

Folgende Elemente **müssen** in allen Applikationen vorhanden sein:

* Ein Eingabeformular.
* Persistenz von Daten/Einstellungen (online oder offline).
* Mehr als ein Screen.
* Die Verwendung mindestens eines Sensors/Eingabe-Elementes, wie beispielsweise Kamera, GPS, Tonaufnahme, Beschleunigungssensor, Gyroskop, ... ist erwünscht.
* Die Verwendung mindestens einer Netzwerkfunktionalität (Authentifizierung, Webservice, Analytics, ...).
* Anpassung an verschiedene Bildschirmgrössen und -ausrichtungen.

# Informieren

Um die Anforderungen zu decken gibt es verschiedene Frameworks, Web APIs und Datenspeicher-Systeme mit Ihren Vor- und Nachteilen. Die App sollte mindestens auf einer gängigen mobilen Plattform wie Android oder iOS ausgeführt werden können und Grundregeln der Platform wie UI-Design einhalten damit der Benutzer schnell die App in Bedienung nehmen kann.

## Anforderungsanalyse

Zuerst wurde eine Anforderungsanalyse erstellt, welche aus der Situationsbeschreibung sowie der Aufgabenstellung der LB stammt, um zu sehen was in der App vorhanden sein muss und danach welche Tools, Schnittstellen und Frameworks eingesetzt werden können um diese zu decken.

|  |  |  |
| --- | --- | --- |
| Nr. | Name | Beschreibung |
| 1 | Eingabeformular | Mindestens ein Eingabeformular ist in der App vorhanden welche Benutzereingaben aufnimmt und verarbeitet. |
| 2 | Daten-Persistenz | Daten wie Einstellungen oder Ähnliches werden entweder online oder offline gespeichert und wieder geladen. |
| 3 | Multi-Screen | Die App besteht aus mehreren Screens/Aktivitäten, durch welche der Benutzer navigiert. |
| 4 | Sensor | Es wird mindestens ein Sensor wie beispielsweise Kamera, GPS, Mikrofon, Beschleunigungssensor, Gyroskop etc. eingesetzt. |
| 5 | Netzwerkfunktionalität | Es wird mindestens eines Web-API verwendet. |
| 6 | Bildschirmgrössen | Die App passt sich an verschiedene Bildschirmgrössen (Tablets/Handys) und -ausrichtungen an. |

## Plattformen

Mobile-Applikationen können entweder spezifisch für eine Plattform entwickelt werden oder mit Frameworks (s. [App-Frameworks](#_App-Frameworks)) für mehrere Plattformen kompatibel gemacht werden. Wenn man sich auf eine Plattform fokussiert, kann man native Features realisieren, welche möglicherweise auf anderen Plattformen nicht zur Verfügung stehen (bspw. Apple Pay, Google ML Kits) und somit die App für das Benutzererlebnis auf dieser Plattform optimieren. Die zwei gängigsten Smartphone Betriebssysteme sind Android und iOS.

### Android

Android ist das quelloffene Betriebssystem für Smartphones und Tablets von Google. Neben der quelloffenen Version gibt es eine angepasste von Google welche auf ihren eigenen Smartphones wie die Pixel Reihe eingesetzt wird und Zugriff auf exklusive Features und Apps erlaubt wie die Google Camera Applikation, eines der marktführenden Kamera Applikationen für Smartphones im Allgemeinen welches nur performant und optimiert auf den Pixel Geräten von Google läuft. Handyhersteller wie Samsung und Huawei verwenden die quelloffene Version welche etwas langsamer Erneuerungen erhält, jedoch von ihnen stark angepasst werden kann um auf ihren Geräten besser zu laufen.

### iOS

Für Apple Smartphones gibt es iOS. Neuere Apple Tablets verwenden iPadOS und haben somit erweiterte Funktionalitäten welche Programmierer ausnützen können, wie beispielsweise der Einsatz von externen Festplatten und dem Apple Pen. Die Entwicklung von Apps für das Apple Ökosystem kann teurer und komplizierter sein wenn man mit dem nativen System arbeiten möchte braucht man ein MacOS Gerät und muss Apples eigene Programmiersprache Swift benutzen, als Alternative dazu stehen Frameworks wie Codename One und Xamarin zur Verfügung welche auch auf Windows mit C# und Java verwendet werden können.

## App-Frameworks

Die App dient als Schnittstelle zwischen dem Benutzer und dem Programmierer sowie den Online Anbietern von APIs und dergleichen. Verschiedene Frameworks erlauben dem Programmierer mit einer ausgewählten Sprache und Umgebung die App zu entwickeln was hauptsächlich GUI und Geschäftslogik beinhaltet. Gängige Sprachen wie Java, C# und Web-Sprachen (JavaScript, HTML, CSS) können in Kombination mit einem Framework wie Codename One, Xamarin oder Nativescript verwendet werden um die App zu realisieren und für die Plattform lauffähig zu machen.

### Codename One

Codename ist das App-Framework, welches die Schule zur Benützung vorschlägt. Der Service kompiliert Java bzw. Kotlin Code zur nativen Umgebung des jeweiligen Betriebssystems. Das heisst dass der grösste Teil der Logik nur einmal programmiert werden muss und so schnell Apps für verschiedene Plattformen entwickelt werden können. Auch grosse Teile des UIs können plattformübergreifend erstellt werden.

Quelle: <https://www.codenameone.com/>

### Xamarin

Xamarin ist sehr vergleichbar mit Codename One. Es ist ein C# Framework welchen Programmierern erlaubt die Businesslogik nur einmal zu programmieren und auch grosse Teile der Benutzer-Interaktion. Im Unterschied zu Codename One muss das UI für jede Plattform individuell gestaltet werden damit UI Komponenten des nativen Systems gebraucht werden können und es unterstütz auch die Verwendung von weiteren nativen Komponenten um Apps mit mehr Gestaltungsfreiheit für die verschiedenen Betriebssysteme zu machen.

Quelle: <https://dotnet.microsoft.com/apps/xamarin>

### Nativescript

Nativescript ist ein quelloffenes App-Framework welches zusammen mit UI Frameworks wie Angular, Vue.js oder React.js die Erstellung von Plattformübergreifenden Applikationen für Web-Entwickler stark erleichtert. Das GUI wird jeweils mit einem der UI Frameworks erstellt und die Logik mit JavaScript sowie ihren Super- und Subsets wie zum Beispiel JSX und Typescript.

Quelle: <https://www.nativescript.org/>

### Android Java/Kotlin

Mit Java können native Apps für die Android Plattform erstellt werden was vollen Zugriff auf plattformspezifische Funktionen und UI Frameworks/Elementen erlaubt und somit für die Android Plattform sehr effiziente und standardisierte Apps entwickelt werden können. Die Logik kann entweder in Java geschrieben werden oder mit Kotlin welches dann von einem Compiler zu Java-Bytecode umgeschrieben wird. Vorteil an der Arbeit mit Kotlin ist die vereinfachte Syntax sowie Erweiterungen durch Bibliotheken von Google und Drittanbietern. Mit der Android Studio Entwicklungsumgebung können dann auch Emulatoren generiert werden um die App auf dem PC zu testen sowie das native UI Tool eingesetzt werden.

Quellen: <https://developer.android.com/guide>, <https://developer.android.com/kotlin>, <https://developer.android.com/studio>

## Web APIs

Um die fünfte Anforderung zu decken muss mindestens eine Web API eingesetzt werden. Web APIs ermöglichen entweder über das REST oder dem SOAP Protokoll die Verbindung zu einem Onlinedienst durch die App. Viele Webseiten bieten neben ihrer Hauptseite eine API damit angepasste Benutzerschnittstellen wie eine Desktop/Mobile-Applikation oder Automationsfunktionalitäten realisiert werden können. Der Datenaustausch passiert meistens über dem HTTPS Protokoll in Form von JSON und XML Dokumenten. Da APIs viel Bandbreite benötigen kostet die Nutzung oft und zum Teil muss man beim Anbieter ein Konto besitzen mit welchem man sich bei jedem Austausch authentifiziert.

Quelle: <https://en.wikipedia.org/wiki/Application_programming_interface>

### Google Firebase

Google Firebase ist eine riesige Platform welche von Google angeboten wird welche über die API an verschiedene Applikationen angebunden werden kann. Es bietet verschiedene Funktionalitäten, an welche die Netzwerkkommunikation einer App und die Anbindung an einen Datenspeicher stark vereinfacht.

Die Hauptfunktionalität von Firebase besteht aus Analytics. Durch eine einfache Einbettung in die Applikation verfolgt Firebase die Nutzung der App von Benutzern und zeigt auf der Onlineumgebung dem Systemadministrator an wie die App verwendet wird. Dieses System ist vergleichbar mit den Google Analytics welches hauptsächlich in Webseiten eingesetzt wird.

Ein weiterer Komponent ist «Cloud Firestore» welche die «Realtime Database» aus vorherigen Versionen ersetzt. Es ist ein Dokumentenbasierter Onlinespeicher welcher aus Collections und in welche Dokumente gespeichert werden. Daten werden in Key-Value Paare aufbewahrt und können gängige Datenformen annehmen. Im Weiteren können Security Rules umgesetzt werden, welche Benutzer der Applikation beschränkt Zugriff auf die Datenbank erlauben können.

Neben Cloud Firestore gibt es noch das Authentication-System mit welchem man schnell Benutzerauthentifikation über gängige Anbieter (z.B. Facebook, Google, Apple) in der Applikation implementieren kann. Wenn man keinen Anbieter verwenden möchte unterstützt Firebase auch normal E-Mail/Passwort Paare, die in der Google Datenbank gespeichert sind.

Mit Firebase Cloud Functions kann man JavaScript Code auf die Plattform hochladen welche durch die API per GET-, POST-, DELETE- oder PUT- aufgerufen werden können und sehr eng mit der Datenbank sowie weiteren Komponenten von Firebase verbunden sind. Die Cloud Functions können auch auf Änderungen in der Datenbank hören (Events) und darauf reagieren.

Weitere Komponenten beinhalten unter anderem das ML Kit, Hosting und Cloud Storage neben Entwicklungstools wie die Crashlytics welche dem Systemadministrator Details über Abstürze der Applikation bei Benutzern anzeigt.

Quelle: <https://firebase.google.com/>

### Myjson

Myjson ist ein Online-JSON-Datenspeicher, welcher über eine sehr einfache REST API von Web- und mobilen Applikationen eingesetzt werden kann. Ein JSON Dokument wird auf der Seite oder über die API erstellt, danach können per GET und PUT HTTPS-Requests Daten angefragt werden beziehungsweise das Dokument verändert werden. Der Service ist momentan in einem Beta Stadium und ermöglicht keine Sicherung der Daten weshalb es nur limitiert eingesetzt werden kann.

Quelle: <http://myjson.com/about>

### Schweizer Transport API

[Opendata.ch](https://Opendata.ch) ermöglicht es Benutzern, die von [search.ch](https://search.ch) angebotenen Fahrplaninformationen über die Transport API abzurufen. Daten können jedoch nur limitiert abgerufen werden da [timetable.search.ch](https://timetable.search.ch) ein Ratelimit besitzt und Daten werden über REST in Form von JSON Key-Value Paaren zurückgegeben.

Quelle: <https://transport.opendata.ch/>

## Datenspeicherung

### Offline

### Online

## Sensoren

### Kamera

### GPS

### Gyroskop

# Planen

Nun kann mit Hilfe des vorherig erarbeiteten Wissens auf eine Grundidee fokussiert werden. Was die App grundsätzlich können sollte sowie Funktionen überlegen, welche die Grundanforderungen decken und auch innerhalb der vorgegebenen Zeit implementiert sowie dokumentiert werden können.

## Projektpool

Als Hilfestellung wurde von der Schule ein Projektpool mitgegeben (s. Anhang) welches einige App-Ideen enthielt, die man erweitern kann um alle Anforderungen zu decken und eine individuelle App zu erstellen. In diesem Projekt werden die Ideen aus dem Projektpool nicht verwendet.

## Grundkonzept

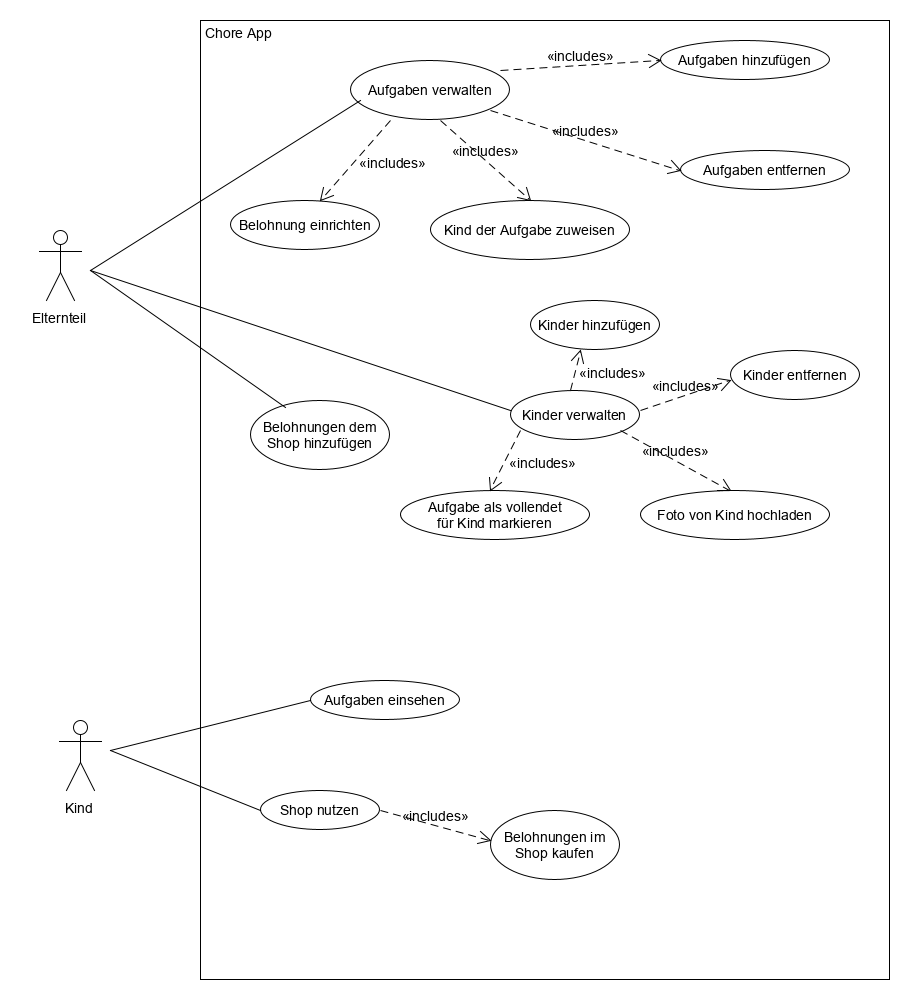
Die Grundidee meiner App ist ein Ökosystem für den Familienhaushalt. Die App sollte für Eltern sowie ihre Kinder zugänglich sein und die Haushaltsaufgaben auf eine übersichtliche Art aufzeigen, sowie Kinder für erledigte Aufgaben belohnen und Eltern die Möglichkeit geben, Aufgaben einzurichten sowie Belohnungen auszugeben, die an die Aufgabe angepasst sind.

Somit sollen Haushaltsaufgaben etwas Spass machen und dank Cloud-Funktionalität auf dem Handy angezeigt und nicht vergessen werden.

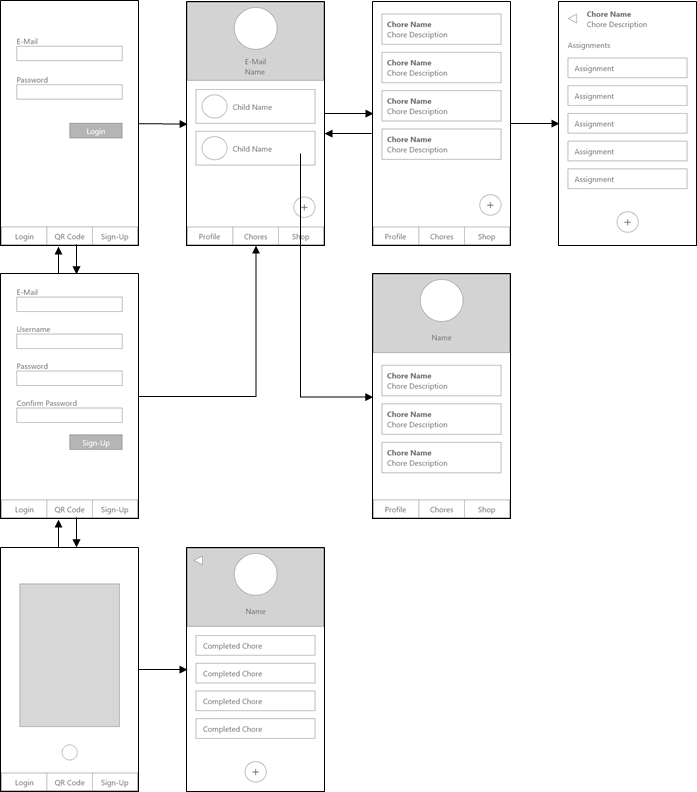
Damit Eltern vollen Zugriff auf ihre «Familien-Datenbank» haben müssen sie sich mit einen der Google Firebase Authentifikationsmethoden anmelden und Kinder können sich mit einem QR-Code anmelden welches über die API ihnen limitierten Zugriff auf die Datenbanken erlaubt und sobald Aufgaben erledigt wurden wird ein QR-Code generiert welche Eltern einscannen müssen um die Aufgabe als erledigt zu markieren.

Da die Firebase Datenbank und Authentifikationssystem eingesetzt werden sowie ein QR-Scanner sind die Anforderungen 2, 4 und 5 gedeckt und da die App verschiedene Bildschirm für das Anmelden, die Aufgabenübersicht und das Erstellen von Aufgaben hat ist auch die Anforderung 3 vorgegeben und das Anmeldeformular stellt die Anforderung 1 dar. Die App sollte auf möglichst vielen Geräten laufen, besonders weil sie von Kindern benutzt werden sollte welche möglicherweise ältere Geräte besitzen oder nur ein Tablet zur Verfügung haben und deswegen sollte per Anforderung 6 die App ihre Bildschirme auf das Gerät anpassen und so viel wie nötig aber so wenig wie möglich anzeigen und die Benutzereingaben für möglichst viele Benutzer anpassbar sein.

### Use-Case Diagramm



## Storyboard



## Anpassung an verschiedene Bildschirmformate

## Aufgabenteilung

|  |  |
| --- | --- |
| Nr. | Aufgabe |
| **1** | **Informieren** |
| 1.1 | Datenspeicherungsmöglichkeiten recherchieren |
| 1.2 | App-Frameworks recherchieren |
| **2** | **Planen** |
| 2.1 | User-Stories machen |
| 2.2 | Use-Case Diagramme erstellen |
| 2.3 | Design konzipieren |
| **3** | **Entscheiden** |
| 3.1 | Datenspeicherungsmöglichkeit auswählen |
| 3.2 | App-Framework auswählen |
| **4** | **Realisieren** |
| 4.1 | App-Projekt erstellen |
| 4.2 | Firebase Projekt erstellen |
| 4.3 | Datenbank in Firebase aufsetzen |
| 4.4 | Design in App umsetzen |
| 4.5 | Firebase in App einbetten |
| 4.6 | Firebase ML Kit in App einbetten |
| 4.7 | Business-Logik in App programmieren |
| 4.8 | App und Firebase Projekt absichern |
| **5** | **Kontrollieren** |
| 5.1 | App testen |
| 5.2 | Firebase Security Rules testen |
| **6** | **Auswerten** |
| 6.1 | Reflexion schreiben |

## Zeitplan

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Aufgabe / Datum** | **30.10.2019** | **06.11.2019** | **13.11.2019** | **20.11.2019** | **27.11.2019** | **04.12.2019** | **11.12.2019** | **18.12.2019** | **25.12.2019** | **01.01.2019** |
| **1 Informieren** |  |  |  |  |  |  |  |  |  |  |
| 1.1 Datenspeicherungsmöglichkeiten recherchieren |  |  |  |  |  |  |  |  |  |  |
| 1.2 App-Frameworks recherchieren |  |  |  |  |  |  |  |  |  |  |
| **2 Planen** |  |  |  |  |  |  |  |  |  |  |
| 2.1 User-Stories machen |  |  |  |  |  |  |  |  |  |  |
| 2.2 Use-Case Diagramme erstellen |  |  |  |  |  |  |  |  |  |  |
| 2.3 Design konzipieren |  |  |  |  |  |  |  |  |  |  |
| **3 Entscheiden** |  |  |  |  |  |  |  |  |  |  |
| 3.1 Datenspeicherungsmöglichkeit auswählen |  |  |  |  |  |  |  |  |  |  |
| 3.2 App-Framework auswählen |  |  |  |  |  |  |  |  |  |  |
| **4 Realisieren** |  |  |  |  |  |  |  |  |  |  |
| 4.1 App-Projekt erstellen |  |  |  |  |  |  |  |  |  |  |
| 4.2 Firebase Projekt erstellen |  |  |  |  |  |  |  |  |  |  |
| 4.3 Datenbank in Firebase aufsetzen |  |  |  |  |  |  |  |  |  |  |
| 4.4 Design in App umsetzen |  |  |  |  |  |  |  |  |  |  |
| 4.5 Firebase in App umsetzen |  |  |  |  |  |  |  |  |  |  |
| 4.6 Firebase ML Kit in App einbetten |  |  |  |  |  |  |  |  |  |  |
| 4.7 Business-Logik in App programmieren |  |  |  |  |  |  |  |  |  |  |
| 4.8 App und Firebase Projekt absichern |  |  |  |  |  |  |  |  |  |  |
| **5 Kontrollieren** |  |  |  |  |  |  |  |  |  |  |
| 5.1 App testen |  |  |  |  |  |  |  |  |  |  |
| 5.2 Firebase Security Rules testen |  |  |  |  |  |  |  |  |  |  |
| **6 Auswerten** |  |  |  |  |  |  |  |  |  |  |
| 6.1 Reflexion schreiben |  |  |  |  |  |  |  |  |  |  |

# Entscheiden

## App Framework

## Datenspeicherung

## Web-Schnittstelle

## Navigation

Für die Navigation innerhalb der App gibt es zwei populäre Möglichkeiten; Die Side-Navigation und Bottom-Navigation. Gängig bei Apps mit vielen Bildschirmen ist die Side-Navigation da sie viel mehr Platz zur Verfügung stellt und zusammenhängende Bildschirme gruppiert werden können. Die Bottom-Navigation ist jedoch besonders bei grösseren mobilen Geräten ergonomischer da man mit wenig Mühe mit ihr interagieren kann. Ein weiterer Vorteil der Bottom-Navigation ist, dass man immer sieht wo man ist und wo man hinkann.

Ich habe mich entschieden, die Bottom-Navigation einzusetzen, da die drei «Haupt-Bildschirme» (Login, Parent, Child; s. Storyboard) nicht mehr als drei Navigationspunkte benötigen. Wenn man als Parent ein Child oder Chore bearbeiten möchte, werden neue Aktivitäten geöffnet, welche die Navigation ausblenden und sobald man eingeloggt ist, ist die Login-Aktivität aufgelöst und wird nicht mehr benötigt bis sich der Benutzer ausloggt. Somit ist die App viel ergonomischer und da die Bottom-Navigation sehr kompakt ist, steht immer noch viel Platz für den Content zur Verfügung.

# Realisieren

# Kontrollieren

## Testen

### QR-Codes

Mit folgenden QR-Codes wurden die Anmeldefunktionen der Applikation getestet:

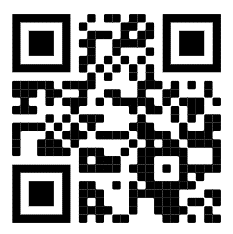
**parentuid:Gsu7QD32l5UajAzOWOTMLiVhtEl2**

[**moravrav@gmail.com**](mailto:moravrav@gmail.com)



**childuid:EEhtqfYn0q2ERtKMCxr0**

**Kind unter RaviAnand Mohabir**



**Weitere QR-Codes…**

# Auswerten

## Reflexion

# Quellen

## Offiziell

* [https://firebase.google.com/docs](https://firebase.google.com/docs/reference/android/com/google/firebase/functions/HttpsCallableReference.html#call(java.lang.Object))
* <https://developer.android.com/>
* <https://codelabs.developers.google.com/codelabs/camerax-getting-started/#7>
* <https://www.android.com/>
* <https://developer.apple.com/ios/>
* <https://kotlinlang.org/>
* <https://material.io/>
* <https://kotlinlang.org/docs/reference/>

## Artikel & Foren

* <https://www.varvet.com/blog/android-qr-code-reader-made-easy/>
* <https://www.c-sharpcorner.com/article/how-to-generate-qr-code-in-android/>
* <https://stackoverflow.com/questions/1898886/removing-an-activity-from-the-history-stack>
* <https://stackoverflow.com/questions/14785806/android-how-to-make-an-activity-return-results-to-the-activity-which-calls-it>
* <https://medium.com/@agrawalsuneet/typecheck-is-and-cast-as-in-kotlin-a2490c567a85>
* <https://blog.mindorks.com/how-to-create-a-singleton-class-in-kotlin>
* <https://stackoverflow.com/questions/2776116/how-do-i-dynamically-choose-which-activity-to-launch-when-opening-an-app>
* <https://stackoverflow.com/questions/5794506/android-clear-the-back-stack>
* <https://stackoverflow.com/questions/19874882/android-view-inflateexception-binary-xml-file-error-inflating-class-fragment>
* <https://stackoverflow.com/questions/40760625/how-to-check-permission-in-fragment>
* <https://stackoverflow.com/questions/53355786/kotlin-open-new-activity-inside-of-a-fragment>
* <https://www.baeldung.com/kotlin-regular-expressions>
* <https://superkotlin.com/kotlin-when-statement/>
* <https://stackoverflow.com/questions/45213706/kotlin-wait-function>
* <https://stackoverflow.com/questions/38157814/request-permissions-based-on-api-level>

### Android UI

* <https://www.journaldev.com/10024/android-recyclerview-android-cardview-example-tutorial>
* <https://medium.com/@stavro96/how-to-setup-android-navigation-component-with-drawer-88b833312661>
* <https://www.raywenderlich.com/1560485-android-recyclerview-tutorial-with-kotlin#toc-anchor-012>
* <https://stackoverflow.com/questions/30450330/cardview-click-state-not-working/30450813>
* <https://www.truiton.com/2015/03/android-cardview-example/>
* <https://stackoverflow.com/questions/44454797/pull-to-refresh-recyclerview-android>
* <https://stackoverflow.com/questions/432037/how-do-i-center-text-horizontally-and-vertically-in-a-textview>
* <https://stackoverflow.com/questions/2020882/how-to-change-progress-bars-progress-color-in-android>
* <https://www.journaldev.com/19911/android-material-design-button-style-design>
* <https://proandroiddev.com/some-tips-to-a-better-experience-working-with-androids-bottomnavigationview-2a224c687c1f>
* <https://guides.codepath.com/android/using-the-app-toolbar>
* <https://inducesmile.com/android-programming/how-to-change-bottom-navigation-view-selected-item-color-in-android/>
* <https://stackoverflow.com/questions/1598119/is-there-an-easy-way-to-add-a-border-to-the-top-and-bottom-of-an-android-view>
* <https://stackoverflow.com/questions/19238738/style-bottom-line-in-android>
* <https://medium.com/@hitherejoe/exploring-the-android-design-support-library-bottom-navigation-drawer-548de699e8e0>

### Google Firebase

* <https://medium.com/@khreniak/cloud-firestore-security-rules-basics-fac6b6bea18e>
* <https://stackoverflow.com/questions/50936424/throwing-new-functions-https-httpserror-on-firebase-cloud-function-rejects-as-in/51287274>
* <https://stackoverflow.com/questions/47876754/query-firestore-database-for-document-id>
* <https://stackoverflow.com/questions/53140913/querying-by-a-field-with-type-reference-in-firestore>
* <https://stackoverflow.com/questions/18125697/how-to-ignore-new-fields-for-an-object-model-with-firebase-1-0-2>
* <https://stackoverflow.com/questions/37368952/what-is-the-best-way-to-save-java-enums-using-firebase>

# Anhang