Fachhochschule Nordwestschweiz
Hochschule für Technik

Cloudbasiertes Praxisrufsystem

IP 5

27. Juli 2021

Abbildung 0.1: Titlebild

Studenten	Joshua Villing, Kevin Zellweger
Fachbetreuer	Daniel Jossen
Auftraggeberin	Daniel Jossen
Studiengang	Informatik
Hochschule	Hochschule für Technik

Zusammenfassung

Das Abstract ist eine Art Zusammenfassung des ganzen Dokuments. Es gibt einen Einblick in die Aufgabenstellung, wie diese umgesetzt wurde und welches Ergebnis erreicht wurde. Aus diesem Grund wird das Abstract immer ganz am Schluss der Arbeit verfasst. Es besteht aus einem zusammengehörenden Absatz und umfasst ungefähr 10 bis 20 Zeilen. Formeln, Referenzen oder andere Unterbrechungen haben im Text nichts zu suchen. Direkt unter dem Abstract folgt eine Liste von drei bis vier Stichworten/Keywords. Diese werden in alphabetischer Reihenfolge aufgelistet und beschreiben das Themengebiet der Arbeit.

Keywords: Anleitung, LaTeX, Thesis, Vorlage

Management Summary siehe PF-IK.

Vorwort

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Fakultativ, siehe PF-IK (URL) Balalbala some edits. [1]

Inhaltsverzeichnis

1 Einleitung **1**

2 Vorgehensweise **2**

2.1 Stakeholder 2

2.2 Projektplan 3

2.3 Organisation 4

3 Anforderungen **5**

3.1 User Stories 5

3.2 Technisch 5

3.3 Features 6

4 Konzept **7**

4.1 Systemarchitektur 7

4.2 Mobile Client 9

4.2.1 Framework Grundlagen 9

4.2.2 Anwendung 9

4.2.3 User Interface 12

4.3 Cloud Service 13

4.3.1 Architektur 13

4.3.2 Domänenmodell 13

4.3.3 Laufzeitmodell 13

4.4 Admin UI 14

4.5 Proof Of Concept 15

5 Evaluation Technologien **16**

5.1 Mobile Client Evaluation 16

5.2 Cloud Service 16

5.3 Betrieb und Platform 16

6 Umsetzung **17**

6.1 Authentifizierung 17

7 Schluss **18**

Abbildungsverzeichnis **21**

8 Anhang **21**

8.1 Benutzerhandbuch 21

8.2 Betriebshandbuch 21

8.3 Entwicklerdokumentation 21

8.4 Ehrlichkeitserklärung 21

1 Einleitung

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Einleitungsbeispiele siehe PF-IK (URL)

2 Vorgehensweise

2.1 Stakeholder

Am Projekt IP5 Cloudbasiertes Praxisrufsystem sind folgende drei Stakeholder beteiligt.

Prof. Daniel Jossen

- Rolle: Auftraggeber und Betreuer
- Kontakt: daniel.jossen@fhnw.ch

Joshua Villing

- Rolle: Student
- Kontakt: joshua.villing@students.fhnw.ch

Kevin Zellweger

- Rolle: Student
- Kontakt: kevin.zellweger@students.fhnw.ch

2.2 Projektplan

Übersicht

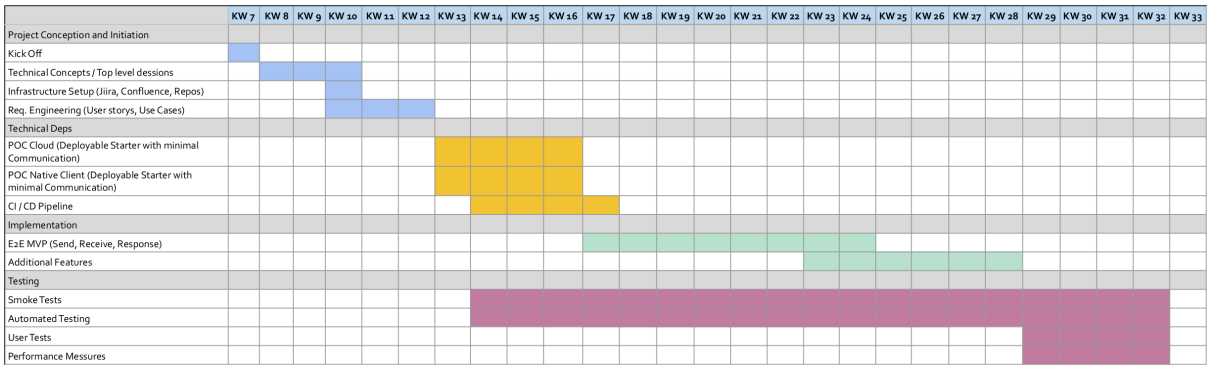


Abbildung 2.1: Projektplan

Milestones

Milestones

POC: Mobile Client ->Cloud Nachricht schicken und etwas persistieren

Wahrscheinlich über HTTP / Rest

POC: Cloud ->Mobile Client Nachricht schicken und etwas anzeigen

Wahrscheinlich über Message Broke

Versenden mit hinterlegter Konfiguration

Konfigurierte Notification Types

1:N Versenden, emfpangen und konfigurieren

Setup Wizard (Neu oder z.B. wie Praxiszimmer XY)

Voice to Speech

Voice Chat 1:n (Out Of Scope?)

2.3 Organisation

Kommunikation

Das Projekt IP5 Cloudbasiertes Praxisrufsystem wurde im FS21 gestartet. Die Organisation und Kommunikation des Projektes mussten dementsprechend für die Einschränkungen wegen Corona angepasst werden. Um sicherzustellen, dass die Kommunikation über die gesamte Projektdauer funktionieren kann, haben wir uns deshalb von Anfang an entschieden die Kommunikation über Remote- und Online Tools zu organisieren. Für Besprechungen und Planungen wurde Microsoft Teams gewählt. Die entsprechende Infrastruktur wurde von der FHNW zur Verfügung gestellt.

Dokumentation

Der Bericht wurde mit LateX und zusammen mit dem Quellcode verwaltet. Kurze Besprechungen, Notizen und interne Dokumentation erfolgten über ein geteiltes One Note Notizbuch.

Sämtliche Diagramme, Mockups und Skizzen wurden direkt in den Tools verwaltet, die zur Erstellung gebraucht wurden. Zum Schluss wurden alle für den Bericht relevanten Darstellungen exportiert und in den Bericht integriert.

Quellcodeverwaltung

Sämtlicher Quellcode der im Rahmen des Projektes entsteht, wurde mit Git verwaltet. Der Quellcode ist für Berechtigte unter dem Projekt IP5-Cloudbasiertes-Praxisrufsystem auf github.com einsehbar. (Referenz https://github.com/IP5-Cloudbasiertes-Praxisrufsystem). Berechtigungen können bei Joshua Villing oder Kevin Zellweger angefordert werden.

- IP5-praxis-mobile-client
- IP5-praxis-cloud-service
- IP5-praxis-admin-ui
- IP5-praxis-documentation

Tools und Werkzeuge

- draw.io
- moqus.com
- Visual Studio Code
- IntelliJ
- Git
- github.com

3 Anforderungen

3.1 User Stories

Projekt Scope

Nummeriert mit U01-Uxx - High Level - Fachliche Anforderung

Benutzer

Id	Anforderung	Features
U01	ALS Benutzer WILL ich relevante Anfragen versenden können, DAMIT ich über Probleme und Bedarf notifizieren kann.	F0x
U02	ALS Benutzer WILL ich relevante Notifikationen empfangen können, DAMIT ich über Probleme und Bedarf notifiziert werde.	F0x
U03	ALS Benutzer WILL will nur Notifikationen sehen die für mich relevant sind, DAMIT nicht unnötig belästigt werde.	F0x
U04	ALS Benutzer WILL ich über eingehende Notifikationen aufmerksam gemacht werden, DAMIT ich Notifikationen nicht übersehe.	F0x
U05	ALS Benutzer WILL ich sehen welche Notifikationen ich verpasst habe, DAMIT ich vergessenes nachholen kann.	F0x

Admin

Id	Anforderung	Features
U10	ALS Administrator WILL mehrere Geräte verwalten kann DAMIT jedes Gerät für Verwendungsort optimiert werden kann.	F0x
U11	ALS Administrator WILL ich definieren können welches Gerät, welche Anfragen versenden kann DAMIT jedes Gerät für Ort und Zweck optimiert werden kann	F0x
U12	ALS Administrator WILL ich die Verwalung und Konfiguration des Praxisrufsystems zentral verwalten DAMIT overhead minimiert wird.	F0x

Out Of Scope

TODO: User Stories für Out Of Scope

3.2 Technisch

Mobile CLient

Id	Anforderung	Features
T01	Die Codebasis des Mobile Client MUSS für Android und IOS verwendet werden können.	F0x
T02	Der Mobile CLient MUSS für IOS auf IPad optimiert werden.	F0x
T03	Der Mobile CLient SOLL für Android Tablets opitimiert werden.	F0x

Betrieb

Id	Anforderung	Features
T10	Der Betrieb aller Cloud Services und Web UIs MUSS über AWS erfolgen.	F0x

3.3 Features

Nummeriert mit F01-Fxx - Lower Level, näher am technischen - Spezifiziert die Anforderungen die Umgesetzt werden sollen

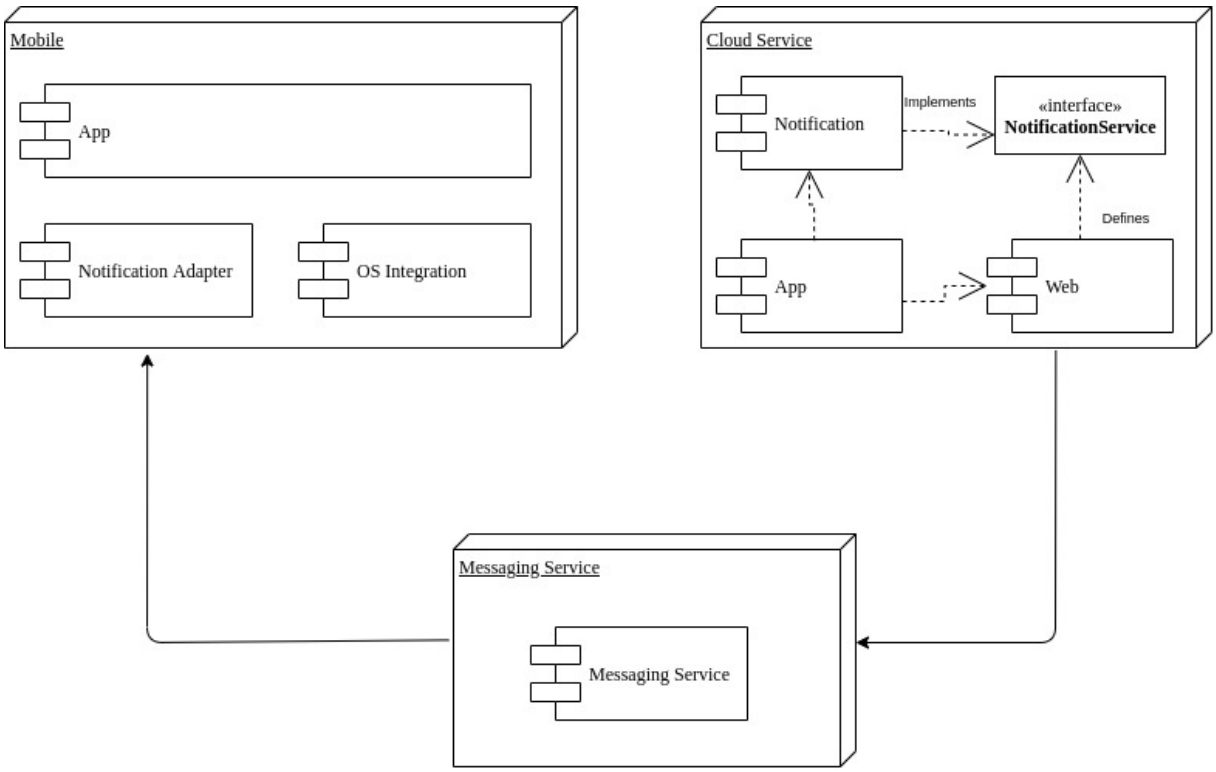
Projekt Scope

TODO: Document Features

Out Of Scope

TODO: Document some features that were discussed but de-prioritized. (e.g. Raspi, Calls)

Weitere Features wurden noch nicht konkreter definiert. Da jeweils je SSprint"die als nächstes umzusetzenden Features definiert wurden.



4 Konzept

4.1 Systemarchitektur

Überblick

Für das Cloudbasierte Praxisruf System sehen wir fünf Komponenten vor:

- Messaging Service
- Cloud Service
- Mobile Client
- Admin UI
- VOIP Mediator

Mobile Client

- Der Mobile Client implementiert die Anbindung an den Messaging Service.
- Als Reaktion auf eine Notification wird eine Rückmeldung im UI angezeigt.
- Als Reaktion auf eine Notification wird eine OS Push Notifikation gesendet. Das UI bietet einen Button der eine Anfrage an die REST Schnittstelle im Cloud Service sendet.

Cloud Service

- Responsibilities (Notification and Configuration)
- Microservice Granularity

Messaging Service

- Dies wird ein externer Service den wir in die Applikationen einbinden. Standard hierfür ist Firebase Notifications.
- Der Messaging Service nimmt Notifikationen vom Cloud Service entgegen und gibt diese an den Mobile Client wieder.
- Dafür müssen auf beiden Seiten Komponenten eingebaut werden, die mit dem Messaging Service kommunizieren.

4.2 Mobile Client

4.2.1 Framework Grundlagen

NativeScript bietet eine Abstraktion zu den nativen Plattformen Android und IOS. Die jeweilige NativeScript Runtime erlaubt es in Javascript (oder einem entsprechenden Application Framework) Code zu schreiben, welcher direkt für die entsprechende native Umgebung kompiliert wird [2].

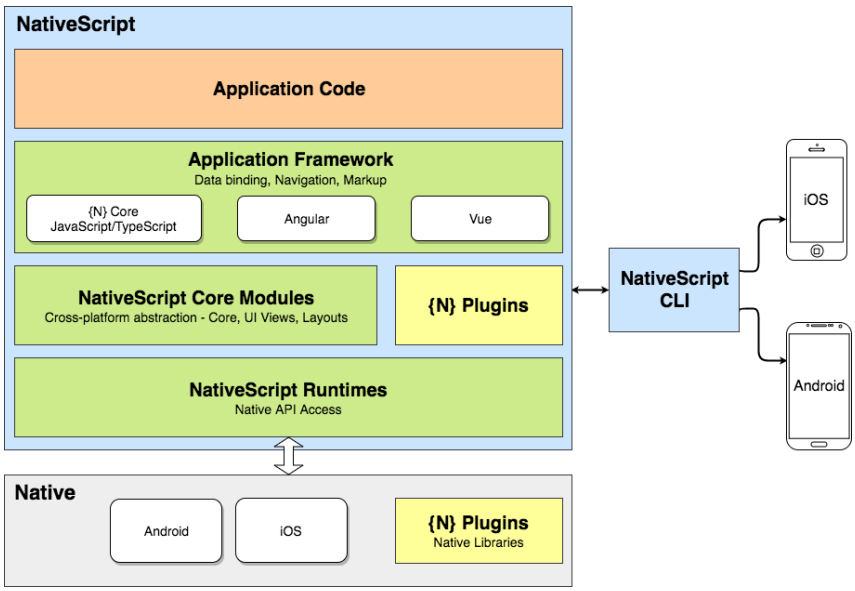


Abbildung 4.1: NativeScript-Overview
©OpenJS Foundation

Die Runtime agiert als Proxy zwischen Javascript und dem jeweiligen Ökosystem. Im Falle von IOS bedeutet dies u.A. das für alle Objective-C types ein JavaScript Prototype angeboten wird. Dies ermöglicht es direkt mit nativen Objekten zu interagieren. Im Umkehrschluss findet eine Typenkonversion via Marshalling Service statt[3].

4.2.2 Anwendung

Wir verwenden NativeScript Core als Framework des Mobile-Clients. In Kapitel *Mobile Client Evaluation* gehen wir auf die weiteren verfügbaren Frameworks ein und erläutern, weshalb wir uns gegen sie entschieden haben.

Die Client-Applikation ist in Module unterteilt. Ein Modul wird aus folgenden Komponenten definiert:

- UI-Markup: Statische Darstellung in XML
- Backend: Verhalten und Dynamisierung in Javascript
- Styling: Layout und Styles in CSS

Ein minimales Modul kann alleine aus einer XML-Datei bestehen. Die optionalen Javascript und CSS Dateien müssen denselben Namen haben wie die XML Datei, um vom Framework korrekt verknüpft zu werden. Dateien mit anderen Namen werden grundsätzlich vom Framework ignoriert. Natürlich steht es Frei dennoch solche Dateien anzulegen und deren Funktionen zu verwenden z. B. als *Services* oder als *Code-Behind Komponenten*.

Zur Veranschaulichung der möglichen Interaktionen gehen wir auf die relevanten Aspekte des Home-Screen Modules ein.

Page Module

```
app
└── home
    ├── home-page.xml
    ├── home-page.css
    ├── home-page.js
    └── home-model.js
```

home-page.xml deklariert die umgebenden Komponenten. Diese Komponenten stellen je nach Typ diverse Properties und Events zur Verfügung. Properties können entweder statisch befüllt oder aus dem Binding-Context geladen werden. Den Events können Callback-Functions zugewiesen werden. Es stehen alle Funktionen zur Verfügung, welche im Backendscript *home-page.js* exportiert werden.

```
1  <Page loaded="onPageLoaded" navigatingTo="onNavigatingTo"
2        xmlns="http://schemas.nativescript.org/tns.xsd"
3        xmlns:profile="components/profile">
4      <StackLayout id="profile" class="home-body">
5          <profile:profile-container/>
6          <StackLayout class="btn-box">
7              <Label textAlignment="center" text="Meldungen" class="section_title"/>
8              <GridLayout loaded="onGridLoaded" columns="auto, auto, auto" rows="auto
       auto auto" horizontalAlignment="center">
9
10             </GridLayout>
11         </StackLayout>
12     </StackLayout>
13 </Page>
```

Listing 1: home-page.xml

Der Binding-Context ist ein JavaScript Objekt welches exklusiv im Page-Context zur Verfügung steht. Es ist allgemein Best-Practice dieses Objekt in einem eigenen Model zu verwalten. Das eigentliche Binding wird vom Backendscript *home-page.js* (Zeilen 8–11) während des ersten Ladens der Seite durchgeführt.

```
1  import {fromObject, Observable, ObservableArray} from '@nativescript/core'
2
3  export function HomeItemsViewModel() {
4      const viewModel = new Observable();
5      viewModel.notificationConfigurations = new ObservableArray([])
6
7      return viewModel
8  }
```

Listing 2: home-model.js

Das Backendscript ist für das dynamische Verhalten der Seite verantwortlich. Hier können die Interaktionen der Benutzer beliebig verarbeitet und der Binding-Context bei Bedarf verwaltet werden.

```
1  import {ApplicationSettings, Button, GridLayout} from "@nativescript/core";
2  import {MessageTrigger} from "~/components/message-trigger/message-trigger";
3  import { HomeItemsViewModel } from './home-model'
4  import {getClientConfiguration} from "~/services/configuration-api";
5  import {showError} from "~/error-dialog/error-dialog";
6
7  const model = new HomeItemsViewModel();
8  export function onPageLoaded(args){
9      const mainComponent = args.object;
10     mainComponent.bindingContext = model;
11 }
```

```
12
13 export function onGridLoaded(args){
14     const grid = args.object;
15     getClientConfiguration(ApplicationSettings.getString("clientId"))
16         .then(result => buildMessageUI(result, grid))
17         .catch((e) => showError("Loading Client Configuration failed", e, "OK"));
18 }
19
20 function buildMessageUI(clientConfiguration, grid){
21     let rowCounter, columnCounter, rowIdx, columnIdx = 0;
22     clientConfiguration.map((conf) => {
23         const messageComp = new MessageTrigger(conf.title, conf.id);
24         grid.addChild(messageComp);
25         GridLayout.setRow(messageComp,rowIdx);
26         GridLayout.setColumn(messageComp,columnIdx);
27         rowCounter++
28         columnCounter++
29         if(columnCounter > 2){
30             columnIdx = 0;
31             columnCounter = 0;
32         } else {
33             columnIdx++;
34         }
35         if(rowCounter > 3){
36             rowIdx++;
37             rowCounter = 0;
38         }else {
39             rowCounter++
40         }
41     })
42 }
```

Listing 3: home-page.js

Code-Behind Komponenten

Code-Behind Komponenten bieten die Möglichkeit zur Laufzeit dynamisch Grafikelemente dem UI hinzuzufügen. Komponenten die das Framework bereits zur Verfügung stellt können direkt mit `new <Component>()` instanziiert werden. Bei Bedarf können diese Komponenten auch erweitert und mit zusätzlicher Funktionalität ausgestattet werden.

Da der Home-Screen dynamisch in Abhängigkeit der Client-Configuration erstellt werden muss, werden eigene `MessageTrigger` Komponenten verwendet.

Services

In Services werden diejenigen Funktionen ausgelagert, welche nicht direkt im Zusammenhang mit der grafischen Representation stehen. So z. B. die REST-Calls zur API.

4.2.3 User Interface

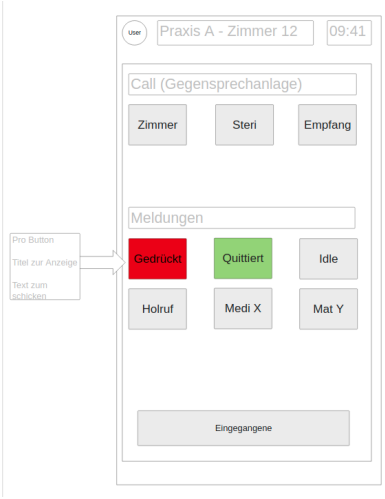


Abbildung 4.2: HomeScreen Mockup

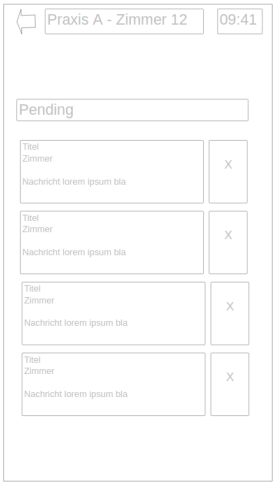


Abbildung 4.3: Inbox Mockup

4.3 Cloud Service

4.3.1 Architektur

4.3.2 Domänenmodell

4.3.3 Laufzeitmodell

4.4 Admin UI

4.5 Proof Of Concept

Anforderungen

- Als <Sender Rolle>möchte ich Notifikationen versenden können.
- Als <Empfänger Rolle>möchte ich Notifikationen in der Applikation sehen, wenn die Applikation geöffnet ist.
- Als <Empfänger Rolle>möchte ich Notifikationen über das OS erhalten, wenn die Applikation minimiert ist.

Restriktionen

- Nur 1 Client.
- Nur 1 fixe Notifikation. Keine Types.
- Notifikation wird vom Client gesendet und vom selben Client empfangen.
- Keine Authentication oder Authorization.

5 Evaluation Technologien

5.1 Mobile Client Evaluation

https://kotlinlang.org/lp/mobile/

+Jet Brains Infrastructure +We like Kotlin

-IoS Env. Needed to develop for Apple -Still has to develop separate API und UI Modules for Platforms

https://web.dev/progressive-web-apps/

+No need of Native Codebase +Perfect for Android -Eventually drawbacks because no entire API Access -PWAs on IOS suck

https://cordova.apache.org/

+ Popular Framework + Tons of plugins to access apis

-Still need to have a Mac for IoS development -Not a truly native app -¿ API Issues

https://nativescript.org/

+Provides a Workaround for nasty X-tools +Claims to be truly Native -Do we really trust it? (sorta new and passion project of a few people)

https://flutter.dev

-Why do you hate me?

SSimply Write Everything twice”

+Would definitely work

-Do most things twice -We don’t have time for that -Kunde wünscht ausdrücklich nur eine Codebasis für beide Clients.

https://stackshare.io/stackups/apache-cordova-vs-nativescript

https://nativescript.org/blog/build-nativescript-apps-remotely-from-windows-or-l

5.2 Cloud Service

https://aws.amazon.com/

https://spring.io/projects/spring-boot

Konfig der Clients könnte sich als No-SQL anbieten.

Config muss nur gelesen und an den Client geschickt oder abgespeichert werden

https://www.mongodb.com/

5.3 Betrieb und Platform

AWS ist MUSS

6 Umsetzung

6.1 Authentifizierung

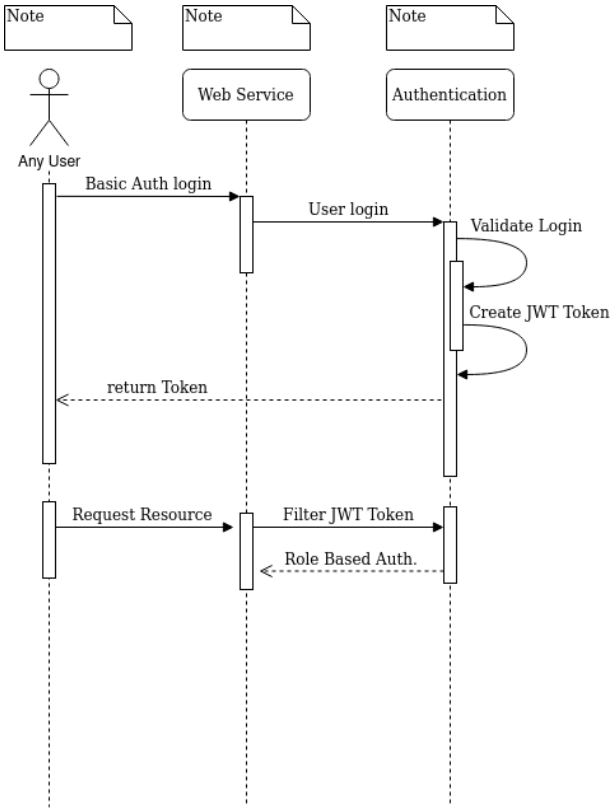


Abbildung 6.1: Authentifizierung-Sequenz

7 Schluss

Literaturverzeichnis

[1] D. E. Knuth, *The TEXbook*. Addison – Wesley, 1990, ISBN: 0-201-13447-0.

[2] O. Foundation. (). How NativeScript Works, Adresse: `https://v7.docs.nativescript.org/core-concepts/technical-overview`.

[3] ——, (). What is iOS Runtime for NativeScript? Adresse: `https://v7.docs.nativescript.org/core-concepts/ios-runtime/overview`.

Abbildungsverzeichnis

0.1 Titlebild 1
2.1 Projektplan 3
4.1 NativeScript-Overview 9
4.2 HomeScreen Mockup 12
4.3 Inbox Mockup 12
6.1 Authentifizierung-Sequenz 17

8 Anhang

8.1 Benutzerhandbuch

8.2 Betriebshandbuch

8.3 Entwicklerdokumentation

8.4 Ehrlichkeitserklärung