Internet Praktikum of Telecooperation Group - Winter Term 2018/2019 Project Topic: Interactive Door Panel for Office Availability Team Foxtrot

Artem Vopilov (email: mister.0026@yandex.ru)
Florian Schunk (email: florian.schunk@stud.tu-darmstadt.de)
Javier Ochoa Serna (email: jochoaserna@hotmail.com)
Pramod Pramod (email: pramod.pramod@stud.tu-darmstadt.de)
Frank Langoulant (email: fl1001@rbg.informatik.tu-darmstadt.de)

TECHNISCHE
UNIVERSITÄT
DARMSTADT

Telecooperation Group
S2|02 A114 Hochschulstraße 10
64289 Darmstadt

Abstract

In this project we implement a door panel application designed to run on Android Tablet as well as the counterpart application designed to run on android smartphones to control the tablet application. The intention is to display information about the office room the doorpanel is assigned to as well as the workers in this office and their availability status. If the workers don't want to be disturbed by visitors coming by, the application provides the means to request an appointment as well as to send a message to a worker.

Contents

1 Motivation 2

2 Structure 2
2.1 Structural Overview 2
2.2 Description of elements and their interaction 2

3 Installation & requirements 2
3.1 Google authentication 2
3.2 Dependencies & Requirements 4

4 User Descripton 5
4.1 Basic Funtionality 5
4.2 Detailed explanation 5
4.2.1 Door panel app 5
4.2.2 Mobile app 6
List of Figures9

1 Motivation

Traditional door panel signs printed on paper may be quite common but they lack a lot of possibilities. Every time there is a change concerning the workers assigned to the office room, the sign needs to be reprinted and replaced. Even more important, the workers have no means to indicate to visitors passing by and maybe wanting to talk to a worker inside the office if they are welcome to enter or if a disturbance is currently inappropriate.

In order to change that we implemented an interactive door panel application to meet the requirements of up to date offices and their personnel. It consists of an application intended to run on an Android smartphone and an application to run on the Android smartphones of the workers.

The tablet application displays the workers who are currently assigned to the office room and provides the possibility to send a message to the worker as well as to request an appointment within timeslots the workers can choose separately.

The smartphone application provides the possibility to the worker to change his availability status, to receive and answer messages, to access his calendar and to change the personal information about them such as picture, phone number and email address. It also allows the worker to choose a Google calendar that is connected to the application and were confirmed appointments are stored. Finally it also allows the worker to choose time slots that are available for appointment request to visitors.

This door panel application is intended to support workers in an office room to handle messages and appointment request of visitors without being disturbed or interrupted during their work.

2 Structure

For easier understanding, we will call the android application designed to run on a tablet placed outside of the workers office 'door panel app' und the counter part application designed to run on workers android smartphone 'mobile app'.

2.1 Structural Overview

This application is designed as a server client architecture. All information is stored using MongoDB, the communication is handled by a NodeJS Server and messages are distributed using Firebase. Google calendar is addressed by the door panel app and mobile app directly. Figure 1 shows an overview over the architecture.

2.2 Description of elements and their interaction

To be able to store any information the application needs access to a MongoDB database. The location is stored in $server/config/default.json$

To enable communication between the elements all elements the NodeJS server needs to be reachable. The IP-Address of the hosting machine is stored in $'RetrofitClient.java'$ as string called $'BASE\_URL'$. The file $'RetrofitClient.java'$ is located at $'src/main/java/.../network/RetrofitClient.java'$ in both applications.

3 Installation & requirements

3.1 Google authentication

Because we use Google APIs for accessing the Google Calendar, we need to follow the Google authentication procedure. Google requires that developers provide the key that is used for compiling in the

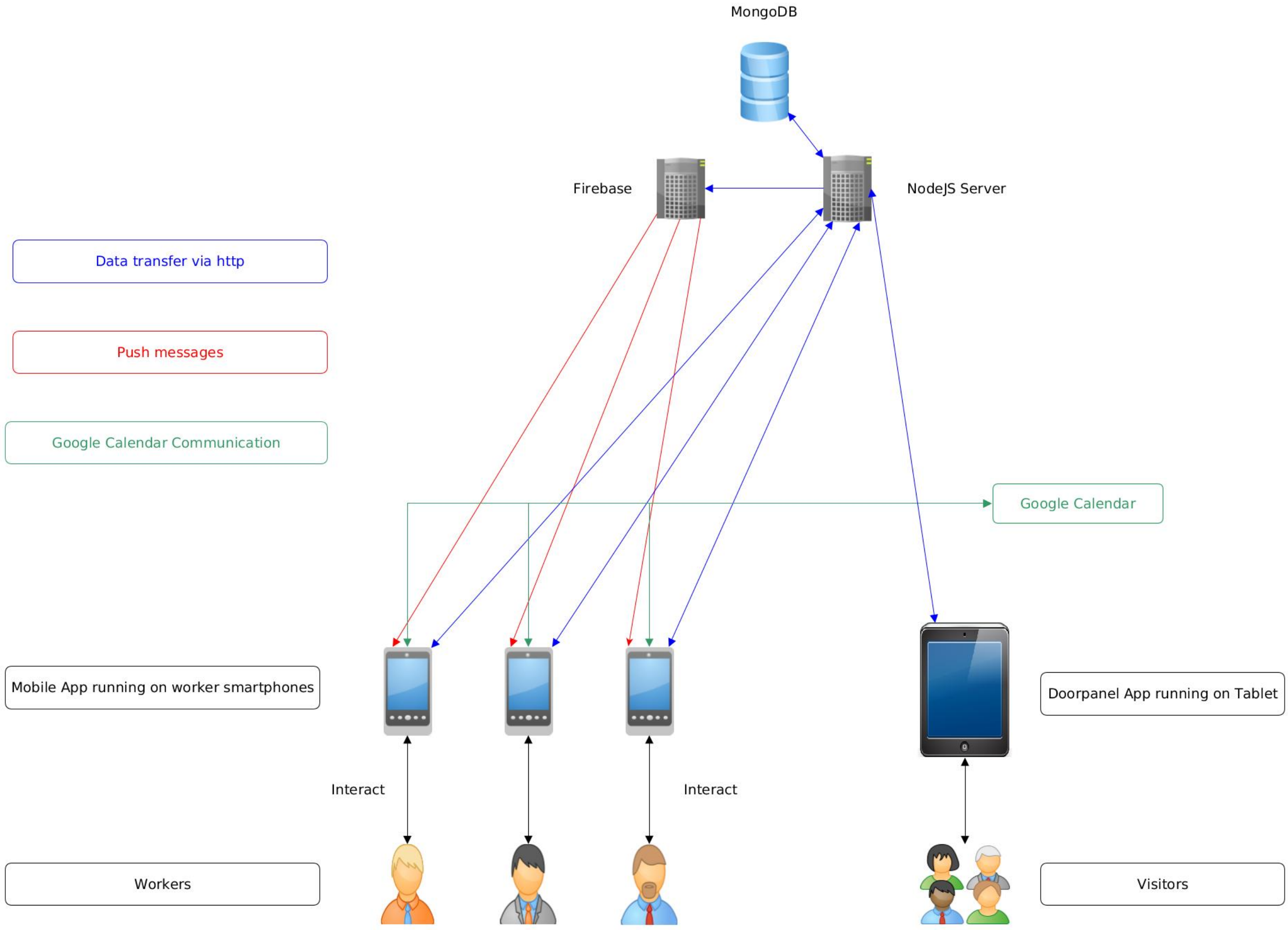


Figure 1: Basic communication

Google Developer Console. To do so, visit the following website:

$https://console.developers.google.com/apis/credentials?folder$
$= \&organizationId = \&project = winter - time - 228212$

Login with email $'doorpaneltest@gmail.com'$ and password $'FoxtroTT'$ There click on $'Create\ Credentials'$ and then on $'OAuth\ Client\ ID'$ Next select $'Android'$. Then you are asked to select a name, here you could use anything, but please be so kind and state your real name here, so it is easier for us to manage. Also you need to provide your key. Google tells you to run

keytool -exportcert -keystore path-to-debug-or-production-keystore -list -v

the path to the keystor should usually be

$/.android/debug.keystore$

however I got an error that I can't give 2 commands at the same time, so I removed the
$-exportcert$ So my total command was the following:

keytool -alias androiddebugkey -keystore /.android/debug.keystore -list -v

More information is under:

$https://support.google.com/cloud/answer/6158849?hl = en - GB\#$
$installedapplications\&android$

alternatively you can run the gradle task signing Report.
Finally you have to choose

$de.tu\_darmstadt.foxtrot.foxtrot\_doorpanel\_app$

as Package name.

3.2 Dependencies & Requirements

This application is designed to work on Android version 9 (API Level 28) The minimum Android version for this application to run is Android version 5.1 (API level 22).

The MongoDB Database is currently located at 'mlab':
$mongodb://Foxtrot\_Artem:12345a@ds163103.mlab.com:63103/foxtrot\_db$
The Address of the MongoDB Database is stored in $'/server/src/app.js'$ $('mongoose.connect((...)())')$

The NodeJS server is currently located at 'Openshift': $http://foxtrot - doorpanel - new - foxtrot - doorpanel - server.7e14.starter - us - west - 2.openshiftapps.com/workers$
Our registration on Openshift ends Mai 15. 2019 at 4:30am.

Packages needed for the NodeJS server to run:

- Koa
- Koa-router

- Koa-bodyparse
- Koa-passport
- Koa-static
- mongoose
- firebase-admin
- nodemon
- passport
- passport-jwt
- passport-local

If the package 'NPM' is installed all those dependencies can easily by executing 'npm install'.

4 User Descripton

4.1 Basic Funtionality

The door panel app running on a tablet displays the room number and the workers currently assigned to this office room. The app surface also displays a picture chosen by the worker, the position of the worker (eg 'Intern', 'Student', 'Professor') and the availability status also selected by the worker. A visitor can select a worker to get further information about the worker (if entered) and also send this worker an instant message. For this a message text is needed as well as a name and an email address.

Next to the worker a calendar symbol offers the visitor the possibility to send an appointment request. For this the visitor needs to choose an available time slot determined by the worker and also enter a name, an email address, an phone number and a message text.

The mobile app is intended as a control and access unit for the worker. It provides the ability to access the linked Google calendar, to select the availability status with a predefined or individual text as well as to see and answer the messages send from visitors and any appointment requests.

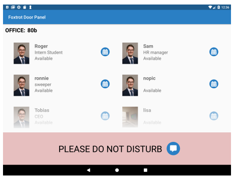
The mobile app is also used to adapt personal settings such as changing the picture, changing phone number, email address and room number, entering further personal information (eg current research or publications), define time slots in which visitors can request appointments, selecting a calendar to link to the application and also to add new workers.

4.2 Detailed explanation

4.2.1 Door panel app

The main level of the door panel app shows the room number and the workers currently assigned to this office room. For each worker is displayed: a picture, a title (if entered), the availability status and a calendar icon. (See Figure 2).

When a visitor selects a worker a screen with further details about the worker is displayed (See Figure 3). There informations are: a picture, a title (if the worker selected one), the availability status as well



OFFICE: 80b

Roger
Intern Student
Available

Sam
HR manager
Available

ronnie
sweeper
Available

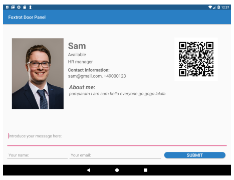
nopic
Available

Tobias
CEO
Available

lisa
Available

PLEASE DO NOT DISTURB

Figure 2: Main Screen - Door panel app



Sam
Available
HR manager
Contact information:
sam@gmail.com, +49000123

About me:
pamparam i am sam hello everyone go gogo lalala

Figure 3: Personal information - Door panel app

as further information (if entered). On the bottom of this view is a possibility to write a message to this worker that is send directly to the smartphone of the worker. That message includes a message text, a name and an email address.
In the right part a QR-code is displayed which indicates the name, phone number and email address of the worker on a visitors smartphone when scanned.

Figure 4 shows the view to request an appointment that opens when a visitor selects the calendar icon displayed next to the worker on the main screen. A calendar view is shown where the visitor can see the time slots declared from the worker. Only within those time slots a visitor can choose a date and time for an appointment. Further more the visitor needs to enter a name, an email address, a phone number and a message text. Only then can the appointment request be send. This request is then send to the worker who can either accept or decline it. The workers response is send to visitors email address and if confirmed the appointment is entered in the linked Google calendar.

In the bottom part, next to the status text for the office, there is a chat symbol. When selected, it shows all replies send by the workers to messages from visitors from the same day.

4.2.2 Mobile app

The main level menu of the mobile app shows the 4 available sections: '$Timeslots$', '$Status$', '$Notifications$' and '$Settings$' (See Figure 5)
Figure 6 shows how arranged time slots are displayed

The button $'List'$ shows the time slots of the day in detail. The Plus-Button at the bottom allows the worker to add new time slots. Visitors who want to request an appointment can only select among the time slots defined by the worker. (–new pic) Figure ... shows how a time slot is defined: it contains a title and a description (location). Below this the worker can select when the time slot starts and when

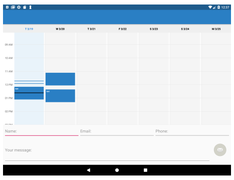


Figure 4: Timeslots - Mobil app

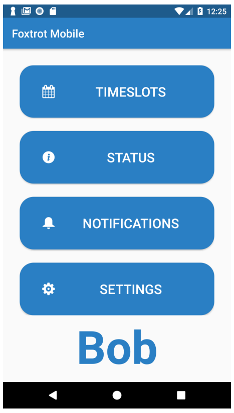
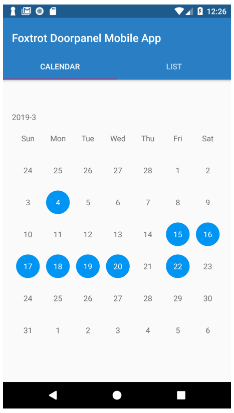


Figure 5: Main Screen - Mobile app



2019-3

Sun	Mon	Tue	Wed	Thu	Fri	Sat
24	25	26	27	28	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Figure 6: Timeslots - Mobil app

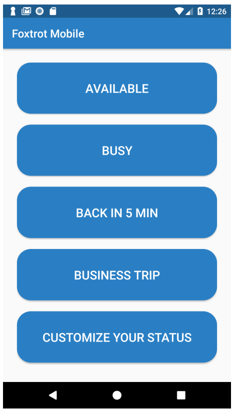
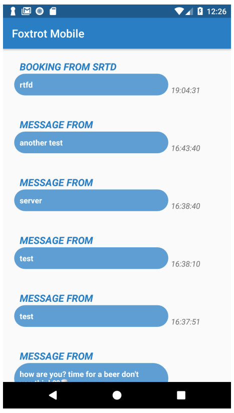


Figure 7: Status Selection - Mobile app



BOOKING FROM SRTD
rtfd 19:04:31

MESSAGE FROM
another test 16:43:40

MESSAGE FROM
server 16:38:40

MESSAGE FROM
test 16:38:10

MESSAGE FROM
test 16:37:51

MESSAGE FROM
how are you? time for a beer don't

Figure 8: Notifications - Mobile app

it end. This is done by selecting a start date and time followed by an end date and time. Above the title entry the worker can select if it is intended to be a single or a repeating one (eg every monday). If *′Automatic repeating timeslot′* is selected, the worker can chose a beginning time and an end time. The option *′Length of a time slot′* allows to worker to chose how long every available time slot within beginning and end time is (eg: one hour between beginning and end time with 30 min time slot length means that there are 2 appointments available).

Figure 7 shows the menu to select a workers availability status. This menu is reached by the button 'status' on the main menu. The worker can choose between some predefined texts or enter an own description. The availability status is changed on the doorpanel app instantly.

Figure 8 shows the list of notifications send from the door panel app to the mobile app. They are sorted chronologically. The worker can reply to the sender by selecting a notification.

Figure 9 shows the main settings menu. There the worker can adjust all available settings concerning personal details and options.

The first button *′Person′* allows the worker to change his/her phone number, the email address and the room number he/she is currently assigned to. Those information can be changed individually or all together. not all fields need to be filled out.

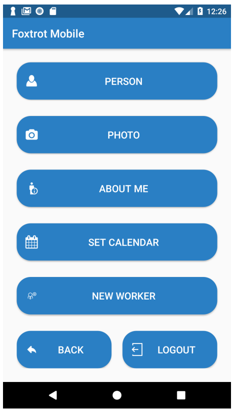


Figure 9: Settings Main Level - Mobil app

The button *′Photo’* allows the worker to select a picture that will be shown on the door panel app.

Under *′Aboutme′* the worker can enter further information (eg ’current research’ or ’publications’) that are shown on the tablet when a visitor selects the worker.

Under *′Set calendar′* the worker can choose what calendar to use for what. All detectable and linked calendars are shown and can be chosen for time slots and for events.

The button *′New Worker′* leads to an interface that allows the worker to enter informations for a new worker.

List of Figures

1 Structural overview 3
2 Main Screen - Door panel app 6
3 Personal information - Door panel app 6
4 Timeslots - Mobil app 6
5 Main Screen - Mobile app 7
6 Timeslots - Mobil app 7
7 Status Selection - Mobile app 8
8 Notifications - Mobile app 8
9 Settings Main Level - Mobil app 9