Smart Personal Organizer

Johann Haag, Tarik Hosic und Simon Blaha

November 18, 2018, Leonding

Contents

1 Introduction

There are a lot of personal organizer providers. These organizers are very similar in their functionality, but they have one thing in common, their functionalities are very limited. The personal organizers are kept very simple. It is nearly not possible to set complex conditions, which are checked automatically. Some organizers have the functionality to share an appointment but with some personal organizer it is not possible to let the organizer automatically find a suitable date for all participants. That is why it is often hard to find a suitable date for all participants. For the distribution of appointments, there are other providers. Here is, for example to mention doodle.

Often appointments have conditions to be able to take place at all. For example, a nice weather is needed to have a barbecue with his friends.

There are also often cases where appointments are interdependent, so it may be that an appointment is not taking place because an appointment with which it was related could not take place.

At an appointment for a lecture, the dependency exists that the lecturer can meet this deadline. It could be the case that the lecturer becomes ill and thereby the appointment fails. There should be an easy way in the organizer to enter such cases and to inform all involved.

2 General Conditions and Constraints

For our project we will use the Framework Ionic. This framework is based on the framework Angular and Apache Cordova. This framework makes it possible to write an application for mobile devices. With the framework, the application should be programmed on the client side. The focus here is on a version for Android.

The server will use 2 APIs, one to poll the weather and another to calculate the time for the path.

The API openweathermap is providing the weather functionality but for free use it has some restrictions, which can be found on https://openweathermap.org/price. In order to determine the time of the way between 2 dates the Google API is used.

3 Project Objectives and System Concepts

Our goal is to create a new personal organizer that will attract the attention of many people with additional features. The new personal organizer should include features such as weather and route, friend function, grouping function, profile and chat, rights system, dependency between appointments, lists and comments, showing birthdays of friends, editing appointments, notifying the group, matching the appointments over the whole group included. Nevertheless, we are aware that for the first time we have to realize the basic functions that an appointment book contains. Appointments have basic functions such as the weekly, monthly and yearly view, add appointments, schedule conflict check, buffer times delete appointment, push notifications, backup and synchronization of appointments (cloud) and local storage. After we have finished the basic functions of a personal organizer, we want to incorporate our additional features, which we have already mentioned, and thus have the new personal organizer realized.

A user's account is there to manage the various groups and profiles. Furthermore, the user is by the Account uniquely identifiable. The profiles are there to save the configurations. There are several profiles because it can be the case that you do not always want to use the same configuration.

A group can also set conditions to what extent a profile must be configured to join the group. Furthermore, a group is able to distribute notifications to its members. The notifications are subdivided in importance levels and that makes it possible to filter them. The groups can be listed or unlisted. If a group is listed, it can be found using a search function of the application.

There is also a functionality with which a user can announce what he is interested in. This allows him to see the groups that correspond to his interests.

Generally there should be a visibility for appointments. The user should be able to set the visibility settings through a configuration.

It should be possible for a person to report sick. As a result, appointments that depend on the person can be deleted. Furthermore, this gives the opportunity to notify all affected.

4 Opportunities and Risks

During the development of our idea we will have to take some risks. For example, one risk would be that the APIs used are unavailable or unrecognized. In addition, they may be unreliable. With our idea we want to get involved in clubs, groups of friends and private persons. In addition, we want to reduce communication problems.

Furthermore, the Framework Ionic is one that is constantly evolving. As a result, it may be necessary to switch to a newer version to be up-to-date. This may mean that the existing code needs to be adapted to new conditions.

5 Planning

The project will start on the November 30, 2018.

5.1 Members

- Johann Haag
- Tarik Hosic
- Simon Blaha (Project leader)

5.2 Milestones of the Project

- Prototype Client
- Prototype Server
- Client Core
- Server Core
- Advanced Client
- Advanced Server
- Client GUI
- Additional Client Functions
- Additional Server Functions
- End of Project

Prototype Client

Description:
A rudimentary prototype will be created. It uses the frameworks and APIs that will be used later in the application.
Goal:
The purpose of the prototype is to determine how the APIs and frameworks used work. As a result, the further course of action in the project can be better defined.
Until: 30.12.2018

Prototype Server

Description:
A rudimentary, little server is created with basic functionality.
Goal:
The purpose of the server is to determine which technology is suitable for programming the server. Furthermore, there is an overview of how to implement the server.
Until: 30.12.2018

Client Core

Description:
The core of the client application has the basic functionality.
Goal:
The core should provide a good foundation upon which the rest of the application can build.
Until: 30.2.2019

Server Core

Description:
The core of the server should have basic functionality.
Goal:
It should provide a good foundation upon which the rest of the server can build.
Until: 30.2.2019

Advanced Client

Description:
The client core with additional functions.
Until: 15.4.2019

Advanced Server

Description:
The server with additional functions.
Until: 30.4.2019

Client GUI

Description:
Updated Client GUI
Until: 20.4.2019

Additional Client Functions

Description:
Additional functions for the client.
Until: 20.5.2019

Additional Server Functions

Description:
Additional functions for the server.
Until: 10.6.2019

End of Project

Description:
End of the project.
Until: 15.6.2019