Travlendar+ Implementation and Test Document

Sinico Matteo, Taglia Andrea

Version 1.0

December 2017

link to the application demo

link to the source code

Table of Contents

1. INTRODUCTION

A.Introduction

This document addresses the implementation and testing deliverable of the travlendar+ application. In the following pages is described how we have implemented and tested our application, based on the what there is in the RASD and DD documents already delivered.

B. Scope

The main scope of this document is to explain the implementation choices taken to deploy a first version of the Travlendar+ application, and to show what kind of tests have been performed on the application, explaining also the automated tools used in order to accomplish this task.

2. REQUIREMENTS/FUNCTIONALITIES IMPLEMENTED

In this section we have included the list of requirements/functionalities implemented in the first version of Travlendar+ application, followed by a brief motivation either for including them or for excluding them. We have reused the abbreviations used in the RASD document to refer to the requirements.

It hasn't been taken care too much of deep usability aspects as the software is intended to be an initial version. Of course improvements can be made from almost any viewpoint.

Here is a list of the requirements/functionalities implemented in the software:

- [R1.1] A log in functionality must be provided in order to authenticate the user.
- [R1.2] User's data (calendar appointments, preferences, personal data) must be kept secret to the user himself and to anybody else.
- [R1.4] Users must have a personal calendar.
- [R2.1] Users must be able to add an event to their personal calendar giving a title, a day, a time at which the event should be scheduled in the calendar.
- [R2.3] Users must be able to delete an already existing event in their personal calendar.
- [R2.4] Users must be able to change their preference about car usage.
- [R2.5] Users must be able to change their preference about bike usage.
- [R2.6] Users must be able to change their preference about walking.
- [R2.7] Users must be able to change their preference about carbon footprint.
- [R2.8] Users must be able to change their preference about public transports.
- [R2.9] Users must be able to edit preferences on break times choosing a lower and upper bound for lunch and dinner times and a minimum break time.
- [R3.1] Computation of optimal route must take into account user preferences.
- [R3.5] No appointment can be successfully scheduled after another if it would not respect flexible break times.

- [R3.6] Change in break times preferences are not taken into account for previously scheduled appointments.
- [R4.1] No overlapping appointments are allowed. A warning would show up and the user may not be able to schedule the event.
- [R4.2] Transport time estimation between appointment location being added and previous location on schedule must be done based on public transports only.
- [R4.3] The application must always be aware of the previous user location with respect to the newly added event.
- [R5.1] The application must compute a travel proposal for the current day to show to the user.

We have implemented these requirements/functionalities because we think that they are all essential in order to captivate future users.

On the other hand here is a list of the requirements/functionalities which has not been implemented in the software with motivation for excluding them

- [R1.3] A password recovery functionality must be available in order for the user not to permanently lose access to the service.
- [R1.5] Users must be able to edit their personal data. (profile data)
- [R2.2] Users must be able to edit an existing event changing the title.

These 3 requirements/functionalities are not seen as necessary because they don't enrich the value of the application, so they are only a loss of time, at least in a first version of the application

- [R3.2] Computation of optimal route must take into account weather forecast.
- [R3.3] Computation of optimal route should be made more times and each with different weight on preferences in order to give different day-travel proposals.
- [R3.4] There must be a way to let the user switch among different proposal.
- [R5.2] A brief description of each Travel Piece of a day Travel Proposal should be shown to the user.
- [R5.5] Each appointment should have an estimated time of arrival (computed based on public transports) attached.

These requirements/functionalities have not been implemented because they are not useful in order to engage future users. But they are nice to have future, so they can be shown as future feature.

- [R5.3] Allow users to check any day scheduled appointments;
- [R5.4] There must be a comprehensive view over any given day.

These requirements/functionalities can't be implemented because we didn't include a calendar view of user's appointments, because we were keen on show a travel proposal for the current day, instead of implement an agenda. These features will be certainly added in a future version of the application.

*explain the home functionalities added in implementation phase.

3. ADOPTED DEVELOPMENT FRAMEWORKS

4. STRUCTURE OF THE SOURCE CODE

5. TESTING:

6. INSTALLATION INSTRUCTIONS:

Please refer to the ReadMe on the official repository located on github.

7. EFFORT SPENT:

8. REFERENCES