

Politecnico di Milano

A.Y. 2017/2018

Software Engineering 2: ***Travlendar+***

**Acceptance Test Deliverable**

Matteo Biasielli - Emilio Capo - Mattia Di Fatta

v. 0.1

**Table of Contents**

**1. Introduction**

1.1. Document purpose..………………………………………………...………………….3

1.2. Description of the problem………………………………………………………….3

1.3. Definitions, Acronyms, Abbreviations………………………………………….3

1.3.1 Definitions……………………………………………………………………….3

1.3.2 Acronyms………………………………………………………………………...4

1.3.3 Abbreviations…………………………………………………………………..4

**2. Requirements and Functionalities**

2.1. Functionalities

2.2. Requirements

**3. Adopted development frameworks**

**4. Structure of the source code**

**6. Test Plan**

**7 Performed tests and analysis**

3.1. Unit Test..………………………………………………...………………….

3.2. Integration testing…………………………………………………….

3.3. Performances evaluation……………………………………………….

**8. External References**

**9. Tools Used**

**10. Effort Spent**

**1. Introduction**

* 1. **Document purpose**

This document focuses on the implementation of details the project Travlendar+. The application’s purpose is to support users in handling out one of the most difficult nowadays’ challenges: organization. No previous versions of this application were developed.

This document is meant to be a reference for any person who has an interest in the project. This includes, but is not limited to, development team members, stakeholders and end users.

* 1. **Description of the problem** *[reported from RASD document]*

The aim of the project is to create an all-in-one system that unites services that are nowadays offered by various different applications (e.g. Calendar, Travel Scheduler). In order to use Travlendar+, final users must be registered and logged in.

Users should be able to schedule their activities directly through the application and, by taking into account travelling times, constraints and preferences expressed by the user, Travlendar+ should:

* Identify the best mobility option;
* Support the user in buying public transport tickets, if necessary;
* Locate the nearest car or bike sharing, if they represent the best solution;
* Warn the user when a place can’t be reached in the available time.

In general, Travlendar+ should make it easier to organize complex schedules, by finding the best compromises between time optimization and the users’ needs and preferences.

* 1. **Definitions, Acronyms, Abbreviations**
     1. **Definitions**
* **User**: actor that is using the application and may want to access all functionalities.
* **Application**: with the term application we are talking about the desktop version, the website and mobile version of the Travlendar+ system.
* **Scheduling**: action performed by a user that is adding a new activity to his personal calendar.
* **Flexible Activity**: An activity with starting and ending time larger than the duration.
* **Fixed Activity**: An activity with fixed starting and ending time.
* **Break**: Flexible Actvity.
  + 1. **Acronyms**
* **RASD:** Requirements Analysis and Specification Document
* **UI:** User Interface
* **API:** [Application programming interface](https://en.wikipedia.org/wiki/Application_programming_interface)
* **UML**: Unified Modeling Language
* **GPS**: Global Positioning System
  + 1. **Abbreviations**
* **[Gn]:** the n-th goal
* **[Rn]:** the n-th requirement
* **[NFRn]:** the n-th non-functional requirement
* **[An]:** the n-th assumption
* **[Cn]:** the n-th constraint
* **[Fn]:** the n-th functionality

**2. Requirements and Functionalities**

* 1. **Functionalities**

In the allotted time for the implementation, we could develop the Server (which structure is discussed later in this document), the Windows Client and Android Client. Considering that **this is a prototype**, we focused on the basic features and what we achieved is that the functionalities we could provide are the following:

* **[F1] Login:** A Login system is available and it’s mandatory to login into the system to use the application;
* **[F2] Registration:** Users that are not registered yet are able to register into the system. Note that for the momentthe registration procedure asks nothing but username and password because those were the only vital data required to provide basic login functionalities. It can be easily extended;
* **[F3] View Calendar:** After the Login, on both clients it’s possible for users to view their calendar with activities that were previously added;
* **[F4] Add a Fixed Activity:** After the Login , users can schedule fixed activities (see definition). After scheduling such an activity, they receive a response from the server containing the result status of the request (OK or an error code) and eventually a notification. There are many different notifications but it’s worth it to focus the attention on a few of them:
  + A notification is received if the added activity does not make the calendar be inconsistent (no overlapping) but the user will not be allowed to be on time to the just added activity;
  + A notification is received if the added activity does not make the calendar be inconsistent (no overlapping) but the user will not be allowed to be on time to the activity that comes immediately after the just added one;
  + A notification is received if the added activity does not make the calendar be inconsistent (no overlapping) but the user will not be allowed to be on time to one activity and the above two cases are not verified.
* **[F5] Add a Flexible Activity (Break)**: After the Login , users can schedule flexible activities (see definition). After scheduling such an activity, they receive a response from the server containing the result status of the request (OK or an error code) and eventually a notification. There are many different notifications but it’s worth it to focus the attention on one of them:
  + A notification is received if the added activity does not make the calendar be inconsistent (no overlapping) but the user will not be allowed to be on time to one activity and the above two cases are not verified.
* **[F6] Update an existing activity:** After the Login, users can update their previously added activities. They can modify any field of activities and they can also change it from fixed to flexible or vice versa. The answer/notification they receive is the same as for [F6] and [F5];
* **[F7] Delete an existing activity:** After the Login, users can delete their previously added activities.
* **[F8] Add a tag:** a tag is a tuple <Position, Address, text>. It represents an address that the user will refer to with a keyword text that he sets as well. Most likely there are some places that users will have to write down really often. The point of the tag system is to allow users to indicate those address by just selecting the tag from the tag list instead of writing down the whole address every time. After the Login, users can add new tags. We completely rely on Google Geocoding and Google Reverse Geocoding APIs for this service, to check that inserted addresses are valid.
* **[F8] Delete a tag:** a tag is a tuple <Position, Address, text>. It represents an address that the user will refer to with a keyword text that he sets as well. Most likely there are some places that users will have to write down really often. The point of the tag system is to allow users to indicate those address by just selecting the tag from the tag list instead of writing down the whole address every time. After the Login, users can delete already existing tags.
* **[F9] Preferences:** After the Login, users can indicate theur travel preferences. There are plenty of possibilities and each of them is always accepted. Though, is a user sets too strict preferences it won’t be possible to find possible routes or to estimate the travel time for an activity in order to be allowed to send notifications to advice the user to get ready to go out. Should this happen, users will be informed.
* **[F10] Notifications:** users are notified every day about the weather. Moreover, users are notified when they should get ready to leave for the next activity. Clients execute the notifications synchronization process every minute;
* **[F11] Travel:** Users can request to be shown a possible travel mean (or more than one) for the next activity. If this is not possible due to their too strict preferences, users are advised to change preferences and retry;