Requirement Analysis and Specification Document

Travlendar+

Authors:

1. Aimi, R. Bigazzi, F. Collini

3/10/2017

Version 0.1.0



1. Introduction

1.A. Purpose

This document is the Requirement Analysis and Specification Document (RASD) of a mobile application called Travlendar+. The purpose of the document is to show the requirements and specification of the new application, considering various aspects like the stakeholders’ needs, domain properties and constrains which the system-to-be is subject to.

*1.A.1 Goals*

1.B. Scope

Travlendar+ is a mobile, calendar-based application that helps the user to manage his appointments and to a greater extent set up the trip to his destination, choosing the best means of transport depending on his needs.

Travlendar+ will choose the most suitable way to get the user to his destination between a large pool of options, considering public transportation, personal vehicles, locating cars or bikes of sharing services and walking to the destination. It will take account of weather, traffic, possible passengers if any, the user-set break times and the potential will to minimize the carbon footprint of the trip, always focusing on taking him on time to his scheduled appointments.

Eventually the user will be able to purchase the tickets he will use to reach his destination in-app. The great customizability is one of the main strengths of Travlendar+, being able to fully comply with the user needs.

*1.B.1 Analysis of the world and of the shared phenomena*

1.C. Definitions, Acronyms, Abbreviations

*1.C.1. Definitions*

*1.C.2. Acronyms*

RASD: Requirements analysis and specification document;

ETA: Estimated time of arrival, it is the time remaining to arrive to destination;

API: Application programming interface;

POI: Point of interest;

1.D. Revision history

1.E. Reference Documents

Documents list:

* Mandatory Project Assignments.pdf
* Requirement Engineering Part III.pdf

1.F. Document Structure

The paper is structured as follows:

* Chapter 1: Explanation of the document purpose and scope
* Chapter 2:
* Chapter 3:
* Chapter 4:
* Chapter 5:
* Chapter 6:

2. Overall Description

2.A. Product perspective

Travlendar+ will be developed as a mobile application that relies on the use of Google maps and Google calendar APIs.

Its user interface will be composed by two main tabs, one with a calendar, to schedule user’s events and the other one with a map to manage the movements of the user.

In the future will have a service of technical assistance via chat.

The application will not provide any API for integration with other systems.

\*\*\*\*-------------------------------------\*\*\*\* Further details on the shared phenomena and a domain model (class diagrams and statecharts)

2.B. Product functions

\*\*\*\*-------------------------------------\*\*\*\* Requirements

2.C. User characteristics

The user of the system-to-be is every person who wants to schedule appointments in a calendar and manage his movements from a location to another at the same time. The application doesn’t have any age limit, or any other restriction applied to the user characteristic. In order to make the application work without limitation the user need to have access to the Internet, but he can access and modify the calendar offline.

2.D. Assumptions, dependencies and constraints

\*\*\*\*--------------------------------------\*\*\*\* Domain assumptions

3. Specific Requirements

\*\*\*\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*\*\*\* More details on all aspects in Section 2 if they can be useful for the development team

3.A. External Interface Requirements

*3.A.1 User Interfaces*

The user interface of the application will appear the same to all users, it will be developed as a mobile application for the main mobile operative systems (iOS and Android), and it must be user-friendly and intuitive. To show how the application will look like some mockups of the user interface are present in the document (mockups are realized for iOS operative system).

After downloading the application from the store of the OS. At the first start it shows a screen where the user must insert all the relevant data (name, surname, important addresses and trip preferences) to fit the most its desires, it is included also the possibility to sync Travlendar+ with a Google account.

Once the application is set, the first screen that will appear to the user at each startup is the main screen that shows up a calendar, where he can add his events.

When the user chooses to add an event pressing the button on the bottom right corner, the application shows a screen where he can insert the details of it.

To manage the trips between home and an event, or between two events, the user can press the button on the bottom left side of the screen and the main screen will change to show the trips.

When a trip is selected the user can manage it, changing its characteristics like the means of transport, the number of passenger various preferences.

When the application need to communicate messages to the user, while it’s open and the screen is turned on, it will show a pop up, otherwise it will send a notification to the user.

*3.A.2 Hardware Interfaces*

The main hardware interface used by the system is the GPS, it’s used in order to correctly position the user in the map. Since the application uses the internet connection, all the hardware required to connect to the internet will be hardware interface for the system

*3.A.3 Software Interfaces*

The mobile application is made up using mainly two Google APIs: Google Maps and Google Calendar. It relies also on other APIs: one for the weather forecast and one for each car sharing, bike sharing service and for the public mobility.

It is developed for the use on the two most common mobile operating systems: Android and iOS.

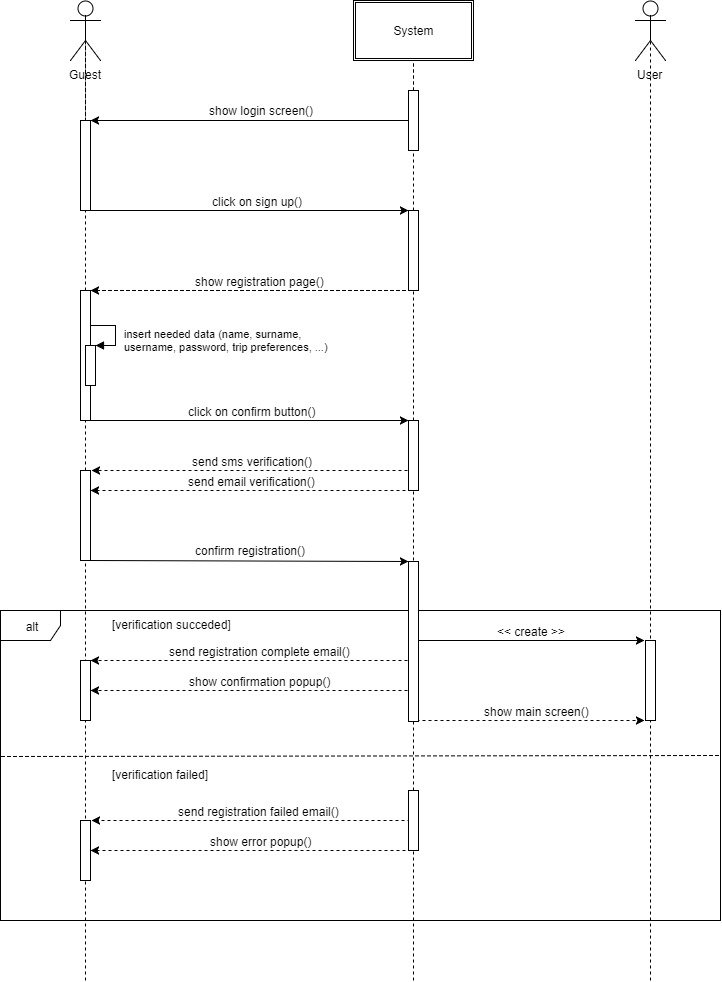
In the future it will employ APIs to buy tickets without using the built-in browser.

*3.A.4 Communication Interfaces*

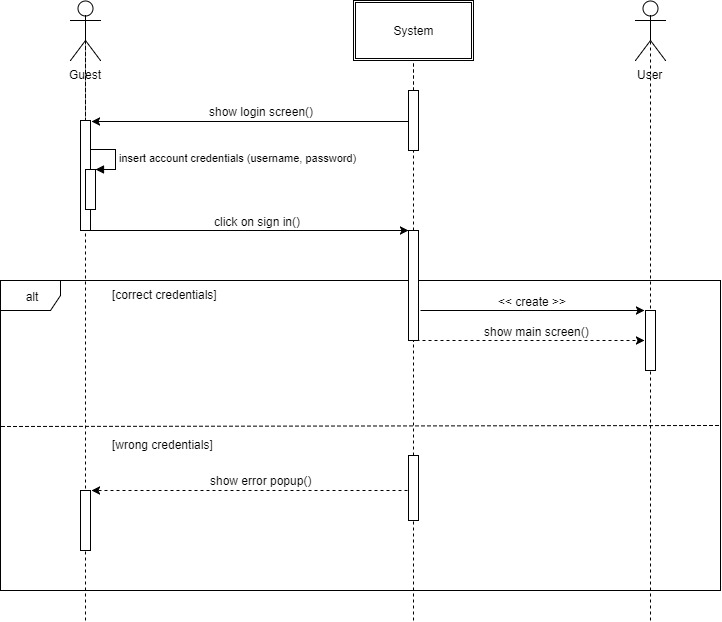
The application communicates with the server using the protocol HTTP (port 80).

3.B. Functional Requirements

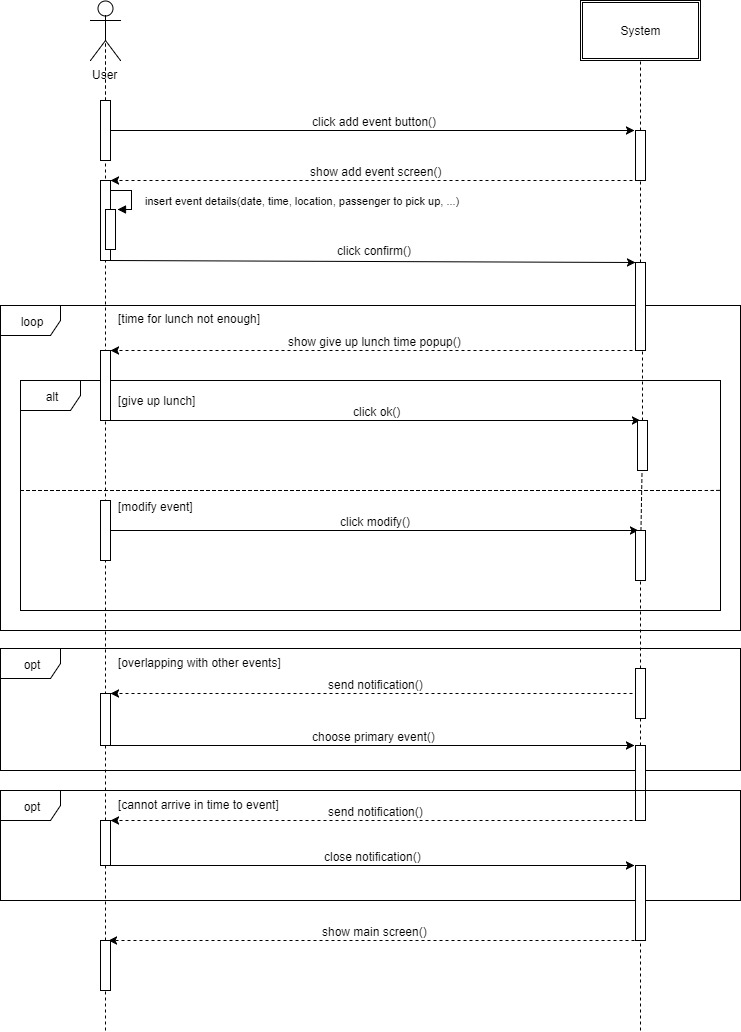
\*\*\*\*\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\*\*\*\* Definition of use case diagrams, use cases and associated sequence/activity diagrams



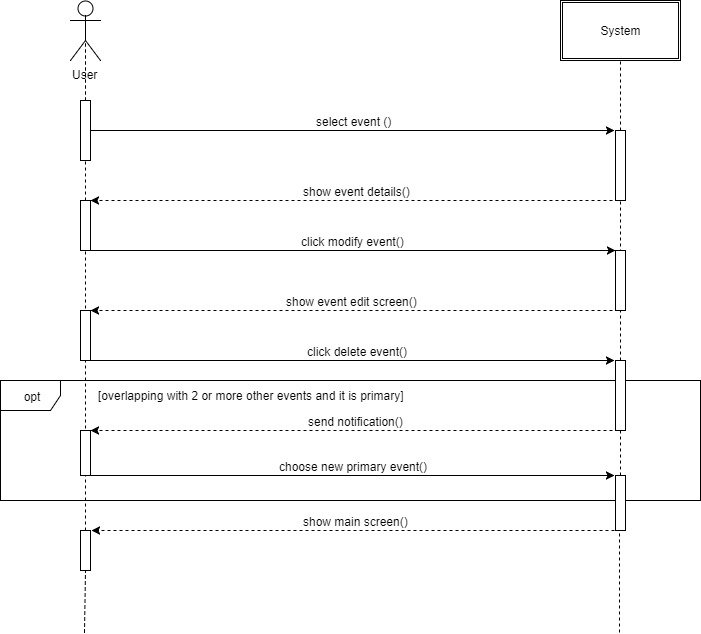
|  |  |
| --- | --- |
| Name | Sign up |
| Primary Actor | Guest |
| Preconditions | The Guest wants to register to “Travlendar+” |
| Postconditions | Guest’s pieces of information are stored in the “Travlendar+” server and locally on the device. The Guest can sign in to use the application, becoming a User. |
| Flow of events | 1. The Guest opens the application for the first time 2. The System shows the Login screen to the Guest 3. The Guest clicks on “Sign Up” 4. The System shows to the Guest the Registration page 5. The Guest inserts his personal details (name, surname, date of birth, …) and his trip preferences (important addresses, owned car, season ticket, …) 6. The Guest reads and accepts the user agreement 7. The Guest taps on “Confirm” 8. The System check the correctness of the data and sends an email and a sms with a verification link 9. The Guest confirms his registration clicking on one of the verification links 10. The Guest is now registered and becomes a User of “Travlendar+” 11. The System shows to the User the Main screen (Calendar) of the application |
| Exceptions | 1. One or more fields of the Registration page are not well formed 2. Username is already in use 3. Email is already in use 4. The verification link is expired (after 24 hours) |



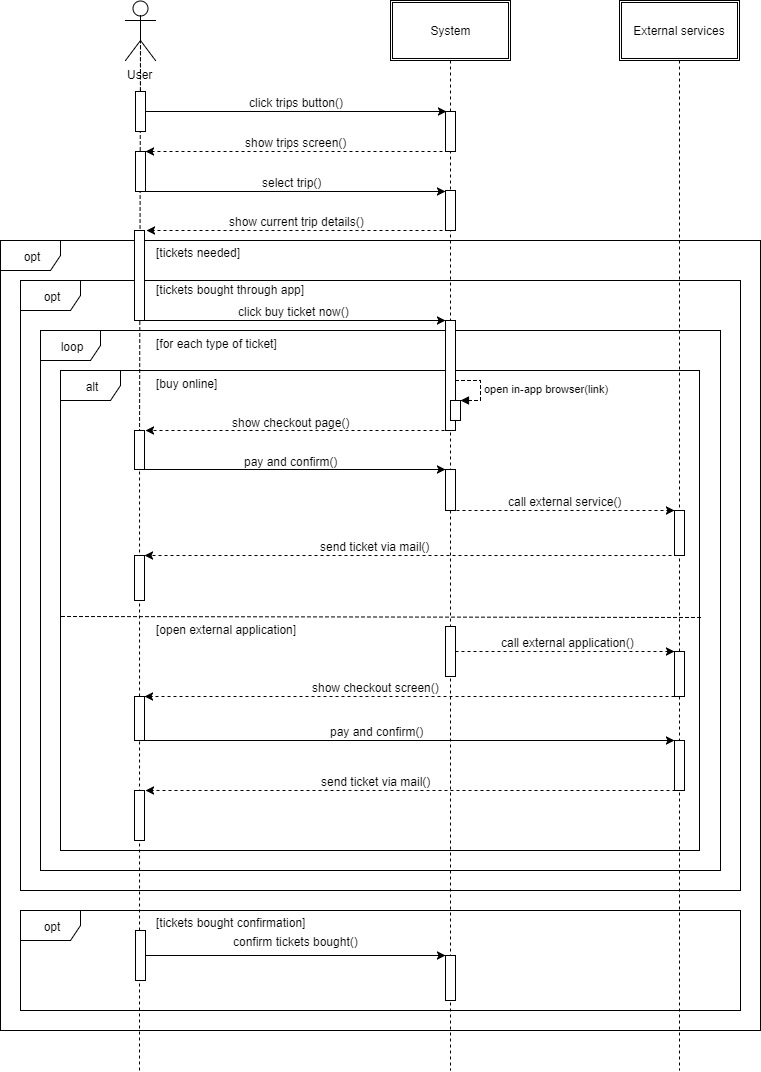
|  |  |
| --- | --- |
| Name | Sign in |
| Primary Actor | Guest |
| Preconditions | The Guest wants to sign in to use “Travlendar+”. He is actually a User of the application because he has a valid account for it |
| Postconditions | The Guest is now logged into the System becoming an User |
| Flow of events | 1. The Guest opens the application 2. The System shows to the Guest the Login screen 3. The Guest inserts account credentials (username and password) 4. The Guest taps on “Sign In” 5. The System checks if the credentials are present in the database 6. The Guest is now logged and becomes an User 7. The System shows to the User the Main screen of the application |
| Exceptions | 1. One or more fields of the Login page are not well formed 2. Username is not present in the database 3. The password associated to the username is incorrect |



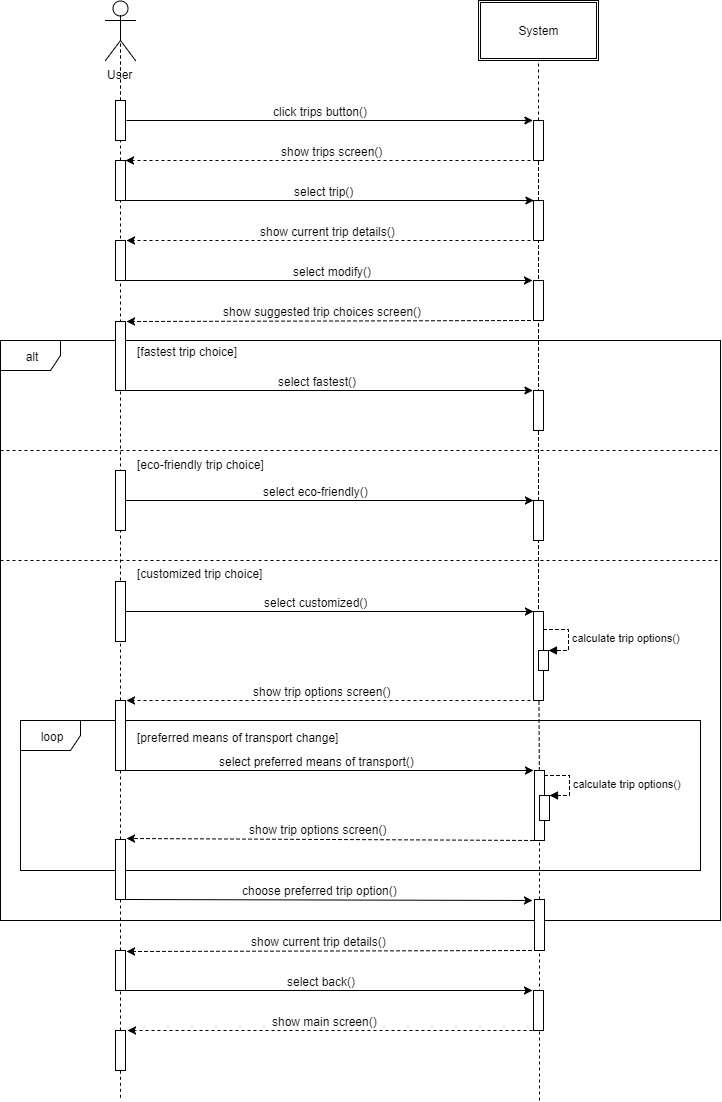
|  |  |
| --- | --- |
| Name | Add event to calendar |
| Primary Actor | User |
| Preconditions | The User wants to add an event to the calendar of his application. He is already logged in into the service and the System is showing the calendar |
| Postconditions | The event is added to the calendar and stored in the database, the System computes the round trip from the location of the event and shows again the calendar to the User |
| Flow of events | 1. The User taps on “Add Event” (the red button with a white cross) 2. The System shows the Add Event screen 3. The User inserts the details in the requested fields 4. The User clicks on “Confirm” 5. The System adds the event to the calendar and shows the updated Main screen to the User |
| Exceptions | 1. The event overlaps with another one 2. The event overlaps with User’s lunch 3. One or more field are not well formed 4. The User cannot arrive in time for the event |



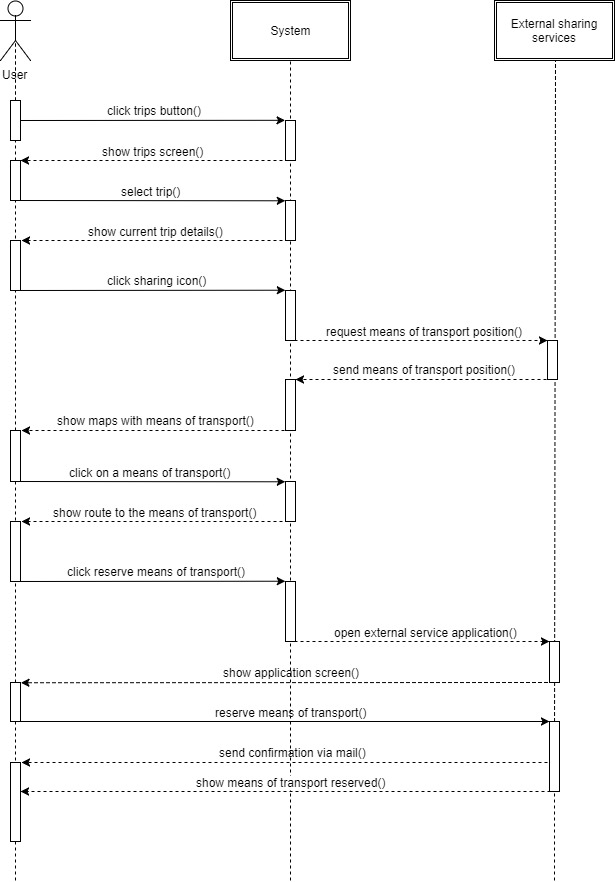
|  |  |
| --- | --- |
| Name | Delete event from calendar |
| Primary Actor | User |
| Preconditions | The User wants to delete an event from the calendar of his application. He is already logged in into the service and the System is showing the calendar |
| Postconditions | The event is deleted from the calendar and from the database, the System shows again the calendar to the User |
| Flow of events | 1. The User selects the event he wants to delete 2. The System shows the Event Details screen 3. The User taps on “Modify Event” 4. The System shows Event Edit screen 5. The User clicks on “Delete Event” 6. The System deletes the event from the calendar and from the database, and shows the updated calendar to the User |
| Exceptions | 1. The event overlaps with at least other two events and it is primary |



|  |  |
| --- | --- |
| Name | Arrange trip |
| Primary Actor | User |
| Preconditions | The User wants to buy the tickets to take a trip. He is already logged in into the service and the System is showing the calendar |
| Postconditions | The User receives the tickets |
| Flow of events | 1. The User clicks on “Trips” (the button with the airplane) 2. The System shows the Trips screen 3. The User chooses the event he wants to buy the tickets for 4. The System shows the Trip details screen 5. The User clicks “Buy Now” 6. The System opens a link to buy the ticket and shows the checkout page to the User for every ticket purchasable online 7. The User pays and confirms for all the tickets 8. The System calls the External services that send the tickets to the User 9. The User communicate to the System that the tickets are bought |
| Alternative flow | 1. The same operations as above until point 5 2. The System calls the External services applications that show the checkout screen to the User, for every ticket only purchasable through the proprietary application 3. The User pays and confirms for all the tickets 4. The External services applications send the tickets to User 5. The User communicate to the System that the tickets are bought |
| Exceptions | 1. One or more payment failed 2. The User doesn’t confirm payment for one or more tickets |



|  |  |
| --- | --- |
| Name | Manage trip options (customized trip) |
| Primary Actor | User |
| Preconditions | The User wants to change the modality of one of his trips, choosing to customize it and changing the preferred means of transport. He is already logged in into the service and the System is showing the calendar |
| Postconditions | The trip is changed as the User prefers |
| Flow of events | 1. The User clicks on “Trips” (the button with the airplane) 2. The System shows the Trips screen 3. The User chooses the event he wants to modify 4. The System shows the Trip details screen 5. The User clicks “Modify” 6. The System shows Suggested Trip Choices screen 7. The User select “Customized” 8. The System calculates all available trip options and shows them to the User in the Trip Options screen 9. The User chooses new preferred means of transport 10. The System recalculate all the trip options with the User selection and show them to the User in the Trip Options screen 11. The User chooses his preferred trip option 12. The System shows the Trip details screen 13. The User selects back (the arrow icon) 14. The System shows the Main screen to the User |
| Exceptions | 1. User chooses to go only by foot but the distance to walk is wider than the one set in the preferences 2. User chooses to go only by bike, but the time is not inside the interval chosen in the preferences |



|  |  |
| --- | --- |
| Name | Reserve a sharing means of transport |
| Primary Actor | User |
| Preconditions | The User wants to reserve a sharing means of transport. He is already logged in into the service and the System is showing the calendar |
| Postconditions | The User has reserved the means of transport chosen and receives a confirmation via mail |
| Flow of events | 1. The User clicks on “Trips” (the button with the airplane) 2. The System shows the Trips screen 3. The User chooses the event that is active 4. The System shows the Trip details screen 5. The User clicks on the “Sharing” button 6. The System requests the position of the means of transport to all External Sharing Services and it shows to the User 7. The User chooses a means of transport 8. The System opens the application of the Sharing Service chosen 9. The User reserves the means of transport 10. The Sharing Service application sends a confirmation via mail and shows the means of transport reserved |
| Exceptions | 1. The Sharing Service application is not installed on the device 2. The User doesn’t have an account for the Sharing Service |

3.C. Performance Requirements

3.D. Design Constraints

*3.D.1 Standards compliance*

*3.D.2 Hardware limitations*

*3.D.3 Any other constraint*

3.E. Software System Attributes

*3.E.1 Reliability*

The system shall have an availability of 99.95% (“three and a half nines”). It means that the application will have at most a downtime per year of 4.38 hours.

*3.E.2 Availability*

The system will run 24/7 in order to make the user manage his events and the trips between them whenever he wants.

*3.E.3 Security*

If the user doesn’t choose to sync Travlendar+ with his Google account, all relevant data will be encrypted and stored locally on the device of the user, so that there won’t be transfers of data potentially subject to security attacks. Otherwise the application relies on Google servers and security system to store and protect relevant data.

*3.E.4 Maintainability*

The application code will be well documented to let future developers understand how it work and to make them able to modify it.

*3.E.5 Portability*

The application will be available for the two most common mobile operating systems: Android and iOS.

*5. Effort Spent*

6. References