TRAVLENDAR+

POLITECNICO
MILANO 1863

Requirement Analysis and Specification Document

29/10/2017 - v1.0

Matteo Colombo   Alessandro Perego   Andrea Troianiello

Deliverable:	RASD
Title:	Requirement Analysis and Verification Document
Authors:	Matteo Colombo, Alessandro Perego, Andrea Troianiello
Version:	1.0
Date:	29-October-2017
Download page:	GitHub - ColomboPeregoTroianiello repository

Contents

List of Figures

List of Tables

1 Introduction

1.1 Purpose

This document is the baseline for project planning and for software evaluation of Travlendar+; it describes the system in terms of functional and nonfunctional requirements, analysing the needs of the customer in order to model the system.

Travlendar+ is a calendar-based application for people who have problems with scheduling working and personal appointments at various locations all across Milan. The application aims at simplifying the life of people by automatically computing travels and by organizing the daily schedules.

1.2 Scope

Travlendar+ is a calendar-based application whose purpose is to help people to easily schedule working or personal meetings around Milan.

The system-to-be will allow users to plan travels and meetings without worrying of being late or to miss lunch and, thanks to its high customizability, people will be able to save a lot of time. Alerts will be given when meetings are created at unreachable locations and travel means will be computed depending on day hours and weather.

1.2.1 Current System

Travlendar+ will rely on many online services provided by other companies.
An existing payment system will be implemented, as well as a maps and weather service that will be used to compute the best travel means.

The journeys will use public transports, trains and bike sharing systems that are already present in the city.

1.2.2 Goals

[G.1] Users can create an account.

[G.2] Allow the users to create meetings.

[G.3] Show warnings in case of unreachable appointments.

[G.4] Suggest travel means depending on the appointment and the day.

[G.5] Allow the users to select preferences and to filter options.

[G.6] Schedule the users' lunch break.

[G.7] Allow the users to schedule breaks during the day.

[G.8] Assist the users during the travel

1.3 Definitions, Acronyms, Abbreviations

1.4 Definitions

- Schedule: set of meetings of the same day.
- Calendar: set of the schedules and from this you can select a specific schedule.

1.4.1 Acronyms

- RASD: Requirement Analysis and Specification Document
- IEEE: Institute of Electrical and Electronic Engineers
- API: Application Programming Interface

1.4.2 Abbreviations

- [G.x]: the goal number x
- [R.x.y]: the requirement number y of the goal x
- [D.x]: the domain assumption number x

1.5 Documents References

- Specification Document: “Mandatory Project Assignments.pdf”.
- IEEE Std 29148-2011 - ISO/IEC/IEEE International Standard - Systems and software engineering

2 Overall Description

2.1 Product Perspective

Travlendar+ will be developed from scratch and it will be a mobile application that will require the user to have a smartphone with an internet connection.

The application will be able to buy transportations ticket in a secure and traceable way by using a trusted payment system. The application also interacts with a partner that give to the user weather information used to calculate the best route to reach the location of the events, in this way depending on the weather conditions different type of transportations are suggested.

The user can also modify the preferences for your trip in such a way to receive suggested of transportation in base of the selected preferences.

2.2 Product Functions

[G.1] Users can create an account.

[G.2] Allow the users to create meetings.

- **[R.2.1]** To create a meeting, the user must be logged in the application with an active account.
- **[R.2.2]** Each meeting must have a location, which can be either an address or coordinates.
- **[R.2.3]** Each meeting must have a starting and ending hour.
- **[R.2.4]** Each meeting must have a name.
- **[R.2.5]** Each meeting must have a type.

[G.3] Show warnings in case of unreachable appointments.

- **[R.3.1]** A warning is given when scheduling a meeting, if the location is unreachable with the available travel means.
- **[R.3.2]** In case of strikes or exceptional events (e.g. train failures), a warning is given to the user.
- **[R.3.3]** Users can create meetings even if a warning is shown.

[G.4] Suggest travel means depending on the appointment and the day.

- **[R.4.1]** Weather conditions must be taken in account when proposing a travel system.
- **[R.4.2]** Different kind of appointments will be reached with different travel means.

[G.5] Allow the users to select preferences and to filter options.

- **[R.5.1]** The application must support many travel means and transport systems.
- **[R.5.2]** Travel means may be subject to constraints (e.g. user can walk for at most 500m).
- **[R.5.3]** Application must show results consistent with the user's preferences.
- **[R.5.4]** Users should be allowed to specify their own passes.

- **[R.5.5]** Users should be able to specify their home address.

[G.6] Schedule the users' lunch break.

- **[R.6.1]** The minimum lunch break duration must be 30 minutes.
- **[R.6.2]** Users must be able to select a time slot in which lunch should be scheduled.

[G.7] Allow the users to schedule breaks during the day.

- **[R.7.1]** The minimum break duration must be 5 minutes.

[G.8] Assist the users during the travel.

- **[R.8.1]** Users should be able to buy transportation tickets.
- **[R.8.2]** The system must locate the nearest vehicle of the selected sharing system.

2.3 User Characteristics

2.3.1 Principal Actors

- Visitor: a person using Travlendar+ without being signed-up. He is able to proceed with registration or log-in.
- User: a person has successful login and can use the app services. He can manage his preferences and his appointments.

2.3.2 Secondary Actors

There are some secondary actors such as third party service providers, that are needed by the system to retrieve information, used to perform payments or to compute the travel options.

2.4 Constraints

2.4.1 Hardware Constraints

1. To use the application, the user must have a smartphone; either Android or iOS.
   For Android the minimum supported OS version is Android 5.0 Lollipop.
   For iOS the minimum required OS version is iOS 8.
2. The application requires an Internet connection.
3. The application requires that devices have an integrated GPS system.

2.4.2 Software Constraints

1. If the users decide to use the bike sharing systems, they must install the systems applications on their phones.

2.4.3 Safety Constraints

1. The system must guarantee that users data are stored in a safety way and that are used only within the application:

2.4.4 Regulatory Constraints

1. The system must ask for the users permissione to acquire, store and elaborate their location.

2. The system must complain with the local laws.

2.5 Assumptions and Dependencies

2.5.1 Domain Assumptions

[D.1] Tickets are not named and they can be bought without owning an account on the service provider's website.

[D.2] During a travel between two meetings, combinations of different travel means could be used.

[D.3] Bus, Trains and all the other public services are always on schedule.

[D.4] An available vehicle of a sharing system service can always be located within a 10 minutes walk.

[D.5] Every vehicle of a sharing system service can be located through GPS.

[D.6] Strikes and system malfunctions are always reported on the companies websites.

[D.7] An account is composed by: username, email and password.

[D.8] Email address is unique for each account.

[D.9] Public transport passes are composed by: validity period, public transport to which they are related and validity area or route.

[D.10] Mobile devices used by the users have GPS and can be located.

[D.11] If "home location" is not set, the position retrieved through GPS is used as initial location for the daily travels.

[D.12] System suggests only the best journey option to the users.

[D.13] Meetings can be of three types: personal, family and working.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The following images represent the mockup of some of the most important pages of the application.

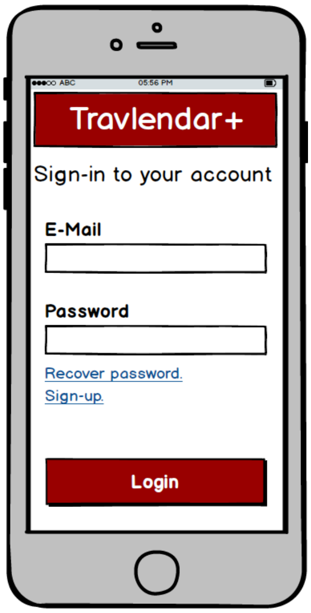


Figure 1: Login page

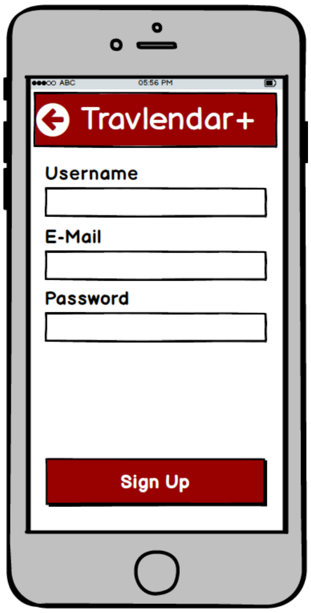


Figure 2: Sign up page

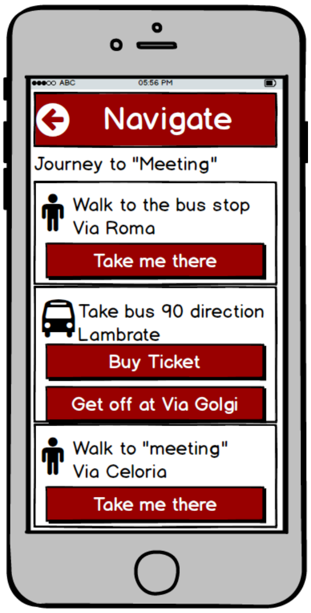

Navigate
Journey to "Meeting"
Walk to the bus stop Via Roma
Take me there
Take bus 90 direction Lambrate
Buy Ticket
Get off at Via Golgi
Walk to "meeting" Via Celoria
Take me there

Figure 3: Navigation page

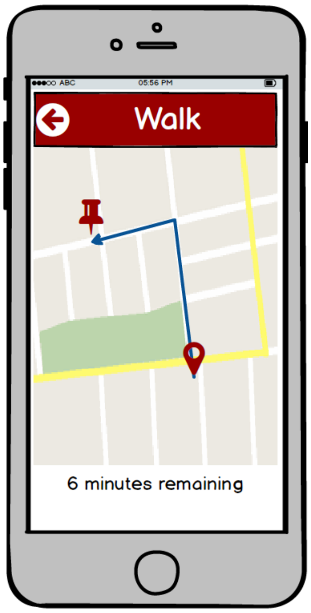


Figure 4: Walking assistant

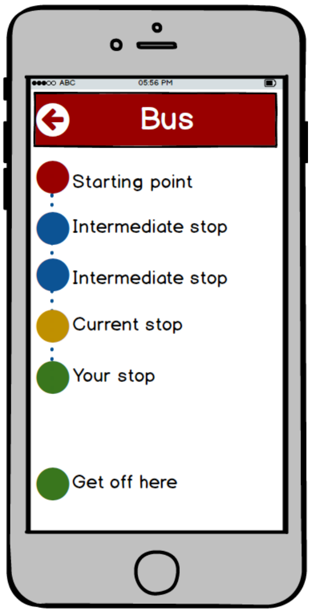
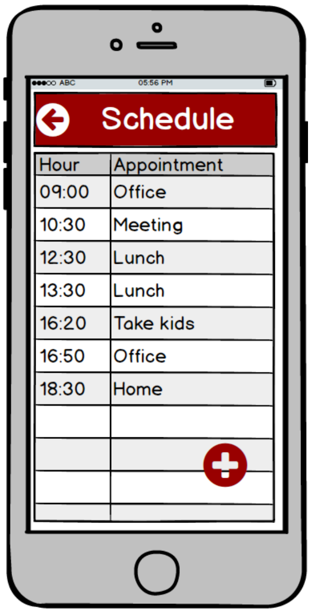
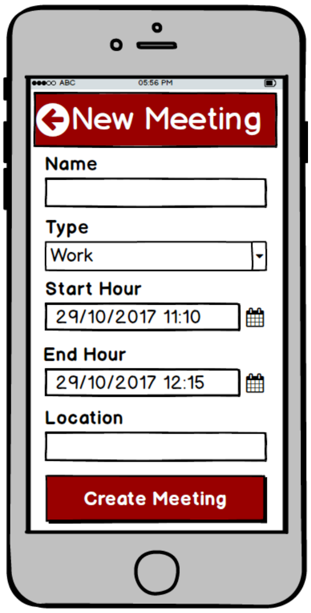


Figure 5: Bus assistant



Hour	Appointment
09:00	Office
10:30	Meeting
12:30	Lunch
13:30	Lunch
16:20	Take kids
16:50	Office
18:30	Home

Figure 6: Schedule page



Name

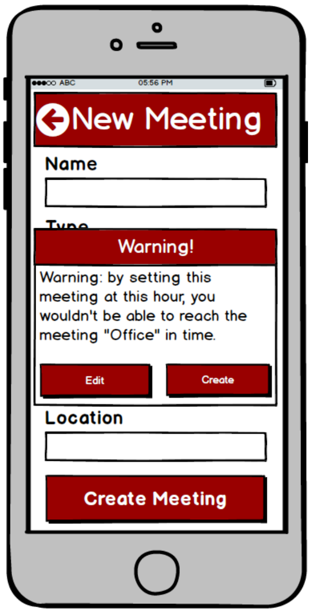
Type
Work

Start Hour
29/10/2017 11:10

End Hour
29/10/2017 12:15

Location

Figure 7: New Meeting



Warning: by setting this meeting at this hour, you wouldn't be able to reach the meeting "Office" in time.

Figure 8: Warning Pop-up

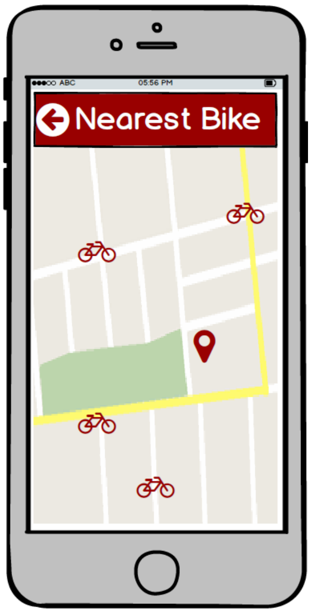
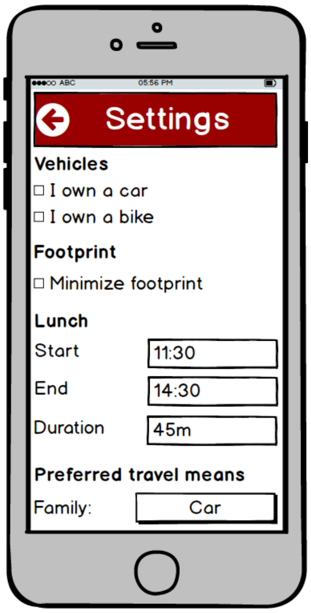


Figure 9: Sharing System



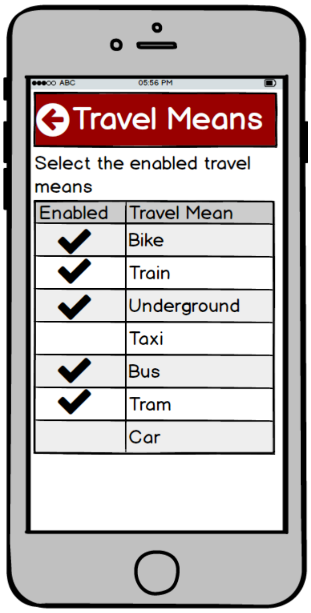
Vehicles
□ I own a car
□ I own a bike

Footprint
□ Minimize footprint

Lunch
Start 11:30
End 14:30
Duration 45m

Preferred travel means
Family: Car

Figure 10: Preferences Page



Select the enabled travel means

Enabled	Travel Mean
✓	Bike
✓	Train
✓	Underground
	Taxi
✓	Bus
✓	Tram
	Car

Figure 11: Travel Means

3.1.2 Hardware Interfaces

This project doesn't require any hardware interface.

3.1.3 Software Interfaces

- Database Management System (DBMS):
  - Name: MySQL
  - Version: 5.6.21
  - Source: https://www.mysql.com
- Operating System:
  - Name: Android
  - Version: 5.0 Lollipop or higher
  - Source: https://www.android.com/

  - Name: IOS
  - Version: 8 or higher
  - Source: https://www.apple.com/ios
- Application Server:
  - Name: Glassfish
  - Version: 4.1
  - Source: https://javaee.github.io/glassfish/

3.1.4 API Interfaces

For map visualization and to show the track of the journey we use the Google Maps API. This API provides the best and most complete maps system. We can retrieve real time information for road traffic and use these to compute a precise travelling time.

For the weather information we use the OpenWeatherMap API. This API gives access to current weather data for moreover than 200.000 cities on the world and it takes data from more than 40.000 wheater stations. The data is avaible in various formats: JSON, HTML and XML. We use the 3 hours weather forecasts. For more information see the OpenWeatherMap website (http://openweathermap.org/api).

To retrive the information of the schedule and anomalies of train, bus, tram and underground we have an agreement with the companies that provide the services. With this agreement we can use their data to calculate the best track for our users and an advise them when for some reason a service is not avaible.

3.1.5 Communication Interfaces

For communications we use the TCP protocol on port 80 for HTTP,port 443 for HTTPS, port 3306 for MySQL database. The data from the server to the application arrive in a JSON format.

3.2 Scenarios

3.2.1 Scenario 1

Mario is the CEO of a manufacturing company in Milan, his company collaborates with many shops located around in the region. Once a month he must visit each shop to get a report of administrative and management activities. To organize and take into account the travel time for the appointments, he uses Travlendar+. After having downloaded the application and registered his data, Mario has the possibility to create a meeting into the app for every appointment that he has in his personal calendar. During the meeting creation, he needs to specify the following information: location for the meeting, starting and ending time, name and type of the appointment. The app automatically calculates if the time between the end of the first meeting and the start of the second one is enough for the journey; in case the time is too short and the second meeting wouldn't be reachable in time, a warning is shown.

3.2.2 Scenario 2

Luca is a young architect of Milan, and in addition to using the latest graphic system to realize his project, he usually produces a demonstration model for his customers scattered around the city. The great capabilities of Luca permits to him to have a lot of appointments, so he uses Travlendar+ to manage the events. Luca has also set the preferences to use car sharing systems for work appointments, in this way he can carry his models around the town without risking of breaking them. The preferences system implemented in the application shows to the user the best itinerary that respects settings and user's preferences.

3.2.3 Scenario 3

Travlendar+ is the perfect application for Giovanni, father of family and financial advisor. With the application Giovanni can easy manage his travel time to reach every type of appointment. Travlendar+ always suggests to him the perfect way to reach a meeting, in case of work meetings the primary suggestion is public transportation, instead in case of family meetings the primary suggestion is his own car. When Giovanni chooses the public transportations for his movement he can also buy the ticket directly on the app, in a simple way he can select the payment system and in a few tap he receives the ticket on his email. Furthermore in case of strike the app sends a notification alert to the user and it will recalculate the travel based on the new options.

3.2.4 Scenario 4

Antonio is an event organizer and meantime he is a food's lover; this is the reason why in his calendar is always presents a lunch in one of the best restaurant of Milan. Antonio is a daily user of Travlendar+ for this reason, the app in fact force a mandatory break of at least 30 minutes for lunch. Antonio can set the length of his lunch break in a period of time between 11.30 am and 2.30 pm. He must set the location where he wants to eat in such way that he can arrive in time for the next appointment. Over the lunch break, Antonio usually plans a break in the afternoon, with Travlendar+ he can do this and the minimum time for the break is 5 minutes

3.2.5 Scenario 5

Alex is a olympic champion of foot race, when he is not training he is often invited to attend some sport conferences. Alex always suggests that it is really important to do physical activity not only during the training sessions but also when we move around the city. For this reason Alex can not do to use Travlendar+, indeed he chooses the bike for his movements and also selects the option to minimize the carbon footprint. Furthermore the application notifies Alex when the weather conditions are not optimal and suggests him to use another travel mean to reach his desiderated location.

3.3 Functional Requirements

3.3.1 Class Diagram

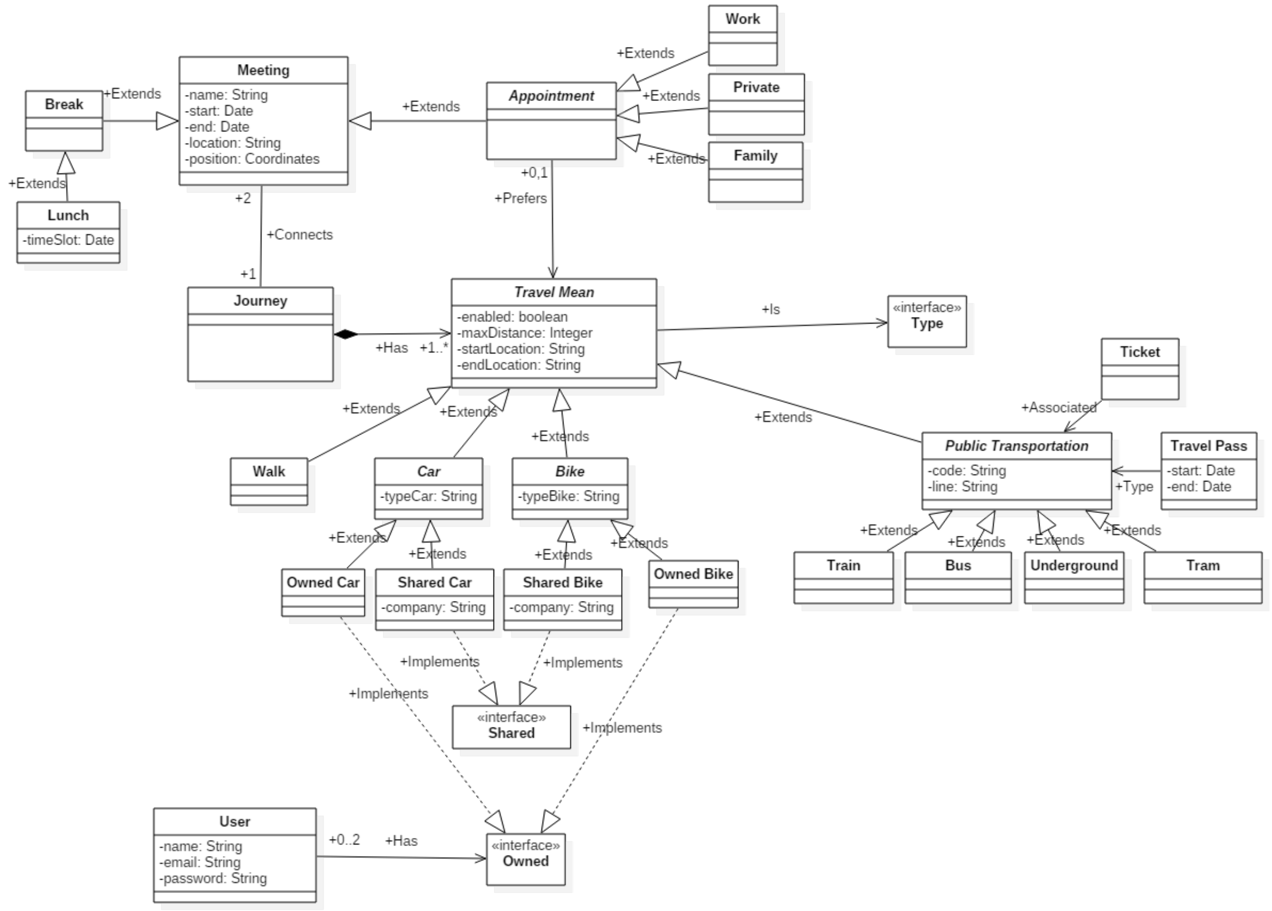


Figure 12: Class Diagram

3.3.2 Use Case Diagram

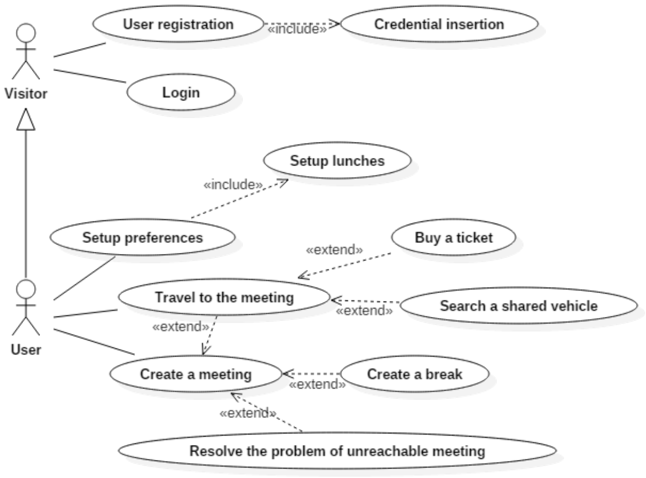


Figure 13: Use Case Diagram

3.3.3 Use Case Descriptions

In this section are listed some common or significant use cases derivable from the Use Case diagram.

User's login

Actor:	User
Goals:	G.1
Input conditions:	The user is on the login page.
Event flow:	1. The user inserts his credentials into the "email" and "password" fields. 2. The user clicks on the "Login" button in order to access.
Output conditions:	The user is successfully redirected to the home page.
Exception:	1. The user inserts a not valid email. 2. The user inserts a not valid password. 3. The server is unreachable. All exceptions are handled notifying the issue to the user and taking back the Event Flow to the point 1.

Visitor's registration

Actor:	Visitor
Goals:	G.1
Input conditions:	The visitor is on the home page.
Event flow:	1. The visitor clicks any button and is redirected to the login page. 2. The visitor clicks on the "Sign Up" button to start the registration process. 3. The visitor fills all the mandatory fields. 4. The visitor clicks on the "Sign Up" button. 5. The system send the data to the server. 6. The system sends a confirmation email to the new user.
Output conditions:	The visitor successfully ends the registration process and is redirected to the login page. From now he can login to the application and start using that.
Exception:	1. The visitor is already an user. 2. The visitor inserts not valid information in one or more mandatory fields. 3. The visitor chooses an email that is associated with another user. 4. The server is unreachable. All exceptions are handled notifying the issue to the visitor and taking back the Event Flow to the point 2.

Table 1: Visitor's registration

Create a meeting

Actor:	User
Goals:	G.2, G.3
Input conditions:	The user is already logged into the system and is into the schedule page.
Event flow:	1. The user clicks on the “New meeting” button to start the creation process. 2. The user inserts the information into the fields and the type. 3. The user clicks on the “Create meeting” button.
Output conditions:	The user is successfully redirected to the meeting page.
Exception:	1. The name isn’t valid. 2. The start hour doesn’t precede the end hour. 3. The start hour and end hour precede the current date. 4. Exists an another meeting with the same hours. 5. Travel mean is unavailable. These exceptions are handled notifying the issue to the user and taking back the Event Flow to the point 2. If the meeting is unreachable, this use case is extended by "Resolve the problem of unreachable meeting".

Resolve the problem of unreachable meeting

Actor:	User
Goals:	G.2, G.3
Input conditions:	The user is creating a meeting and this is unreachable.
Event flow:	The applicaiton shows a warning message and the user can either click "Edit" to change the meeting information or click "Create" and continue creating the meeting even if it is unreachable in time.
Output conditions:	The user is successfully redirected to the meeting page.
Exception:	All exceptions are the same as those the use case "Create a meeting".

Travel to the meeting

Actor:	User
Goals:	G.4, G.8
Input conditions:	The user is already logged into the system and is into the schedule page.
Event flow:	1. The user clicks on the desired meeting and is redirected into the meeting information. 2. The user clicks on “Navigate” and goes into navigate page. 3. The navigate page helps the user with indications. 4. When the user arrives at the desired location, the application notifies him.
Output conditions:	The user is successfully redirected to the schedule.
Exception:	1. The GPS is unavailable. 2. Internet connection isn’t working. 3. The location isn’t found. All exceptions are handled notifying the issue to the user and taking back the Event Flow to the point 1.

Setup preferences

Actor:	User
Goals:	G.5, G.7
Input conditions:	The user is already logged into the system and is into the home page.
Event flow:	1. The user clicks on “Setting” button and is redirected into setting page. 2. The user inserts his preferences about travel means and lunch timetable. 3. The user confirms the new information with “Confirm” button.
Output conditions:	The user stays in the setting page.
Exception:	1. The user doesn’t insert numbers in the lunch fields. All exceptions are handled notifying the issue to the user and taking back the Event Flow to the point 2.

3.3.4 Sequence Diagrams

3.4 Performance Requirements

1. 95% of the requests should be processed within 5 seconds.
2. 100% of the requests should be processed within 20 seconds.
3. There is no limit to the total number of the registered users.
4.

3.5 Design Constraints

3.6 Software System Attributes

3.6.1 Reliability

The system is designed to run on a single server and its reliability depends on the server one. The server is required only for logins an backups of the users’ schedules. In case of downtime, as long as the user is logged in the application, the fault wouldn’t affect the functions and backups would be carried out when the service returns available.

3.6.2 Availability

The system is required to have a 99% uptime. Most scheduled downtimes must occur in the weekends at night time.

3.6.3 Security

All the communications between the clients and the server will be encrypted and protected by using the SSL protocol. All the information will be stored on the server and security will have high priority; users' passwords will not be stored in plain text.

3.6.4 Maintainability

3.6.5 Portability

The backend will be developed in Java so that it will be possible to run the application on every machine that supports Java Virtual Machine. The frontend will be developed in Java for Android and in Swift for iOS and the application must support at least the last three version of each operative system.

4 Formal Analysis Using Alloy

5 Effort Spent