POLITECNICO DI MILANO

Software engineering 2 project:

Requirement Analysis and Specification Document
(R.A.S.D)

version 1.0.2

Travlendar+

Authors:
Giovanni Probo 893698
giovanni.probo@mail.polimi.it

Giovanni Tommasi 900074
giovanni2.tommasi@mail.polimi.it

Academic year 2017-2018

Contents

1. INTRODUCTION **3**
1.1 - PURPOSE 3
1.2 - SCOPE 4
1.3 - DEFINITIONS, ACRONYMS, ABBREVIATIONS 4
1.3.1 - DEFINITIONS 4
1.3.2 - ABBREVIATIONS 6
1.D - REVISION HISTORY 7
1.E - REFERENCE DOCUMENTS 7
1.F - DOCUMENT STRUCTURE 7

2. OVERALL DESCRIPTION **8**
2.A - PRODUCT PERSPECTIVES 8
2.B - PRODUCT FUNCTIONS 9
2.B.1 - CLASS DIAGRAM 11
2.B.2 - STATECHARTS 12
2.C - USER CHARACTERISTICS 13
2.D - DOMAIN ASSUMPTIONS 13

3. SPECIFIC REQUIREMENTS **14**
3.A - EXTERNAL INTERFACE REQUIREMENT 14
3.A.1 - USER INTERFACES 14
3.A.1.1 - LOG IN AND SIGN UP 14
3.A.1.2 - DEFAULT PREFERENCES 15
3.A.1.3 - MAIN PAGE 16
3.A.1.4 - DAILY SETTINGS 17
3.A.1.5 - CALENDAR 18
3.A.1.6 - MEETING DETAILS 19
3.A.1.7 - SUGGESTED ROUTES 20
3.A.2 - HARDWARE INTERFACES 21
3.A.3 - SOFTWARE INTERFACES 21
3.A.4 - COMMUNICATION INTERFACES 21
3.B - FUNCTIONAL REQUIREMENTS 22
3.B.1 - SCENARIO IDENTIFICATION 22
3.B.1.1 - MARQUES AND SIGN UP 22
3.B.1.2 - CLYDE AND CREATING A MEETING 22
3.B.1.3 - XAVIER AND RAINY DAY 22
3.B.1.4 - TIFFANY AND BROKEN CAR 23

3.B.1.5 - JESSE AND BEING LATE 23
3.B.1.6 - MARTHA AND PASSENGERS 23
3.B.2 - USE CASE IDENTIFICATION 24
3.B.2.1 - REGISTRATION PROCESS 24
SATISFIED REQUIREMENTS 25
USE CASE DIAGRAM 25
SEQUENCE DIAGRAM 26
3.B.2.2 - LOGIN PROCESS 27
SATISFIED REQUIREMENTS 28
USE CASE DIAGRAM 28
SEQUENCE DIAGRAM 29
3.B.2.3 - CREATION OF A MEETING 30
SATISFIED REQUIREMENTS 31
USE CASE DIAGRAM 31
SEQUENCE DIAGRAM 32
3.B.2.4 - EXAMINE BEST MOBILITY OPTIONS 33
SATISFIED REQUIREMENTS 34
USE CASE DIAGRAM 34
SEQUENCE DIAGRAM 35
3.B.2.5 - SETTING PREFERENCES 36
SATISFIED REQUIREMENTS 37
USE CASE DIAGRAM 37
SEQUENCE DIAGRAM 38
3.B.2.6 - NAVIGATING THROUGH CALENDAR 39
SATISFIED REQUIREMENTS 40
USE CASE DIAGRAM 40
SEQUENCE DIAGRAM 41
3.B.2.6 - STARTING A JOURNEY 42
SATISFIED REQUIREMENTS 43
USE CASE DIAGRAM 43
SEQUENCE DIAGRAM 44
3.C - PERFORMANCE REQUIREMENTS 45
3.D - DESIGN CONSTRAINTS 45
3.D.1 - STANDARDS COMPLIANCE 45
3.D.2 - HARDWARE LIMITATIONS 45
3.E - SOFTWARE SYSTEM ATTRIBUTES 46
3.E.1 - RELIABILITY 46
3.E.2 - AVAILABILITY 46
3.E.3 - SECURITY 46
3.E.4 - MAINTAINABILITY 46

3.E.5 - PORTABILITY 46

4. FORMAL ANALYSIS USING ALLOY 47

4.A - ALLOY CODE 47

4.B - RESULTS 52

4.C - ALLOY WORLD 53

4.C.1 - METAMODEL 53

4.C.2 - SHOW USER 54

4.C.3 - SHOW 55

4.C.4 - CREATE MEETING 56

4.C.5 - NO DOUBLE MEETING 57

4.C.6 - NO SAME USERNAME 58

5. EFFORT SPENT 59

6. REFERENCES 60

1. INTRODUCTION

1.A - PURPOSE

Travlendar+ is a calendar-based application that automatically computes and accounts for travel time between appointments to make sure that the user is able to reach all of them without being late.
The application suggests travel means, taking in account appointment specifications and other variables based upon a specific day, as weather forecast.
Travlendar+ allows users to define various kinds of preferences, like choosing which travel mean use or setting constraints on distance, departure time or carbon footprints emissions.

1.B - SCOPE

Travlendar+'s aim is to provide some functionalities which should help users organizing meetings during the day.

[G1] - Manage his/her own meetings

[G2] - Move through the city with best mobility options taking in account personal preferences as choice criteria

[G3] - Buy transportation tickets

[G4] - Locate nearest bike from a bike-sharing service

[G5] - Locate nearest car from a car-sharing service

[G6] - Organize lunch breaks

[G7] - Know when get prepared for the journey

[G8] - Know whenever there is a delay in the schedule and know alternative options to reach the meeting in time

1.C - DEFINITIONS, ACRONYMS, ABBREVIATIONS

All the definitions and acronyms listed below represent common words and abbreviations that developers related to a particular meaning in Travlendar+ project context.

1.C.1 - DEFINITIONS

ALTERNATIVE OPTIONS:
Mobility options the system calculates violating user preferences

CALENDAR:
Part of the application showing weekdays and all the event occurring during that period.

CREDENTIALS (USERNAME AND PASSWORD):
Information required in sign up and login operations. Username correspond to the user E-mail while Password is a serie of alphanumeric characters invented by the user whose length is equal or greater than 6 characters.

DAILY PREFERENCES:
Set of settings the user can enable and modify which last until the day ends.
They try to meet user necessities imposed by random external variables (i.e. broken car, less money than expected)

DEFAULT PREFERENCES:
All the possible settings the user can decide the first time he/she logs into the applications. They can be modified in every moment from the settings menu

DEFAULT SETTINGS:
Initial state of settings menu options before user choices.

HURRY-UP NOTIFICATION:
A notification that is sent from the system to the user, that indicates to be ready for the travel.

INCONSISTENCY
The condition of irregularity in which an option get in conflict with system settings or creates a paradox, an antinomy (15 minutes of travelling, necessity to be at the arrival point in 10).

LATE NOTIFICATION:
A notification that is sent from the system to the user when he/she is late.

LATE OPTIONS
Mobility options the system calculates violating time constraints and user preferences in order to reach the meeting as fast as possible.

MEETING:
Event of flexible duration and placement

REGISTRATION:
Operation the user accomplishes providing his/her e-mail and creating a password.
A message will be sent to the user with a proper link to confirm the email address.

SECONDARY OPTIONS:
Not optimal mobility options the system calculates preserving user preferences and time constraints

USER:
Any person who interacts with the application after the registration process.

VISITOR
Any person who interacts with the application before the registration process.

WEATHER FORECAST
Prevision based weather data that combine specific locations with weather categories (i.e sunny, rainy, windy) during a period of time.

1.C.2 - ABBREVIATIONS

3G:
mobile communications standard that allows mobile phones, computers, and other portable electronic devices to access the Internet wirelessly.

API:
a set of functions and procedures that allow the creation of applications which access the features or data of an operating system, application, or other service.

DAT:

Departure advance time.
It is a parameter the user can set to impose the number of minutes
he/she wants to receive a notification before the departure time.

GPS:
Global Positioning System.
It is a radio navigation system that allows land, sea, and airborne users to determine their exact location, velocity, and time 24 hours a day, in all weather conditions, anywhere in the world.

OS:
Operative systems (Android, IOS).

RASD:
Requirements Analysis and Specification Document.

1.D - REVISION HISTORY

- 1.0.0 version - 29/10/2017
- 1.0.1 version - 14/11/2017
- 1.0.2 version - 17/11/2017

1.E - REFERENCE DOCUMENTS

- Specification document : Mandatory Project Assignments.pdf

- Theory: Requirement Engineering.pdf

- Alloy reference: Alloy.pdf

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

- IEEE Std 1016 tm -2009 Standard for Information Technology-System Design-Software Design Descriptions.

1.F - DOCUMENT STRUCTURE

- Section 2: OVERALL DESCRIPTION
  This section explains all the main system aims and functions aside from a basic structure of the project using UML class diagram and statecharts. Some details regarding users identification and domain assumptions are included too.

- Section 3 : REQUIREMENTS
  This section presents all the application interfaces, describing the interaction with users through mock-ups.
  Every functional requirement is analyzed by means of scenarios identification, use case description, use case diagrams and sequence diagrams.
  Performance requirements, standards, limitations, constraints and system attributes are included too.

- Section 4: FORMAL ANALYSIS USING ALLOY
  Formalization and analysis of the project using Alloy Analyzer

- Section 5: EFFORT SPENT
  Time needed by development team to complete RASD document

- Section 6: REFERENCES
  This section lists all the citations and sources of information

2. OVERALL DESCRIPTION

2.A - PRODUCT PERSPECTIVES

Travlendar+ is a mobile application built for the most common OS, available for Android 4.2.2 , IOS 8 and further versions.
Accessibility will depend on users' interactions with a touch-screen smartphone or tablet.
The application will be based on an internal calendar service, programmed in Java.
External data, such as map service and weather forecast, will be retrieved from external web sources (Map Service, Weather forecast service, Car and Bike sharing service servers).

2.B - PRODUCT FUNCTIONS

Ways to interact with the system:

- In the first page users will be able to log in and sign up to the service.
- During registration users will set default preferences.
- Users will be able to create, update and delete meetings.
- During meeting creation users will provide time and location information.
- In "Calendar" page users will be able to visualize all their meetings chronologically ordered and divided by week.
- In "Settings" page users will be able to change default preferences or to restore default settings.
- In "Settings" page users will find "Lunch break" options to set the interval of time for a lunch break.
- In the main page users will find "Change daily settings" button to set daily preferences

System functionalities:

- The system will take in account preferences to create best mobility options and locating nearest bikes or cars from vehicle-sharing systems.

- In the main page the system will show next meetings and best mobility options for each travel.

- The system will provide alternative mobility options in case of strikes, transports delays and bad weather. The user will be warned with a push notification too.

- Clicking on a meeting the system will display secondary mobility options, a small map and a section for buying tickets or booking bikes and cars.

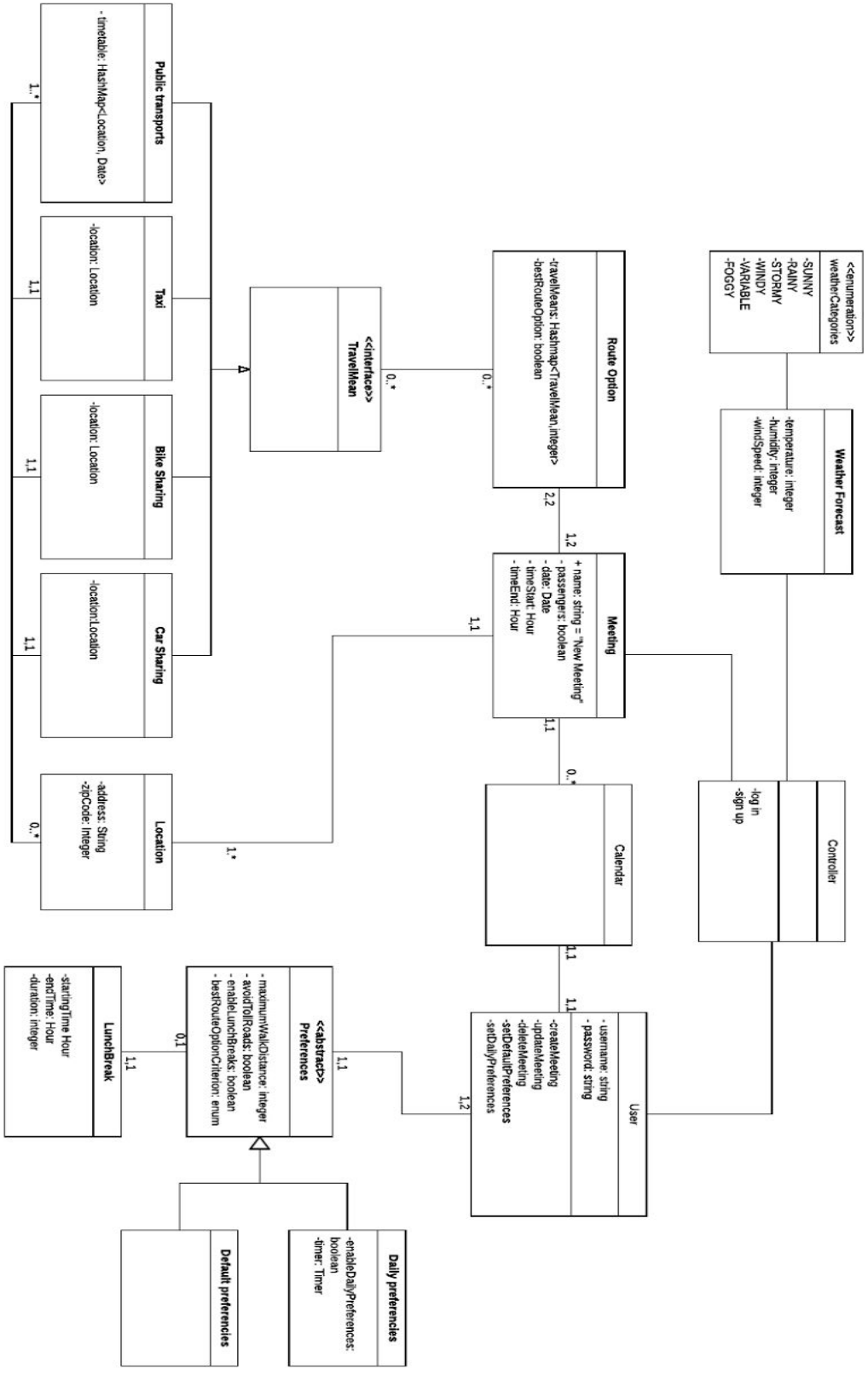
- The system will send a “Hurry Up” notification for departure according to DAT.

- The system will send a “Late” notification when it’s not possible to reach the location in time anymore.

- The system will warn the user with an error message whenever creating a meeting there will be the impossibility to reach the location in time.

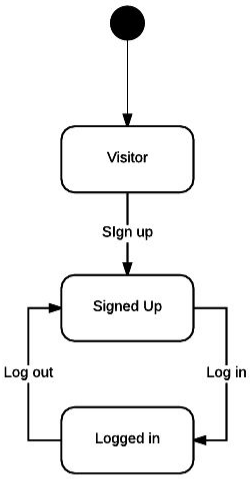
- The system will warn the user with an error message whenever a meeting will overlap lunch break time.

2.B.1 - CLASS DIAGRAM

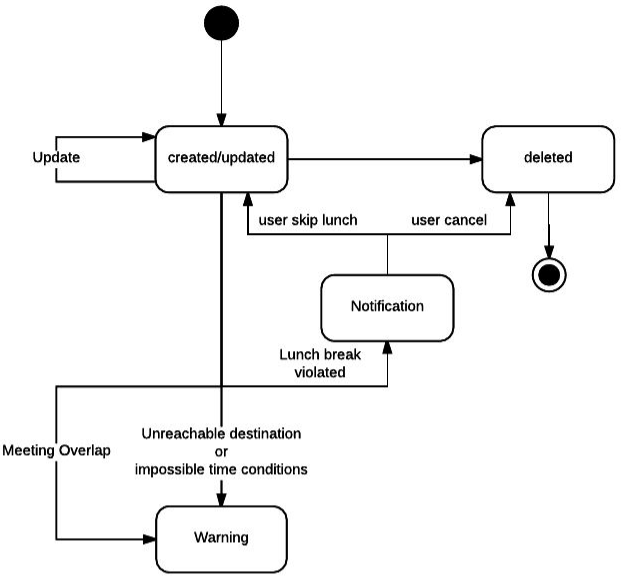


2.B.2 - STATECHARTS

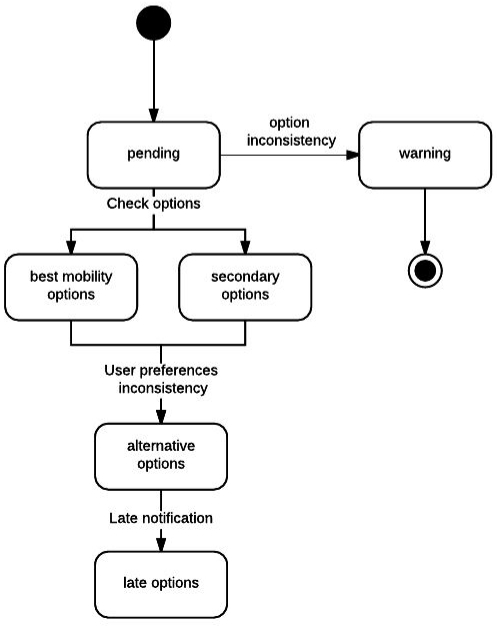
User



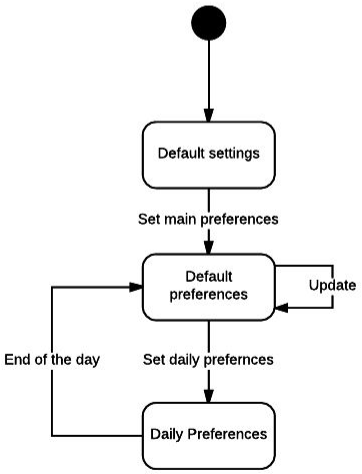
Meeting



Mobility options



Preferences



2.C - USER CHARACTERISTICS

For the application we identified some typologies of actors:

- Final users : people who wish to plan all of their meetings and who live in Lombardy. Users have already signed up in the system and own a smartphone with GPS and internet connection.

- Map service
  provider of maps and mobility options

- Weather forecast service
  provider of weather forecast

- Bike sharing service
  bike sharing system used by the application

- Car sharing service
  car sharing system used by the application

2.D - DOMAIN ASSUMPTIONS

The following hypothesis involve some ambiguous aspect of the specification.
In order to avoid subjectivity we will assume those facts to be always true.

- Users will always try to attend all created meetings
- Users create meetings in existing locations
- It’s always possible to buy ticket of public transports
- Gps on and Internet connection always work properly
- Maps are always updated
- Transport timetables are available online
- Strikes or delays are always known in advance
- Meetings take place in locations for which It’s possible to retrieve weather forecast
- Weather forecasts are always reliable and identifiable with the use of categories

3. SPECIFIC REQUIREMENTS

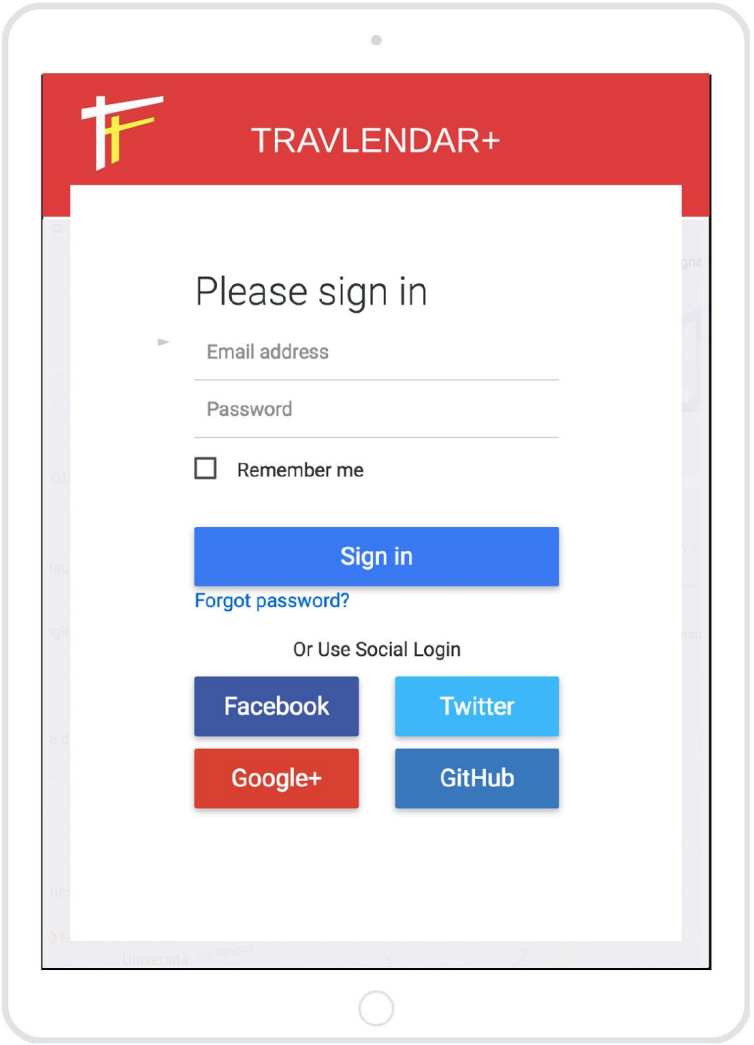
3.A - EXTERNAL INTERFACE REQUIREMENT

3.A.1 - USER INTERFACES

Travlendar+ is designed to be used as a mobile application due to its need to be a portable service.
The following mock-ups show how the user will be able to interact with the application.

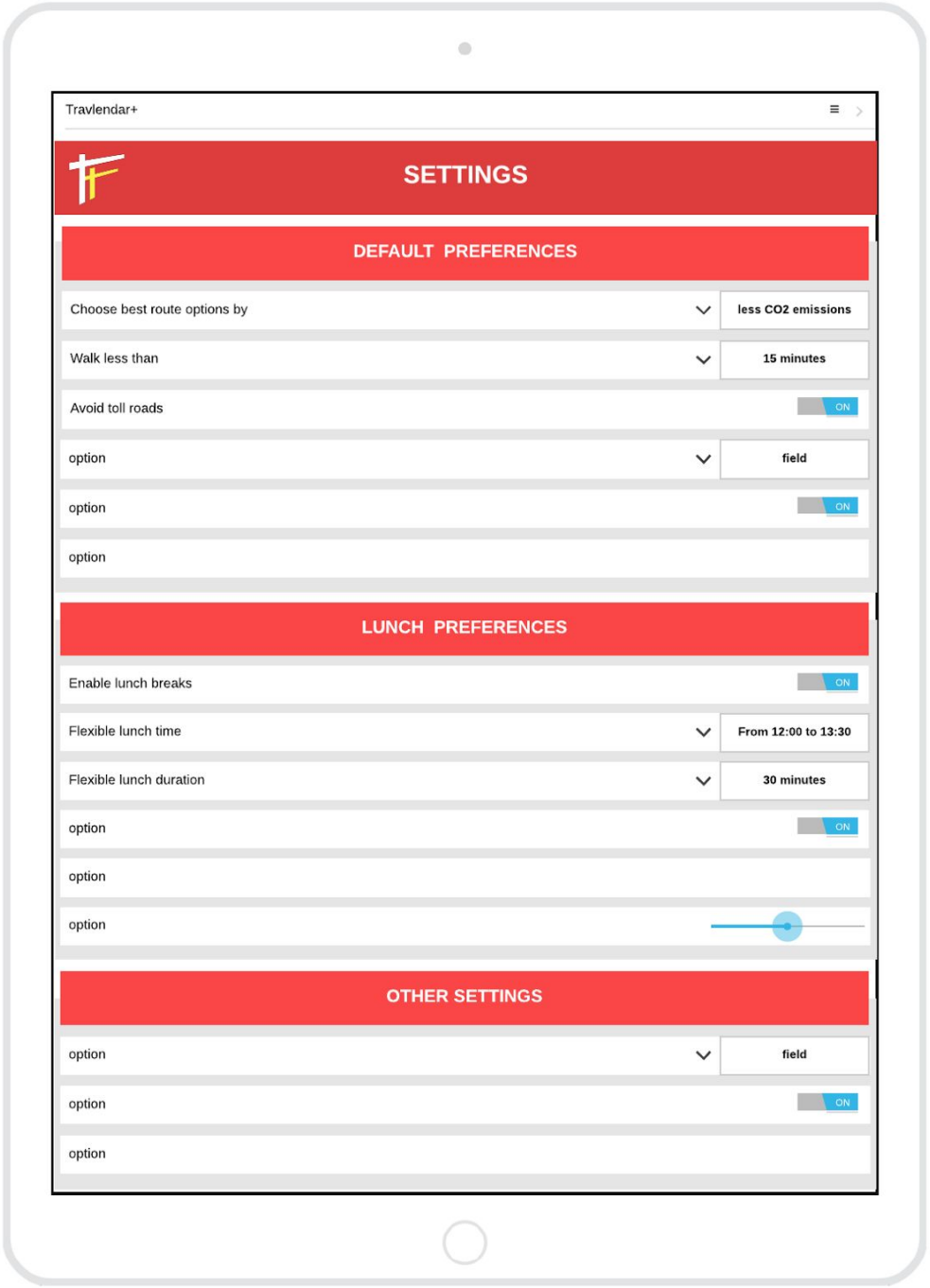
3.A.1.1 - LOG IN AND SIGN UP

This is the first page that will be displayed by the user. Here it is possible to log into the system by inserting credentials, or to sign up and provide e-mail, Username and Password.



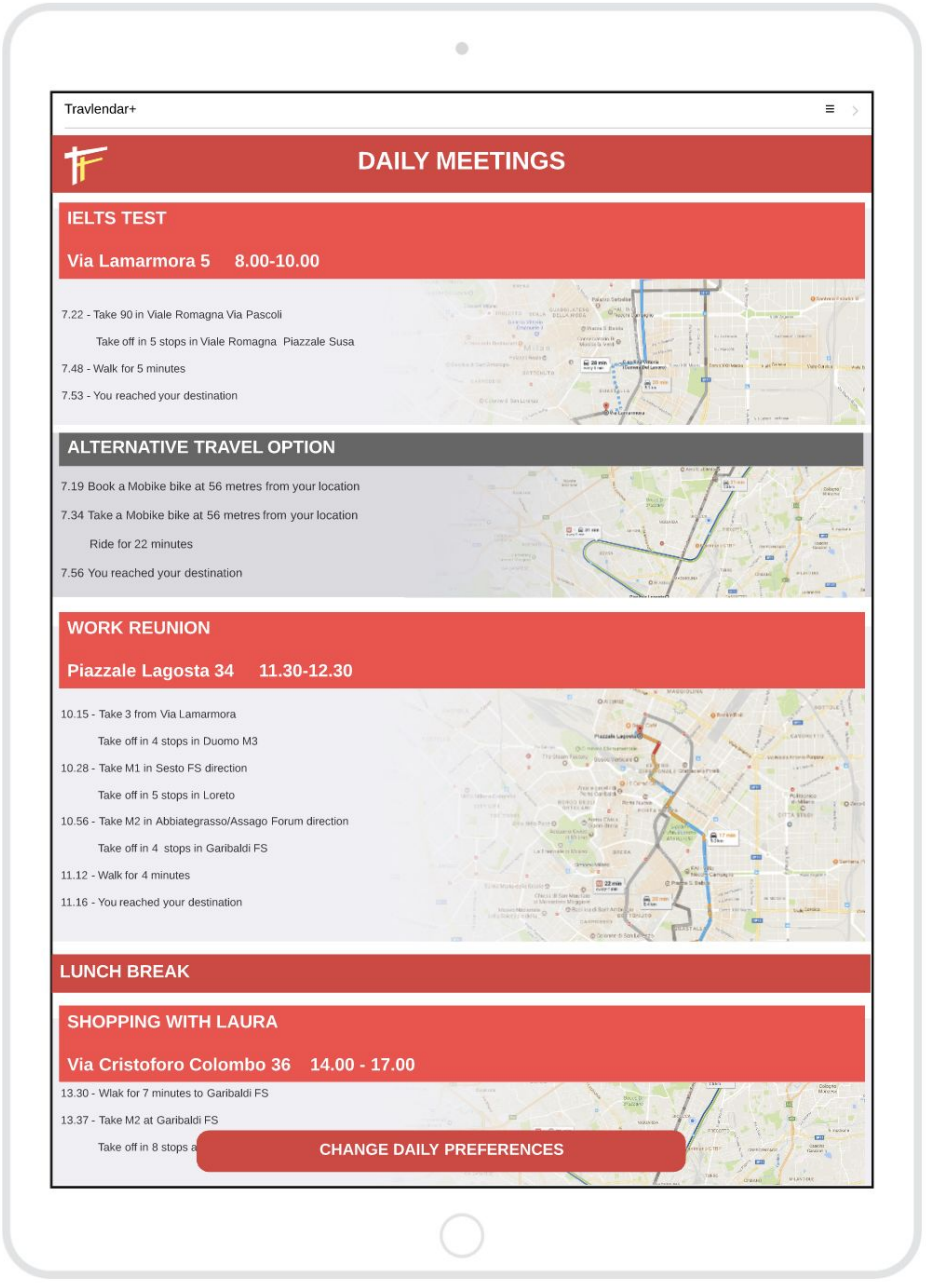
3.A.1.2 - DEFAULT PREFERENCES

When a new user sign up to Travlendar+, he/she should be able to set the his/her default preferences or accept the recommended ones.
This page can also be reached using ≡ button whenever the user desires to change one of the preferences.



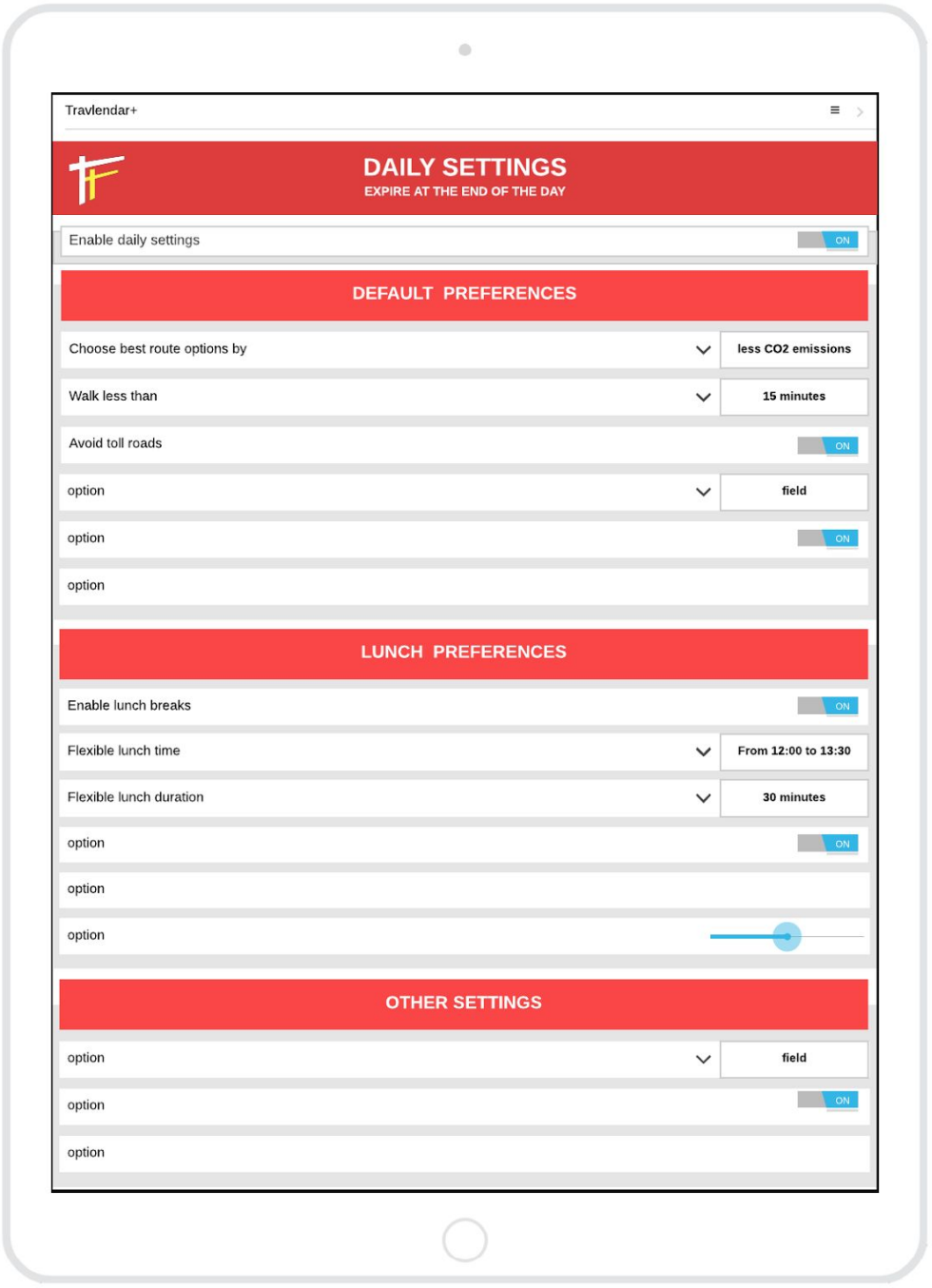
3.A.1.3 - MAIN PAGE

This page will be shown once logged in. It contains all the incoming meetings and each best mobility option. The first incoming meeting will show a secondary mobility option and the lunch break will appear if set.
Clicking on a meeting “Suggested routes” page will be opened.
On the bottom of the page “Change daily settings” button will redirect the user to Daily Settings page.
On the top right ≡ button will let the user moving through the pages (Main, Meetings, Settings).



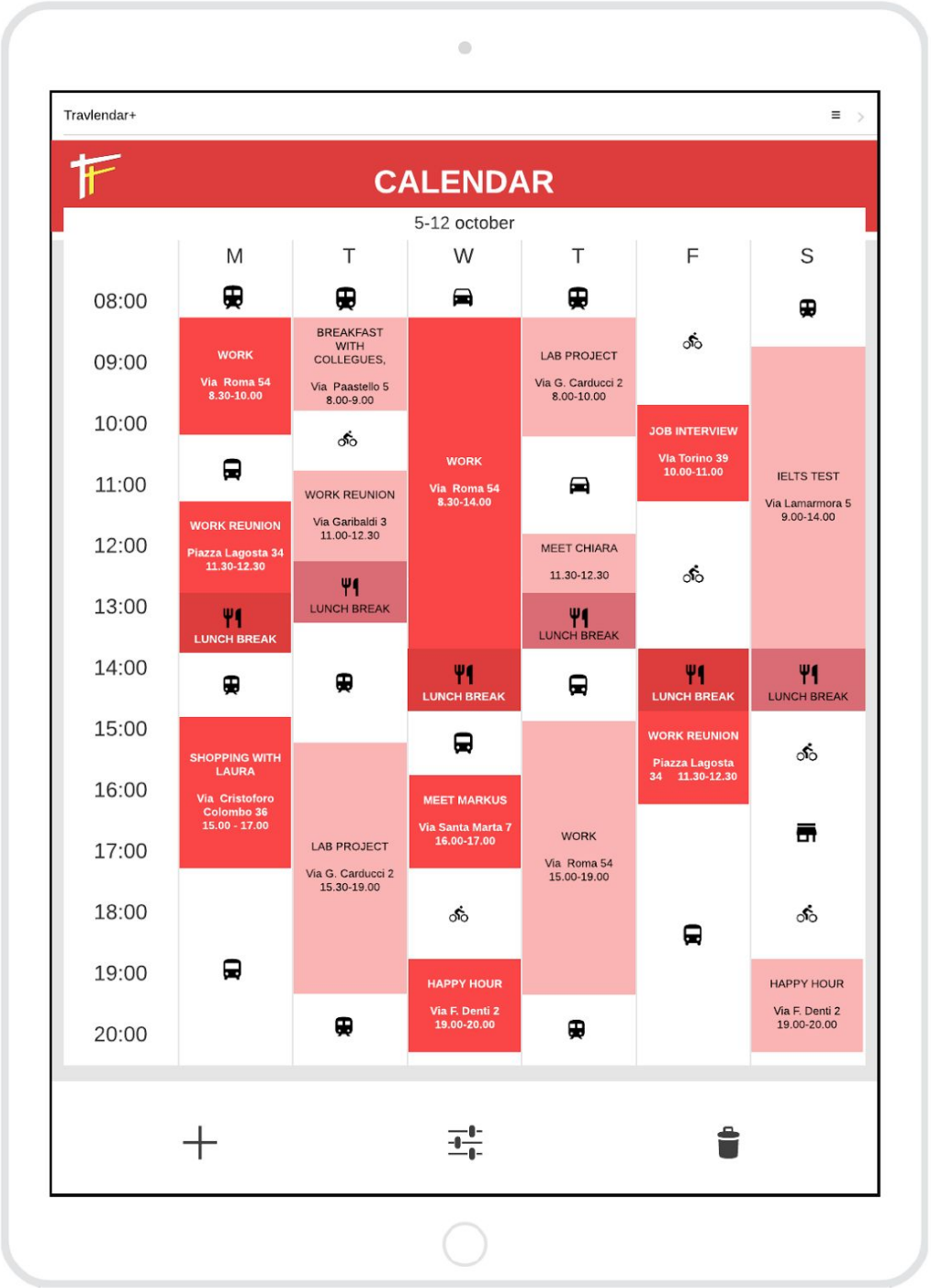
3.A.1.4 - DAILY SETTINGS

This page is quite similar to Default Preferences page. In this page user can modify the preferences for the current day because of unexpected events (i.e. broken car, no money to buy transport tickets etc.).



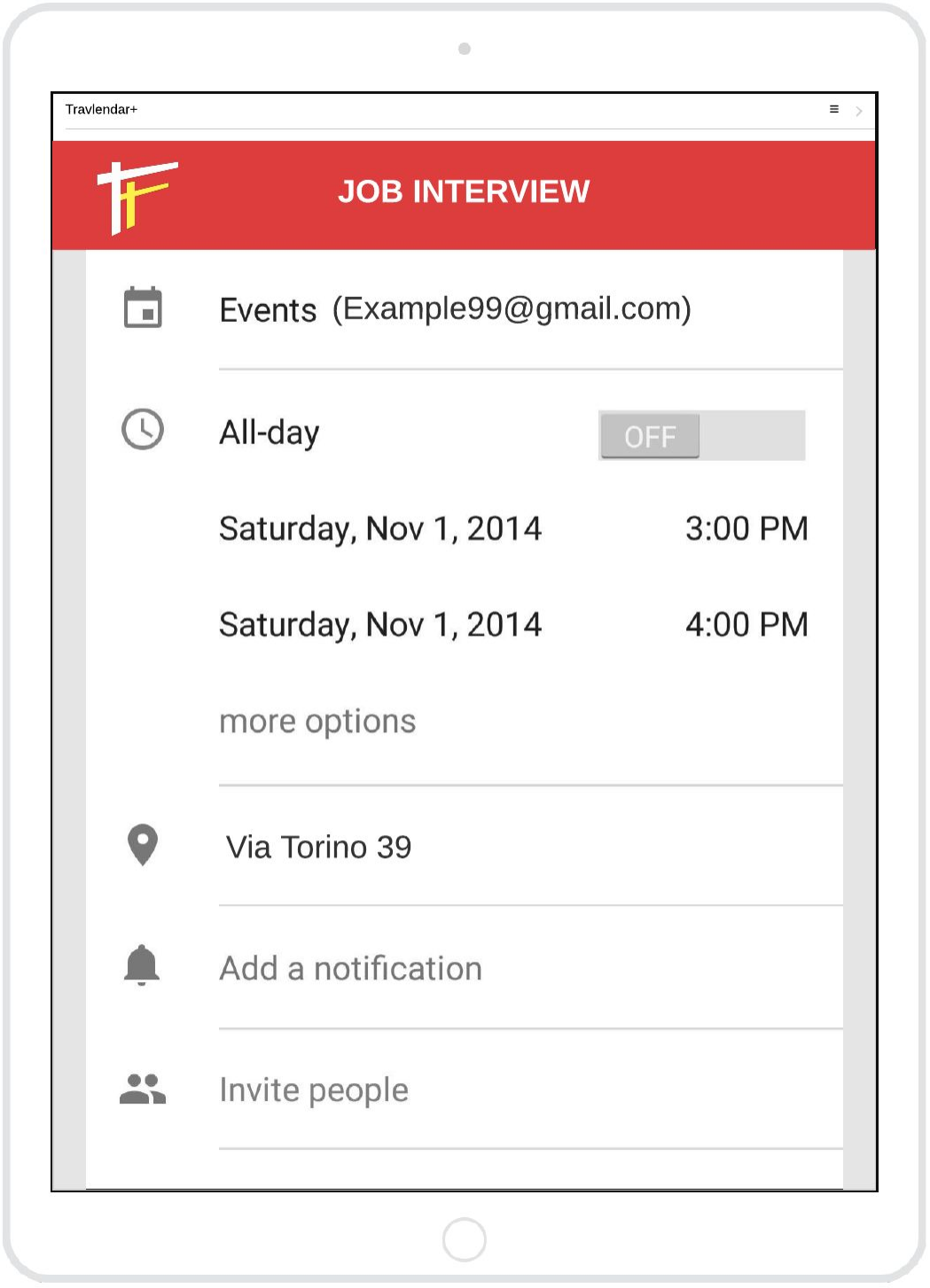
3.A.1.5 - CALENDAR

Calendar page shows a weekly calendar with all the created meetings and some details.
On the bottom three button could let the user to create, modify or delete meetings.
Clicking on a meeting “Suggested routes” page will be shown.
Using “Create” or “Modify” buttons user will be redirected to “Meeting details” page.



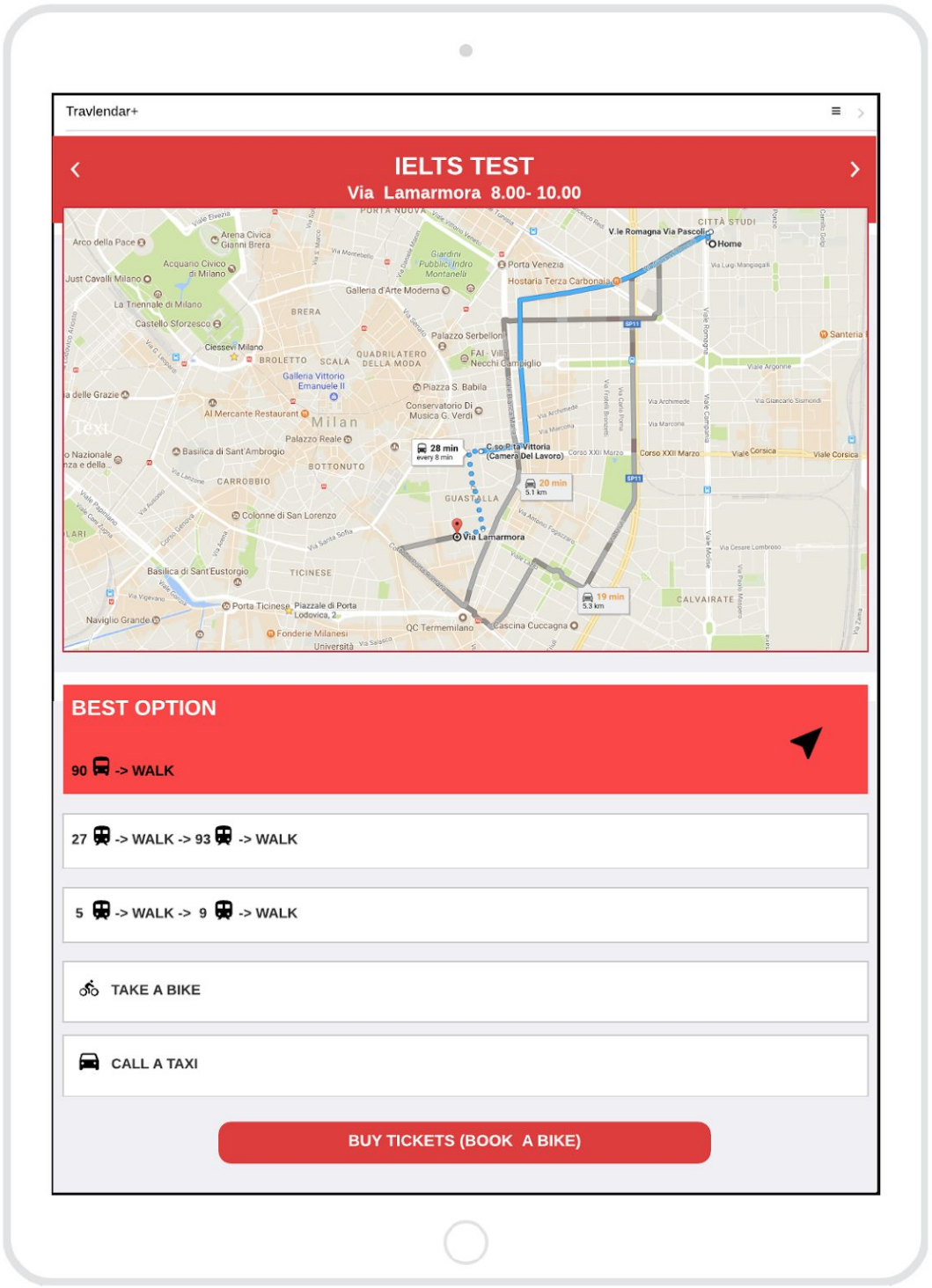
3.A.1.6 - MEETING DETAILS

It contains the name of the meeting, the location, time and further information about the event. Using this page it is possible to create or modify meetings.



3.A.1.7 - SUGGESTED ROUTES

This page shows all the possible travel means that user can use to reach the location
It is also possible to buy transports tickets or to book a bike.



3.A.2 - HARDWARE INTERFACES

The application is designed to guarantee a fluent user experience with touch input devices such smartphones and tablets. Those should even have enough memory space to install the application.

3.A.3 - SOFTWARE INTERFACES

Travlendar+ interfaces with:

- operative systems using his own APIs (need permissions to work properly)
- Map service via Map APIs
- Weather forecast via Weather Channel API
- Car sharing service via its own API
- Bike sharing service via its own API

3.A.4 - COMMUNICATION INTERFACES

Travlendar+ interfaces with:

- operative systems using his own APIs for push notifications
- e-mail provider via Mail API for news and registration process

3.B - FUNCTIONAL REQUIREMENTS

3.B.1 - SCENARIO IDENTIFICATION

3.B.1.1 - MARQUES AND SIGN UP

Marques has just downloaded the application on his smartphone and wants to sign up in Travlendar+ system.
He opens the application and clicks on sign up button. He fulfills all the requested fields with his personal information and then clicks on ‘Submit’.
After a few seconds he receives an e-mail from the system.
The e-mail message contains some welcome information and a link to confirm his e-mail address. Marques opens the link completing the registration process.
Launching the application the default settings is presented, he sets his personal preferences and he’s finally ready to use Travlendar+.

3.B.1.2 - CLYDE AND CREATING A MEETING

Clyde wants to organize a meeting he has arranged with his friend.
He navigates to Calendar page and clicks on “Create” button. He creates a meeting, named “Shopping with Bonnie” in Via Torino 37, from 12:00 pm to 15:30 pm. Travlendar+ notificates Clyde that in Default Preferences he had set at least 25 minutes for lunch from 12:30 pm to 14:00 pm. Travlendar+ asks him with a push notification if he wants to create the meeting anyway or cancel the creation. Clyde choose not to lunch and to create the meeting.

3.B.1.3 - XAVIER AND RAINY DAY

Xavier wakes up at 7:30 am and checks his daily meetings on Travlendar+. His next meeting is “Training with Lionel” at 9:15 am in Parco Lambro. On Default Preferences he sets to use as much as possible his own bike. The app checks on Weather forecast service the weather forecast and gets a ‘Rainy’ result. Travlendar+ sends a push notification to Xavier and creates new mobility options, using only public transportation. In this case he has to leave his house at 8:45 am and take 81 bus for 17 stops. At 8:35 am the application sends a Hurry-up notification. At 8:40 am Xavier goes to the bus stop. At 9:04 am he reaches Parco Lambro and meet Lionel.

3.B.1.4 - TIFFANY AND BROKEN CAR

Tiffany is using Travlendar+ to organize her work meetings. The application suggests her to use her own car for a morning reunion but unfortunately it has broken the day before. She opens the Main page and clicks on “Change daily preferences” button. Tiffany gets redirected to Daily settings page where she can untick “use your own car” checkmark. The system recalculate all the mobility options for her reunion, proposing the bus as the best travelling option.
Tiffany gets the bus and she is able to reach her destination in time.

3.B.1.5 - JESSE AND BEING LATE

Jesse is at work. His next meeting is “Spritz with James” at 7:15 pm, Travlendar+ suggests him to take 9 tram at 6:35 pm. Unfortunately, he had to stay at work until 6:40 pm. At 6:37pm Travlendar+ send a Late Notification to Jesse. After work he opens the meeting page. The app suggest him to book a bike as best late mobility option. Clicking on “Book a bike” button Jesse reserves the bike. He rides for 43 minutes and he reaches the destination at 7:23 pm. Thankfully, James is not angry for the delay.

3.B.1.6 - MARTHA AND PASSENGERS

Martha is a Travlendar+ user who has planned her work meetings for the day. The application creates a mobility option that use only her own bike to avoid the urban traffic, the expected duration of the travel is 24 minutes. In the morning she adds a new meeting called “Take the kids home from school” and puts a checkmark on the preference “meeting with passengers”.
The system recommends her taking the car instead of using her bike, the new expected duration to go to work is higher, around 43 minutes, but at least she can take her kids back home.

3.B.2 - USE CASE IDENTIFICATION

Use Case name:

3.B.2.1 - REGISTRATION PROCESS

Participating actors:

Visitor, User

Entry condition:

A visitor opens the application and wants to sign up in the system.

Flow o events:

- The application display the visitor the signup page
- The visitor submit his/her e-mail and creates a password
- The application send an email to the visitor
- The visitor clicks on the confirmation link becoming a user
- The application displays the default settings page
- The user sets his/her default preference

Exit condition:

The use case terminates when the user confirms all the preferences and is able to use Travlendar+.

Exceptions:

If the visitor sets:

- A wrong e-mail
- A password less than 8 characters long
- A password not containing both upper and lower case letters

Then the application shows a pop-up directly on sign up page.

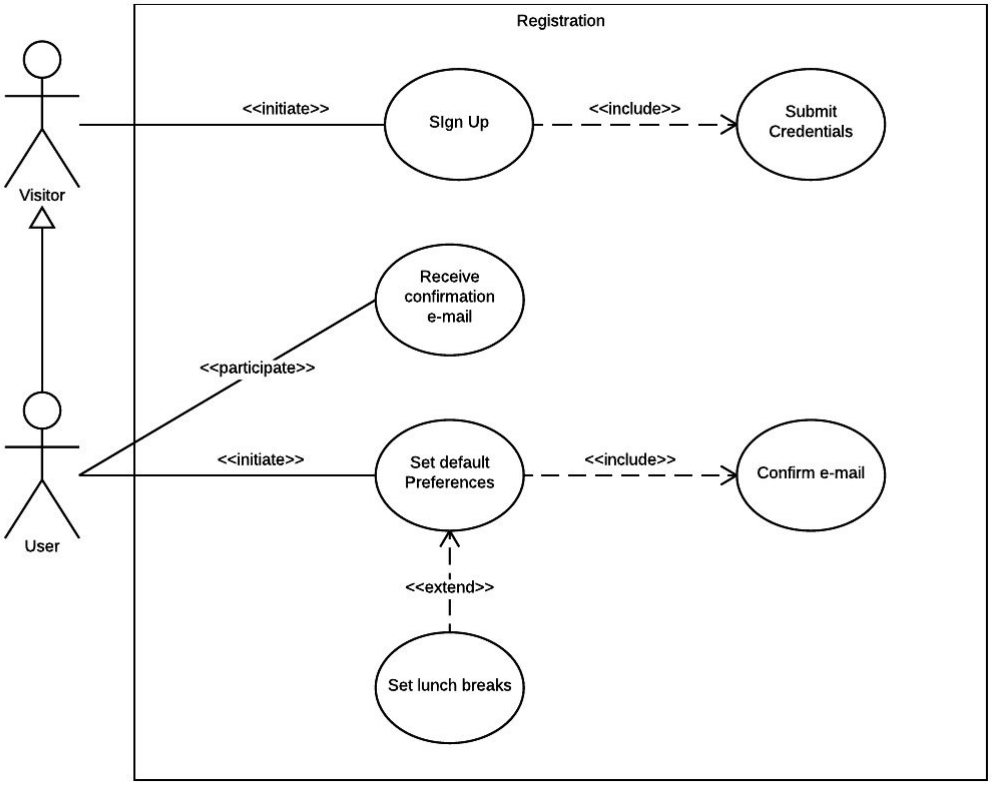
Special requirements:

- User not already signed in
- Application downloaded correctly on user device

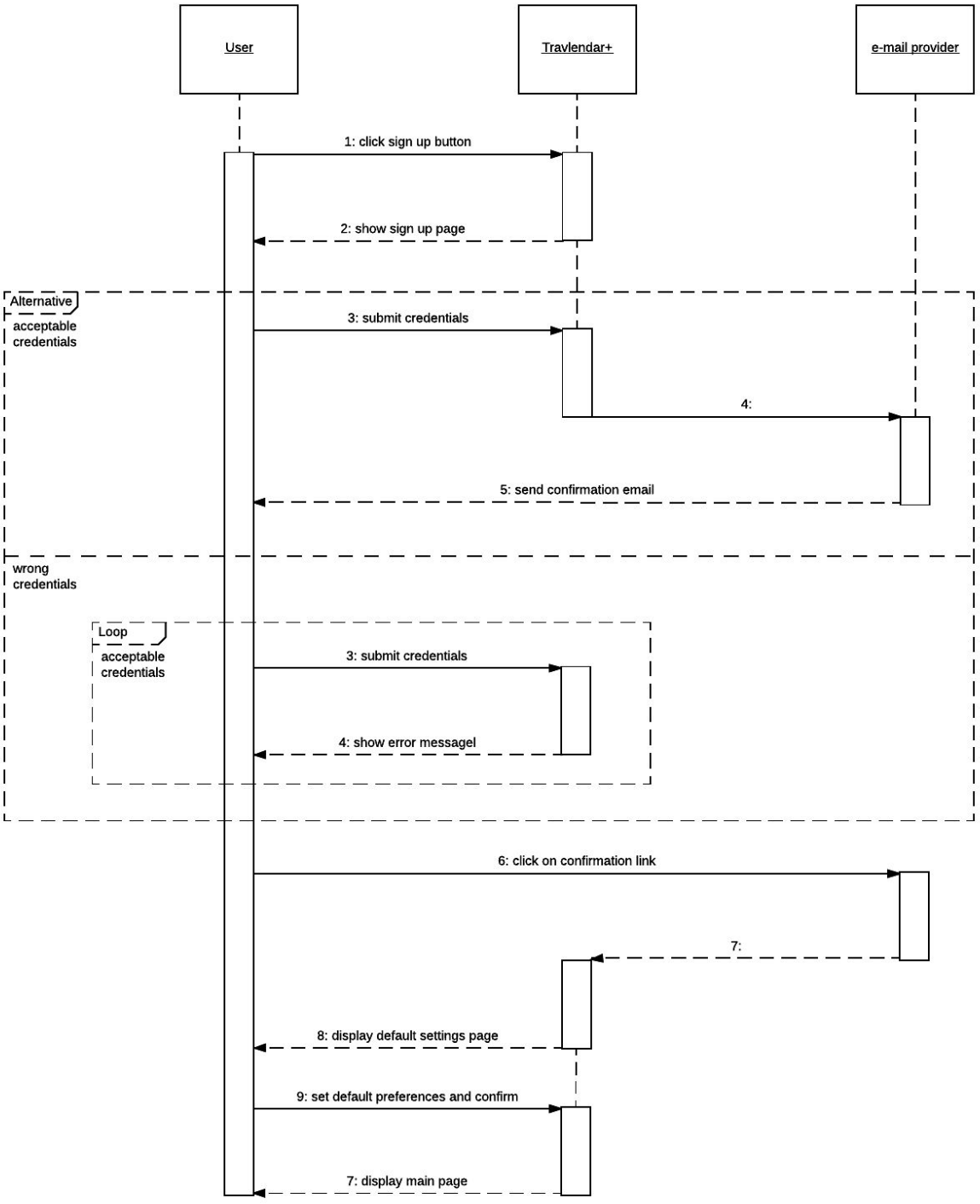
SATISFIED REQUIREMENTS

- The user will be able to sign up to the service.
- During registration users will set default preferences.

USE CASE DIAGRAM



SEQUENCE DIAGRAM

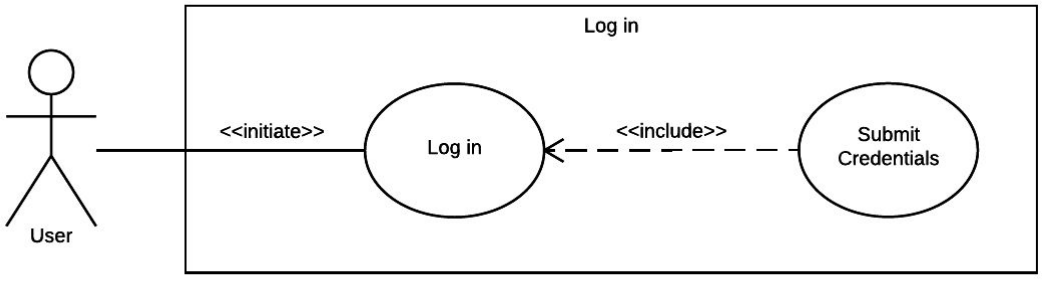


Use Case name: **3.B.2.2 - LOGIN PROCESS**
Participating actors: User
Entry condition: A user opens the application and wants to log in the system.
Flow o events: ● The application display the login page ● The user submit his/her e-mail and password
Exit condition: The use case terminates when the user submit all the credentials and is able to use Travlendar+.
Exceptions: If the visitor submits: ● A wrong e-mail ● A wrong password Then the application shows a pop-up directly on login page.
Special requirements: ● User not already logged in ● Application downloaded correctly on user device

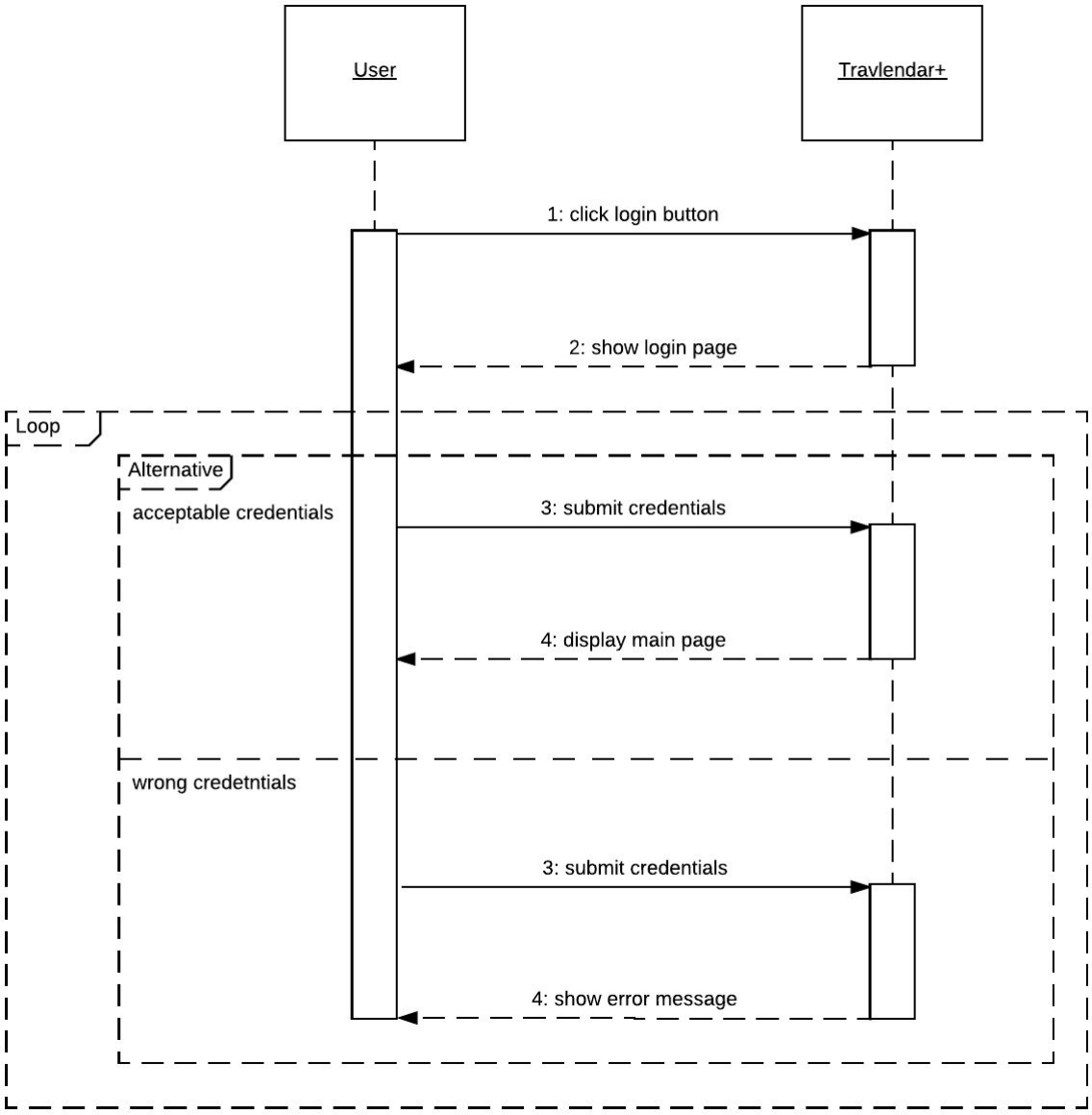
SATISFIED REQUIREMENTS

- In the first page the user will be able to log into the service.

USE CASE DIAGRAM



SEQUENCE DIAGRAM



Use case name:

3.B.2.3 - CREATION OF A MEETING

Participating actors:

User, Map service

Entry condition:

A user wants to create a new meeting and clicks on "New Meeting" button.

Flow of events:

- The application display new meeting page
- User select meeting name, tick multiple passenger checkmark, select date, time and location
- The application checks location and time consistency taking in account meeting overlaps and lunch breaks constraints and give a confirmation

Exit condition:

The use case terminates when the user fill in all the fields and create the meeting.

Exceptions:

If time and location consistency are not respected due to:

- Location too far to be reached by transports or in time
- Overlaps with other meetings
- Lunch break constraints violated (i.e. meeting time not respecting minimum lunch duration)

Then the application sends a warning notification to the user which make him decide if delete the meeting or keep constraints violated.

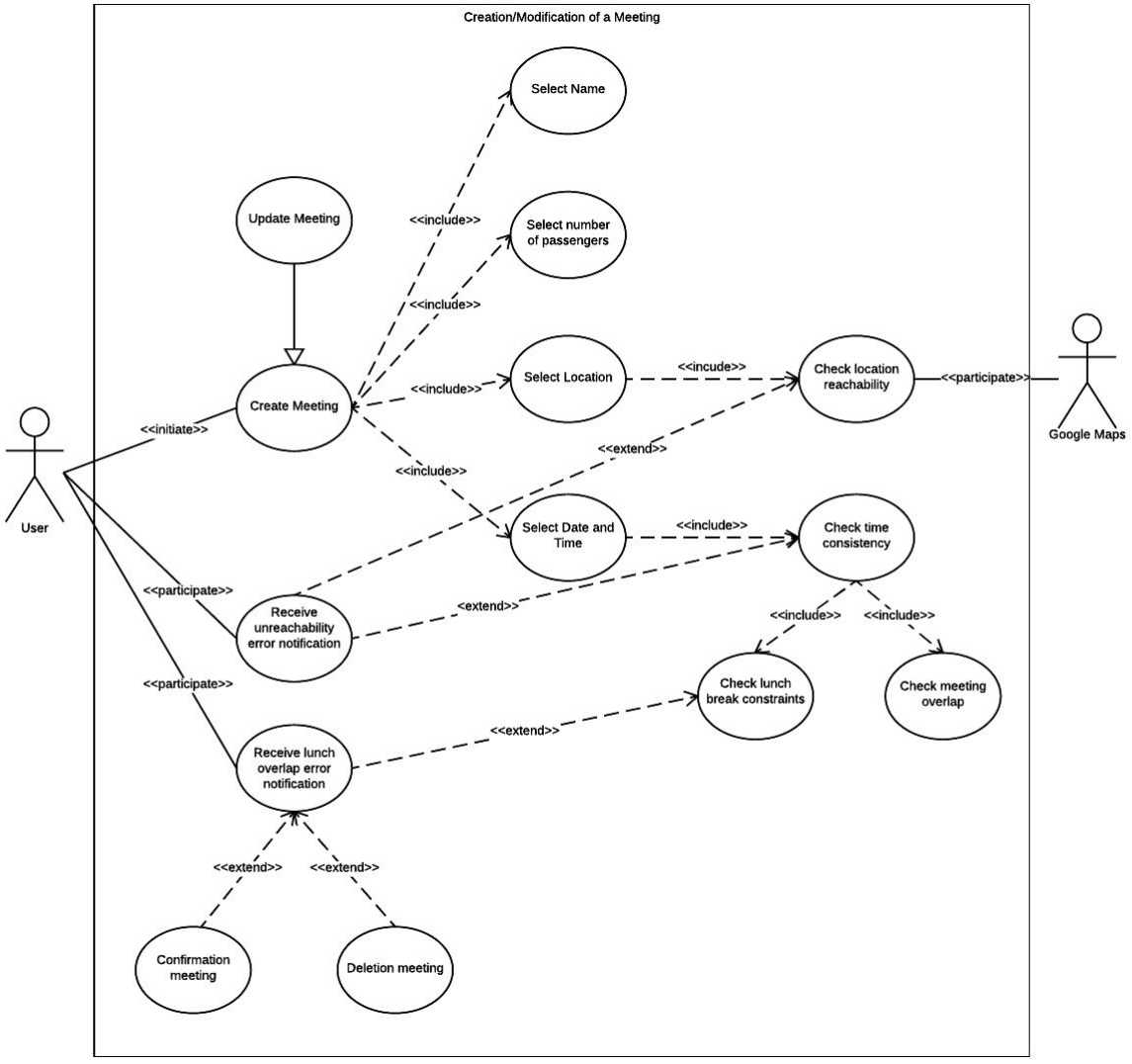
Special requirements:

- User logged in
- Application downloaded correctly on user device

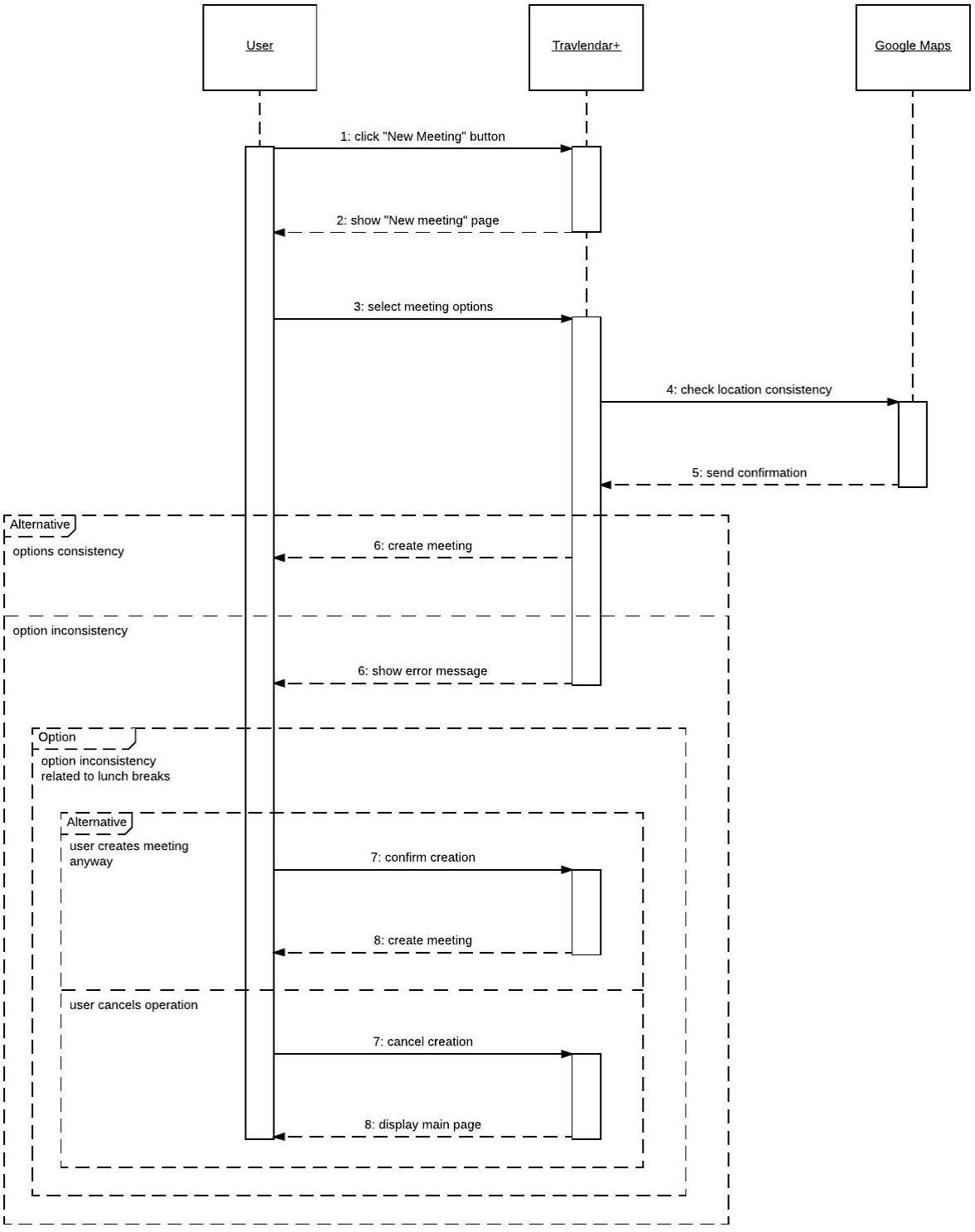
SATISFIED REQUIREMENTS

- Users will be able to create, update and delete meetings.
- During meeting creation users will provide time and location information.
- The system will warn the user with an error message whenever creating a meeting there will be the impossibility to reach the location in time.
- The system will warn the user with an error message whenever a meeting will overlap lunch break time.

USE CASE DIAGRAM



SEQUENCE DIAGRAM



Use case name: **3.B.2.4 - EXAMINE BEST MOBILITY OPTIONS**
Participating actors: User, Map service, Weather forecast service, Car sharing service, Bike sharing service
Entry condition: A user wants to examine the best mobility options of a personal meeting.
Flow of events: ● The application checks: ○ Weather conditions retrieving data from Weather forecast service ○ Public transportation strikes or delays from Map service ○ Bike of a bike sharing system nearby from Bike sharing service ○ Car of a car sharing system nearby from Car sharing service ○ User default preferences or daily preferences if enabled ● The application shows the user the best mobility options
Exit condition: The use case terminates when the user can see displayed all mobility options for his/her meeting.
Exceptions: If there are no mobility options available the system shows an error message.
Special requirements: ● User logged in ● Application downloaded correctly on user device

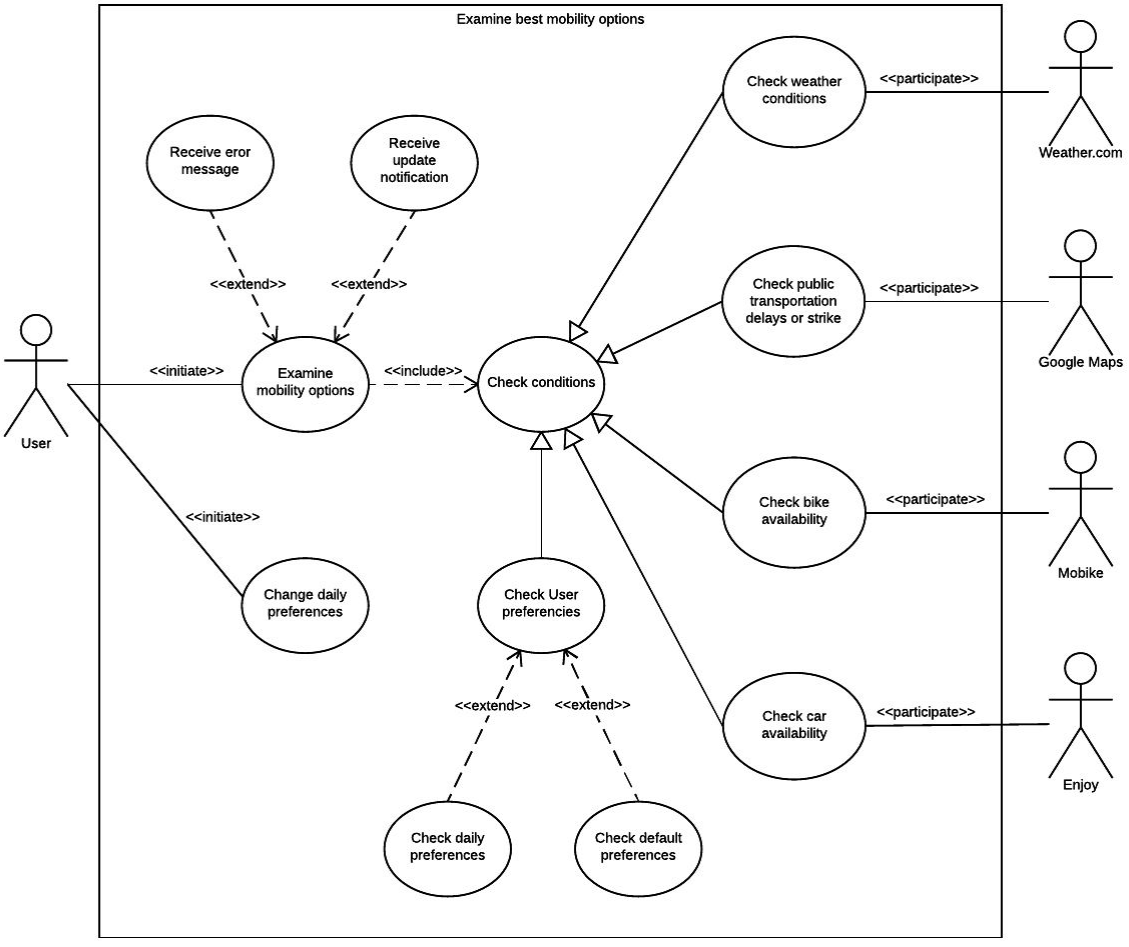
SATISFIED REQUIREMENTS

- The system will take in account preferences to create best mobility options and locating nearest bikes or cars from vehicle-sharing systems.

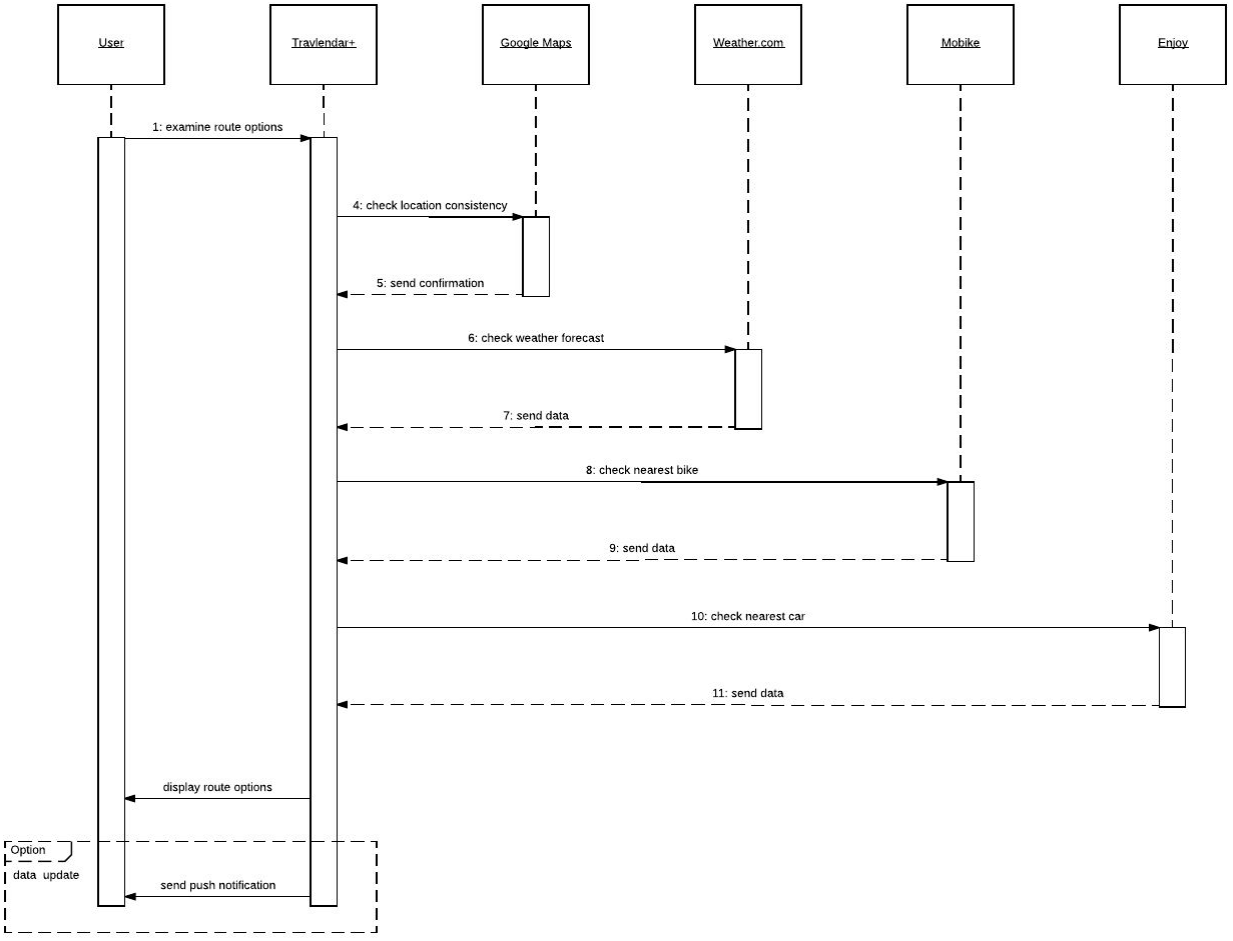
- The system will show next meetings best and secondary mobility options for each travel.

- The system will provide alternative mobility options in case of strikes, transports delays and bad weather. The user will be warned with a push notification too.

USE CASE DIAGRAM



SEQUENCE DIAGRAM



Use case name: **3.B.2.5 - SETTING PREFERENCES**
Participating actors: User
Entry condition: A user wants to change any of the preferences.
Flow of events: ● The application display settings page ● User sets his/her preferences fields ● The application applies the changes
Exit condition: The use case terminates when the user exits from the settings page.
Exceptions: if changing account information: ● password is shorter than 8 character ● password does not contains both upper and lower case letters Then the application shows a pop-up directly on account information page.
Special requirements: ● User logged in ● Application downloaded correctly on user device

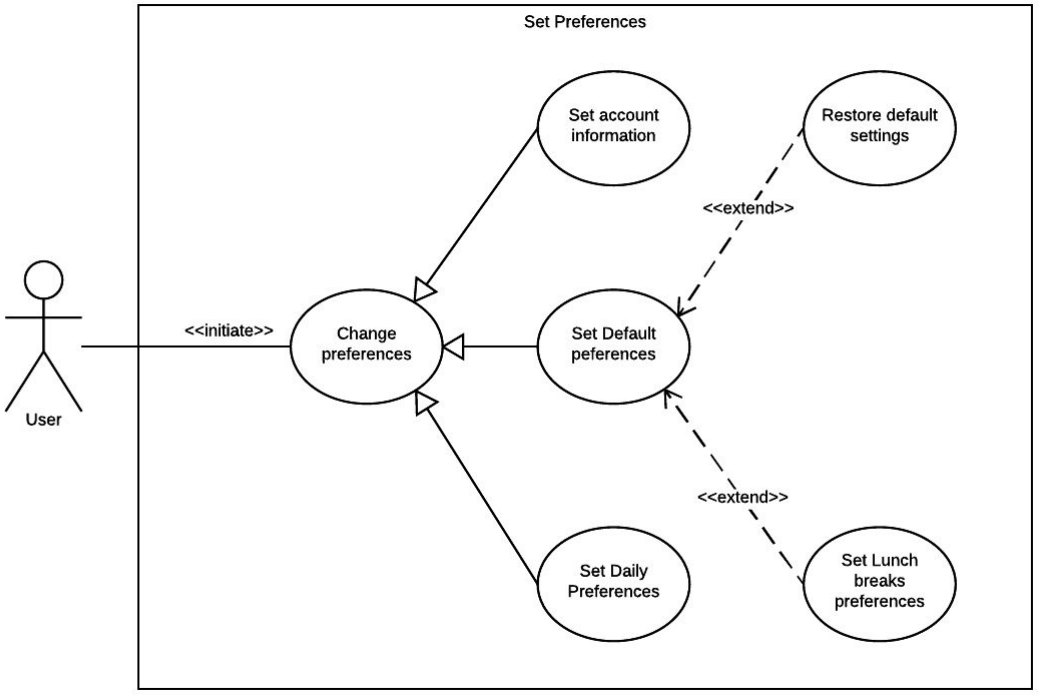
SATISFIED REQUIREMENTS

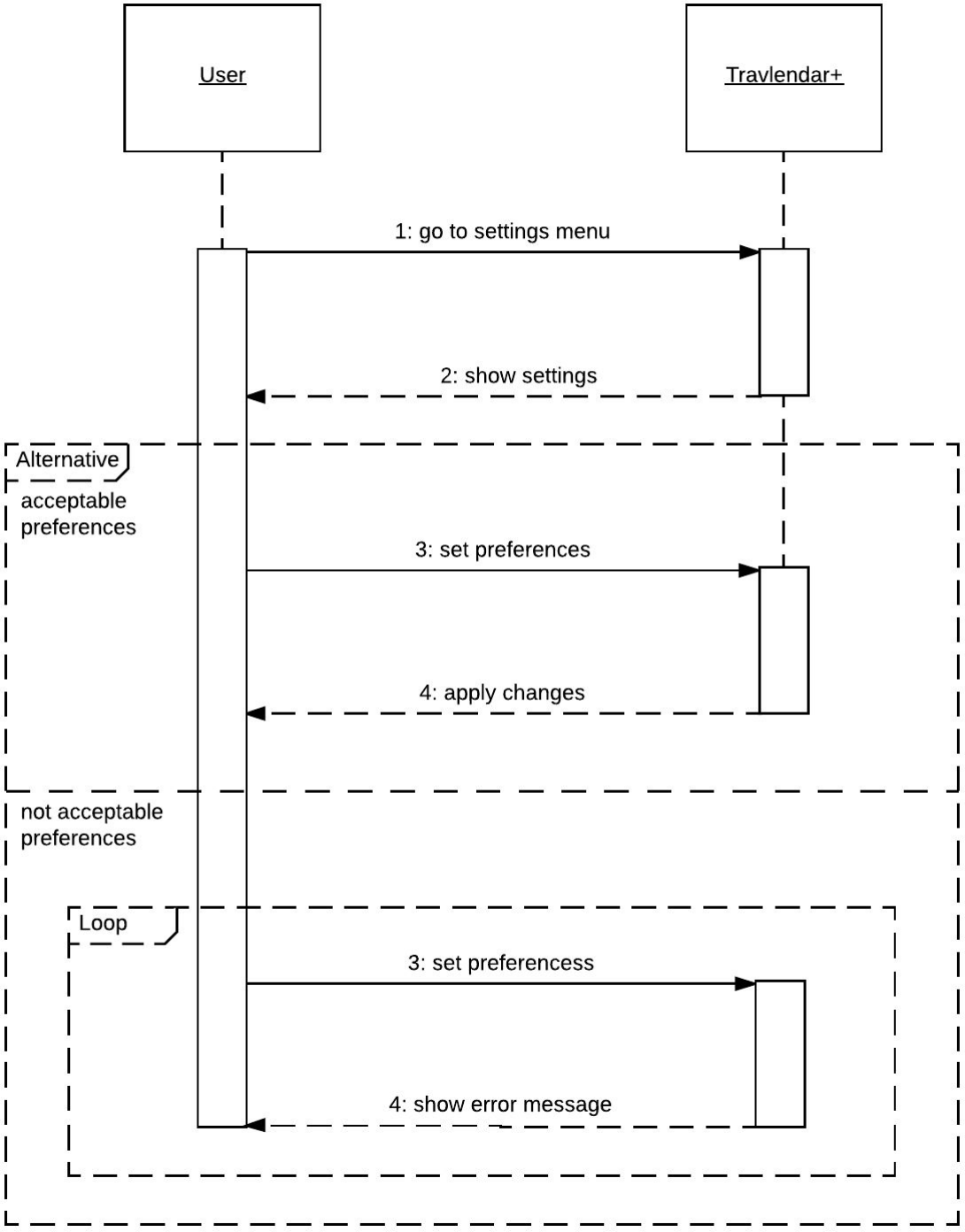
- In “Settings” page users will be able to change default preferences or to restore default settings.

- In “Settings” page users will find “Lunch break” options to set the interval of time for a lunch break.

- In the main page users will find “Change daily settings” button to set daily preferences

USE CASE DIAGRAM



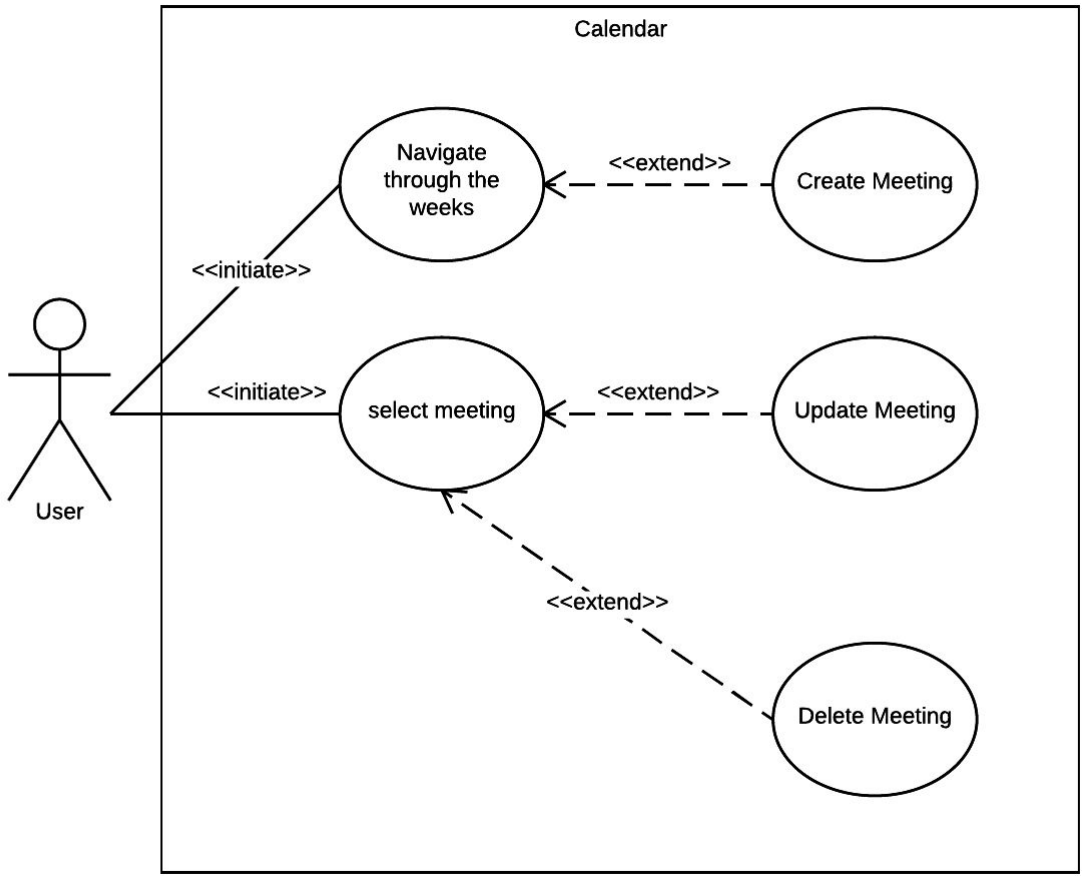

User
Travlendar+
1: go to settings menu
2: show settings
Alternative
acceptable
preferences
3: set preferences
4: apply changes
not acceptable
preferences
Loop
3: set preferencess
4: show error message

Use case name: **3.B.2.6 - NAVIGATING THROUGH CALENDAR**
Participating actors: User
Entry condition: A user wants to have a look to all his meetings during the week moving to calendar page.
Flow of events: ● The application display calendar page ● The user moves through weeks using arrows icons in top corners ● The application shows different weeks ● The user clicks on a meeting ● The application shows the meeting ● The user has the possibility to create update or delete meetings using specific buttons
Exit condition: The use case terminates when the user gets redirected to another page.
Exceptions: On calendar page data is only displayed, exceptions can only occur due to data inconsistency or system bugs.
Special requirements: ● User logged in ● Application downloaded correctly on user device

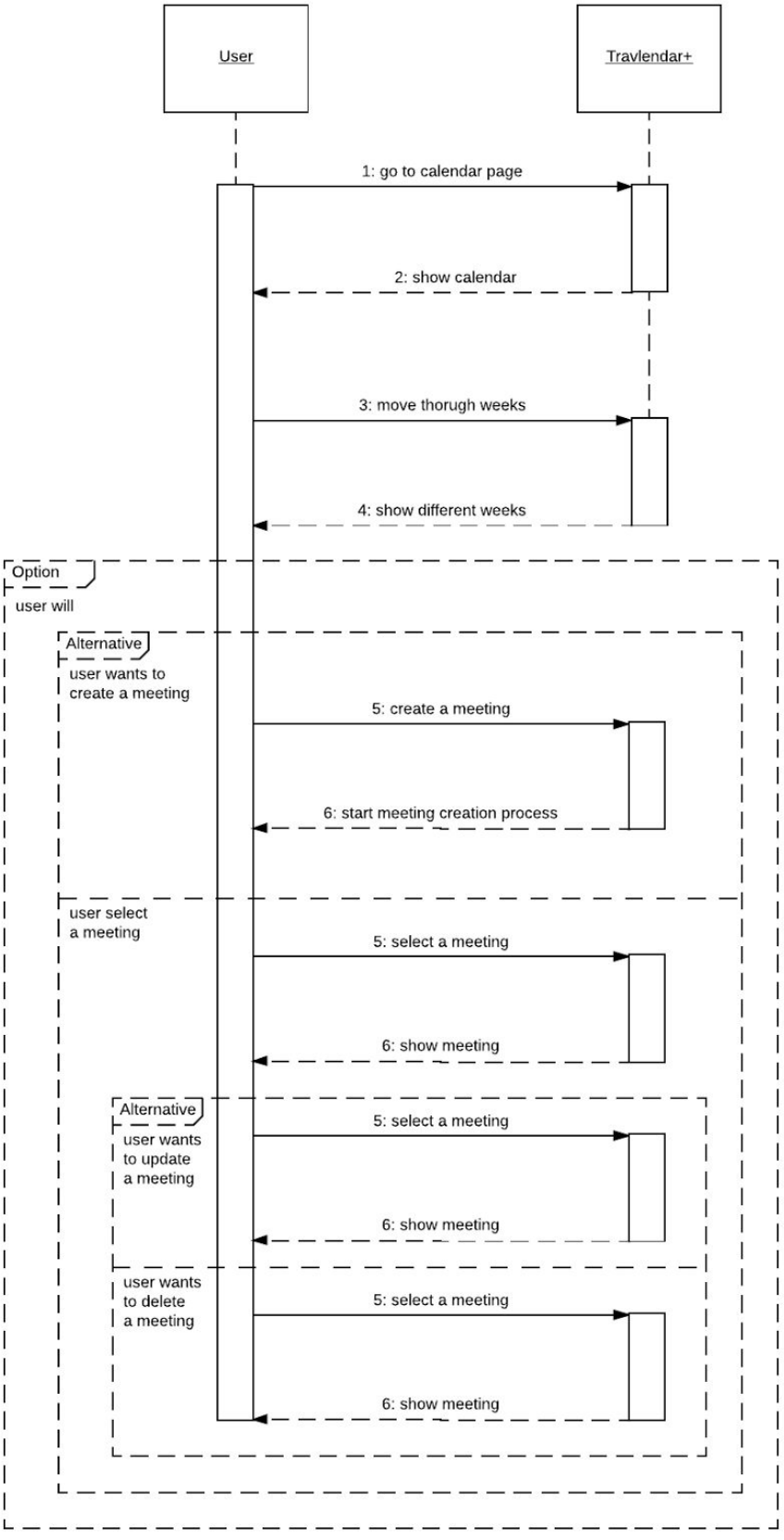
SATISFIED REQUIREMENTS

- In “Calendar” page user will be able to visualize all his/her meetings chronologically ordered and divided by week.

USE CASE DIAGRAM



SEQUENCE DIAGRAM



Use case name: **3.B.2.6 - STARTING A JOURNEY**
Participating actors: User
Entry condition: A user receives an Hurry Up notification.
Flow of events: ● The user opens the “Route options” page ● The system displays best mobility options ● The user buy the tickets for his/her travelling mean ● The system gives a confirmation
Exit condition: The user start his/her journey.
Exceptions: if the user preferences cannot be respected alternative mobility options are calculated If the user is late he/she receive a Late notification and late mobility options are calculated.
Special requirements: ● User logged in ● Application downloaded correctly on user device ● User has planned a meeting ● User has set DAT

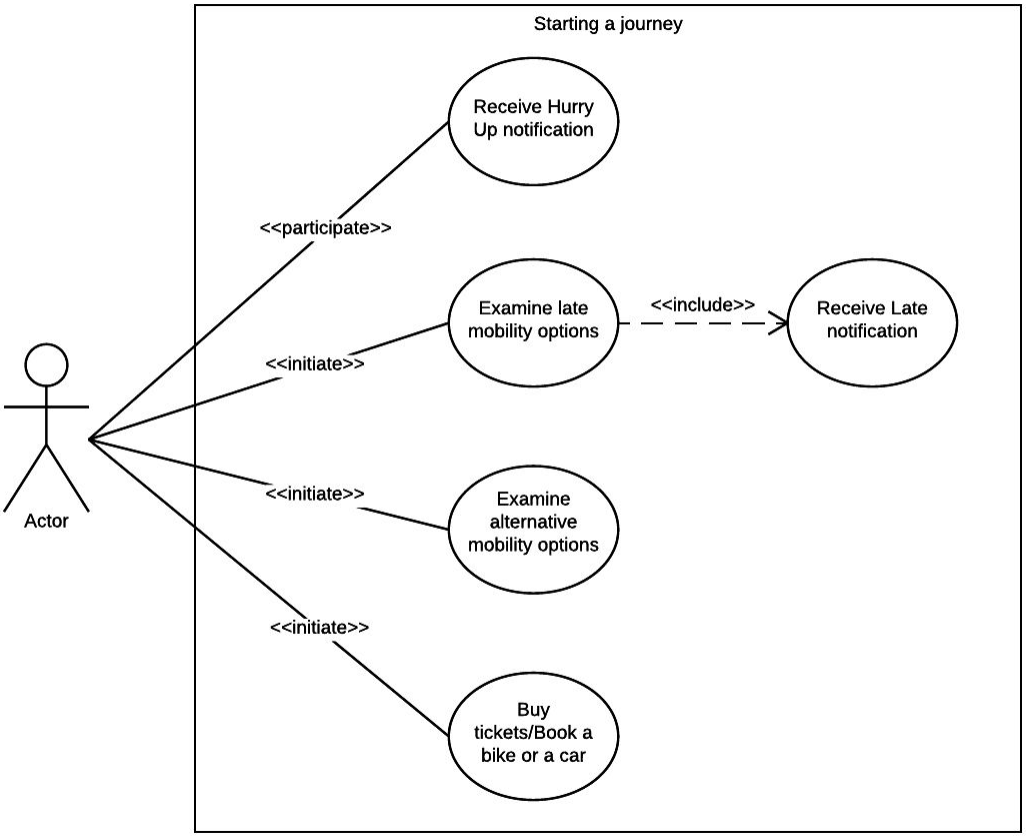
SATISFIED REQUIREMENTS

- Clicking on a meeting the system will display secondary mobility options and a section for buying tickets or booking bikes and cars.

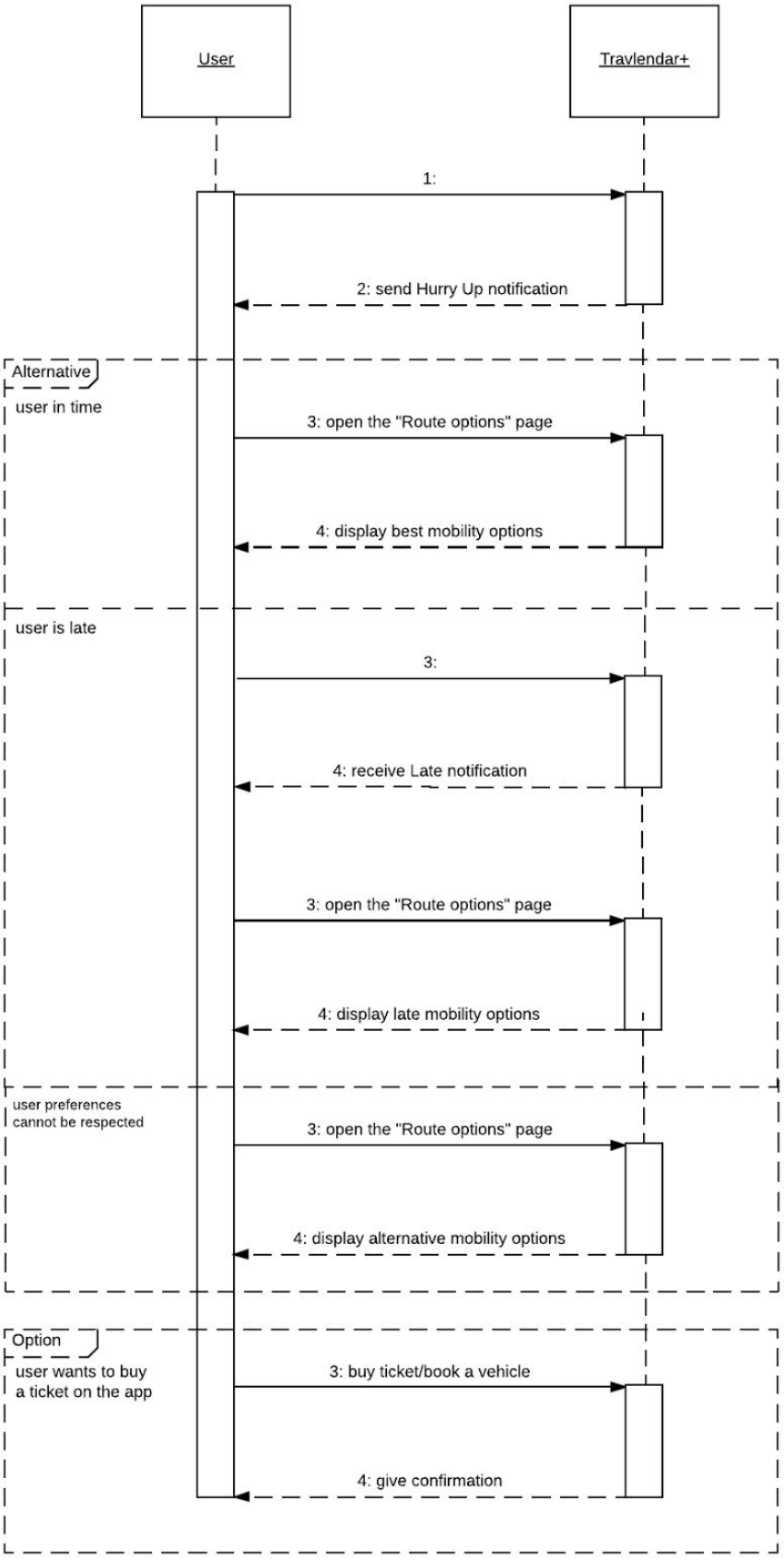
- The system will send a “Hurry Up” notification for departure according to DAT.

- The system will send a “Late” notification when it’s not possible to reach the location in time anymore.

USE CASE DIAGRAM



SEQUENCE DIAGRAM



3.C - PERFORMANCE REQUIREMENTS

The application presents some performance lower-bounds to let the user have a fluent experience navigating in the system.
The response time for basic requests (i.e. navigating through the pages) cannot take an amount of time greater than 50 ms while calculating mobility options requests cannot take more than 0.5 s.
Performance depends on internet connection speed, it is assumed that the average user will have access to high level 3g-4g services or wifi connections during the day.
Moreover, it is also assumed that every device that respects o.s. version limitations will have hardware specifications that permits such performances.

3.D - DESIGN CONSTRAINTS

3.D.1 - STANDARDS COMPLIANCE

For a correct functionality all the following conformities are applied:

- all java EE standards are inherited
- wifi, 3g/4g connection communication work using IEEE standards
- operative system interactions follow Android and IOS functions and user interfaces
- external services communicate with their own APIs
- more in general any web communication complies with w3c standards

3.D.2 - HARDWARE LIMITATIONS

Hardware limitation are linked to device GPS signal and internet connection. In fact they are very unlikely to show up. As mentioned in the paragraph 3.C devices are assumed to have hardware specification that permits the application to have good performances.
Ram memory allocation problems can be omitted due to lack of need of creating big temporary files.

3.E - SOFTWARE SYSTEM ATTRIBUTES

3.E.1 - RELIABILITY

Travlendar+ should be enough reliable to allow any user to have a full day session without interruptions. This can be achieved preventing the application to have downtimes during the day and with redundant subsystems.

3.E.2 - AVAILABILITY

Travlendar+ must reach a “2-nines” level of availability (99%). The system will be down only for quick upgrades and serious failures. This should also happen during the night, as many user tend to use the application during the day.

3.E.3 - SECURITY

Account information and data stored are encrypted and saved in a safe environment.
User must log in the system using a password and personal details can only be seen once logged in.

3.E.4 - MAINTAINABILITY

Travlendar+ maintenance should be as little as possible. This can be obtained with good quality code and an exhaustive documentation.

3.E.5 - PORTABILITY

Travlendar+ is designed as a mobile application to support its transportability intent, a tablet version will be developed as well.
A web version of Travlendar+ will not be deployed as the user is intended to move around the city using the application with his/her portable device.

4. FORMAL ANALYSIS USING ALLOY

4.1 - ALLOY CODE

```
// ------------------- SIGNATURES ------------------- //

abstract sig Bool{}

one sig True extends Bool {}
one sig False extends Bool {}

sig Strings{}

sig User{
   username: one Strings,
   password: one Strings,
   calendar: one Calendar,
   preferences: one Preferences
   }

sig Date{
   day: one Int,
   month: one Int,
   year: one Int
   }

sig Hour{
   minutes: one Int,
   hour: one Int
   }

sig Location{
   address: one Strings,
   zipCode: one Int
   }


sig Travel{
   travelmean: one TravelMeans,
   duration: one Int,
 }

sig RouteOption{
   departure: one Meeting,
   arrival: one Meeting,
   travelmeans: set Travel
   }
```

```
abstract sig TravelMeans{}

sig Taxi extends TravelMeans{
   location: one Location
   }

sig BikeSharing extends TravelMeans{
   location: one Location
   }

sig CarSharing extends TravelMeans{
   location: one Location
   }

sig PublicTransportation extends TravelMeans{
   location: one Location,
   hour: one Hour,
   linenumber: Int
   }

sig WeatherForecast{
   location: one Location,
   humidity: Int,
   temperature: Int,
   windspeed: Int
   }

sig Calendar{
   meetings: set Meeting
   }

sig Preferences{
   maximumwalkdistance: Int,
   avoidtollroads: one Bool,
   bestrouteoption: one Strings,
   lunchBreak: lone LunchBreak
   }

sig LunchBreak{
   startingTime: one Hour,
   endTime: one Hour,
   duration: Int
   }
```

```
// ---------- FACTS ----------//

//All meetings belong to a calendar
fact MeetingCalendar{
   all m: Meeting | some c: Calendar | m in c.meetings
    }

//Different users have different usernames
fact OneUserPerUsername{
   no disj u1, u2: User| u1.username = u2.username
   }

//Password and Username must be different
fact PasswordAndUsername{
   all u: User | u.username != u.password
   }

//All preferences belong to a user
fact PreferenceUser{
   all p: Preferences | some u: User | p = u.preferences
   }

//One user per calendar
fact OneUserPerCalendar{
   all c: Calendar | (one u: User | u.calendar = c)
   }

//Correct date
fact CorrectDate{
   all d: Date |
      d.day > 0 &&
      d.month > 0 &&
      d.year > 0
   }

//Correct Time
fact CorrectTime{
   all h: Hour | (
      h.hour >= 0 &&
      h.minutes >= 0 )
   }

//Positive duration of a travel
fact CorrectTravel{
   all t: Travel | t.duration > 0
   }

//A meeting must end after it starts
fact EndAfterStart{
   all m: Meeting | ((m.timeStart.hour < m.timeEnd.hour) ||
      ((m.timeStart.hour = m.timeEnd.hour) &&
         (m.timeStart.minutes < m.timeStart.minutes)))
   }
```

```
//Different meetings in a route option
fact NoSameMeeting{
   all r: RouteOption | r.departure != r.arrival
   }

//In a route option departure must be before than arrival
fact RightOrderMeetings{
   all r: RouteOption |
      ((r.departure.timeEnd.hour < r.arrival.timeStart.hour) ||
      ((r.departure.timeEnd.hour = r.arrival.timeStart.hour) &&
         (r.departure.timeEnd.minutes < r.arrival.timeStart.minutes)))
   }

//No meeting that start in the same moment in a calendar
fact NoSameStart{
   all c: Calendar | all disj m1, m2: Meeting | (m1 in c.meetings && m2 in c.meetings) implies
         ((m1.date != m2.date) || ((m1.date = m2.date) && (m1.timeStart != m2.timeStart)))
   }

//No overlap between meetings
fact NoOverlap{
   all c: Calendar | all disj m1, m2: Meeting | (m1 in c.meetings && m2 in c.meetings)  iff
      ((m1.date.year != m2.date.year) ||
      ((m1.date.year = m2.date.year) && (m1.date.month != m2.date.month)) ||
      ((m1.date.year = m2.date.year) && (m1.date.month = m2.date.month) && (m1.date.day != m2.date.day)) ||
      (((m1.date.year = m2.date.year) && (m1.date.month = m2.date.month) && (m1.date.day = m2.date.day)) &&
         ((((m1.timeStart.hour < m2.timeStart.hour) ||
            ((m1.timeStart.hour = m2.timeStart.hour) && (m1.timeStart.minutes < m2.timeStart.minutes))) &&
         ((m1.timeEnd.hour < m2.timeStart.hour) ||
            ((m1.timeEnd.hour = m2.timeStart.hour) && (m1.timeEnd.minutes < m2.timeStart.minutes)))) ||
         (((m2.timeStart.hour < m1.timeStart.hour) ||
            ((m2.timeStart.hour = m1.timeStart.hour) && (m2.timeStart.minutes < m1.timeStart.minutes))) &&
         ((m2.timeEnd.hour < m1.timeStart.hour) ||
            ((m2.timeEnd.hour = m1.timeStart.hour) && (m2.timeEnd.minutes < m1.timeStart.minutes)))))))
   }


//StartingTime before EndTime in LunchBreak
fact StartBeforEndLunch{
   all l: LunchBreak | ((l.startingTime.hour < l.endTime.hour) ||
      ((l.startingTime.hour = l.endTime.hour) && (l.startingTime.minutes < l.startingTime.minutes)))
   }
```

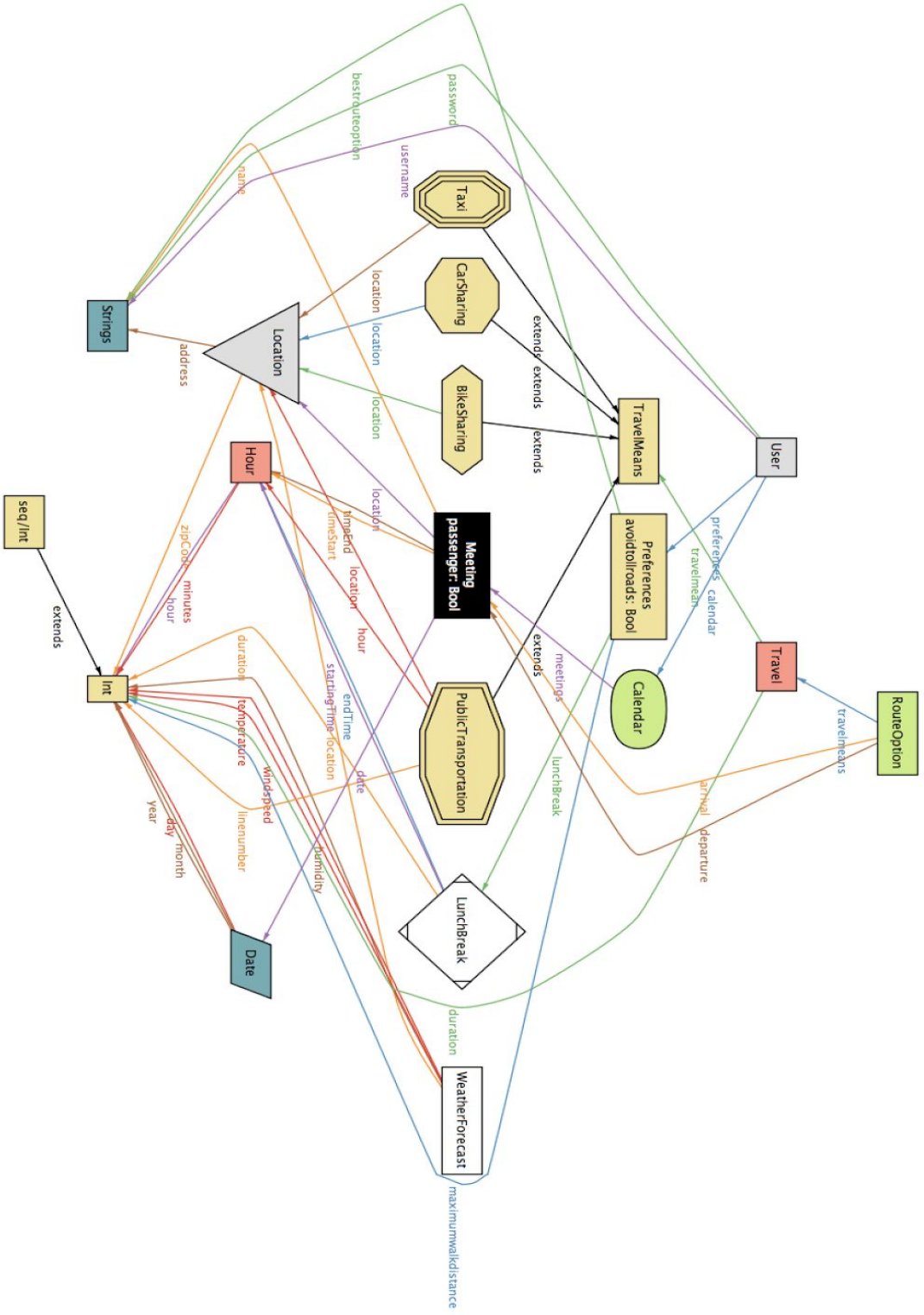
```
// ------------ PREDICATES ------------//

//Create a new Meeting
pred CreateMeeting [m: Meeting, u: User]{
   u.calendar.meetings = u.calendar.meetings + m
   }

//Delete a Meeting
pred DeleteMeeting [m: Meeting, u: User]{
   u.calendar.meetings = u.calendar.meetings - m
   }

pred NoDoubleMeeting[u: User]{
   all m1: Meeting | no m2: Meeting | m1 in u.calendar.meetings && m2 in u.calendar.meetings && m1 = m2
   }

pred NoSameUsername[u: User]{
   some u1: User | u.username != u1.username
   }

pred PositiveRouteOption[r: RouteOption]{
   (sum t: Travel | t.duration) >0
   }

pred show{}

pred showuser{
   #User > 1 &&
   #Meeting > 2
   }


// ---------------- ASSERTION ----------------//

assert MeetingWithoutOverlaps{
   all u: User | no m1, m2: Meeting | m1 in u.calendar.meetings && m2 in u.calendar.meetings &&
      (m1.date = m2.date) && (m1.timeStart.hour < m2.timeStart.hour) && (m1.timeEnd.hour > m2.timeEnd.hour)
   }

assert NoDoubleMeeting{
   all u: User | no disj m1, m2: Meeting | m1 = m2 && m1 in u.calendar.meetings && m2 in u.calendar.meetings
   }

// ------------- CHECK ---------------- //

check MeetingWithoutOverlaps

check NoDoubleMeeting

run showuser for 3

run show for 4  but exactly 1 User

run CreateMeeting for 4

run PositiveRouteOption for 3 but 6 Int

run NoDoubleMeeting for 3 but 6 Int

run NoSameUsername for 3 but 6 Int
```

4.2 - RESULTS

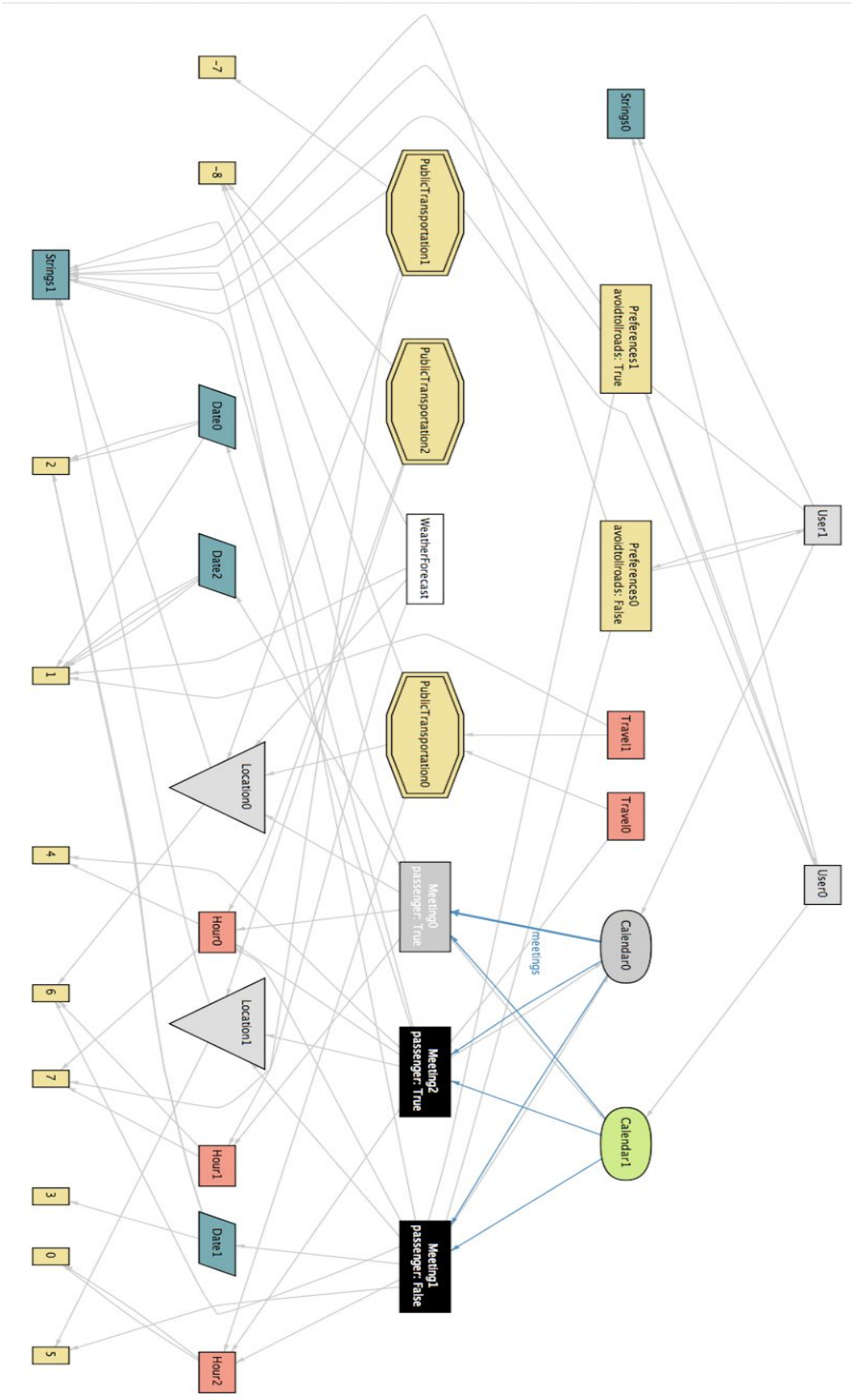
```
Executing "Check MeetingWithoutOverlaps"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   18400 vars. 936 primary vars. 47121 clauses. 1193ms.
   No counterexample found. Assertion may be valid. 943ms.

Executing "Check NoDoubleMeeting"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   17505 vars. 936 primary vars. 44297 clauses. 191ms.
   No counterexample found. Assertion may be valid. 3ms.

Executing "Run showuser for 3"
   Solver=sat4j Bitwidth=4 MaxSeq=3 SkolemDepth=1 Symmetry=20
   17438 vars. 927 primary vars. 44216 clauses. 148ms.
   Instance found. Predicate is consistent. 161ms.

Executing "Run show for 4 but exactly 1 User"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   25367 vars. 1280 primary vars. 66111 clauses. 614ms.
   Instance found. Predicate is consistent. 131ms.

Executing "Run CreateMeeting for 4"
   Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
   26422 vars. 1352 primary vars. 68043 clauses. 160ms.
   Instance found. Predicate is consistent. 241ms.

Executing "Run NoDoubleMeeting for 3 but 6 int"
   Solver=sat4j Bitwidth=6 MaxSeq=3 SkolemDepth=1 Symmetry=20
   72717 vars. 2802 primary vars. 217177 clauses. 311ms.
   Instance found. Predicate is consistent. 105ms.

Executing "Run NoSameUsername for 3 but 6 int"
   Solver=sat4j Bitwidth=6 MaxSeq=3 SkolemDepth=1 Symmetry=20
   72741 vars. 2805 primary vars. 217261 clauses. 362ms.
   Instance found. Predicate is consistent. 184ms.

7 commands were executed. The results are:
   #1: No counterexample found. MeetingWithoutOverlaps may be valid.
   #2: No counterexample found. NoDoubleMeeting may be valid.
   #3: Instance found. showuser is consistent.
   #4: Instance found. show is consistent.
   #5: Instance found. CreateMeeting is consistent.
   #6: Instance found. NoDoubleMeeting is consistent.
   #7: Instance found. NoSameUsername is consistent.
```

4.3 - ALLOY WORLD

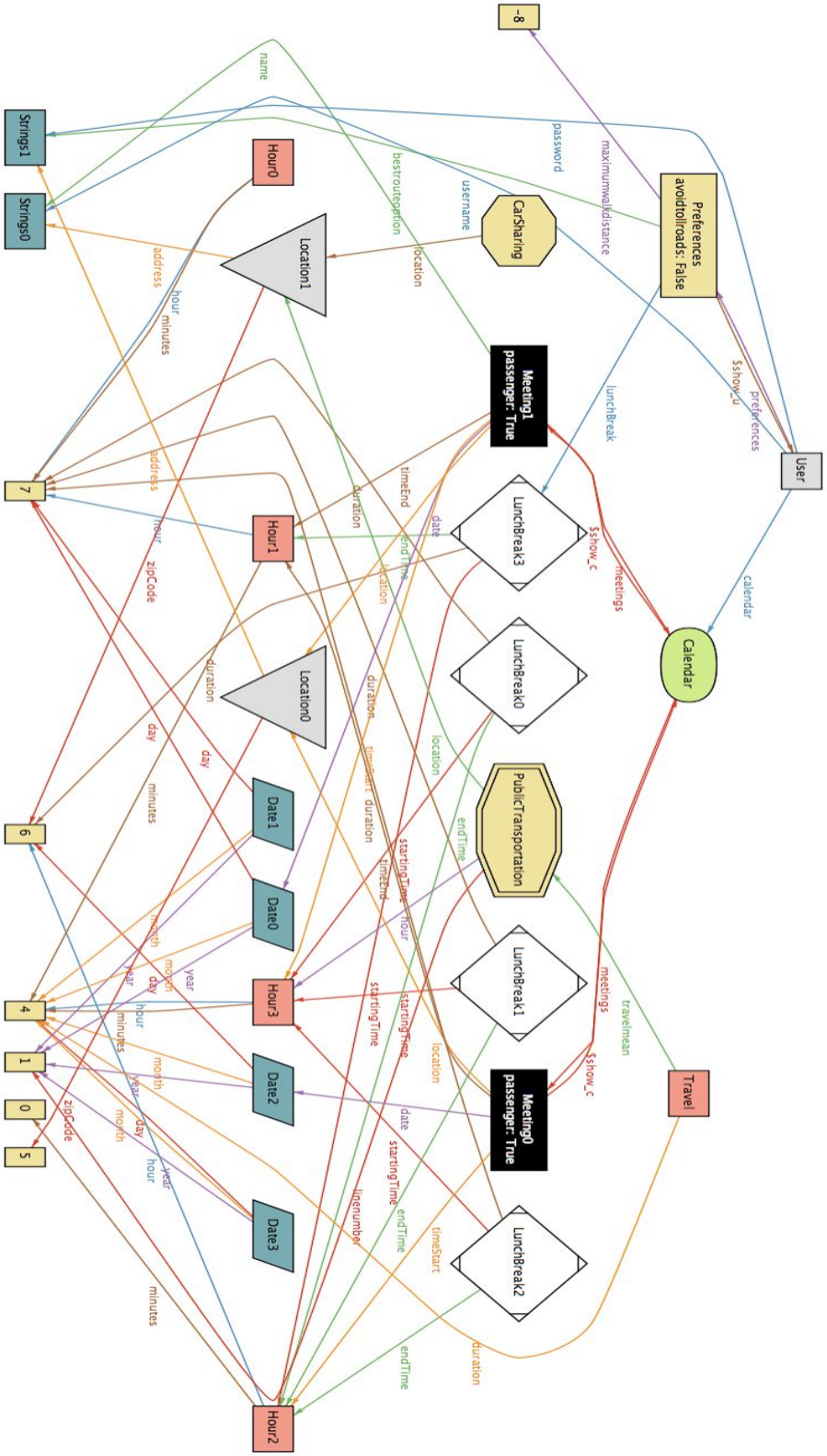
4.3.1 - METAMODEL



4.3.2 - SHOW USER

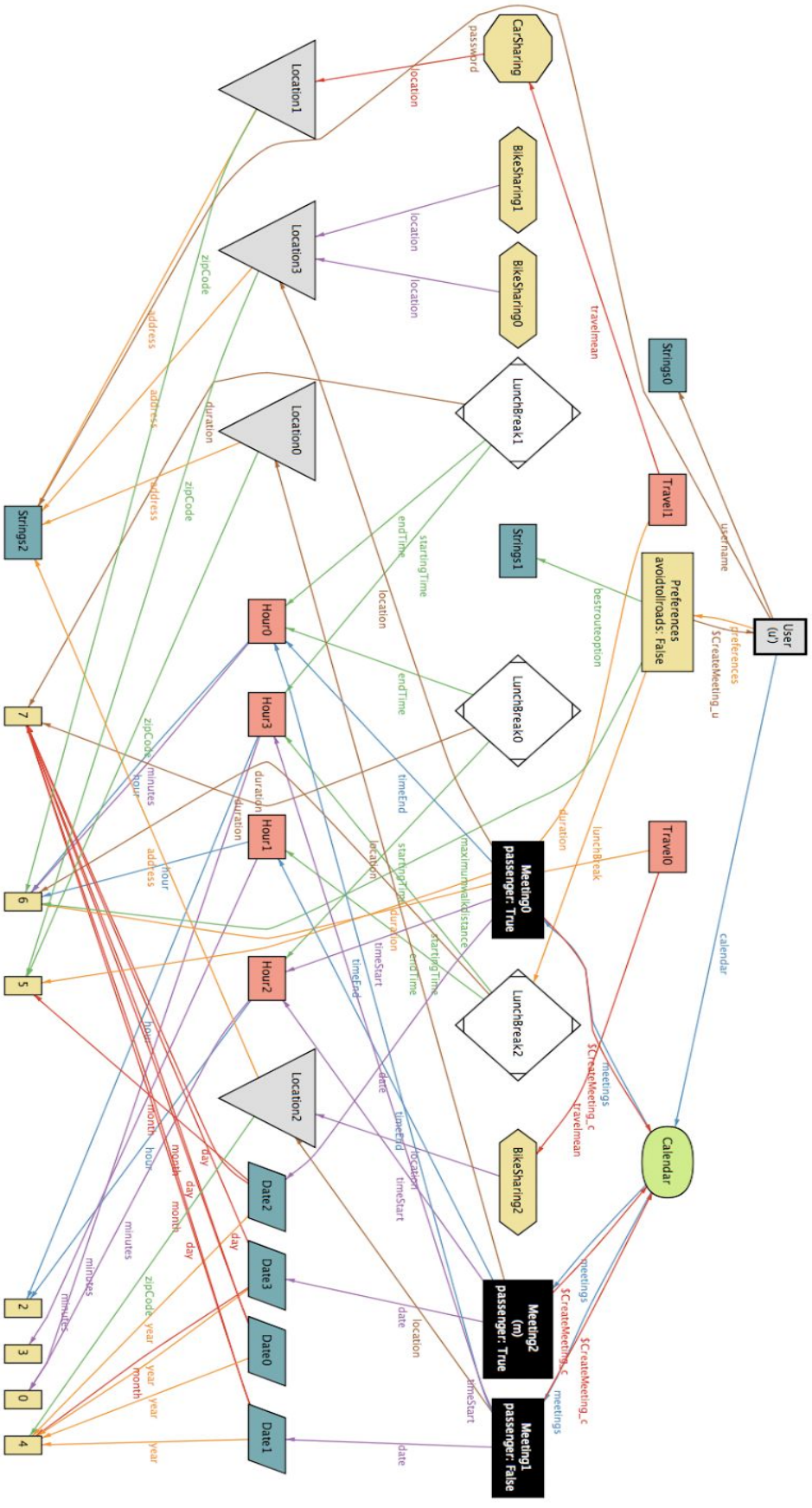


4.3.3 - SHOW

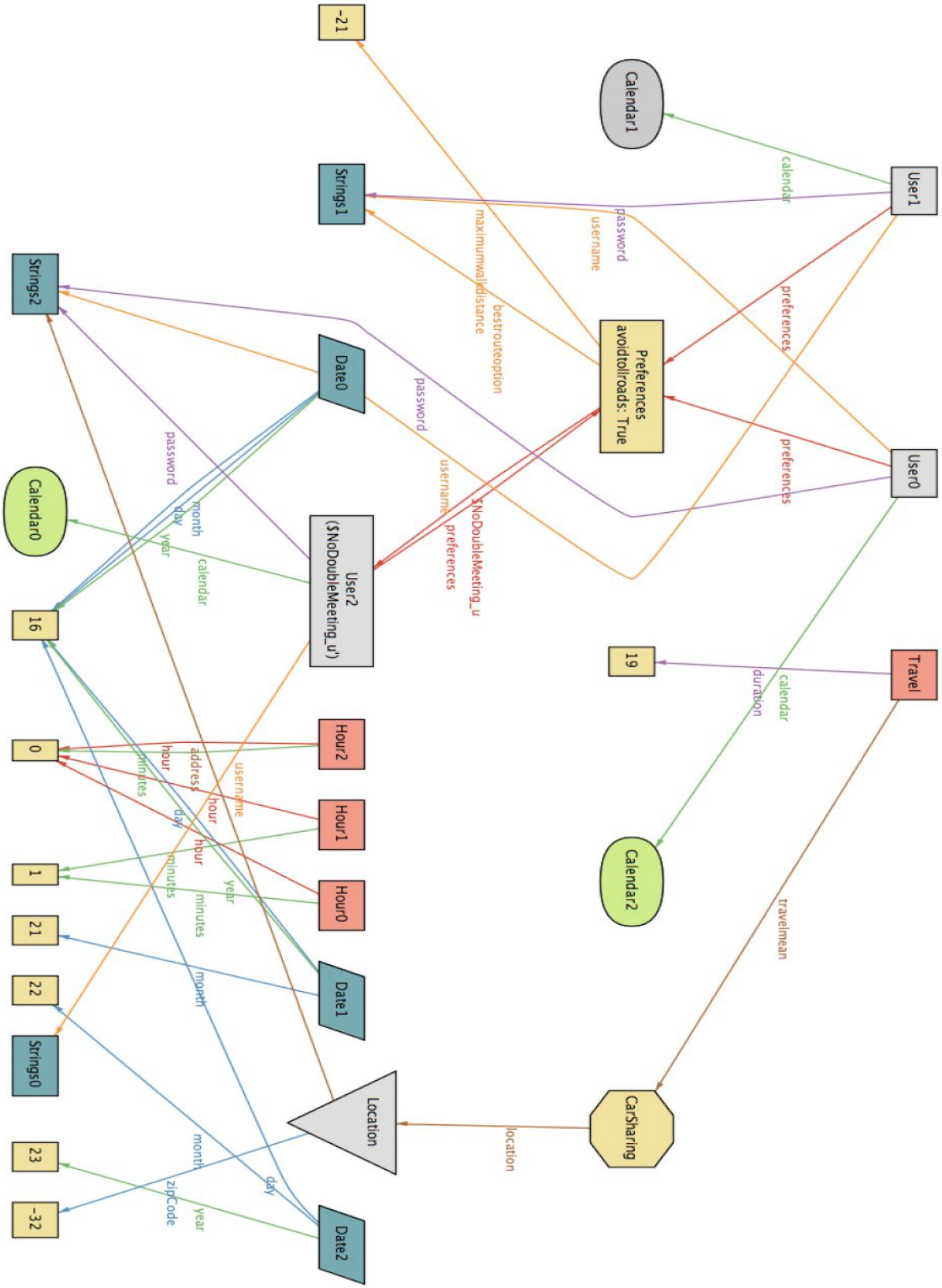


-8
Strings1
Strings0
Hour0
Location1
CarSharing
Preferences avoidtollroads: False
User
Meeting1 passenger: True
LunchBreak3
LunchBreak0
Calendar
Hour1
Location0
PublicTransportation
Date1
Date0
LunchBreak1
Hour3
Date2
Meeting0 passenger: True
Travel
Date3
LunchBreak2
Hour2
7
6
4
1
0
5
name
password
maximumwalkdistance
bestrouteoption
username
location
address
hour
minutes
lunchBreak
$show_u
preferences
timeEnd
duration
date
endTime
$show_c
meetings
calendar
zipCode
day
month
year
startingTime
timeStart
linenumber
travelmean

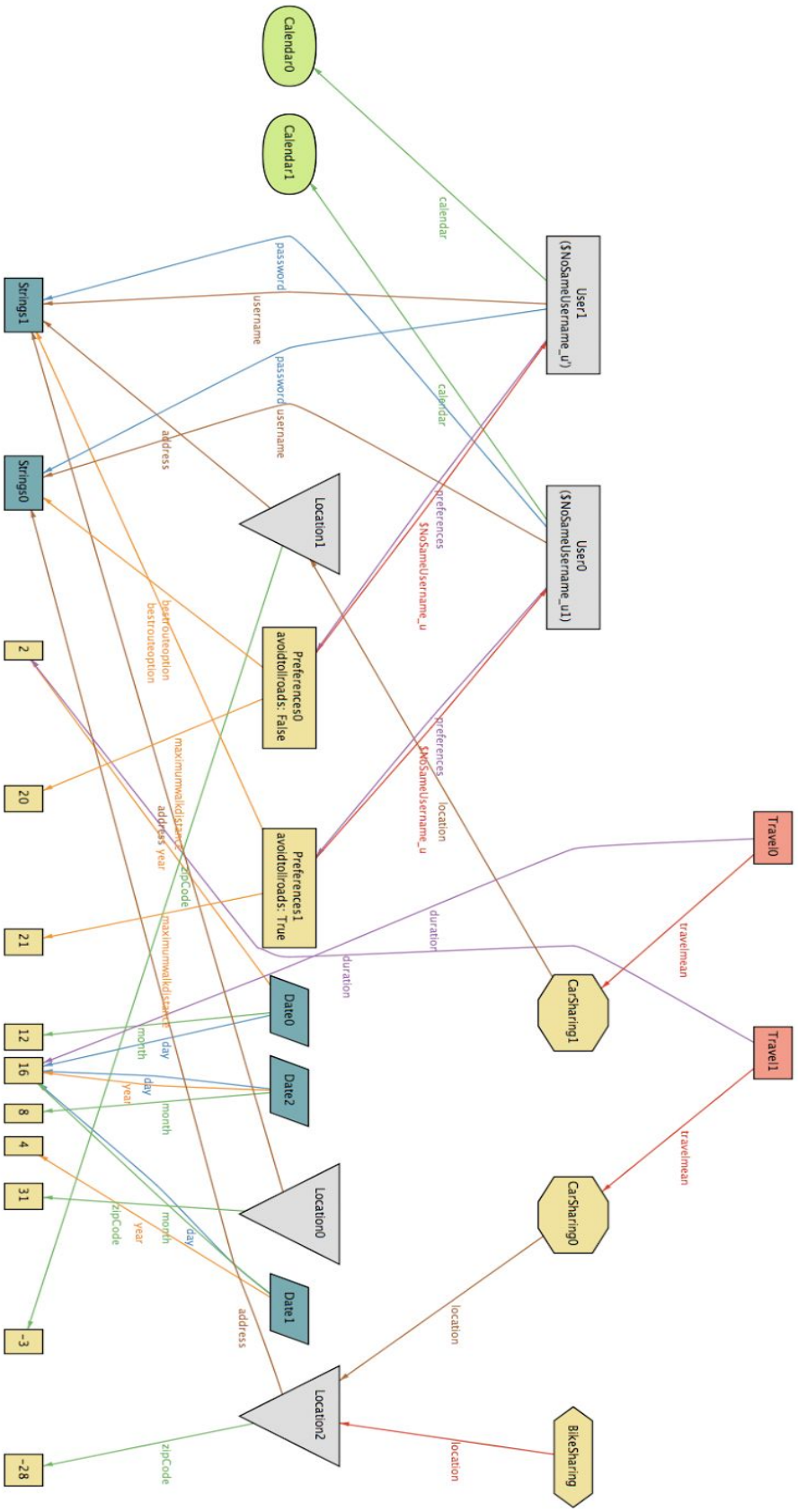
4.3.4 - CREATE MEETING



4.3.5 - NO DOUBLE MEETING



4.3.6 - NO SAME USERNAME



5. EFFORT SPENT

Day	Probo Giovanni	Tommasi Giovanni
6 ottobre	2 h	2 h
8 ottobre	1 h	0.5
9 ottobre	3 h	4 h
11 ottobre	3 h	3 h
14 ottobre	1.5 h	-
15 ottobre	-	5 h
16 ottobre	2 h	-
18 ottobre	0.5 h	2.5 h
20 ottobre	2.5 h	-
21 ottobre	-	3 h
22 ottobre	5 h	5 h
23 ottobre	3.5 h	2.5 h
25 ottobre	5 h	5 h
26 ottobre	3 h	-
27 ottobre	1.5 h	1.5
29 ottobre	1 h	1 h

6. REFERENCES

- Specification document : Mandatory Project Assignments.pdf

- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

- IEEE Std 1016 tm -2009 Standard for Information Technology-System Design-Software Design Descriptions.