Politecnico di Milano
A.A. 2017/2018

TRAVLENDAR+

RASD

Requirements Analysis and Specification Document

Sara Pidó 894744
Chiara Plizzari 893901
Giuseppe Severino 898458

Contents

1 Introduction

1.1 Purpose

1.1.1 Description of the RASD

RASD stands for Requirements Analysis and Specification Document. The main goal of this document is to describe the system by clearly specifying functional and non-functional requirements in a structured though informal form in order to provide a guideline. RASD takes into account the limits and the constraints of the problem and its possible solutions. It provides feedback to the customer, it serves as an input to the design specification, as a product validation check and as a contractual basis between the customer and the developer. Therefore, RASD should be a model that is directed towards ensuring that the final system conforms to client needs. The document is addressed to all customers and users, system and requirement analysts, developers and programmers who participate in the implementation of the requirements, testers and project managers.

1.1.2 Purpose of the application

The application should be useful for busy people that have to travel from a place to another one because of their engagements. It helps people by organizing their own calendar: it finds the best solution to reach a certain place in a specific time basing on user's preferences. Users are able to see their meetings, their journey between them, their breaks and in every time they can modify their day calendar by modifying an activity or deleting it.

1.2 Scope

1.2.1 Description of the given problem

We want to create a calendar-based application which name is Travlendar+. It is a software that provides a support to everyone that has scheduling meetings at various locations. It helps the user in finding the best option to reach the destination in the optimal conditions and at a fixed time. Once the user has registered and has inserted time and place of his/her meetings, the system automatically computes and accounts for travel time between appointments, to make sure that the client will not be late. Moreover, the

user will be warned if the location is not reachable at the allotted time. Other services are provided, such as the possibility to identify the best mobility option basing on user's preferences (preference or avoidance for a determined mean) and on external conditions (weather, strikes), the opportunity to buy public transportation tickets and to locate the nearest bike of a bike sharing system or the nearest car of a car sharing system. It is also possible to select combinations of transportations means that minimize carbon footprint and to specify the maximum walking distance. Thanks to this application the users can also organize in a customizable way their break time between an appointment and one other, for example by managing a flexible lunch.

1.2.2 Actual system

Even if there already exist applications that allow users to find the best travel solution, this is a new kind of application for the innovative idea of managing the time. Therefore, we assume that the whole system will be created by new. However, the application will exploit applications or websites to allow the user to buy transportation's tickets and to use sharing services. Moreover, it will profit by these websites to have real-time news on weather, strikes...

1.2.3 Goals

[G_1] A user should be able to go to a meeting.

[G_2] A user should be able to know if he/she cannot arrive on time to a meeting.

[G_3] A user should be able to have a break that has a minimum duration between two meetings.

[G_4] The time or the location of a meeting can be modified.

[G_5] A user should be able to unsay a meeting.

[G_6] A user should be able to have some preferences for the trip.

- [$G_{6.1}$] A user should be able not to use some travel means.
- [$G_{6.2}$] A user should be able not to walk more than a specific distance.
- [$G_{6.3}$] A user should be able to minimize carbon footprint when he/she goes to a meeting.

[G_7] A user should be able to know every possible option to go to a meeting with different travel means in order to choose the suitable one.

[G_8] A user should be able to buy public transportation tickets or day/week/season pass basing on his/her needs.

[G_9] A user would like to see on Tripadvisor advised places for his/her breaks.

[G_{10}] A user would like to locate the nearest vehicle of a vehicle sharing system if he/she wants to use that type of travel means.

[G_{11}] A user would like to have real-time news on strikes, weather and traffic.

1.2.4 Actors

There are two types of actors:

- Visitors: people that download the application and that are not registered in the system but they have free access to the login page, the sign-up page.
- Registered users: they can see all the pages available to visitors and after successful login, they can take advantage of all the services of the application.

1.3 Definitions, acronyms, abbreviations

1.3.1 Definitions

- Visitor: a person that is not registered yet, but has the access to the application's information.
- Registered user: a person that is logged in the system and can create meetings.
- Activity: an event that happens in the real world and that could be a meeting or a break.
- Meeting: an activity among the registered user and other people. It can be created, modified and deleted by the meeting's creator.

- Break: an activity that a registered user can insert in order to manage it in a customizable way.
- Trip: it indicates the route and the travel means chosen, based on user's preferences.
- Location: fixed place where a user stands or where he/she has to attend a meeting.
- Global preferences: they are global attributes that registered users can modify and those are valid for all trips (i.e. minimize carbon footprint).
- Creation screen: the screen of the application in which the registered user create a meeting or a break and enters its related details.
- Blocked travel means: it is a travel means that the user has selected as unwanted.
- Warning: a message directed to the user that arrives to him in form of a notification and can be shown on the application screen. It is generated by the system when there are some impediments for a trip (bad weather, strikes, traffic...) or when there are some problems (invalid data during the registration process or during the creation of an activity etc).

1.3.2 Acronyms

- RASD: Requirements Analysis and Specification Document

1.3.3 Abbreviations

- Gi: i-goal
- Ri: i-requirement
- Di: i-domain assumption
- Ai: i-assumption
- DEi: i-dependency
- Ci: i-constraint
- Ui: i-use case

1.4 Revision history

1.5 Reference documents

- Specification Document: Mandatory project assignment;
- IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications;
- "Alloy come strumento di analisi e progettazione di sistemi software" (graduate thesis of Claudio Vallorani);
- Alloy Language Reference;
- Example RASD of previous years;
- "Introduzione all'arte della composizione tipografica con LATEX".

1.6 Document structure

This document is structured as follows:

Section 1: Introduction In this section it is described the purpose of this document, the main goals of the given problem and a brief description of its main characteristics.

Section 2: Overall Description It provides further information about the application with a summary of major functions and it states all the assumptions and the constraints.

Section 3: Specific Requirements In this part we include more details about the requirements.

Section 4: Formal Analysis using Alloy This section provides the Alloy model and all the proves that it supplies.

Section 5: Effort spent Here are reported the information about the hours of work spent by each member of the group by doing this project.

Section 6: References

2 Overall description

2.1 Product perspective

Our application requires a smartphone (iOs/Android) to be executed and it requires an internet connection in order to benefit from the application's main services. It also requires an active GPS connection to identify the user's position. Travlendar+ is similar to other pre-existing applications that compute the best route with the best means to reach a specific location (e.g. Moovit), and like them, it is supported with updated timetables of all the travel means and with an estimation of travel time.

2.1.1 Class diagram

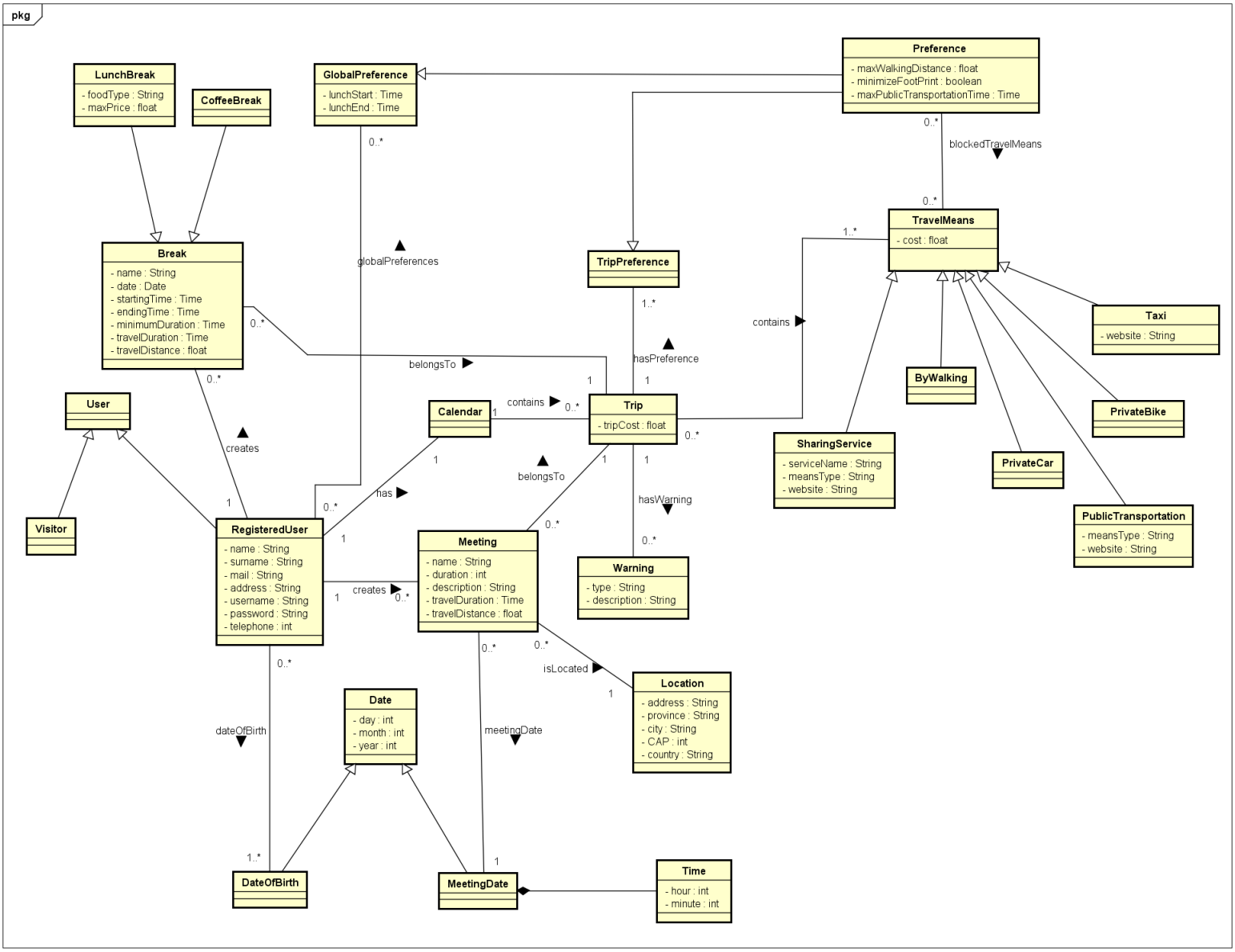


Figure 1: Class Diagram

2.1.2 Statechart diagram

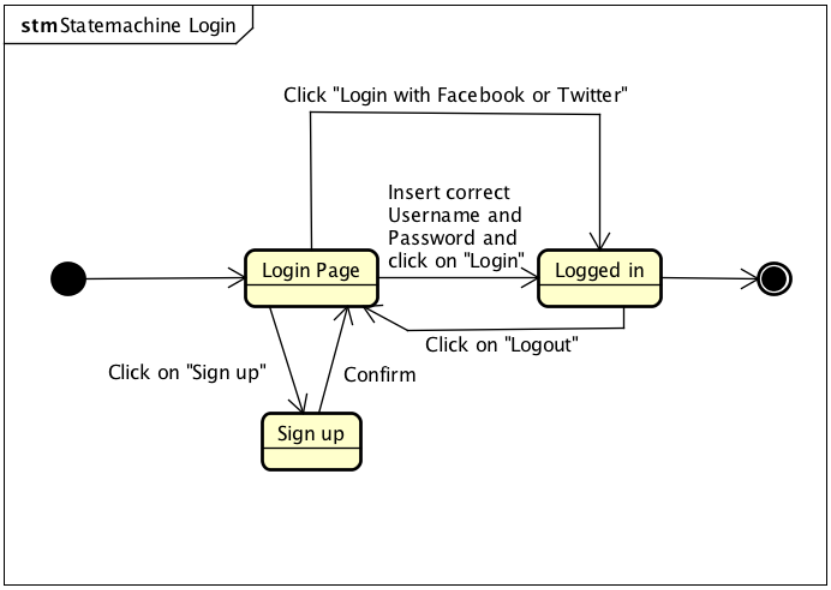


Figure 2: Login

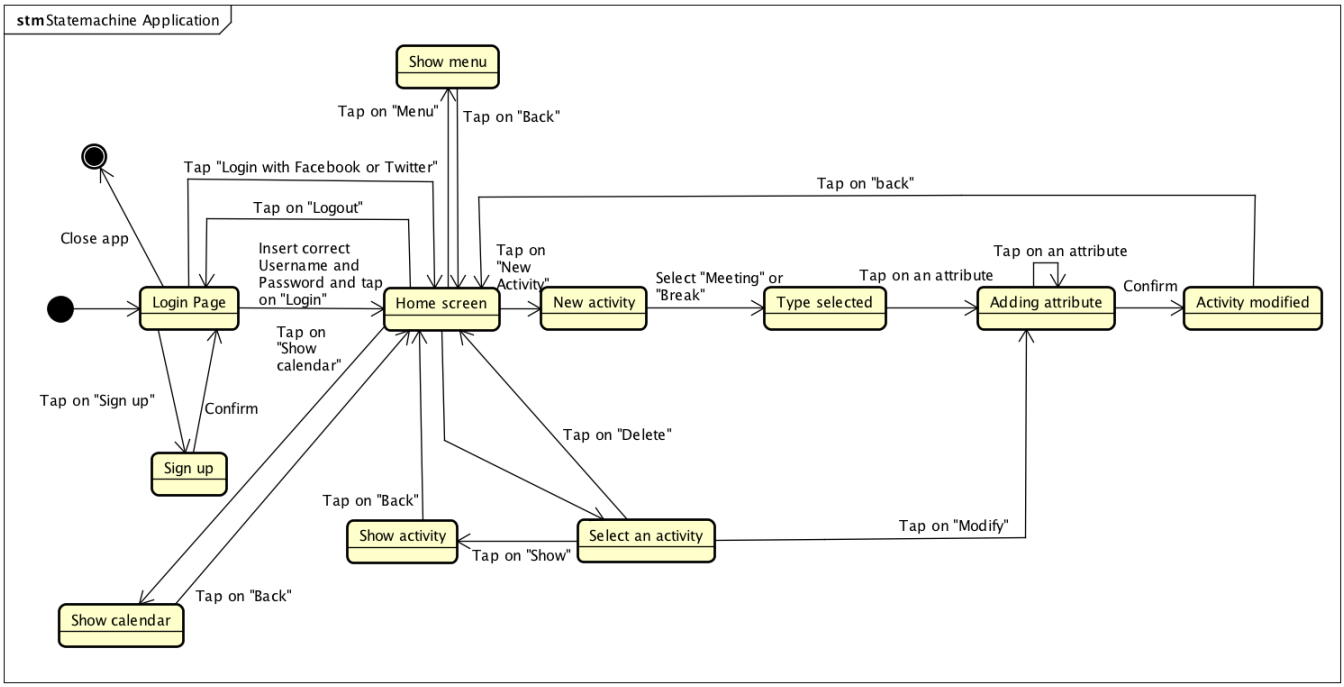


Figure 3: Application

2.2 Product functions

n this section, we will offer a specific list of the main features of the application in order to explain better its utility.

Offline mode Travlendar+ could be used in offline mode simply as a basic calendar. Users' can visualize on the main screen all their activities for the day, by clicking them they can see all details about the location and the time. In this mode, it isn't possible to see the map with the computed trips. Users' can also insert new activities by pressing the plus button, whereas if they click on the calendar icon, they are able to visualize the whole calendar.

Map The application offers the possibility to visualize the map of the interested area. If there is any activity saved in the calendar for the current day, an indicator is shown in its location. On the map, a blue line highlights the trip between the different meetings of the day. By pressing an activity appears also the chosen path to reach it from the actual location. By pressing "GO" it can be visualized the whole path of the day.

Personalization Travlendar+ offers a lot of options that can be customized, for example, the user can:

- specify a minimum duration of the lunchtime;
- specify everyday lunchtime;
- activate or deactivate travel means;
- insert the maximum walking distance;
- select combinations of transportation means that minimize carbon footprint;
- deactivate public transportation in a certain time zone.

The best route's computation Once a meeting is created Travlendar+ suggests the best route between the previous, the current and the next activity of the day that satisfies the constraints of the user. If the user is not satisfied with the proposed trip, he/she can choose another one between those suggested by the app. Moreover the user can buy tickets online or find the nearest vehicle of a sharing system if he/she needs.

Real-time information Every time the system is updated with the information about the traffic, the weather, strikes etc. The user can either receive the respective notification or visualize all the warnings on the main screen. All these warnings influence the choice of the travel means for the trip.

Possible future implementations There are some interesting ideas that can be applied to Travlendar+ in the near future. Here we make some examples.

- Users can specify in the calendar also how they want to spend their break time. If they have a lot of time between an appointment and another, they can decide to have an "entertainment break". In this case they will select it, and Travlendar+ will inform them of the entertainment activities near the meeting point (i.e. cinemas, museums...).
- The external websites on which Travlendar+ deals could be integrated. In this way users are not redirected to other websites, but they can manage all from the same application. For example, they would not need to use the Trenitalia's website to buy train tickets (this implies that they also have to register and insert their payment method to other websites ecc..), or they will not need to install other applications to use a vehicle-sharing service. In order to do this, it will be requested to the user during the registration process to insert the payment data. Since the payment data need more protection, it could be implement a system that sends the password to the user at the moment of the registration to guarantee more security.
- Users can invite friends to an event at which they participate.
- Users can ask some friends that participate at the same meetings to share part of the trip with them.

2.3 User characteristics

Travlendar+ has no specific user target: everyone, that is able to use a device with an internet connection, can take advantage of this new type of calendar application. However, the user that we expect to benefit most is a busy person who wants to organize his/her daily activities in the best way, finding the fastest and the most comfortable way to travel between two activities and

to reach the destination on time. To own a personal calendar, the user must have a device with an internet connection and register with all necessary data or sign in with Facebook or Twitter.

2.4 Assumptions, dependencies and constraints

Assumptions

[A_1] Accurate registered user's location is known by GPS.

[A_2] The user knows the right details about the activities.

[A_3] The public transportations are always in time.

[A_4] The external third part services (e.g. vehicle sharing systems, google maps, travel means' websites ecc...) work correctly.

[A_5] The news about the traffic, the weather and strikes are correct and known in advance.

[A_6] Maps are always updated.

[A_7] The user attends meetings in existing locations.

[A_8] Public transportations have services that allow a user to buy tickets online.

Dependencies

[DE_1] The application depends on external application and websites for some services, such as: the purchase of public transportation's tickets, the use of vehicle sharing systems.

[DE_2] The application bases on some other services like Google maps, meteo services and traffic services and uses the received information to guarantee its services.

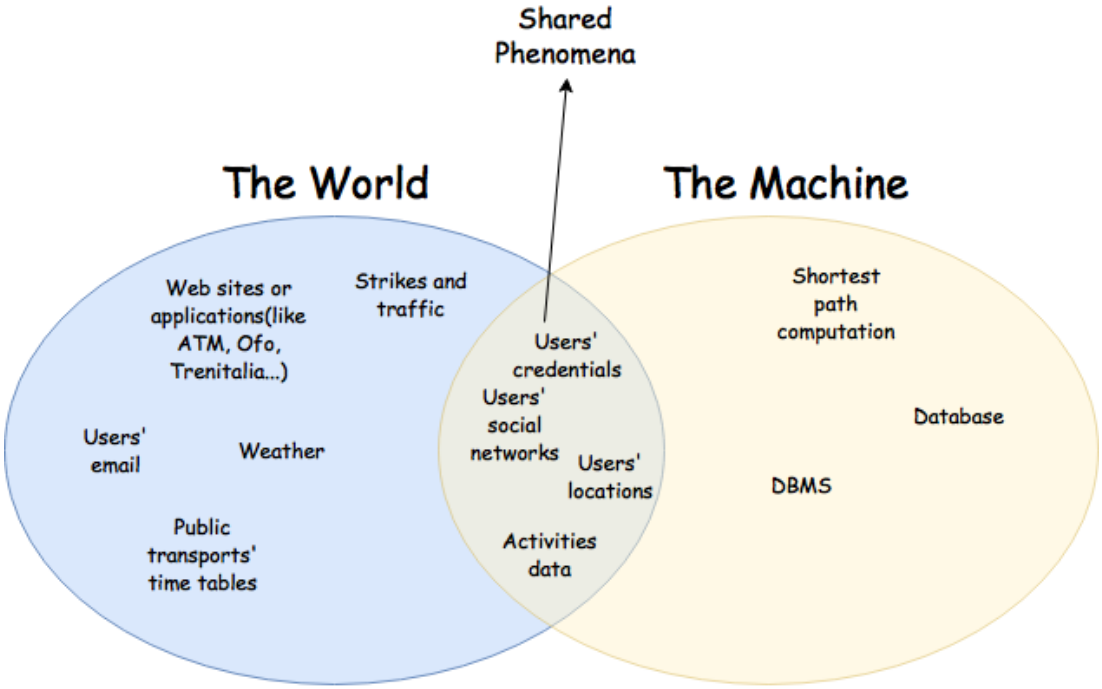
Constraints

[C_1] Travlendar+ requires a smartphone or a tablet with internet connection in order to offer its functionalities.

[C_2] Travlendar+ needs a GPS service to find the correct location of the user.

2.5 The World and The Machine

In order to analyze Travlendar+, we would like to consider The World and Machine model. It was presented in 1995 by M.Jackson and P.Zave and it is useful to identify requirements. In this way, we can divide them into three parts: the portion of the real-world that interact with the application (the World), the portion of the system to be developed (the Machine) and the intersection between the two (Shared phenomena), that are all world information known or managed by Travlendar+. We insert also an example of these components, even if it is not complete.



3 Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

Our application has been designed to be used through a smartphone or a tablet. The following mock-ups show the screens of the main features offered by the smartphone version.

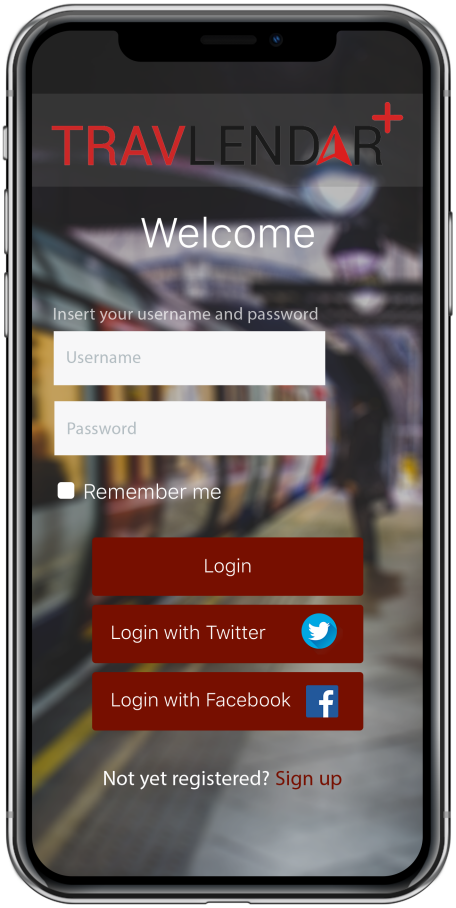


Figure 4: This is the login screen that appears when a visitor opens the application.

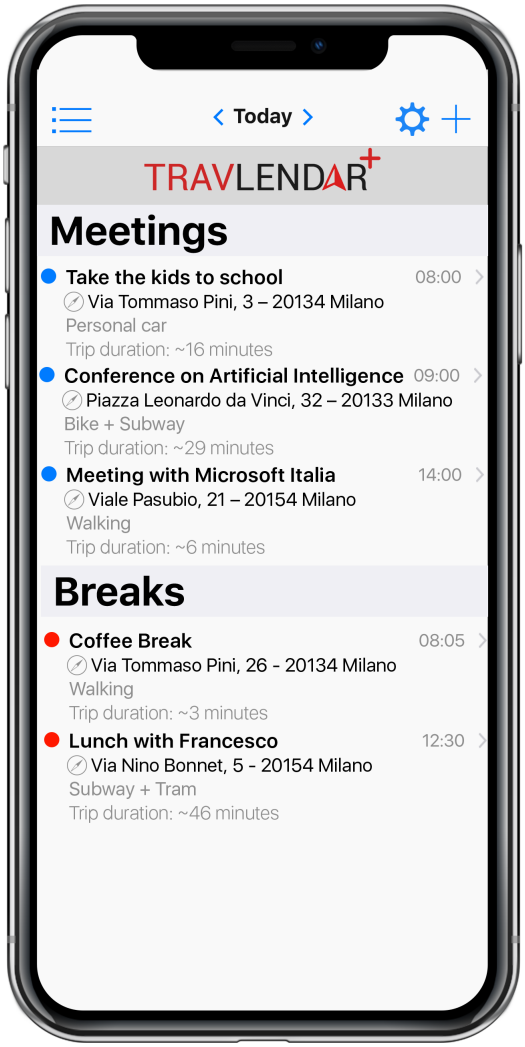


Figure 5: This is the main screen that appears when a registered user logs in.

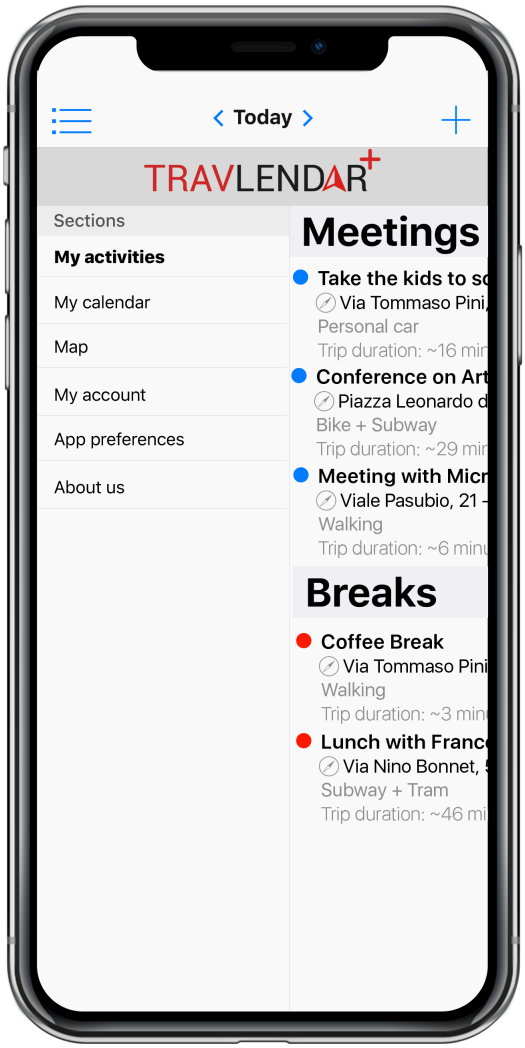


Figure 6: To modify user's preferences, to go to the map screen and for other features, the user should tap on the menu icon on the left top corner.

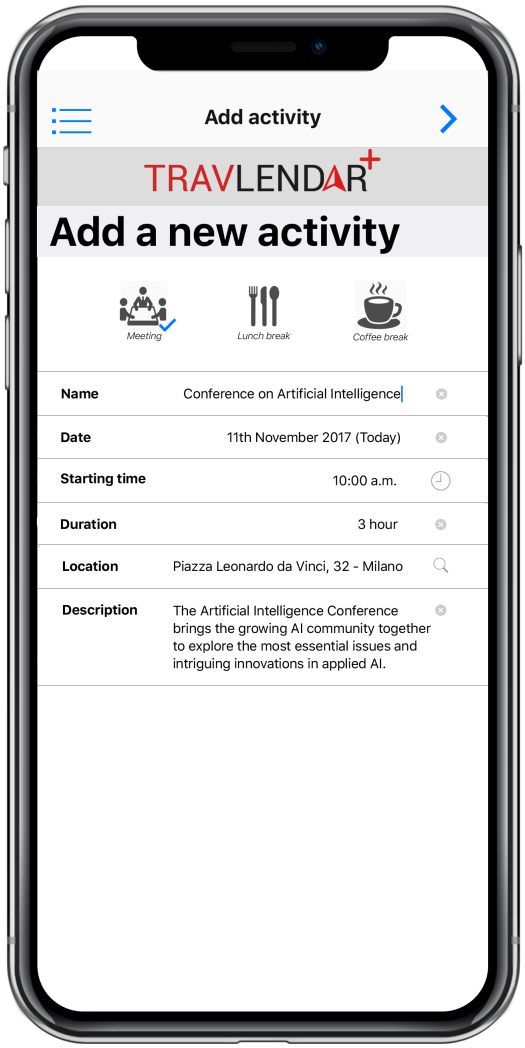


Figure 7: This is the screen that allows the registered user to add a new activity, by specifying its type and details.

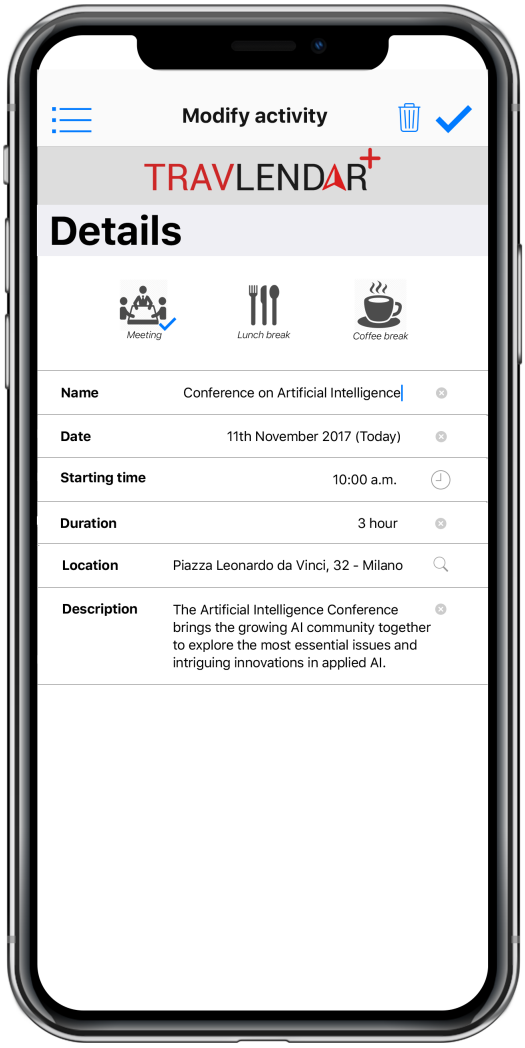


Figure 8: This is the screen that is shown to the registered user when a tap on an existing activity is performed. This allows the registered user to modify or remove the entire activity.

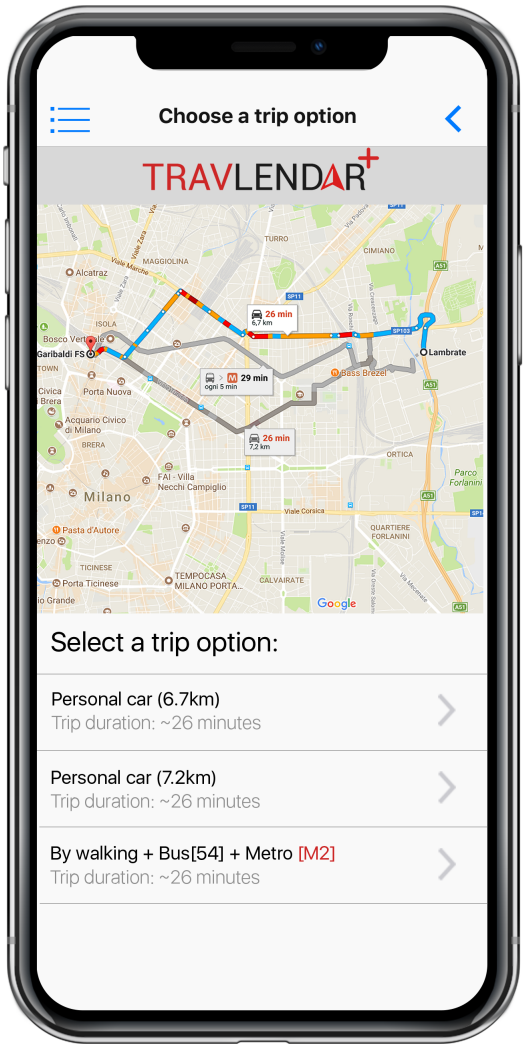


Figure 9: This is the screen that shows the various trip options suggested by Travlendar+ after that the registered user has inserted a new activity.

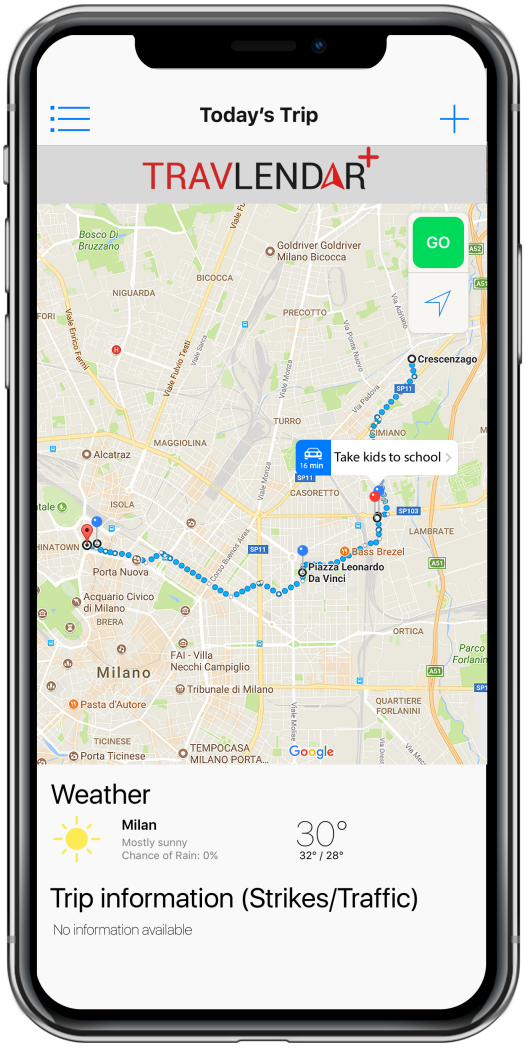


Figure 10: This is the screen that shows the map with all activities of the current day. Moreover, it shows the different trips that the user should do. The activities are shown through a tap on the pointer.

3.1.2 Hardware interfaces

The application does not require any hardware interface.

3.1.3 Software interfaces

Travlendar+ does not provide for itself the possibility to directly buy public transportation's tickets and the possibility to use sharing systems, but redirects the user to the corresponding website or, if it's already installed in the device, to the corresponding app.

3.1.4 Communication interfaces

The application needs an internet connection on the device in order to receive real-time information about traffic, strikes and weather. Furthermore, it's required also for the communication with third-party services that are provided in Travlendar+.

3.2 Functional requirements

Prerequisites

[R_1] The system should allow the visitor to begin the registration process by asking him/her all mandatory data: name, surname, date of birth, email, username, password.

[R_2] The system should be able to extract data from Facebook or Twitter if the visitor signs in with that application.

[R_3] The system must not allow a registered user to perform registration process.

[R_4] The system must allow visitors only to see the login page and the registration form.

[R_5] The system should allow only registered users to log in.

[R_6] Username and password inserted must be correct to perform the login.

Requirements

[G_1] A user should be able to go to a meeting.

- [R_1] The system should allow registered users to create meetings.
- [R_2] Users must complete the form by filling the mandatory fields and confirm the creation.

[G_2] A user should be able to know if he/she cannot arrive on time to a meeting.

- [R_3] The system should be able to calculate necessary time for each trip option.
- [R_4] The system should be able to notify the user if there is not any option that makes him/her arrive on time.
- [R_5] Users should be able to close the warning.

[G_3] A user should be able to have a break that has a certain duration between two meetings.

- [R_6] The system should allow the user to create an activity with type "Break".
- [R_7] The system should allow the user to specify the minimum duration of the break.
- [R_8] The system should be able to provide a solution to have a break if a user specifies it.

[G_4] The time or the location of a meeting can be modified.

- [R_9] The activity that the user wants to modify must be already created.
- [R_{10}] Users must confirm the modification to conclude the process.
- [R_{11}] Modified data will not be available any more after the update.

[G_5] A user should be able to unsay a meeting.

- [R_{12}] The activity that the user wants to delete must be already created.
- [R_{13}] Users must confirm the elimination of the activity.

[R_{14}] Activity will not be seen on the calendar and on the map.

[R_{15}] Deleting process is not reversible.

[G_6] A user should be able to have some preferences for the trip.

[R_{16}] The system must keep track of user's preferences.

[R_{17}] A travel means that should be deactivated must be active.

[R_{18}] A travel means that should be activated must be non-active.

[R_{19}] The system should not allow users to deactivate all travel means otherwise the computation of the trip is not possible.

[R_{20}] The system should be able to calculate the carbon footprint and to minimize it.

[R_{21}] The system should not allow users to impose constraints on deactivated travel means.

[G_7] A user should be able to know every possible option to go to a meeting with different travel means in order to choose the suitable one.

[R_{22}] The system should be able to compute the different routes with several travel means.

[R_{23}] The system should be able to know the activities of the day and provide a route between them.

[R_{24}] The system should be able to calculate the necessary time for each trip option and shows it to the user.

[G_8] A user should be able to buy public transportation tickets or day/week/season pass basing on his/her needs.

[R_{25}] The system should be able to open the application or the website of the service that provides that type of trip.

[R_{26}] Users must confirm if they want to open an external application.

[G_9] A user would like to see on Tripadvisor advised places for his/her breaks.

[R_{27}] The system should be able to open the application or the website of Tripadvisor.

[R_{28}] Users must confirm if they want to open an external application.

[G_{10}] A user would like to locate the nearest vehicle of a vehicle sharing system if he/she wants to use that type of travel means.

[R_{29}] The system should be able to open the application or the website of the service.

[R_{30}] Users must confirm if they want to open an external application.

[G_{11}] A user would like to have real-time news on strikes, weather and traffic.

[R_{31}] The system should be informed about weather, strikes and traffic in real time exploiting other systems.

[R_{32}] The system should be able to notify the user if there are problems that can change his/her trips.

[R_{33}] Users should be able to close the warning.

[R_{34}] Users should be able to change the trip.

[R_{35}] The system should be able to open again the screen with the different trip options

3.2.1 Scenarios

Scenario 1 Dybala is a very busy football player that lives in Turin. Tomorrow morning he is going to have an important meeting with one of the main sponsor of the Juventus team, the Adidas, in Milan, and in the evening he has the Champions League's final. It is the most important match of the year, so he needs to organize his day at the best! He decides to use Travlendar+. He inserts the two activities in the list of meetings: "Meeting with Adidas", on 20th of May, from 11 am to 1 pm, in Corso Buenos Aires 40 (Milan) and "Match prep for the Final", on 20th of May, from 3 pm to 7 pm, at the Allianz Stadium (Turin). Since he wants to preserve energies for the Final, he selects "Walking" and "Bike" as blocked travel means in the Preferences' area. The application suggests him a trip with the train from Turin to Milan in the early morning because the estimated time for the route by car or taxi is reported to be slower due to the traffic. The application

redirects him to the Trenitalia's website to buy the ticket. Then, for coming back, Travlendar+ suggests him to take a taxi because public transport timetables for the given time are not convenient.

Scenario 2 Giuseppe is a student at Politecnico of Milan. In view of the exams session, he has not much time but he does not want to give up a good lunch with his friends. Therefore, he specifies in the "My Account" section of Travlendar+ that every day he needs at least 30 minutes free to have a comfortable lunch between 11.30 am and 2.30 pm. He inserts the schedule of all his lessons. After having inserted the Formal Languages and Compilers' lesson of Wednesday, a warning appears to report an overlap with the fixed time reserved for the lunch. He can ignore the warning and decide what he prefers to do.

Scenario 3 Carlo Cracco is a famous chef that has two restaurants in Milan. He would like to open a new restaurant in Verona. So he has to go to Verona to search for a place that can be restored as he likes and to meet architects and designers to decide all details of the new restaurant. He would like to have a coffee break between the afternoon appointments to restore himself. Therefore he uses Travlendar+ to organize the full day. He inserts all the meetings and he also inserts the coffee break: pressing the plus symbol on the main screen, Carlo selects the type "Break", he inserts the minimum duration for the break and he is readdressed to the website "TripAdvisor". In this way, the chef can see what coffee is opened near the two appointments that he has before and after the break and he can choose the best one.

Scenario 4 Lorenzo Fragola is a young boy with a strong passion for singing, so he decides to go to an audition for XFactor, the most famous music talent show in Italy. The audition will take place on the 30th of July in Milan, at Arena Civica. Lorenzo lives in Catania and decides to use Travlendar+ to program the trip for the audition. He inserts the meeting in the system, with the date, the location and the time, and visualizes the trip. The two available means suggested by the application are the train, which takes 13 hours, and the car, which takes 14 hours. Since both take too long, he decides to organize himself the trip with the plane. Once he will be in Milan, he will use the application to go to the Arena Civica. The

trip suggested is by autobus from the airport to the railway station, and then by metro to the Arena. Unfortunately, one week before the audition, a warning appears on his phone: on the day of the audition, there will be a strike of ATM! At first Lorenzo is a little bit worried, but immediately sees that Travlendar+ has suggested him an alternative trip by car (taxi or car sharing) from the railway station to the Arena. He will choose at that day the best option. When the moment arrives, he selects car sharing as favourite means, and the application redirects him on the play store to download the Enjoy's application (the car-sharing service). Through Enjoy's application, he can see that there is a car near the station and he can book it.

Scenario 5 Travlendar+ has become a very well known application. A lot of people start to use it and find Travlendar+ the most useful application of 2017. Also, the USA president has become curious because of the fame of this new type of smart calendar. He decides to try it: he has an iPhone X and he goes to the AppStore and installs it. Once it is installed, Trump opens it and he finds out that he has to sign in. He clicked the "Sign in" button and a new screen appears. He has to insert a lot of personal information: name Donald, surname Trump, date of birth 14/06/1946, address 1600 Pennsylvania Ave NW, Washington, username Tump, password ******** and at the end the telephone number +1 202-456-1111. Once he finishes compiling, he clicks "Sign in" and the screen for the registered users appears: there is the screen with the schedule of the day that is obviously empty for new users.

Scenario 6 Mr. Bean is a very busy man that uses Travlendar+ to manage his great number of appointments. Once he inserts a work meeting in the morning between 10.00 and 12.00, he decides to go with his loved car. After that, he knows that he has to be present to a ceremony for his teddy bear's birthday that should start at 11.45 a.m.. This cannot be possible!!! Travlendar+ notifies him with a warning that there is an overlap between two appointments. Therefore Mr. Bean closes the warning and decides to go out of the conference before to arrive on time at the birthday. Mr. Bean is very excited and decides to have a stop in a toy shop to buy a present. So he inserts in the application the new activity but there is another problem. With a warning, Travlendar+ warns him that if he wants to arrive on time at the ceremony, he cannot go to the shop. Hence he does not confirm the creation of the activity.

Scenario 7 Leonardo DiCaprio is a famous actor that has always been active in fighting climate change. To raise awareness among the people about the environmental problems, he has the idea to live stream one of his ordinary day to show people the importance of using means of transport that minimizes carbon footprint. Leonardo decides to use Travlendar+ in order to arrange his trips in the day. Once the application is installed and he creates his personal account, Leonardo inserts three activities (all in New York) to be performed during the day. The application now suggests the ideal route that minimize travel durations, but in order to pursue his goal Leonardo enter the "My Account" section of the app and then he enables, in the preferences' area, the option "Minimize footprint". The route suggested by the application seems to be more eco-friendly, for example, Travlendar+ proposes to him to rent a bike or to use a service of electric car sharing.

Scenario 8 Claire is going to have a marry of her best friend on Sunday morning. Since after the ceremony there will be a big party with food and drink, she decides not to drive. To discover the best option to reach the church, she uses Travlendar+. The application suggests her to take the metro and then to walk for ten minutes from the underground station to the destination. It's all perfect for her, but unfortunately, she wakes up on Sunday morning and it is raining. She can not afford to walk under the rain because she has just gone to the hairdresser! She immediately logs in to the application to check an alternative trip. Travlendar+, that has an updated weather forecast, has already changed the trip: it suggests her to take the tram once exited from the underground station instead of walking. She notices looking at the tram schedule in the activity's details that she will not take much more to go there with this alternative because trams of line 34 stops every 13 minutes. She will reach the destination in time.

Scenario 9 Guglielmo is a Computer science engineer. He works for a big company: he earns a lot of money. Because he often had to move, he used a lot his car. Guglielmo was very reckless, in fact, he went on taking fines, even very expensive. He didn't care about it. But once the police gave him another big fine and withdrew his driving license and confiscate his car. Since he lives in Milan and he hates public transports, he decides to try the application Travlendar+. He enrolls in the app and in the section "My Account" he deactivates every public transportations and sharing cars. He

has to go to work: when the trip is suggested, he presses on bike sharing's option because he really love the speed and he has a lot of time to reach the location. Guglielmo is redirected to the Ofo's application and from that application, he can see where is the nearest bike. Finally, he can go fast without taking any fine!

3.2.2 Use case diagrams

Registration

Use case ID	$[U_1]$
Actor	Visitor
Input condition	NULL
Event flow	The registration's process is: 1. The visitor tap on "sign up" on the home page to start the registration process. 2. The visitor fills in at least the mandatory fields, that are: name, surname, username, password, email, address, date of birth. The non-mandatory field is the telephone number. 3. The visitor tap on "ok". 4. The application saves the data and redirects him to the home page where he can proceed to the login.
Output condition	Registration process is completed. The visitor becomes a registered user.
Exception	The possible exceptions are: 1. The visitor is already a registered user. 2. The visitor fills one or more fields with invalid information. 3. The visitor chooses an already existent username. 4. The visitor inserts an email that is already associated with another user. In case of an exception the invalid fields are coloured with red. The visitor can't press "ok" until all fields are correctly filled.

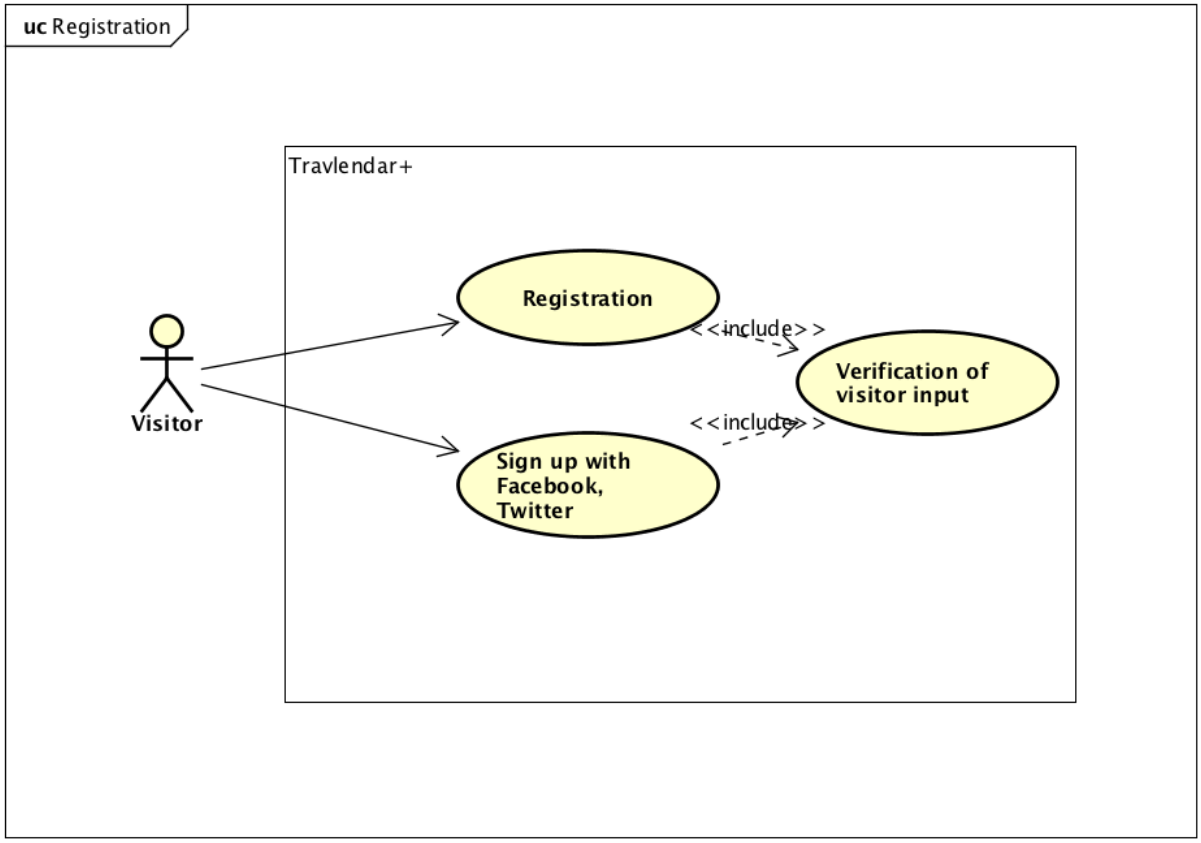


Figure 11: Use case of the registration process

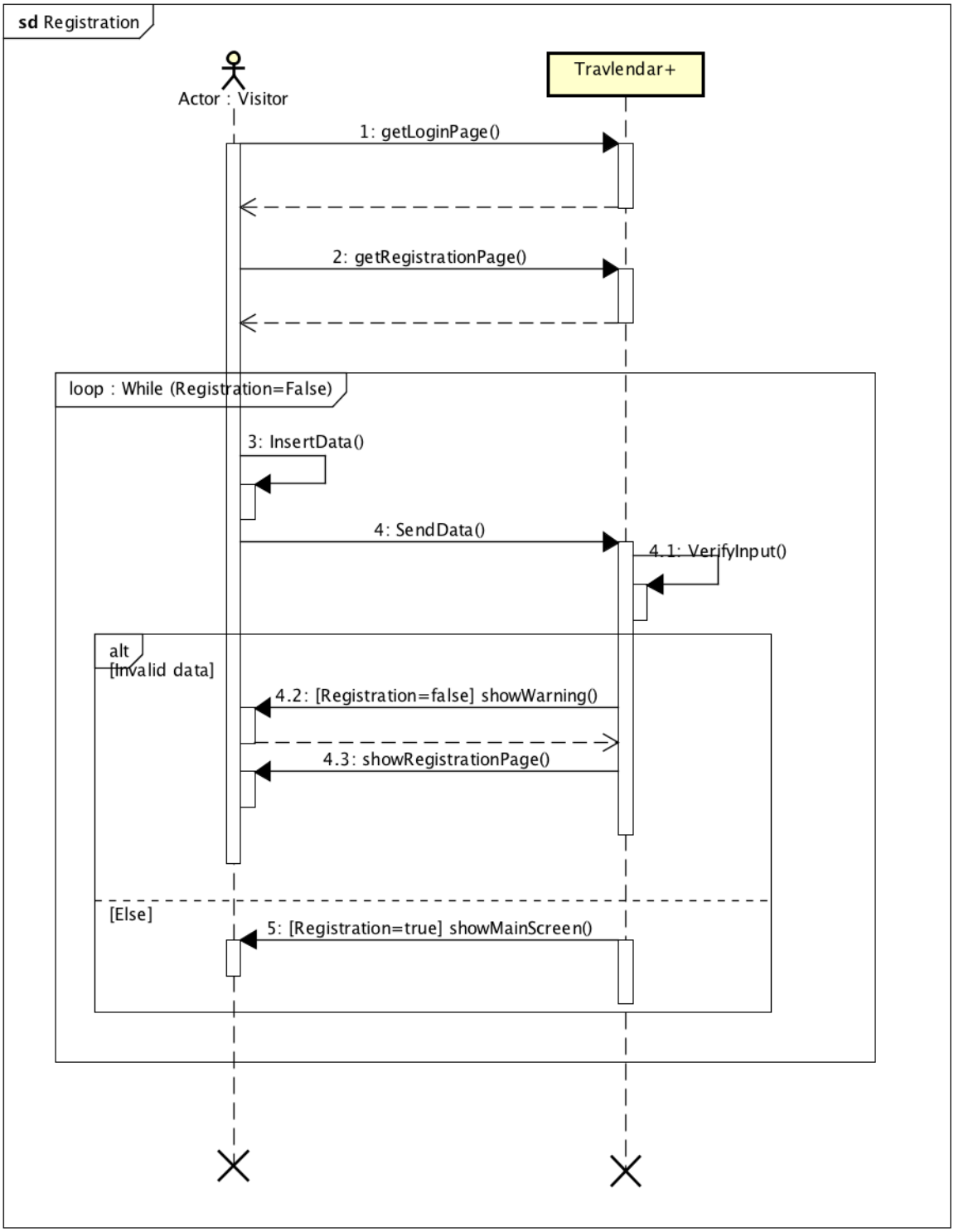


Figure 12: Sequence diagram of the registration process

Login

Use case ID	$[U_2]$
Actor	Visitor, registered user
Input condition	The user is on the homepage.
Event flow	The login's process is: 1. The registered user fills the fields "username" and "password" with her/his username and password. Alternatively, a simple visitor can tap on "Login with Facebook" or "Login with Twitter" in order to login directly without having completed the registration process. 2. If the access is not with Facebook or Twitter, after having inserted the username and password the registered user must tap on "Login".
Output condition	Travlendar+ verifies user's credentials (either they have been directly inserted by the user or obtained through Facebook or Twitter) and if they are correct shows the main screen with the map with the activities.
Exception	The possible exceptions are: 1. The user inserts an incorrect username or password. 2. The user's data obtained by Facebook or Twitter are invalid. In both the cases, the user is notified with an error message and invited to try again.

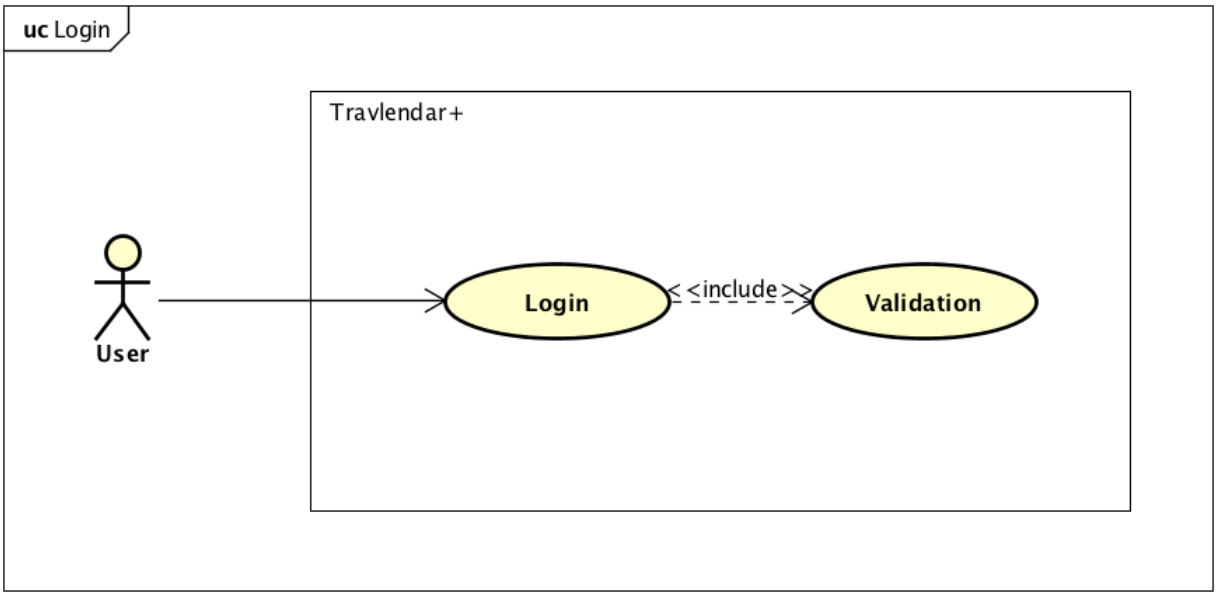


Figure 13: Use case of the login process

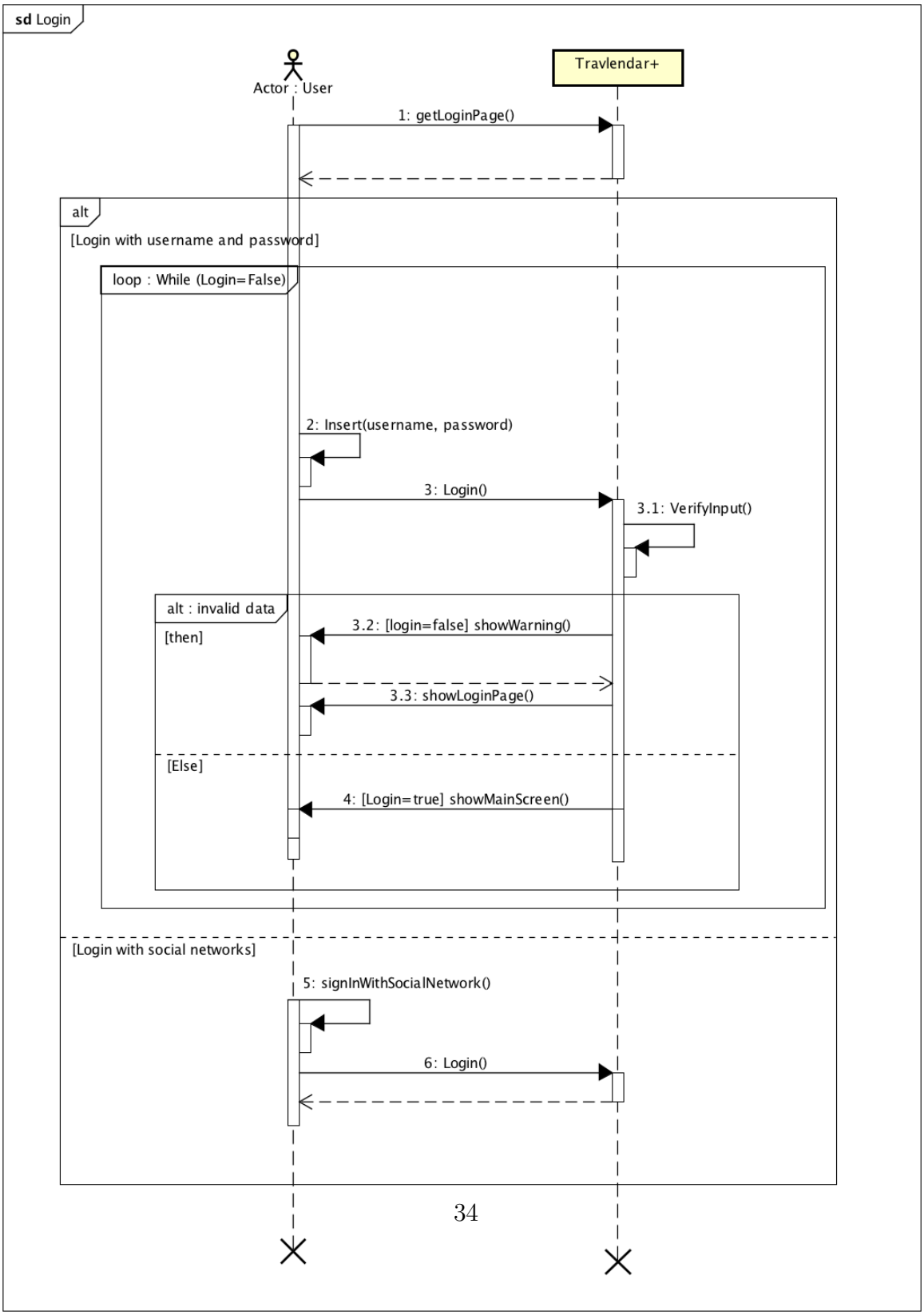


Figure 14: Sequence diagram of the login process

New activity

Use case ID	$[U_3]$
Actor	Registered user
Input condition	The registered user is already logged into Travlendar+.
Event flow	The process for the creation of a new activity is: 1. The registered user tap on the plus icon ("new activity button"). 2. Travlendar+ shows a new page with the form, and the creation process starts. The user is required to specify the type of activity (meeting or break) and to insert all the attributes about the activity, such as name, date, time, location and favourite means. If the activity is of type "break", the registered user must also specify a minimum duration of the activity. 3. The registered user tap on "ok".
Output condition	Travlendar+ redirects the registered user on the main screen. On the map is appeared the new activity on its location.
Exception	The possible exception is that the user does not fill a mandatory field or fills it with invalid data. In such cases, the invalid fields are coloured with red. The user must insert the correct data in order to complete the creation of the activity.

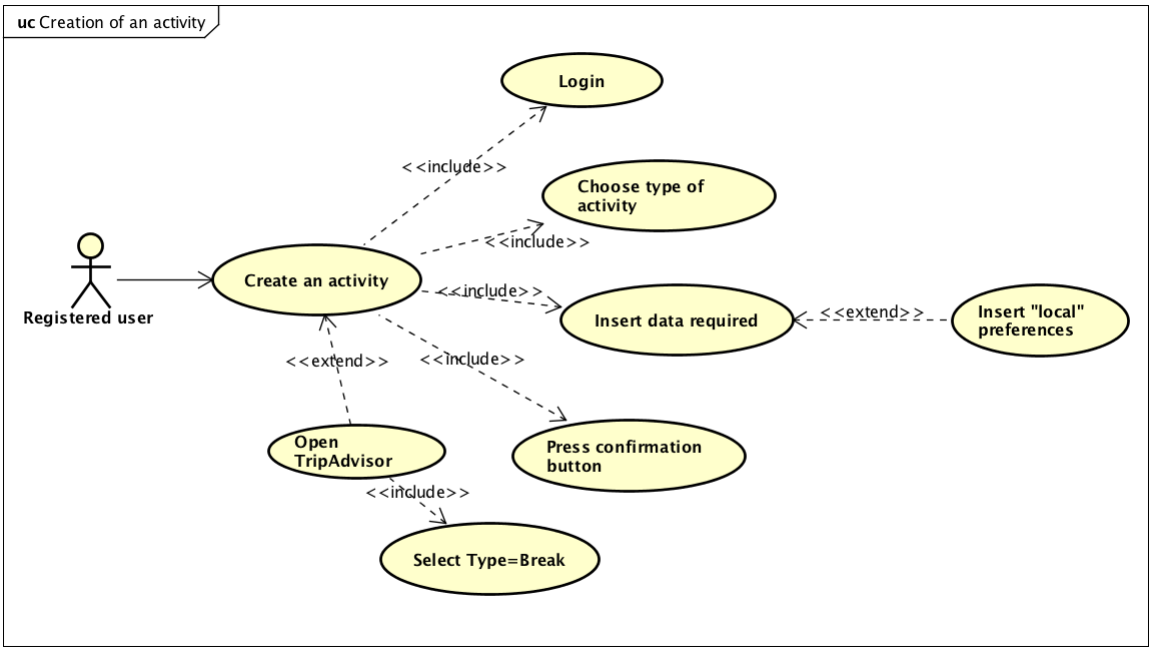


Figure 15: Use case of the creation of an activity

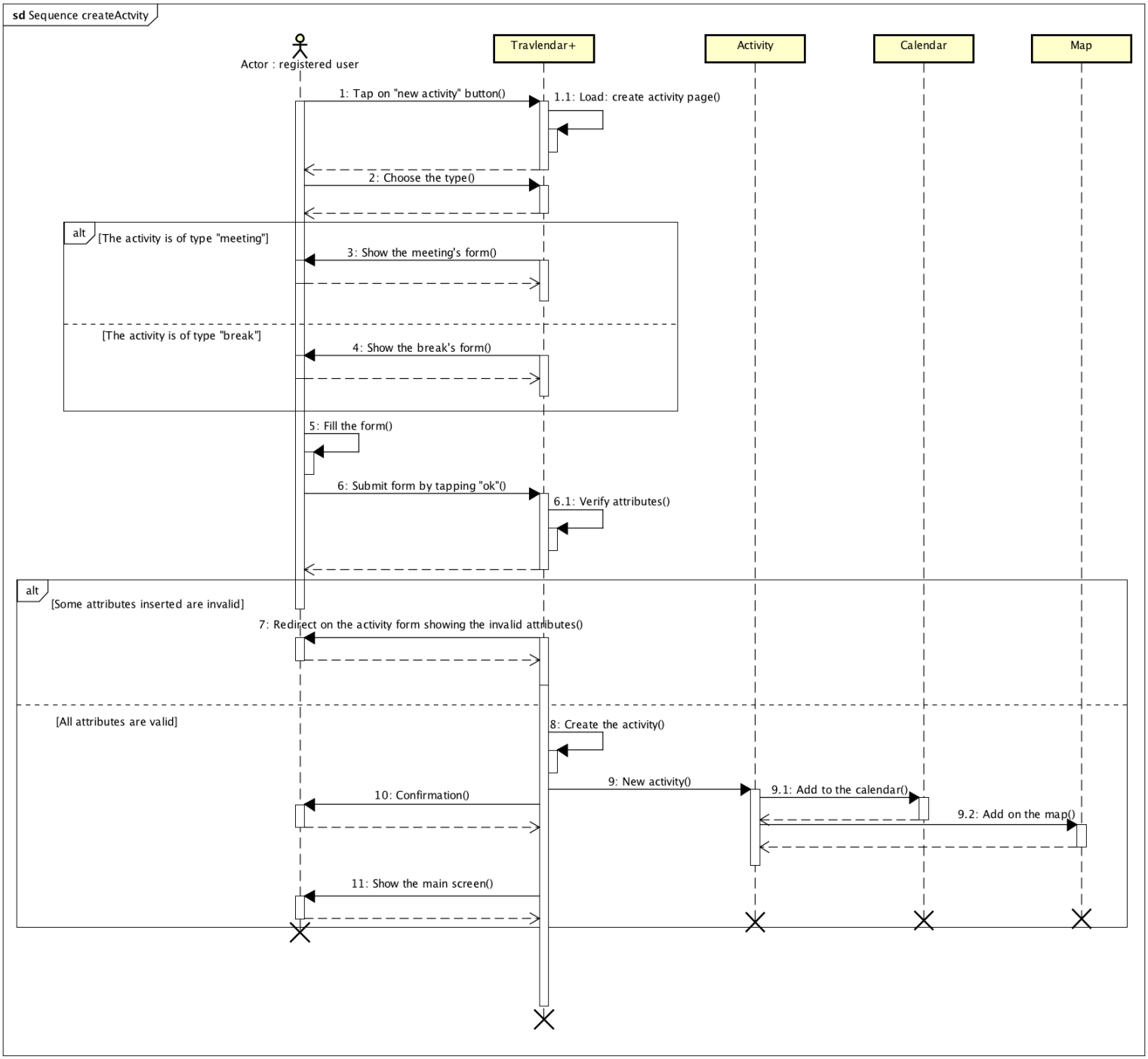


Figure 16: Sequence diagram of the creation of an activity

Modify activity

Use case ID	$[U_4]$
Actor	Registered user U
Input condition	U is the creator of the event A and is logged in.
Event flow	The process for the modification of a new activity is: 1. U taps on the activity A on the map (coloured with blue if A is a meeting, with red if A is a break), in the section "My activities" or in the main screen with the activity of today. 2. U taps on the "modify" symbol. 3. The application shows the screen form of the activity that contains the fields to fill with the attributes. 4. U can modify the fields by tapping on it. 5. U taps on "ok" when he/she has completed the modification. 6. The application updates the calendar with the new information. 7. The application redirects U to the main screen.
Output condition	Travlendar+ updates the calendar with the modified activity, then redirects U on the main screen. If the location of the location has been modified, on the map on the main screen the activity appears in the new location.
Exception	The possible exceptions are: 1. U inserts an invalid modification. 2. U does not want to apply the modifications. In the first case, the invalid field is highlighted with red. In the second one, the U can tap "cancel" when the system asks for confirmation and no changes will be made.

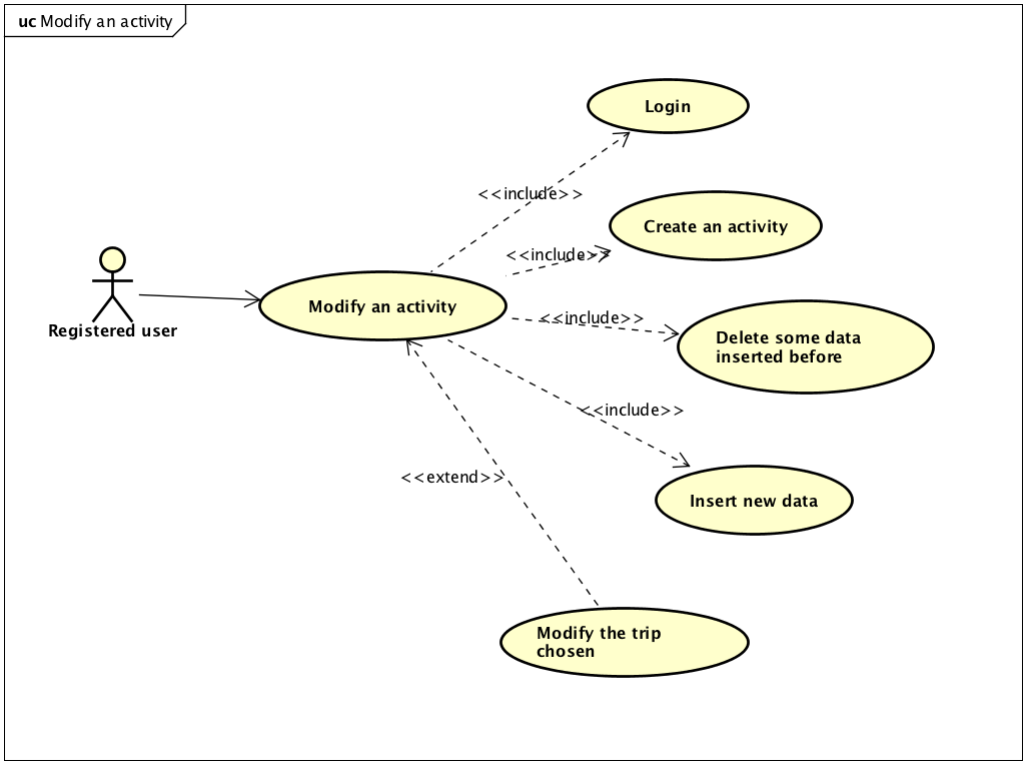


Figure 17: Use case of modifying an activity

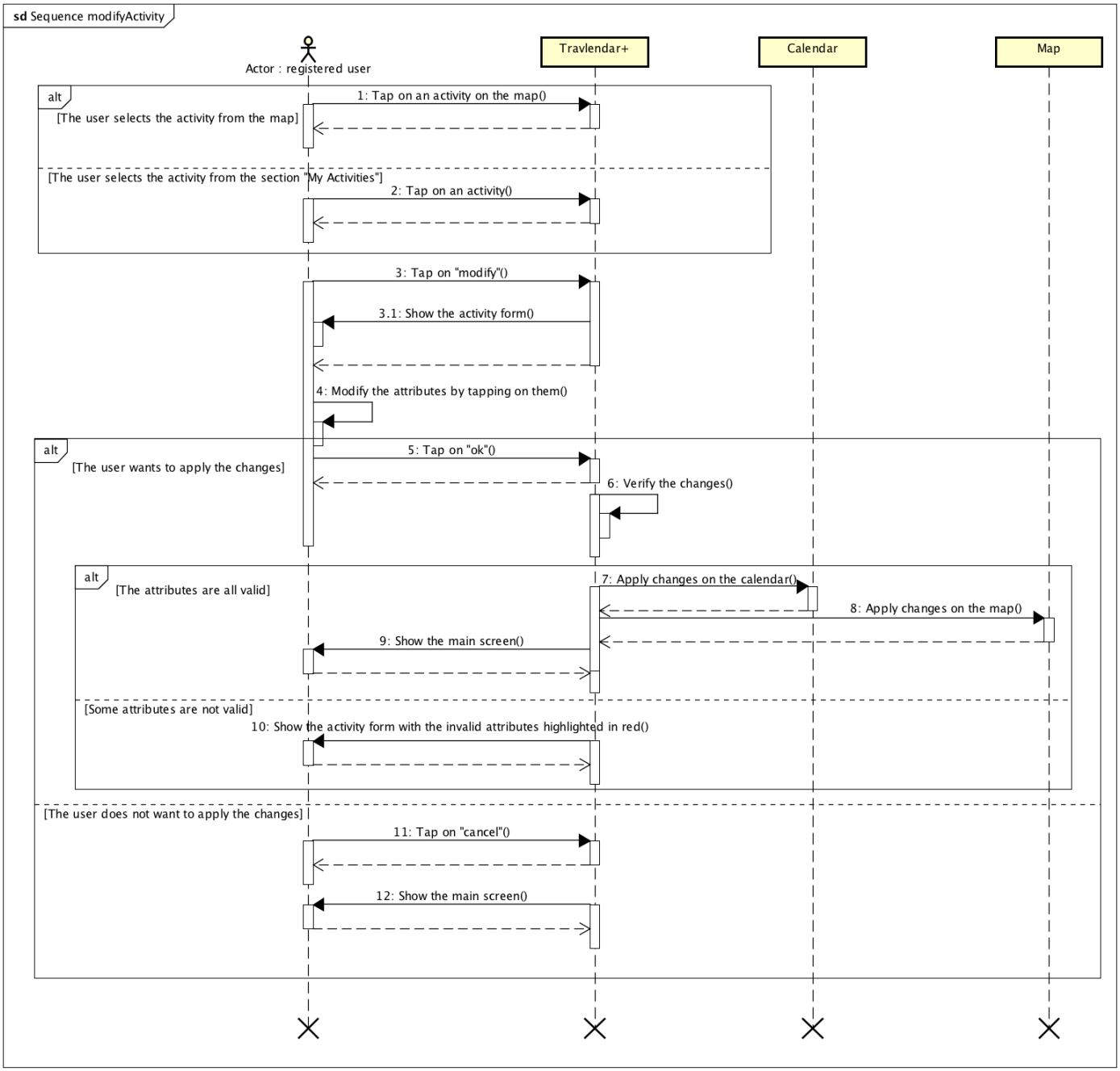


Figure 18: Sequence diagram of the modification of an activity

Delete activity

Use case ID	$[U_5]$
Actor	Registered user U
Input condition	U is the creator of the event A and is logged in.
Event flow	The elimination of an activity's process is: 1. U selects the activity from the map, from the section "My activities" or from the main screen with the activity of today. 2. U taps on "delete". 3. The application asks a confirm for the elimination. If U answers "ok", the activity is deleted from the calendar.
Output condition	Travlendar+ updates the calendar, in particular, it deletes the activity from the map and from the calendar.
Exception	The possible exception is that U has already tapped on "delete" but he/she does not want to delete the activity. In that case he/she can tap on "cancel" when the system asks for the confirmation.

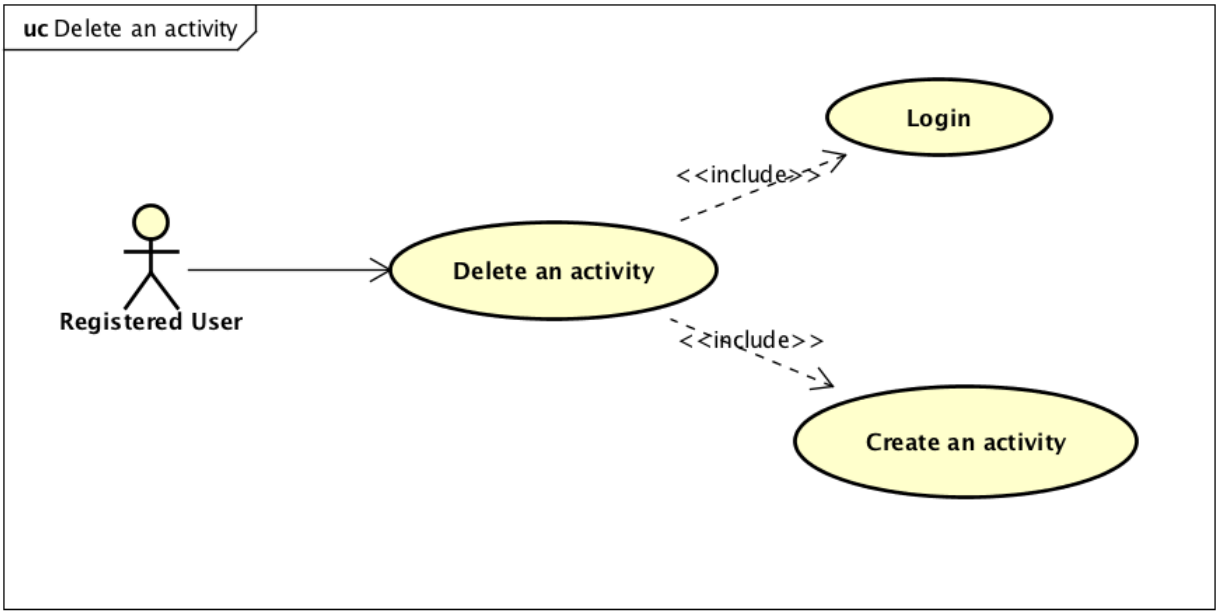


Figure 19: Use case of deletion of an activity

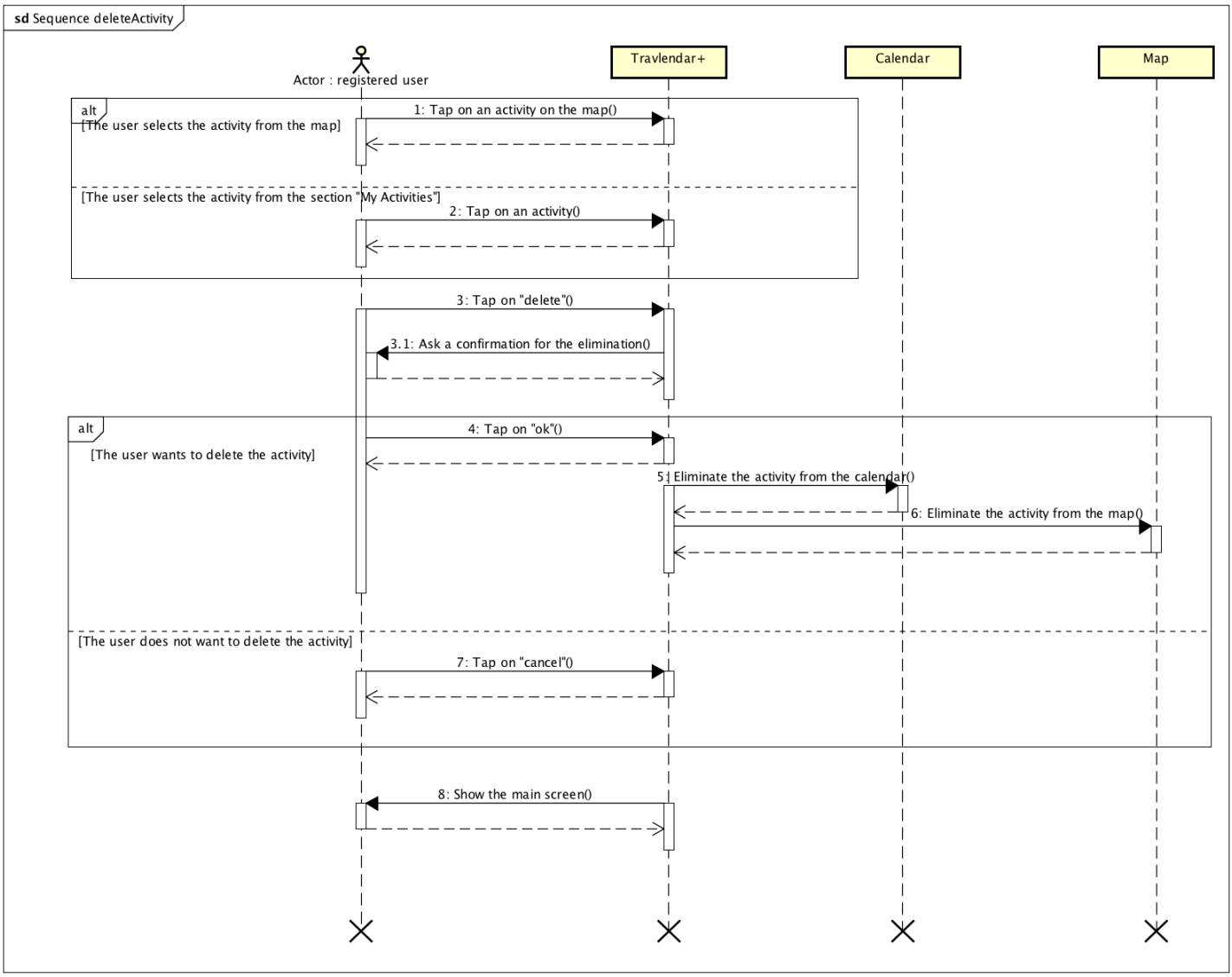


Figure 20: Sequence diagram of deletion on an activity

Choose a favourite trip

Use case ID	$[U_6]$
Actor	Registered user U
Input condition	U is the creator of the event A and is logged in.
Event flow	Once the activity has been created, the screen with the choice of the trip between all meetings of that day appears. U can choose one by tapping on it.
Output condition	Travlendar+ updates the calendar with all the details about the trip in the corresponding activities and highlights the trip between the meetings of the day in blue.
Exception	The possible exceptions are: 1. The trip chosen takes too much time, and U risks to not arrive in time. 2. The trip chosen is not available because of traffic or in case of a strike. In both cases, a notification arrives on U's screen and Travlendar+ invites him/her to do another trip.

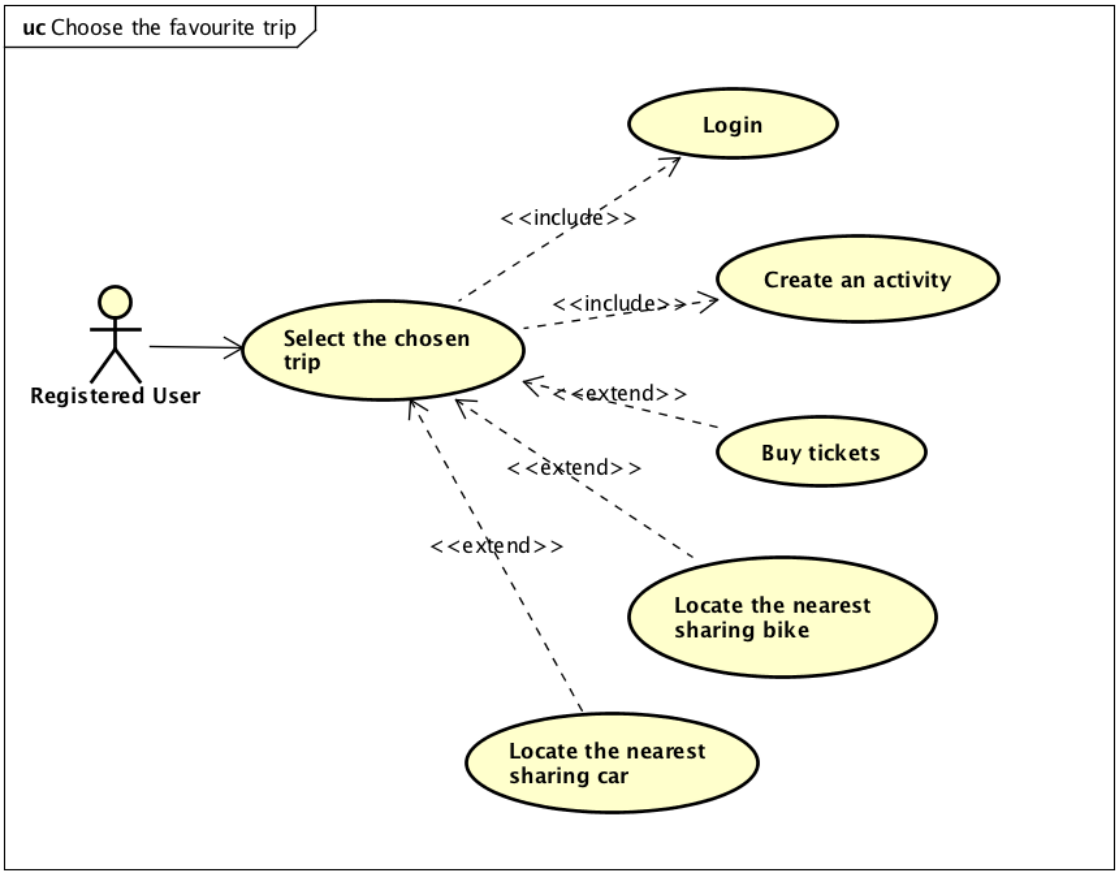


Figure 21: Use case of choosing the trip

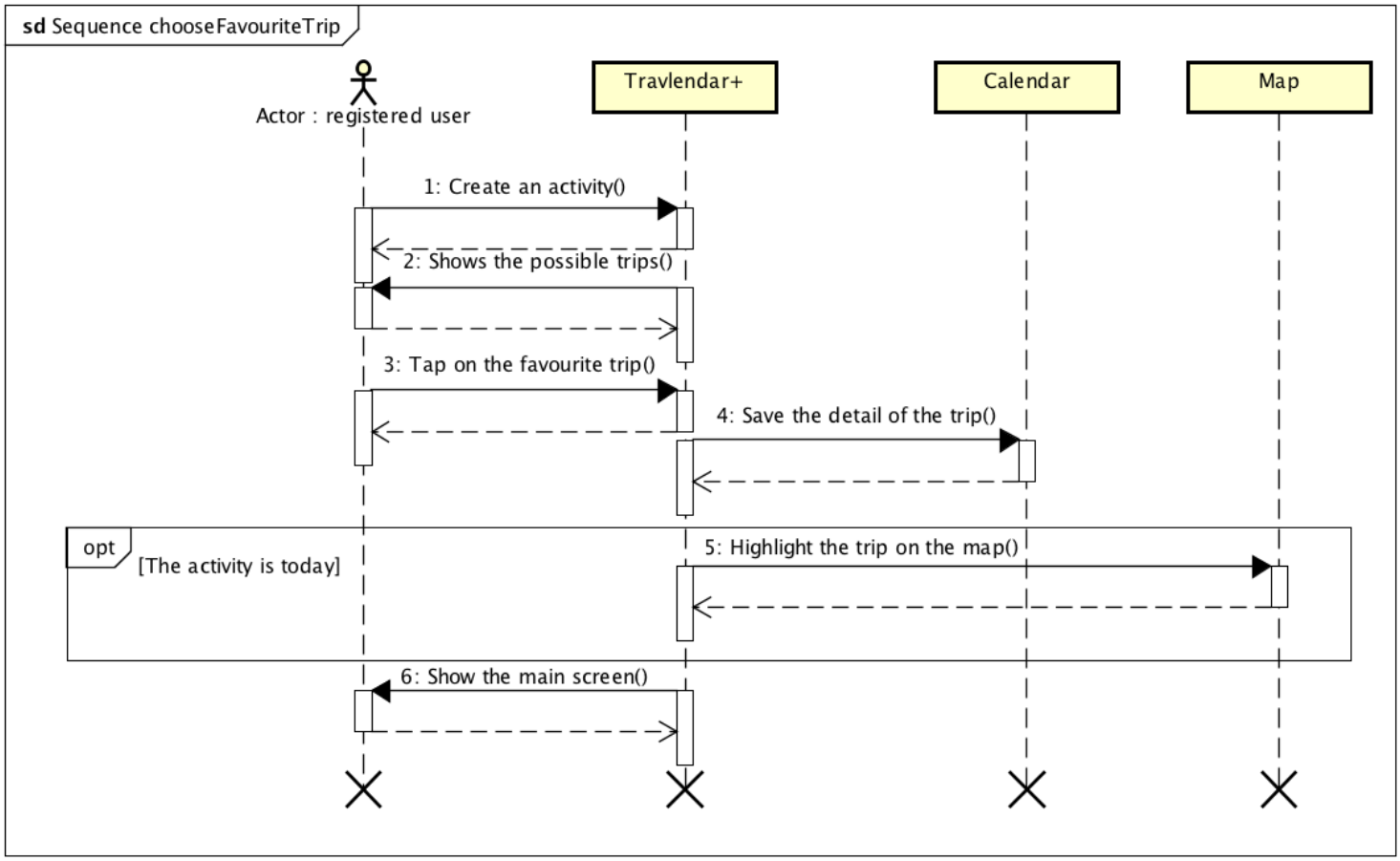


Figure 22: Sequence diagram of choosing the trip

Modify global preferences

Use case ID	$[U_7]$
Actor	Registered user U
Input condition	U is logged in.
Event flow	Once U is on the main screen, to modify the preferences he can: 1. Tap on "menu" symbol. 2. Tap on "App preferences". 3. Tap on one of the preferences to modify it.
Output condition	The application saves the changes. In the trips proposed to U, it will not comprehend trips with the deactivated travel means. If the option of minimizing carbon footprint is activated, Travlendar+ will propose trip with the minimum carbon footprint, and it will not exceed the minimum walking distance in finding the routes.
Exception	The possible exceptions are: 1. U wants to activate/deactivate a travel means that is already active/non-active. 2. U wants to activate/deactivate the option of minimizing the carbon footprint, but this option is already active/non-active. In both cases, the new change will overwrite the last one.

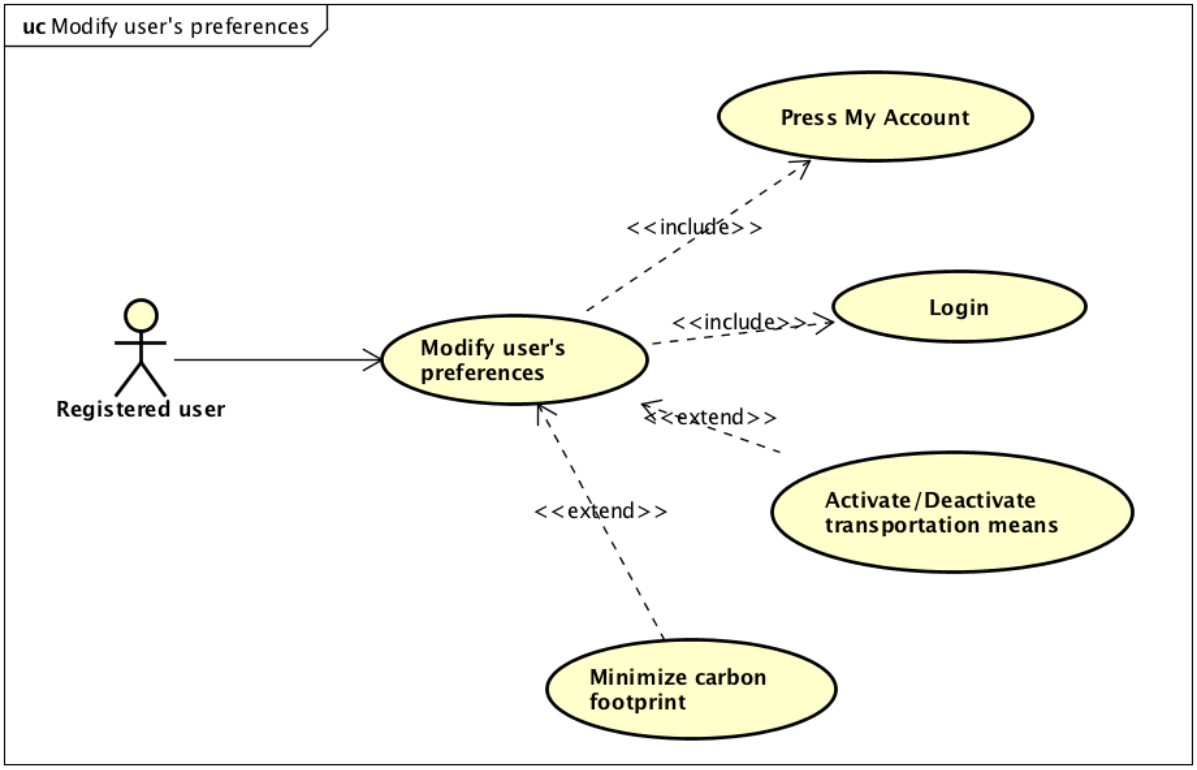


Figure 23: Use case of modifying user's preferences

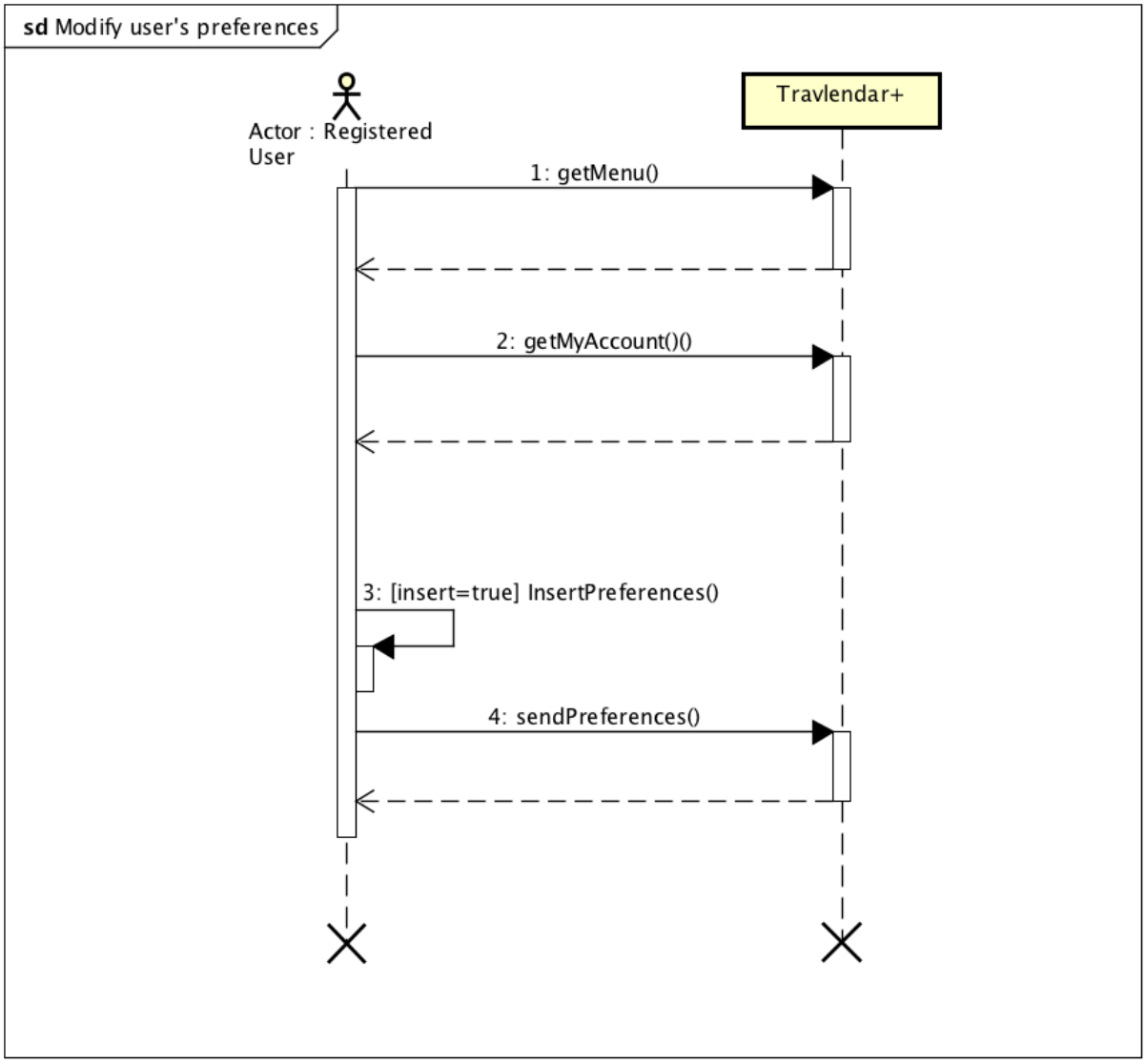


Figure 24: Sequence diagram of modifying user's preferences

3.2.3 Traceability matrix

Goal ID	Req ID	Assumption ID	Use case ID
G1	R1, R2	A2, A7	U3
G2	R3, R4, R5	A1, A3, A4, A5	-
G3	R6, R7, R8	-	U7
G4	R9, R10, R11	A2	U4
G5	R12, R13, R14, R15	A2	U5
G6	R16, R17, R18, R19, R20, R21	-	U7
G7	R22, R23, R24	A1, A4, A5	U6
G8	R25, R26	A4	-
G9	R27, R28	A4	-
G10	R29, R30	A1, A4	-
G11	R31, R32, R33, R34, R35	A5	-

3.3 Performance requirements

In order to guarantee the performance of our application, we have to specify:

- Response Time
- Workload
- Scalability
- Platform

To be reactive and able to answer to a large number of requests, we assume that the response time is close to 0 (from Jakon Nielsen book on Usability 0.1 seconds is about the limit to have the user feel that the system is reacting instantaneously), so it depends mostly on the internet connection of the platform used. We assume that there will be no problem with scalability even if it is a new software and it could suffer an unexpected growth in popularity and from an increase in workload.

3.4 Design constraints

In order to be compatible both with Android and iOs, the application will be developed within the following programming languages:

- Swift 4 (for the iOs version)

- Java 8 (for the Android version)

We have chosen to use the last version of the programming languages to increase robustness and stability of the application, despite the fact that a lot of devices in the market will not be compatible because they're not up to date with the latest OS version.

3.4.1 Hardware limitation

Since it is a mobile application, Travlendar+ requires a smartphone or a tablet with internet connection and with the GPS to find the location of the user. Moreover, users must have enough storage space to install Travlendar+.

3.5 Software system attributes

3.5.1 Reliability

The system must be active 24/7 to guarantee all the services in every occasion.

3.5.2 Availability

Since Travlendar+ is a portable application, its offline functionalities are always guaranteed. As far as online functionalities, their availability depends on that of third part services.

3.5.3 Security

Travlendar+ application is equipped with a login authentication to protect the information of users. Some precautions about the password are necessary to limit vulnerability and to guarantee a complete security of the user's data. In order to avoid brute force attacks, it is necessary to develop a system that requires a strong password, for example containing at least 8 characters comprehensive of numbers and capital letters.

3.5.4 Mainteinability

Travlendar+ will be periodically updated in order to correct eventual malfunction reported by the users and to prevent compatibility problems with new operative systems.

3.5.5 Portability

In order to reach the highest number of devices on the market, the application could be used on every smartphone or tablet provided with iOs or Android.

4 Formal analysis using alloy

This is our Alloy model for our application. It has been created starting from the representation of the Class Diagram. Below the code there are some examples of worlds generated by using the Alloy Analyzer tool to verify some important properties of the model.

4.1 Data type signatures

```
open util/integer

sig Strings{}
sig TravelMeansType{}

abstract sig Boolean{}
one sig True extends Boolean{}
one sig False extends Boolean{}

sig Time{
   hour: one Int,
   minute: one Int
}
{
   hour>=0 and hour<=23
   minute>=0 and minute<=59
}
sig PositiveFloat{
   leftPart: one Int,
   rightPart: one Int
}
{
   leftPart>=0
   rightPart>=0
}

abstract sig Date{
   day: one Int,
   month: one Int,
   year: one Int
}
{
   day>=1 and day<=31
   month>=1 and month<=12
   year>=1
}
sig SimpleDate extends Date{}
sig MeetingDate extends Date{
   time: one Time
}
```

4.2 Signatures

```
abstract sig User{}
sig Visitor extends User{}
sig RegisteredUser extends User{
   name: one Strings,
   surname: one Strings,
   mail: one Strings,
   address: one Strings,
   username: one Strings,
   password: one Strings,
   telephone: one Int,
   dateOfBirth: one SimpleDate,
   createdMeetings: set Meeting,
   createdBreaks: set Break,
   calendar: one Calendar,
   globalPreferences: one GlobalPreference
}

sig Calendar{
   trip: set Trip
}

abstract sig Break{
   name: one Strings,
   date: one SimpleDate,
   startingTime: one Time,
   endingTime: one Time,
   minimumDuration: one Time,
   location: set Location
}
sig LunchBreak extends Break{
   foodType: one Strings,
   maxPrice: one PositiveFloat
}
sig CoffeeBreak extends Break{}
```

```
sig Meeting{
   name: one Strings,
   date: one MeetingDate,
   duration: one Int,
   description: one Strings,
   location: one Location,
}
{
   duration>0
}
sig Location{
   address: one Strings,
   province: one Strings,
   city: one Strings,
   cap: one Int,
   country: one Strings
}
{
   cap>0
}

sig Trip{
   meetings: set Meeting,
   breaks: set Break,
   travelMeans: set TravelMeans,
   cost: one PositiveFloat,
   warning: set Warning,
   preference: set TripPreference,
}

sig Warning{
   type: one Strings,
   description: one Strings
}

abstract sig Preference{
   maxWalkingDistance: one PositiveFloat,
   minimizeFootPrint: one Boolean,
   maxPublicTransportationTime: one Time,
   blockedTravelMeans: set TravelMeans
}
sig TripPreference extends Preference{}
sig GlobalPreference extends Preference{
   lunchStart: one Time,
   lunchEnd: one Time
}

sig TravelMeans{
   type: one TravelMeansType,
   cost: one PositiveFloat,
   website: one Strings
}
```

4.3 Facts

```
//Every calendar belongs to exactly one user
fact OneCalendarPerRegisteredUser{
   all cal: Calendar | (one ru : RegisteredUser | ru.calendar = cal)
}

//Every travel means type belongs to exactly one travel means
fact OneTravelMeansTypePerTravelMeans{
   all tmt: TravelMeansType | (one tm : TravelMeans | tm.type = tmt)
}

//Every meeting is created by exactly one user
fact OneMeetingCreator{
   all mee: Meeting | (one ru : RegisteredUser | mee in ru.createdMeetings)
}

//Every break is created by exactly one user
fact OneBreakCreator{
   all br: Break | (one ru : RegisteredUser | br in ru.createdBreaks)
}

//Every warning belongs to exactly one user's trip
fact OneWarningPerTrip{
   all war: Warning | (one tr : Trip | tr.warning = war)
}

//Every day has exactly one trip associated
fact DayAndTripCorrispondence{
   all disj mee1, mee2: Meeting|
      (one tr: Trip |
         (mee1 in tr.meetings) and (mee2 in tr.meetings)
         and (mee1.date.day=mee2.date.day)
         and (mee1.date.month=mee2.date.month)
         and (mee1.date.year=mee2.date.year)
      )
}

//Every activity is associated to exactly one trip
fact ActivitiesAndTripCorrispondence{
   all disj tr1, tr2: Trip |
      tr1.meetings & tr2.meetings = none and tr1.breaks & tr2.breaks = none
}

//Every date belongs to exactly one activity (break or meeting)
fact OneDatePerActivity{
   all da: Date |
      (
         (one br : Break | br.date = da) or
         (one me: Meeting | me.date = da) or
         (one re: RegisteredUser | re.dateOfBirth=da)
      )
}
```

```
//All the meetings created by a registered user are present in his/her calendar
fact CreatedMeetingsInRegisteredUserCalendar{
      all ru: RegisteredUser |
         ru.createdMeetings in ru.calendar.trip.meetings
}

//All the global preferences are created by a registered user
fact GlobalPreferencesCreatedByUser{
   all gp: GlobalPreference | (one ru : RegisteredUser | ru.globalPreferences = gp)
}

//All the breaks created by a registered user are present in his/her calendar
fact CreatedMeetingsInRegisteredUserCalendar{
      all ru: RegisteredUser |
         ru.createdBreaks in ru.calendar.trip.breaks
}

//Prevents the use of the same username or email for more than one user
fact NoDuplicatedUser{
   no disj ru1, ru2: RegisteredUser |
      (ru1.username = ru2.username) or (ru1.mail = ru2.mail)
}

//Every travel means type is unique (e.g. only one 'Taxi' type, only one 'Private car' type etc...)
fact NoDuplicatedMeansType{
   no disj tm1, tm2: TravelMeans |
      (tm1.type =  tm2.type)
}


//Travel means blocked in the global preferences cannot be used
//for the trips of the registered user
fact NoGloballyBlockedTravelMeansInTheTrips{
   all ru: RegisteredUser|
      ru.calendar.trip.travelMeans.type & ru.globalPreferences.blockedTravelMeans.type = none
}

//Travel means blocked in the trip preferences cannot be used
//for the current trip of the registered user
fact NoBlockedTravelMeansInTheTrips{
   all tr: Trip |
      tr.travelMeans.type not in tr.preference.blockedTravelMeans.type
}
```

```
//If a user inserts two (or more) meeting with an overlapping time, there must be
//(at least) a warning in the trip associated to the meetings
fact GenerateWarningWithOverlappingMeetingTime{
   all tr: Trip, mee1, mee2: Meeting |
      ((mee1 in tr.meetings) and
      (mee2 in tr.meetings) and
         ((mee1.date.time.hour.add[mee1.duration] > mee2.date.time.hour) or
            (
               (mee1.date.time.hour.add[mee1.duration] = mee2.date.time.hour) and
               (mee1.date.time.minute > mee2.date.time.minute)
            )
         ))
      implies
      (#tr.warning>=1)
}

//If a user inserts two (or more) breaks with an overlapping time, then there must be
//(at least) a warning in the trip associated to the meetings
fact GenerateWarningWithOverlappingBreakTime{
   all tr: Trip, br1, br2: Break |
      ((br1 in tr.breaks) and
      (br2 in tr.breaks) and
         (br1.endingTime.hour > br2.startingTime.hour or
            (br1.endingTime.hour = br2.startingTime.hour and
            (br1.endingTime.minute > br2.startingTime.minute)
         ))
      )
      implies
      (#tr.warning>=1)
}
```

4.4 Assertions

```
//Check that there all the calendars are created by a registered user
assert NoCalendarWithoutUser{
   no cal: Calendar |
      (no ru : RegisteredUser |
         ru.calendar = cal)
}

check NoCalendarWithoutUser

//Check that all the meetings created by a registered user are present in his/her calendar
assert AllMeetingsAreInTheMeetingCreatorCalendar{
   no ru: RegisteredUser |
      ru.createdMeetings not in ru.calendar.trip.meetings
}

check AllMeetingsAreInTheMeetingCreatorCalendar

//Check that all the breaks created by a registered user are present in his/her calendar
assert AllBreaksAreInTheMeetingCreatorCalendar{
   no ru: RegisteredUser |
      ru.createdBreaks not in ru.calendar.trip.breaks
}

check AllBreaksAreInTheMeetingCreatorCalendar

//Check that all the travel means selected as "blocked" by the registered user
//in the global preferences don't appear in the trip's travel means
assert AllTheMeansUsedForATripAreNotGloballyBlocked{
   no ru: RegisteredUser|
      ru.calendar.trip.travelMeans.type in ru.globalPreferences.blockedTravelMeans.type
}

check AllTheMeansUsedForATripAreNotGloballyBlocked
//Check that all the travel means selected as "blocked" by the registered user
//for a specific trip don't appear in the trip's travel means
assert AllTheMeansUsedForATripAreNotBlocked{
   no tr: Trip |
      tr.travelMeans.type in tr.preference.blockedTravelMeans.type
}

check AllTheMeansUsedForATripAreNotBlocked
```

4.5 Predicates

```
//This predicate show that if a registered user specify in his/her preference that a
//certain travel means type is blocked, it will not be used in the meeting's travel means
pred showBlockedTravelMeansInMeetings(){
   #RegisteredUser=1
   #Trip=1
   #Meeting=1
   #Break=0
   #TravelMeansType=2
   #(GlobalPreference.blockedTravelMeans)=1
   }
run showBlockedTravelMeansInMeetings for 4 but 7 int

//This predicate show that if two meetings have an overlapping time, then (at least) one
//warning associated to the trip that contains the two meetings is generated
pred showWarningWithOverlappingTime(){
   #RegisteredUser=1
   #Trip=1
   #Meeting=2
   #Break=0
   #TravelMeansType=2
   #Time=1 //The two meetings starts at the same time
   }
run showWarningWithOverlappingTime for 4 but 7 int

//This predicate show a simplified instance of the world
pred showWorld(){
   #RegisteredUser=1
   #Trip=1
   #Meeting=1
   #Break=1
   #TravelMeansType=2
   }
run showWorld for 4 but 7 int
```

4.6 Execution results

```
8 commands were executed. The results are:
  #1: No counterexample found. NoCalendarWithoutUser may be valid.
  #2: No counterexample found. AllMeetingsAreInTheMeetingCreatorCalendar may be valid.
  #3: No counterexample found. AllBreaksAreInTheMeetingCreatorCalendar may be valid.
  #4: No counterexample found. AllTheMeansUsedForATripAreNotGloballyBlocked may be valid.
  #5: No counterexample found. AllTheMeansUsedForATripAreNotBlocked may be valid.
  #6: Instance found. showBlockedTravelMeansInMeetings is consistent.
  #7: Instance found. showWarningWithOverlappingTime is consistent.
  #8: Instance found. showWorld is consistent.
```

4.7 Generated worlds

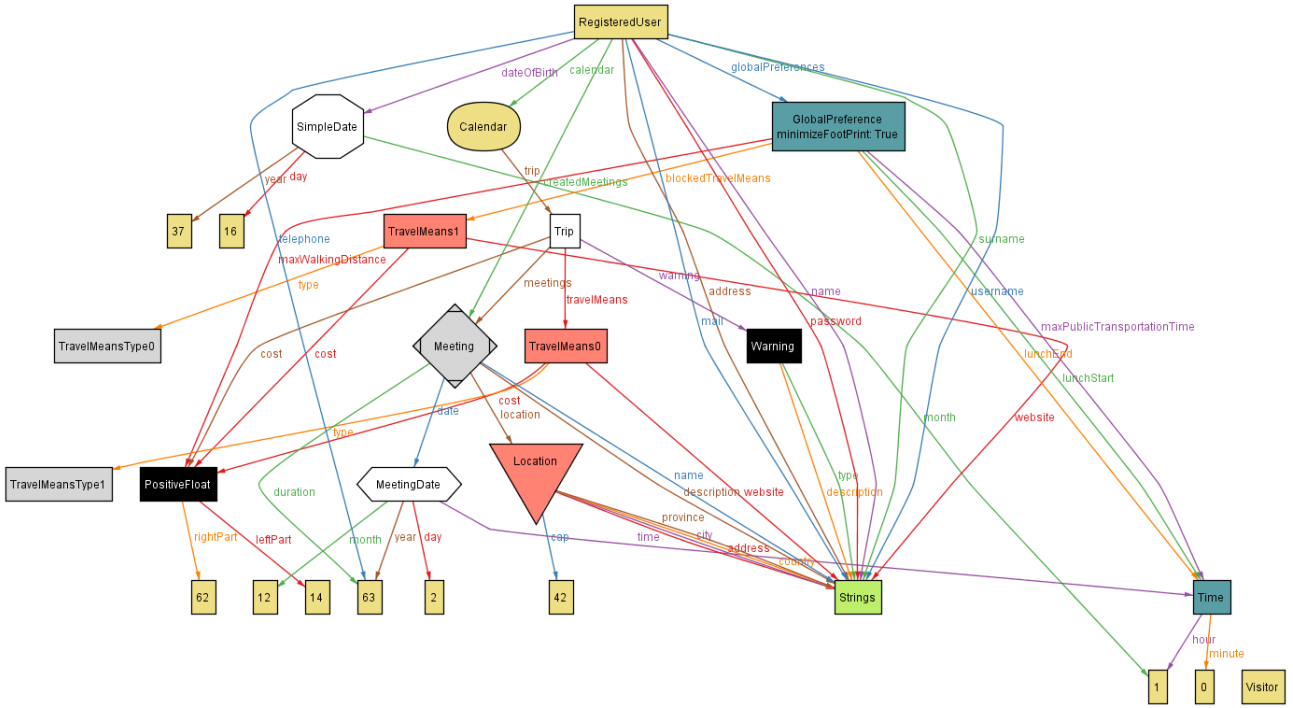


Figure 25: This instance shows the "showBlockedTravelMeansInMeetings" predicate

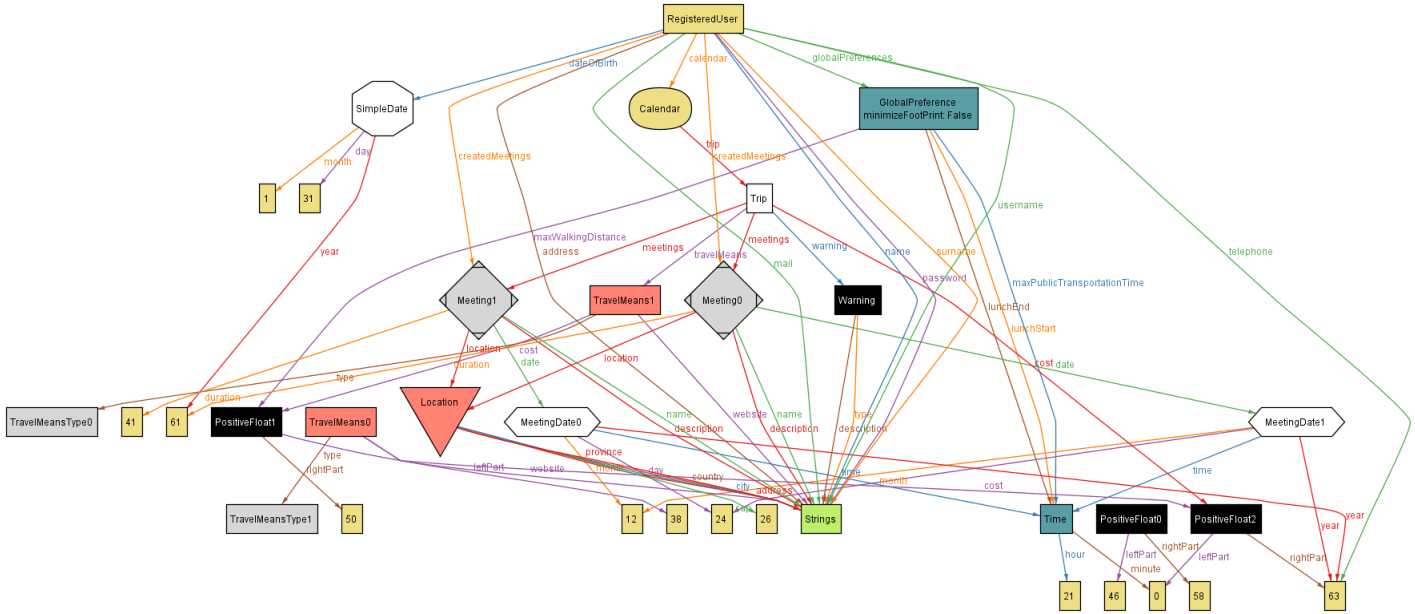


Figure 26: This instance shows the "showWarningWithOverlappingTime" predicate

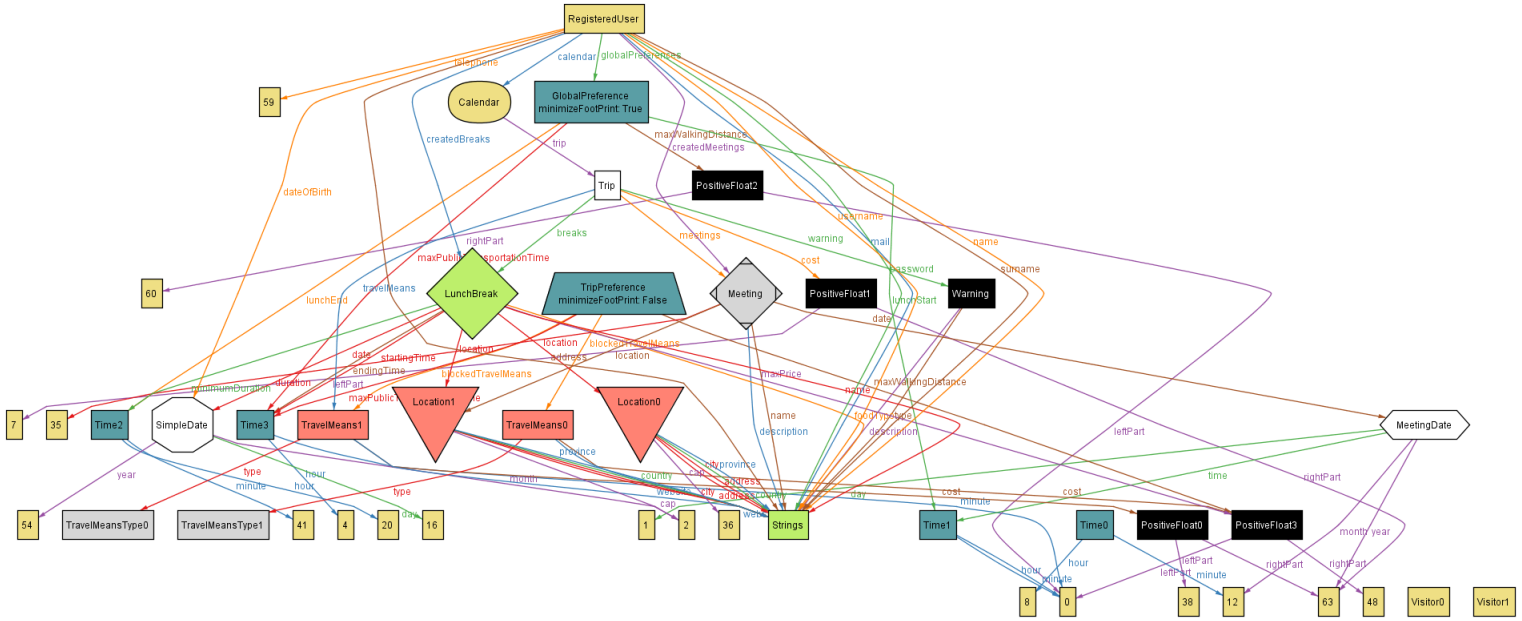


Figure 27: This instance shows the "showWorld" predicate

5 Effort spent

We managed to distribute the workload fairly between days and team members in a way that allows us to finish a few days before the deadline and have time for an accurate check in the last days. We worked together most of the times and trying to do something every day, as it can be shown by Git commits. The total amount of time required to build this document is about 34 hours of work per person.

6 References

- Professor Rossi's slides;

[1] `http://www.1202performance.com/` (To understand the part of performance requirements);