Politecnico di Milano
AA 2017-2018

Software Engineering Project

TRAVLENDAR+

RASD

Requirements Analysis and Specification Document

Version 1.01

Marco Prosdocimi - 898395
Enrico Ruggiano - 900040
Giacomo Vercesi - 899928

Contents

depth 2

Introduction

General Description

Travlendar+ is an appointment management system capable of efficently scheduling a user's calendar. Thanks to this application it is *possible to calculate in advance the estimated time to reach a meeting point based on the transportation choices made my the user.*

The application not only is used for interregional travels but also for personal or work appointments. Once the user is registered he or she will be able to set their preferences regarding transportation choices to reach their destination. The application will suggest different itineraries to the user based on different transportation methods, based on metrics such as transportation time and cost.

Travelendar+ is a mobile and web application capable of improving its user's everyday life with simplicity, efficency and safety.

Purpose

In these document will be analyzed all the requirements and specifications needed for the design and developing of Travlendar+ System. The document is divided in sections to be more readable and linkable in external paper.

Goals

The main functions offered by Travlendar+ are substantially three:

- **Calendar** App manager functions
- **Maps Geolocalization** App functions
- Customizable **Personal User** page and preferences functions.

So we can group the goals in three categories, one for a specific function.

Calendar Functions

1) Show a personal calendar of 'Events' submitted.
2) Let the User submit 'Events' on the calendar to plan his/her future appointments.
3) Notify the User when the 'Event' is about to start and she/he needs to leave to be in time to the 'Event Meeting Location'.
4) Let the User to create 'Flexible Event' for everyday short and low priority appointments.

Maps Geolocalization Functions

1) Geolocate the gps coordinates of the "Starting Location" and the "Meeting Location" on a virtual map.

2) Calculate a list of possible shortest routes from a “Starting Location” and the “Meeting Location” with the ‘Vehicles’ available.
3) Calculate the extimation time of arrival at the destination for each route.
4) Let the User choose a route from the list of routes provided by the algorithm
5) Let the User modify the routes adding ‘Costraint’ for ‘Intermidiate Locations’, preferred ‘Vehicle’, max distance with a specified ‘Vehicle’ or max time on a specified ‘Vehicle’.
6) Consider on the possible ‘Vehicle’ avaible all the public transports of the city, railway stations, aeroports, train stations, car and bike sharing systems, bike, car and by foot.
7) notify with a Warning message if the route chosen by the User is not good and he/she may arrive on late at the ‘Meeting’ because of its Extimation Time too long.
8) Suggest a “Best Route” to the User with a ‘Vehicle’ which is appropriate for the day time of the appointment, the geographical location, the type of the meeting and the weather.

User Profile Manager Functions

1) Let the User to sign in to the Service filling an online form.
2) Let the User to login to their personal User page and update their informations.
3) Let the User fill their ‘Vehicle’ preferences or dislike for best result on Algorithm ‘Best Route’ calculus.
4) Let the User buy online tickets for the majority of the public transports involved on the ‘Best Route’ chosen.
5) (**optional**) Let the User modify settings for the Algorithm ‘Best Route’such activating ’Green Mode’ or enabling options like “No traffic lighters”, “No Schools at 16:00”, “Show Autovelox”.
6) (**optional**) Let the User submit on his/her page the availability of public transports subcription, driver licence, coupons for special transports for best result on Algorithm route calculus.

Scope

Travelandar+ has a simple scope, helping the Users to planning efficiently their appoitnments. We can analyze all the shareded phenomena by the App and the Users.

Shared Phenomena:

1) Registration on the Travlendar+ Service
2) Submitting the ‘Event’

3) Submitting the preferences of the User
4) Alarm function
5) Calculus of the ‘Best Route’

Not Shared Phenomena:

1) Geographical positions of the ’Event’s Locations.
2) Presence of roads, streets, railway stations, public transports in the city.
3) Preferences of the User.

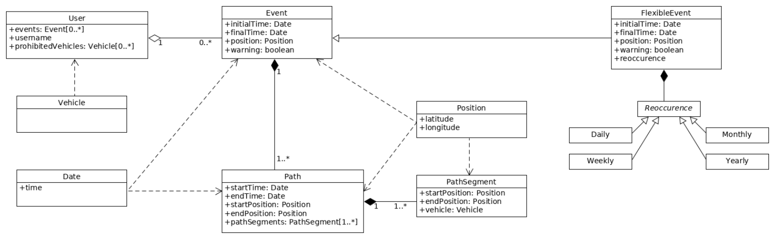
Definitions

- *Alarm function*: a way in which the phone can notify the user that something important is happening. It is often a sound or a vibration according to the phone used.
- *API:* Application programming interface; it is a common way to communicate with another system.
- *Best Route*: The best route calculated by the algorithm to reach a given event without any delay. The algorithm also considers the user’s preferences.
- *Costraint*: Something that controls what you do by keeping you within particular limits.
- *Event*: The users can create some events and submit them to the calendar. The application checks if there are some overlaps and calculates the best route to reach the events.
- *Green Mode*: A user’s preference which means that the user prefers to keep his Ecological Footprint as lower as possible. (for instance the user prefer to use a bike instead of a bus)
- *Itermidiate Location*: Any locations between the start point and where the event take place.
- *Meeting Location*: The place or position that the user specifies during the submission of a new event.
- *Route*: The roads you follow to get from one place to another place. The routes are calculated after an event submission.
- *Starting Location*: The position where the algorithm starts to calculate the routes to reach the event.
- *Vehicle*: Something such as a car, bike or bus that takes people from one place to another, also “by foot” is considered a vehicle.
- *Warning*: If there are some problems with an event (like overlapping) the application produces a notify that is attached to the event. The user can select the warning and see more details.

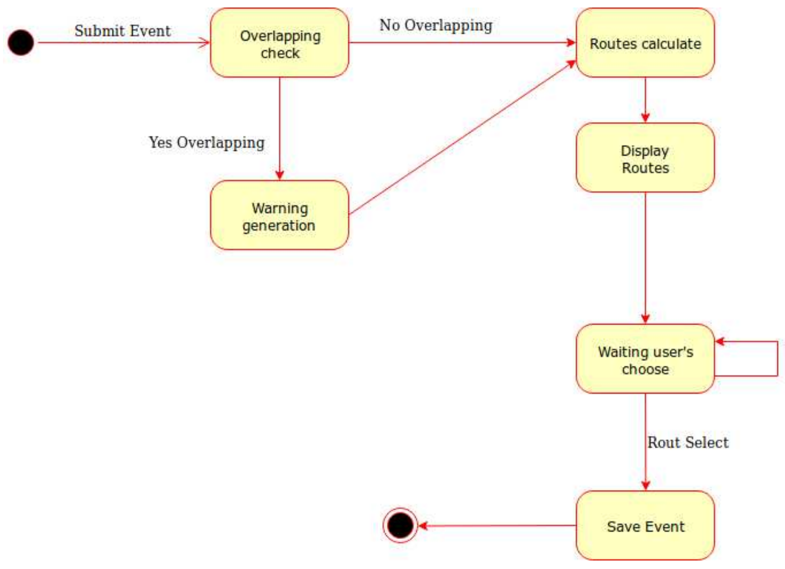
Overall Description

Product Perspective

Class Diagram



State Chart



Product Functions

Focusing on each category of the Goals of Travelander+ the principal requirements can be summarized in this sections

Calendar Functions Requirements

1) *Show a personal calendar of 'Events' submitted.*
    1) The system must store the submit of 'Event' of the User
    2) The system must have a visual calendar showing the 'Event' of the User.
    3) The system must have a page navigation system.
2) *Let the User submit 'Events' on the calendar to plan his/her future appointments.*
    1) The system must store an "Event" when the User specifies a "Starting Location", a "Meeting Location" a date and the time.
    2) The system must not let the User to create 'Event' in the past days.
    3) The system must provide a message of 'Warning' if the User is creating an 'Event' which can be overlapped to an existing 'Event'.
3) *Notify the User when the 'Event' is about to start and she/he needs to leave to be in time to the 'Event Meeting Location'.*
    1) The system must have an 'Alarm System' which warns the User that can miss or be late to an appointment if he/she doesn't leave.
    2) The 'Alarm System' can be configurate by the User and can have different ways of notify.
    3) The 'Alarm System' if activated must start before the start time of the 'Event'
4) *Let the User to create 'Flexible Event' for everyday short and low priority appointments.*
    1) The system must provide a 'Flexible Event' flag when the User is creating a new 'Event'
    2) A 'Flexible Event' can be overlapped to an another 'Event'
    3) A 'Flexible Event' can be copied and pasted on the Calendar and be repeated on several days of the same Week.
    4) A 'Flexible Event' can be easly suppressed.

Maps Geolocalization Requirements

1 *Geolocate the gps coordinates of the "Starting Location" and the "Meeting Location" on a virtual map.*

1) The system must provide gps API and be able to locate the position on a graphical map.

2 *Calculate a list of possible shortest routes from a "Starting Location" and the "Meeting Location" with the 'Vehicles' available.*

1) The 'Best Route' Algorithm must return a list of shortest routes.

3 *Calculate the Extimation time of arrival at the destination for each route.*

1) The system with the support of external API can calculate an Extimation time of arrival for a specified 'Best Route'.

4 *Let the User choose a route from the list of routes provided by the algorithm*

1) The system must provide a grphical list in which are presented all the possible 'Best Routes' and details of the itinerary.
2) The system must wait a choice of the User to save the route for the specified 'Event'.

5 *Let the User modify the "Best Route" adding 'Costraint' for 'Intermidiate Locations', preferred 'Vehicle', max distance with a specified 'Vehicle' or max time on a specified 'Vehicle'.*

1) The system must provide a graphical feature in which the user can modify the path adding location on the virtual maps.
2) The 'Best Route' Algorithm must update the Extimate time of arrival at destination depending on the geographical position of the 'Intermidiate Locations' added or the new 'Vehicle' speed average chosen.
3) In case of 'Costraint' too much strict the system can return a 'Warning' message notifing the User that a 'Best Route' does not exist with that 'Costraint'.

6 *Consider on the possible 'Vehicle' avaible all the public transports of the city, railway stations, aeroports, train stations, car and bike sharing systems, bike, car and by foot.*

1) The system must query information on timetables of the public transports of the city.
2) The system must notify on the virtual map stations of the public transports of the city.

7 *Notify with a Warning message if the route chosen by the User is not good and he/she may arrive on late at the 'Meeting' because of*

its Extimation Time too long.

1) Before subitting the ‘Event’, the system must check if the time of the ‘Event’ and the ‘Extimation’ time of Arrival of the corrisponding ‘Best Route’ overlap with other ‘Event’ time start.

8 *Suggest a “Best Route” to the User with a ‘Vehicle’ which is appropriate for the day time of the appointment, the geographical location, the type of the meeting and the weather*

1) The system must provide a “Suggested Route”

User Profile Manager Requirements

1) *Let the User to sign in to the Service filling an online form.*
    1) The system must provide a registration form to the User.
    2) The User is not signed in until all the fields of the form are not filled and valid.
    3) The system must verify if the information on the registration form are valid.
2) *Let the User to login to their personal User page and update their informations.*
    1) The system must provide an update function on the User profile page.
    2) The system must verify if the new informations are valid.
3) *Let the User fill their ‘Vehicle’ preferences or dislike for best result on Algorithm ‘Best Route’ calculus.*
    1) The system must store the preference or dislike of the User
4) *Let the User buy online tickets for the majority of the public transports involved on the route chosen.*
    1) The system must provide an “Arrange System”.
    2) The “Arrange System” can query external systems and reserve vehicles for user.
    3) The “Arrange System” can redirect the user to secure pages in which can be buyed tickets for tranports inolved on the route chosen.
5) (**optional**) *Let the User modify settings for the Algorithm ‘Best Route’such activating ’Green Mode’ or enabling options like “No traffic lighters”, “No Schools at 16:00”, “Show Autovelox”.*
    1) The system must store all the setting of the Algorithm chosed by the User
6) (**optional**) *Let the User submit on his/her page the availability of public transports subcription, driver licence, coupons for special transports for best result on Algorithm ‘Best Route’ calculus.*

1) The system must use if available those information when calculating the 'Best Route'

User Characteristics

- Travelendar+ was made to aid organizations to effectively plan appointments throughout the year for its registered users. Its simplicity makes it versatile and easily accessible for all users. *

There are 3 user categories that travelendar is aimed at:

- *Business men*
- *Travelers*
- *City Lovers*

Businessmen

Businessmen are all individuals that use the app for business appointments and meetings. Their behaviour will be characterized by:

1) submit rate of meetings per week and month *very high.*
2) meetings location *very distant* and often *different.*
3) *high* interest on arriving on time at meetings
4) *high* interest on buyng via internet tickets for the vehicle
5) *medium* interest on additional feature, such as interconnecting other technologies for a better organization (email notifiation, smart alarm, phone and/or smart clock notifications. . .)
6) *minimal* interest on user interface and graphical feature
7) *vehicle preferences* are public transport for city meetings or *train* and *airplane* for outside city meetings.
8) Long term users (will have a prolonged use the app consistently)

Travelers

Travelers are those who use the app for planing their trip or work conference. Their main goals are to reach airports, hotels, train stations or museums. Their behaviour will be:

1) submit rate of meetings per week and day *very high*
2) Two important 'Event' which are the 'Departure' and the 'Arrival' Event. For these events the interest on arriving on time is *crucial.*
3) Several intermediate 'Event' on the week between the 'Arrival' and the 'Departure'. For these event the interest on arriving

on time is *medium* since most of the locations are museums, restaurants, hotels.
4) The ‘Routes’ have often intermediate ‘locations’ and the ‘Vehicle’ used is often ‘By Foot’.
5) Sometimes ‘Event’ planned for a day can be modified and switched with other ‘Event’ scheduled for the next days.
6) *medium* interest on graphical feature and user interfaces. The ‘Events’ could have useful verbose information attached to them.
7) Short term users. Once the trip is over, they will probably uninstall the Application.

City Lovers

City Lovers are people that will use the application to schedule free time activities. They are tech savvy enthusiast who needs to annotate all their appointment during the week and are often curious of the limit of the Application. For example their ‘Events’ are linked with their social activities like going to the movies with friends or going shopping with their girlfriends, or sport activities, like going to the gym or jogging on Sunday morning. So their ‘Behaviour’ is characterized by:

1) submit rate of meetings per week and day *very high*
2) *low* interest on arriving on time on their appointments.
3) *high* interest on user interface and additional feature, like vocal message warning.
4) *high* rate of modified ‘Event’
5) *high* interest on the ‘Personal Profile’ page of the App or feature like adding secondary information, uploading profile images, recording all the kilometers of his/her ‘Routes’ and all the location visited.
6) *high* interest on ‘Green Mode’
7) *Vehicle* preferences are often bike, public transports and car/bike sharing.
8) They are *often* young age users, university students and sportmen.
9) *High* interest on Technical performance of the App, like memory storage consuption, cpu memory usage, heat burst.
10) *Short* term and *occasional* user

Assumptions

Algorithm ‘Best Route’ Calculation Assumptions:

1) The Algorithm will take into account statistics from the user to determine its walking pace and better optimize the algoritm.

2) The Algorithm doesn't take into account for a 'Vehicles' various ground impacts that could slow down the walk, such as stairs, rough terrain, long street climbs.
3) The Algorithm doesn't take into account for a 'By Foot' vehicle preference if the sidewalk is crowded in that day and time which could slow down the walk of the user.
4) For a 'By foot' or a 'Bike' the Algorithm avoids to track the route across a park or a green area on the map if it is not specified by the user.

Query external DBs Assumptions:

1. **The Application can access informations on:** -Local public transportations timetables such tram, bus, Coach. -Positions and availability of Car and Bike sharing *private* and public service stations -Positions of public transportations stops and stations like railway stations, train stations, bus stops.
2. The Application can redirect the user during the navigation on secure Payments service page allowing the user to buy tickets online for public transports.

Specific Requirements

External Interface Requirements

In these section it will presented in the details all the specific interface of Travelandar+.

User Interface

UI and Graphical features are suited for all kind of users. It is essential a simple and immediate design which is characteristic of nowday applications.

The User Interface of the Broswer Application and of Mobile Application must be as similar as possible like the most popular application web based.

> *Except for the first image we will assume that all the following mock up presented are for the Broswer Application and Mobile Application*

Features

The Gui elements are:

1) Slide left menu

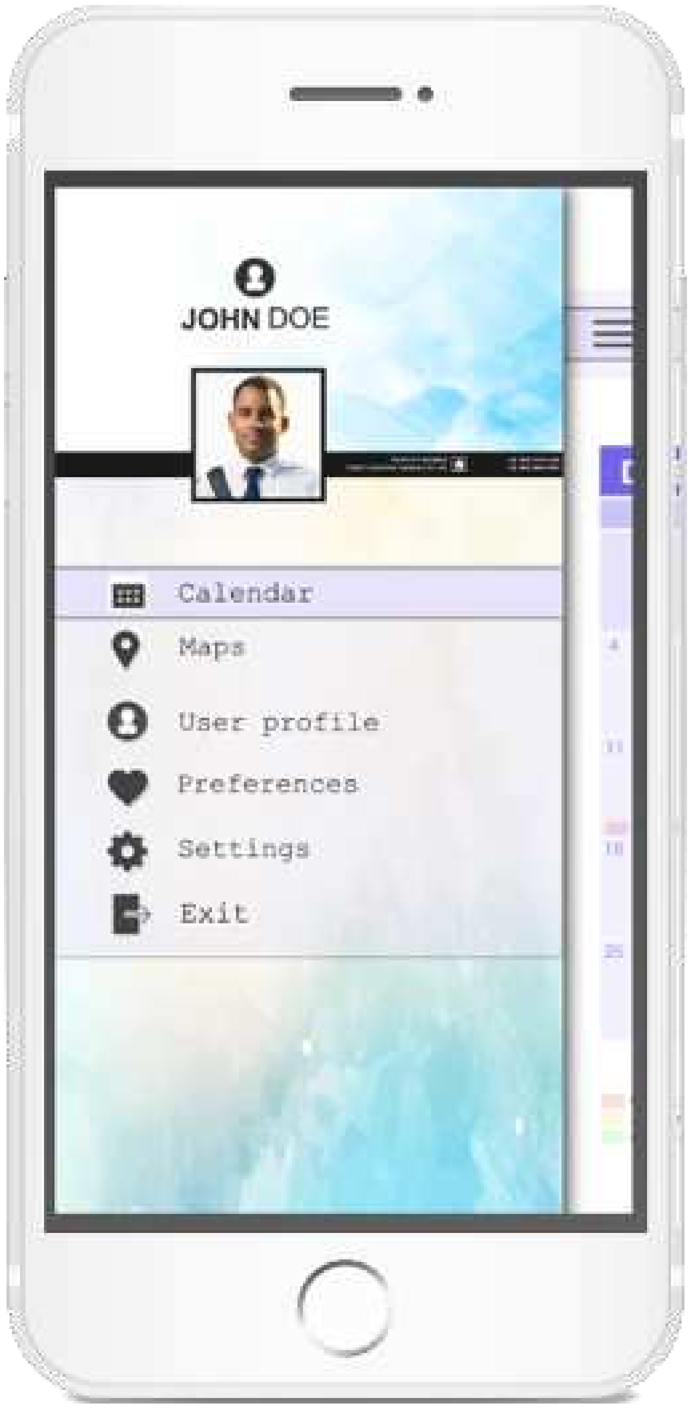


Figure 1: The App Gui is intuitive and easy to use.

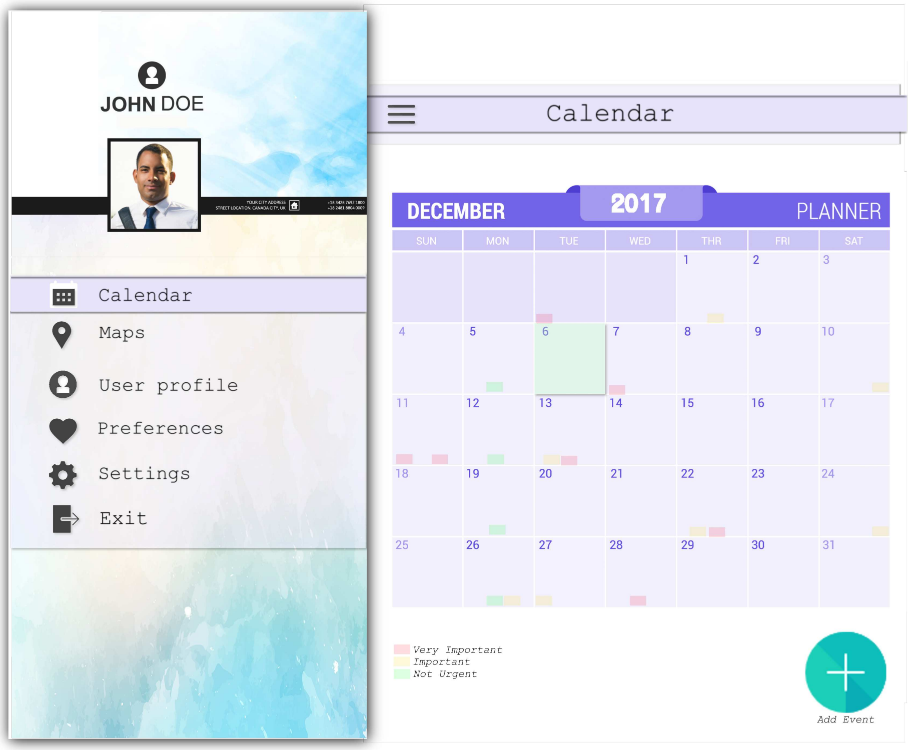


Figure 2: The Broswer Gui is the same and extended version of th App Gui.

2) First loading screen which shows the Travlendar Logo;
3) Push buttons.
4) Input text field.

The main screen are:

1) Calendar screen
2) Submit Event Screen
3) User Page
4) Preferences Page

The Main Screens

Figure 3: The Loading page shows the Travlendar+ logo and has a nice interface.

The Calendar Page shows a calendar with all the event submitted

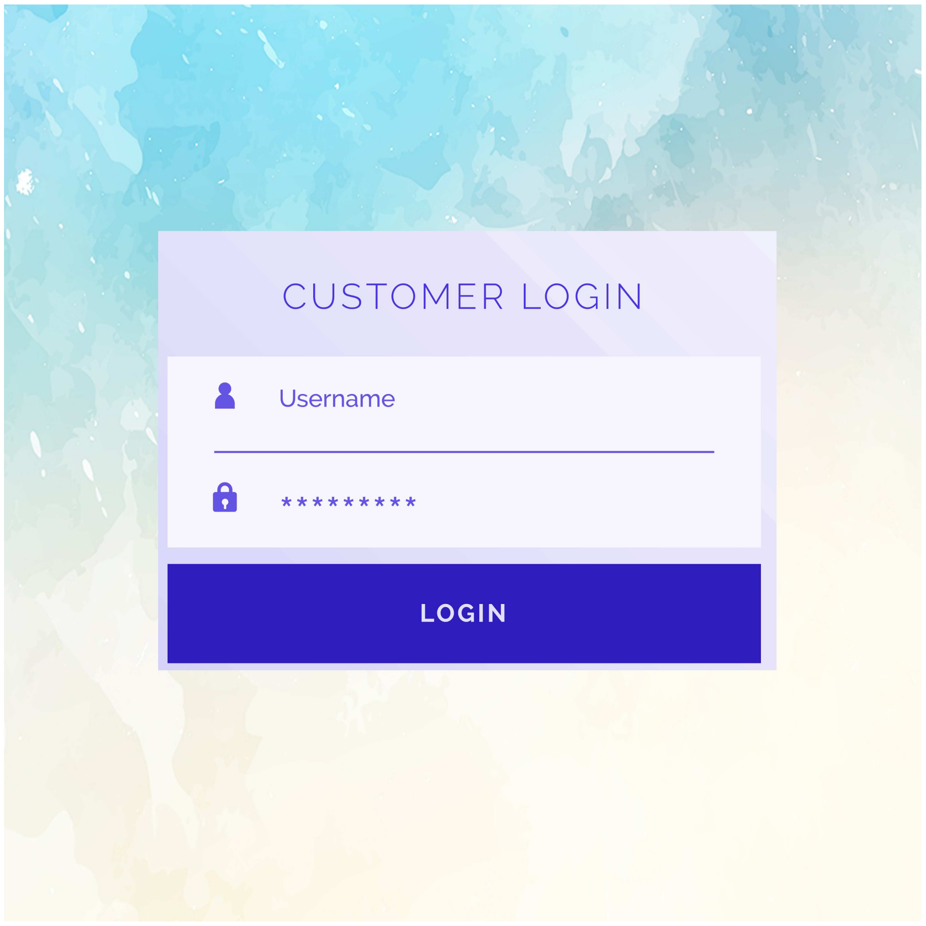


Figure 4: The Login Page in which the user must enter his user name and a password.

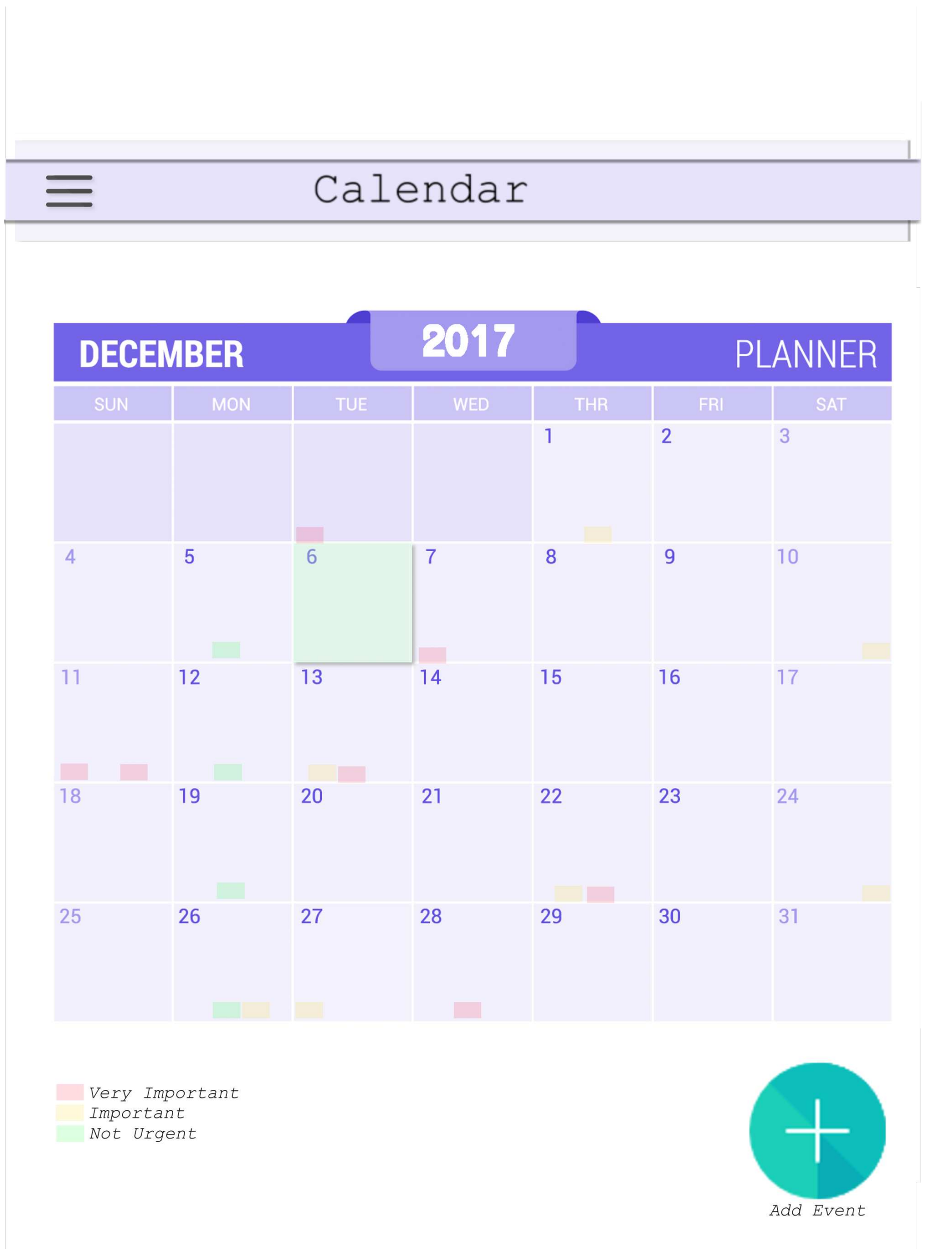

Calendar
DECEMBER
2017
PLANNER
SUN
MON
TUE
WED
THR
FRI
SAT
Very Important
Important
Not Urgent
Add Event

Figure 5:

by the user. There is a submit button in which the user can submit a new event.

The User page in which the user can modify his/her personal informations

The Preference page in which the user can modify and update his/her preferences on the transport to use.

The Submit Event Screen

The Submit event screen has a first page in which the user submits the destination, the date, the starting location and the time of the event.

In the second page there is a list of the routes calculated by the algorithm with the specific extimational time of arrive.

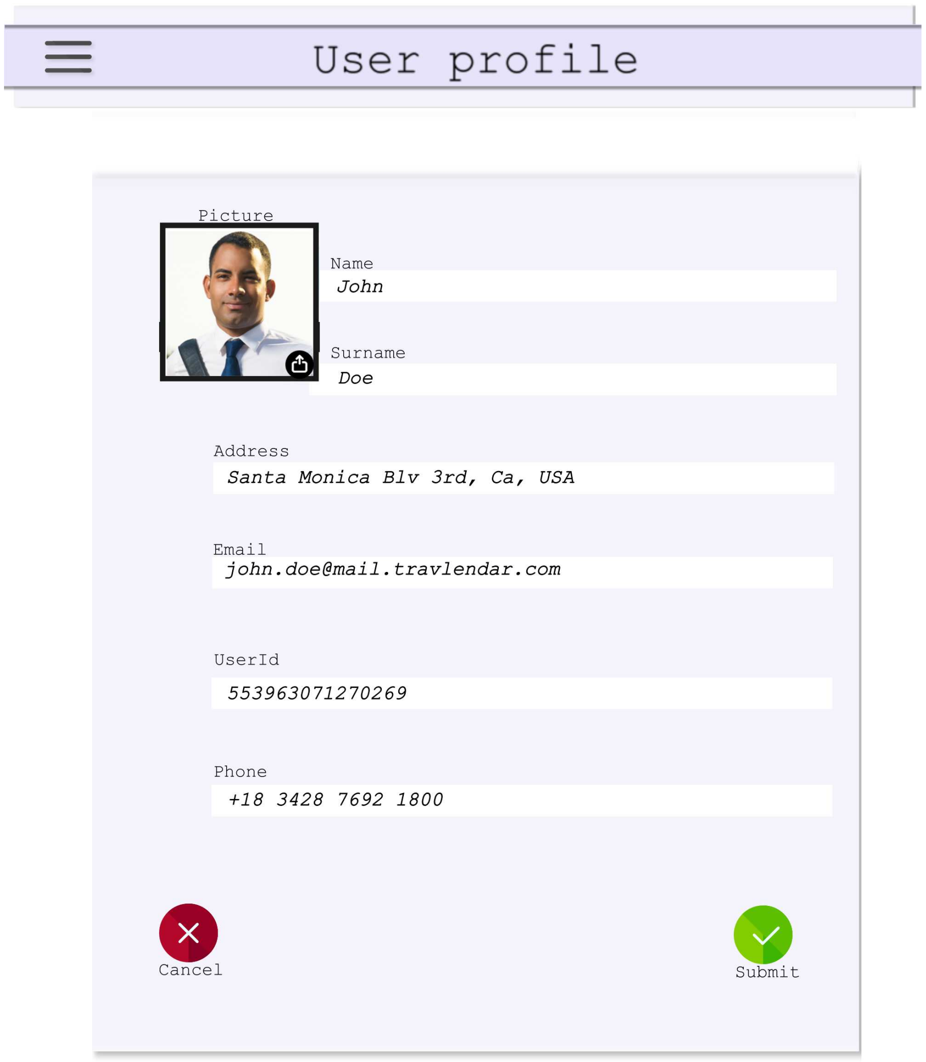
The warning shows if with the extimational time of arrive the user can be late for the meeting. The first and the second page are linked by a Scroll Down animation.

The First page in which the user submits the main informations.

The Second page in which the user can modify the routes and has a graphic map feature.

Software Interface

The software will not present interoperable software interfaces in its first release.



Name
John

Surname
Doe

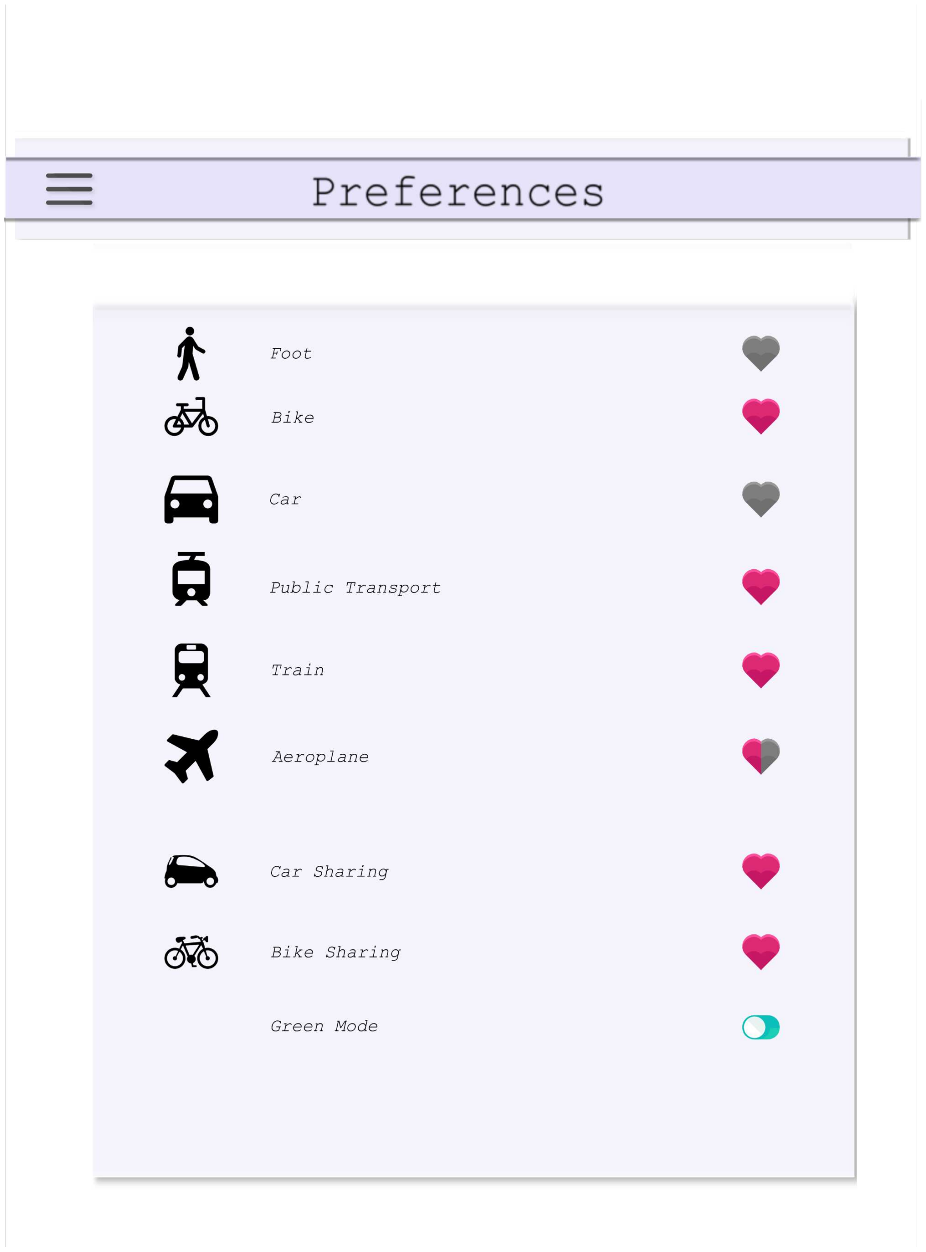
Address
Santa Monica Blv 3rd, Ca, USA

Email
john.doe@mail.travlendar.com

UserId
553963071270269

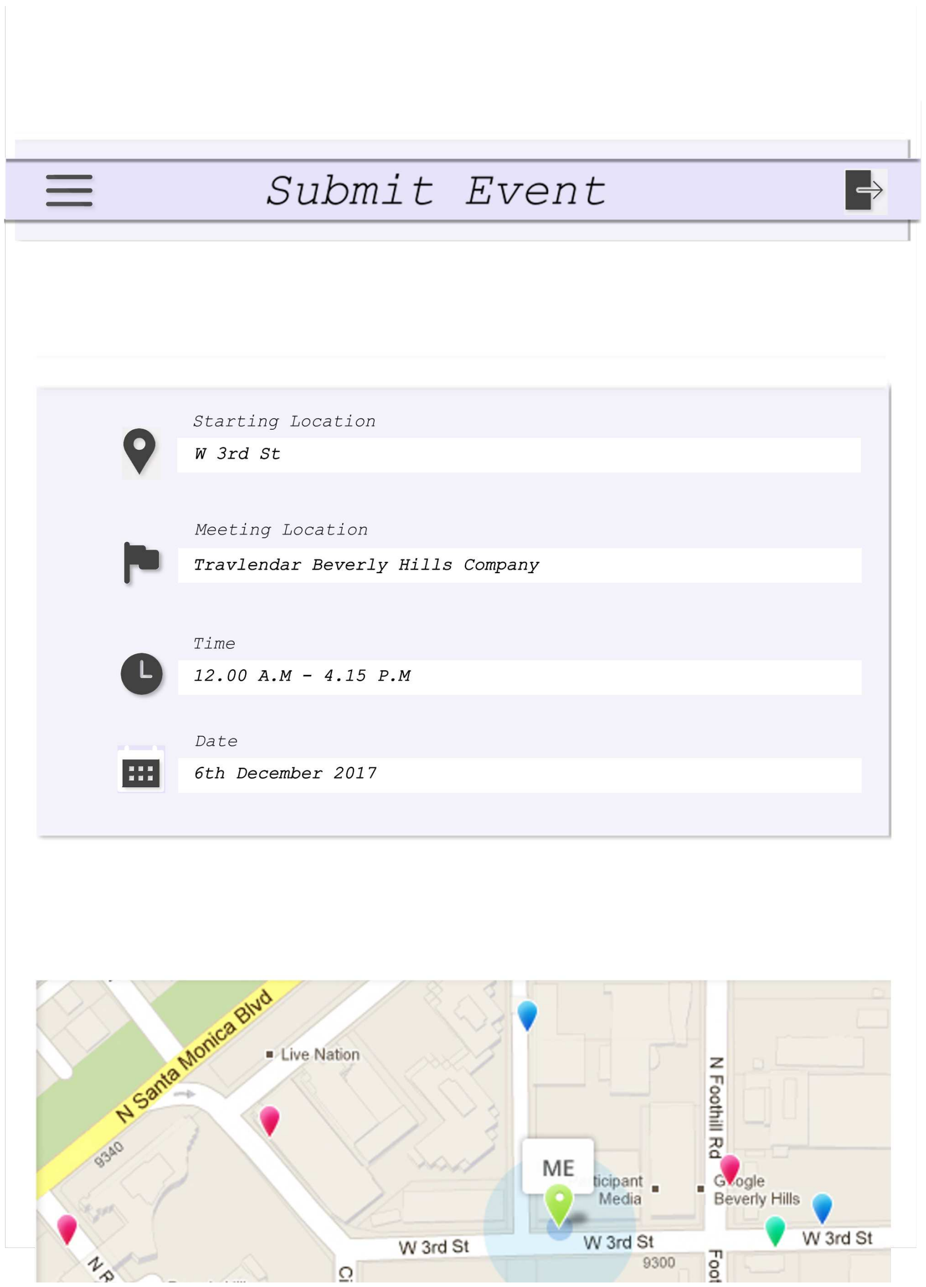
Phone
+18 3428 7692 1800

Figure 6:



Preferences

Figure 7:



Starting Location

W 3rd St

Meeting Location

Travlendar Beverly Hills Company

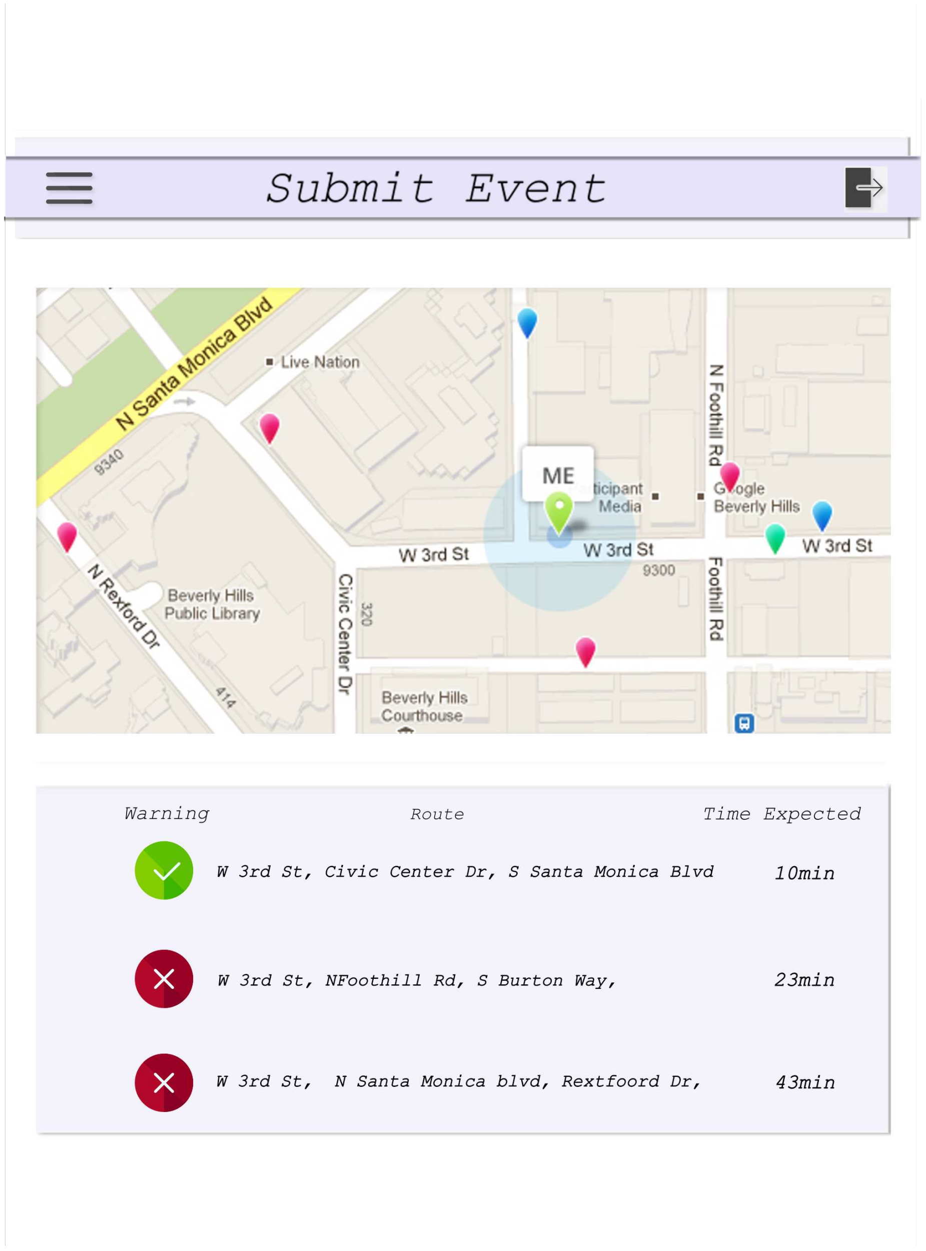
Time

12.00 A.M - 4.15 P.M

Date

6th December 2017

Figure 8:

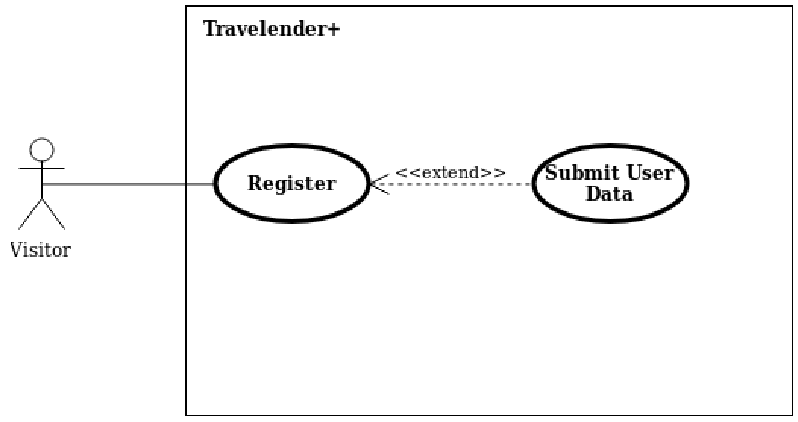


Warning	Route	Time Expected
✓	W 3rd St, Civic Center Dr, S Santa Monica Blvd	10min
✗	W 3rd St, NFoothill Rd, S Burton Way,	23min
✗	W 3rd St, N Santa Monica blvd, Rextfoord Dr,	43min

Figure 9:

Functional Requirements

Use Case about User Profile

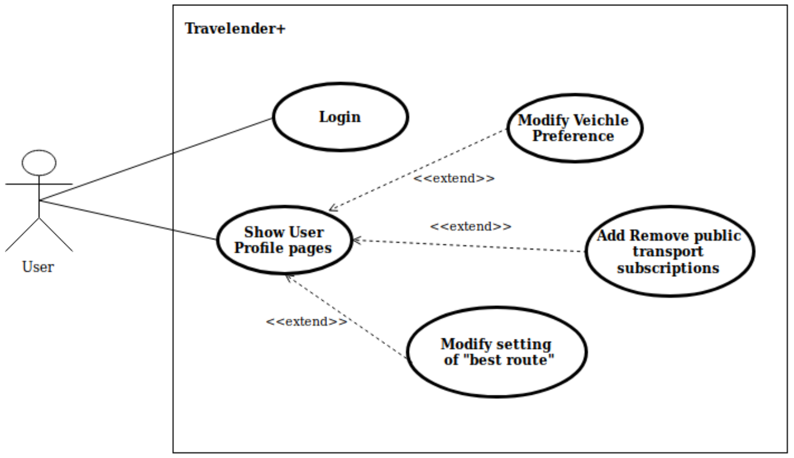


Name	Register Proces
Actors	Visitor
Goals	G2
Entry Condiction	There are no entry conditions
Flow Event	1) The visitor on the home page clicks on the register button to start the registration process. 2) The visitor fields the form and provides the informations. 3) The System saves the data 4) The system sends an e-mail with a link for verify the accuracy of the information provided by the user
Exit Condiction	1) after the user verifies the e-mail address

Exceptions

1) The visitor is already an user.
2) The visitor does not provide all the informations.
3) The visitor chooses an email address that has been associated with another user.
4) The visitor does not verify the email address in a period of 10 days

---



Travelender+
Login
Modify Veichle Preference
<<extend>>
Show User Profile pages
<<extend>>
Add Remove public transport subscriptions
<<extend>>
Modify setting of "best route"
User

Name

Login

Actors

User

Goals

G3

Entry Condiction

User is in the login page or in start screen of the app.

Flow Event

1) The User inserts his credential into "Username" and "Password"

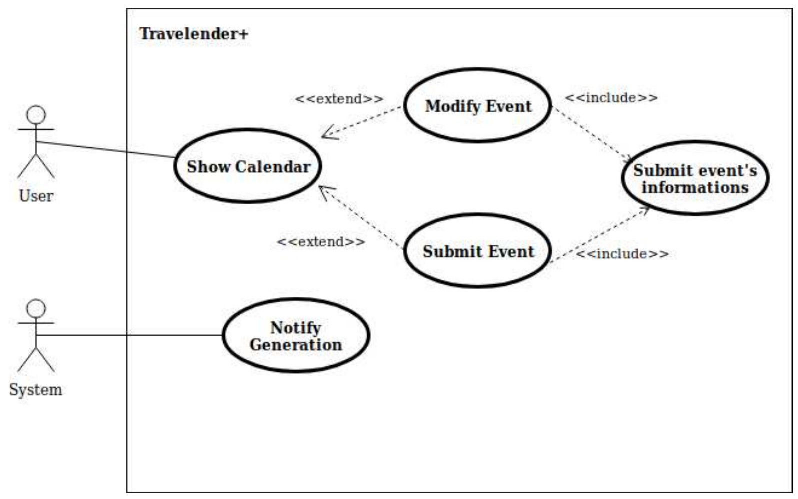
Exit Condiction

1) after the user has inserted the right credentials.

Exceptions	1) The user inserts the wrong credentials.

Name	User Profile modify
Actors	User
Goals	G1,G4,G5,G6
Entry Condiction	The Login in of the User was valid
Flow Event	1) The user visits his profile pages 2) The user chooses the tab with the information that he wants to change 3) The user changes his informations 4) The user chooses the best path from a list showed by the system 5) The user presses the save button
Exit Condiction	1) when the user presses the save button
Exceptions	1) The user inserts not valid informatons

Use Case Calendar Functions



Name	Show,Modify or delete the events
Actors	User
Goals	G1
Entry Condiction	The Login in of the User was valid
Flow Event	1) The user visits the calendar of the events and see the events. 2) The user chose to delete/change an event **or** 3) The user clicks on a warning to see what is the problem and the possible solution offert by the system. 4) The user presses the save button 5) The Sistem stores the changes
Exit Condiction	1) when the user psess the save button at the end of modify.
Exceptions	

Name	Submit a new event
Actors	User
Goals	G2,G4
Entry Condition	User has been already logged
Flow Event	1) The user visits the calendar of the events. 2) The user chooses to add an event. 3) the user submits all the information about the events 4) The user presses the save button
Exit Condition	1) when the user presses the save button at the end of modify
Exceptions	1) The user misses to fill important informations.

Name	Notify generations
Actors	System
Goals	G2
Entry Condition	The system has one or some notifies for the user

Flow Event

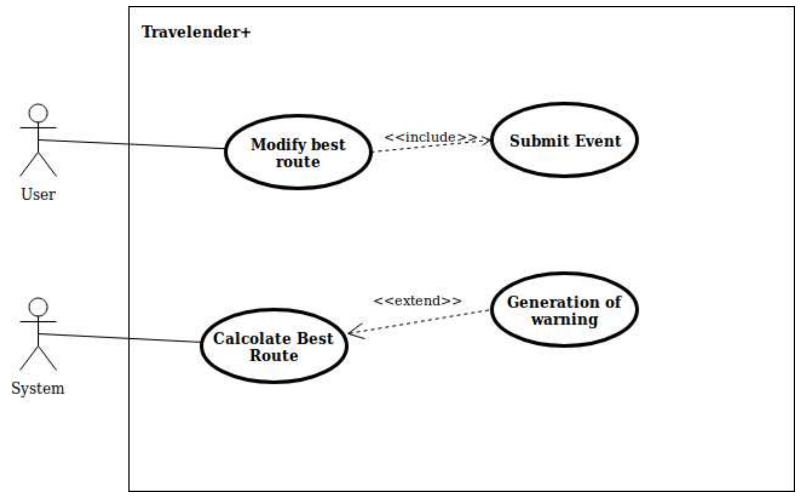
1) The system checks the calendar of the user.
2) The system generates a notify when the user needs to leave to be in time at an event
3) The system generates a notify if there are one or more warning about the events.

Exit Condition

1) When the system has finished to generate the notifies

Exceptions

Use Case Map Geolocalization Functions



Name

Best Route Alogitms

Actors

System

Goals

G1,G2,G3,G6,G8

Entry Condition

The User submits an events

Flow Event	1) After the user submits the system proceeds to apply the Best Route Algorithm 2) The system scans the user reference about the vehicle 3) The system creates a list of path which the user can choose to go to the meeting 4) The system waits the user's choice. 5) The system stores the event and the best route chosen by the user
Exit Condiction	1) when the user presses the save button at the end of the proccess
Exceptions	1) The user closes the application before the save 2) The system can't calcolate the best path for external reason

Name	Warning generation
Actors	System
Goals	G7
Entry Condiction	The User submits an event
Flow Event	1) The system can't calcolate the best path because there are no way to attend the event in time 2) The system generates a warning for the event 3) The system generates a notify for the user
Exit Condiction	1) When the system has finish to generate the notifies
Exceptions	

Name	Modify the best route
Actors	User
Goals	G4,G5
Entry Condiction	the system has calcolated a list of best path for the user
Flow Event	1) The user chooses one of the route suggested by the system 2) The user chooses to modify the path 3) User waits a positive response of the system which it checks if the modify is valid 4) User presses the save button to submit the changes
Exit Condiction	1) when the user presses the save button at the end of modify.
Exceptions	1) The system gives a negative response to the user modify

Performance Requirements

Performance for Apple iOS and Android App:

1) Battery Consuption should be not greater than 0.96mah.
2) 'Best Route' Calculation time should be not graeter 2.0 seconds
3) 'Alarm' function ('Event-reminder') should be configurable to be active even if the cellphone is Power Off. - tech difficulty to implement
4) The graphical effects of the 'Virtual Map' should not slow down the runtime execution of the App.
5) Memory Storage Consumption of the application should not be greater than 128MB.
6) (**optional**)'Alarm function' should be configurable to be active even if the cellphone is Power Off.

Performance for Browser Application:

1) Javascript Animations should be performed after that the login bar is loaded.

Design Constraints

see dd document for more informations

Standards Compliance

The software will use the following standards when deployed:

- Client Server protocol communications.
- Utilize a REST standard for communication between the backend and frontend

Hardware Limitations

The System will have the following hardware limitations:

- Semi-continuous [1] internet access (3G/4G/WiFi)
- GPS or trackable Geographical position.

Mobile Systems Limitations

1) Android system.
2) Android Mobile Systems should have installed the latest Google Play Service avilable API level 21 or more.

Other Constraints

Since the system relies on confidential information in order to work the system will need to store the data securely, especially regarding saved addresses. **None of the information provided by the user will be used for commercial purposes**.

[1]Semi-continuous meaning that the system can loose connection briefly but overall needs to be able to access the internet on a reoccurring basis, mostly to allow traffic/weather updates

Software System Attributes

Reliability

The main focus is on the 'Best Route' Algorithm and the calcolation of the 'Expected time' of Arrival at the 'Meeting Location'.

1) 'Best Route' Algorithm should be tested and have a coverage greater than 80%
2) 'Expected time' of arrival at the 'Meeting Location' should be have a relative error of 5% of the time exstimated.
3) If the user does not have internet connectivity on the mobile, he/she still could open the app and access to 'Calendar' function and view the meetings submitted.
4) The 'Expected time' of arrival at the 'Meeting Location' should be updated constantly in case of changing of weather forecast or (**optional**) unexpected event (public transport goes off. . .)

Availability

Travlendar+ helps its users to schedule their personal life appoinment and shold be as much open and accessible as possible even with the absence of Internet

1) The 'Calendar' function should be accessible on the App even if the phone is without Internet.
2) (**optional**) A pdf description of the 'Best Route' can be downloaded on the Travelndar+ Broswer.
3) (**optional**) User can import a 'Calendar' configuration package and simply update his/her personal schedule of appointments.

Security

Travlendar+ manages personal informaton of the user registered. For this reason it is very important the Security issue and some achievements have to be taken.

- https protocol used for all the calls
- salted password hashes are used in the backend DB
- use of firewall

Scenarios

Scenario 1

Karla is a businesswoman that needs an app to help her manage all her appointments; following her friend's advice she downloads the Tavelendar+ app. Karla registers herself onto the app by inserting her personal data (username, password, mail) and her transportation preferences. The app sends an email with a verification link in order to verify the existance and ownership of the email address. After Karla presses the link inside the email the registration process will be completed, she will be shown a quick tutorial on how to use the app and add/change her preferences. From this point onward she can start using Travelendar+.

Scenario 2

John wants to add an appointment to a day that has none. After pressing on the 'add appointment' button, a form is presented to him asking date, time, place and name. Moreover the application asks the starting position by offering a choice between the current position, from a list of saved locations or the position of the preceding appointment. Afterwards Travlendar+ checks if there are any overlaps with other appointments and the possibility of having lunch given the current schedule status. If no overlap is found then it computes the optimal paths to reach the appointment location, presenting a ranked list of alternatives. John picks one of the proposed itineraries, which is saved by the app.

Scenario 3

Jennifer submits a new event in her calendar. Travelander + verifies the present of an overlay and if there is the app generates a warning signal on that day. Jennifer by pressing on the signal can decide to modify the date of one of the appointments, in order to avoid overlapping, or delete appointments considered by her least essential, always in order to avoid overlapping.

Scenario 4

Greg has inserted an appointment that involves using a bicycle. The app check the day previous to the appointment weather condition, discovering that rain is forecasted. Knowing that the itinerary contains a part on bicycle, it sends a notification to Greg asking whether or not he wants to change the itinerary/transportation mean, offerring him alternatives.

Scenario 5

Mary added its lunch time between 12:00 and 2:30, with a duration of 45 min, during the configuration of the app. Mary knows that Travelendar+ will automatically add a 'Lunch' event to each day's timetable and, if needed, will shift the event in case of overlap. One day Mary adds enough appointments to make it unfeasible to have lunch, due to this the app will generate a warning, giving Mary the choice to reschedule the appointments or to skip lunch.

Scenario 6

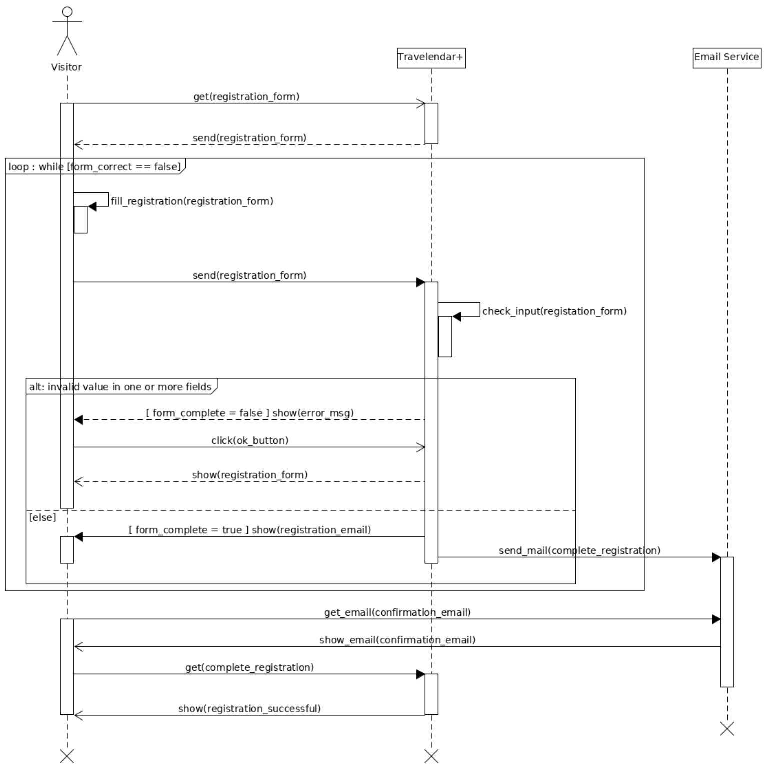
James is a man who is particularly concerned with ecology. He decides to use Travelander + to schedule his appointments. James chose Tralevander + because he knows that application is able to calculate the best path to reach the destination in order to respect the timetables and its preference. Among them James has put in the fact that he wants to keep low his ecological footprint. Travelander + can calculate James's routes so that they can be Green as much as possible.

Scenario 7

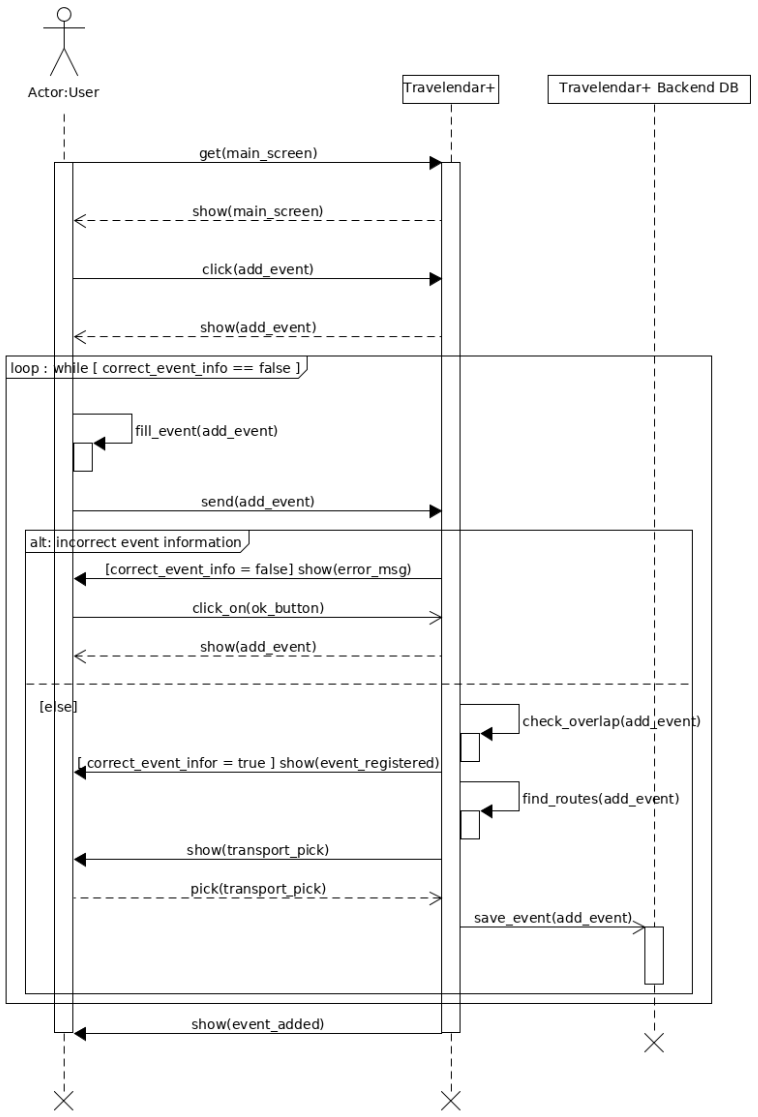
Kevin uses Travelendar+ to schedule a series of out of town appointments. After having picked the best path, the app offers Kevin the option to directly purchase the tickets needed for the trip.

Sequence Diagrams

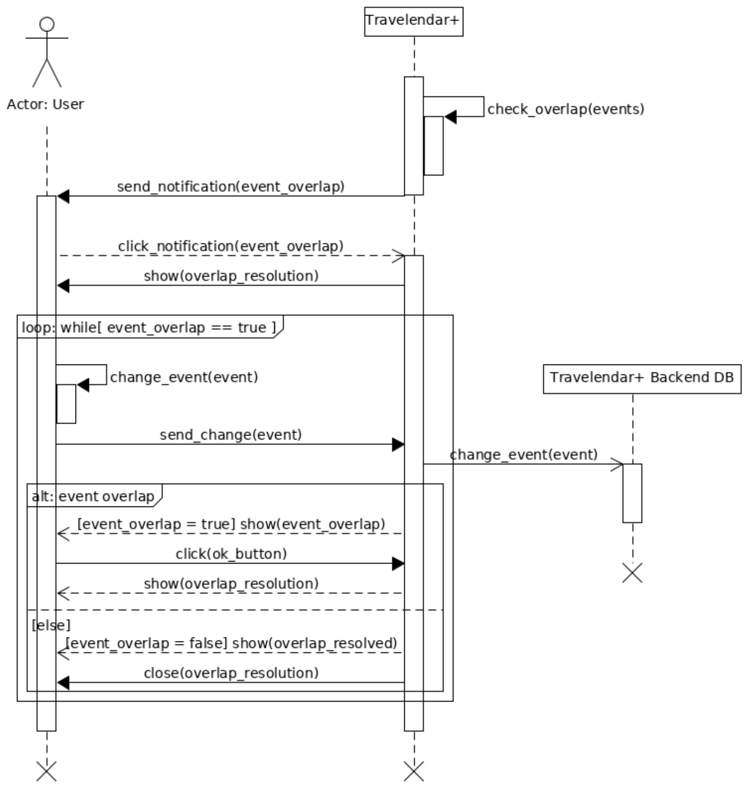
User Registration


Visitor
Travelendar+
Email Service
get(registration_form)
send(registration_form)
loop : while [form_correct == false]
fill_registration(registration_form)
send(registration_form)
check_input(registation_form)
alt: invalid value in one or more fields
[form_complete = false] show(error_msg)
click(ok_button)
show(registration_form)
[else]
[form_complete = true] show(registration_email)
send_mail(complete_registration)
get_email(confirmation_email)
show_email(confirmation_email)
get(complete_registration)
show(registration_successful)

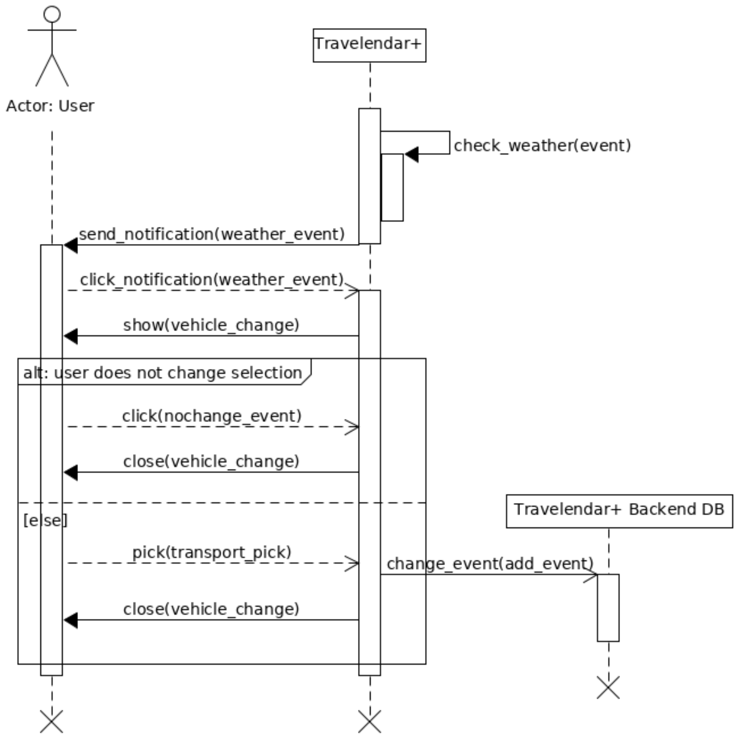
User Event Creation



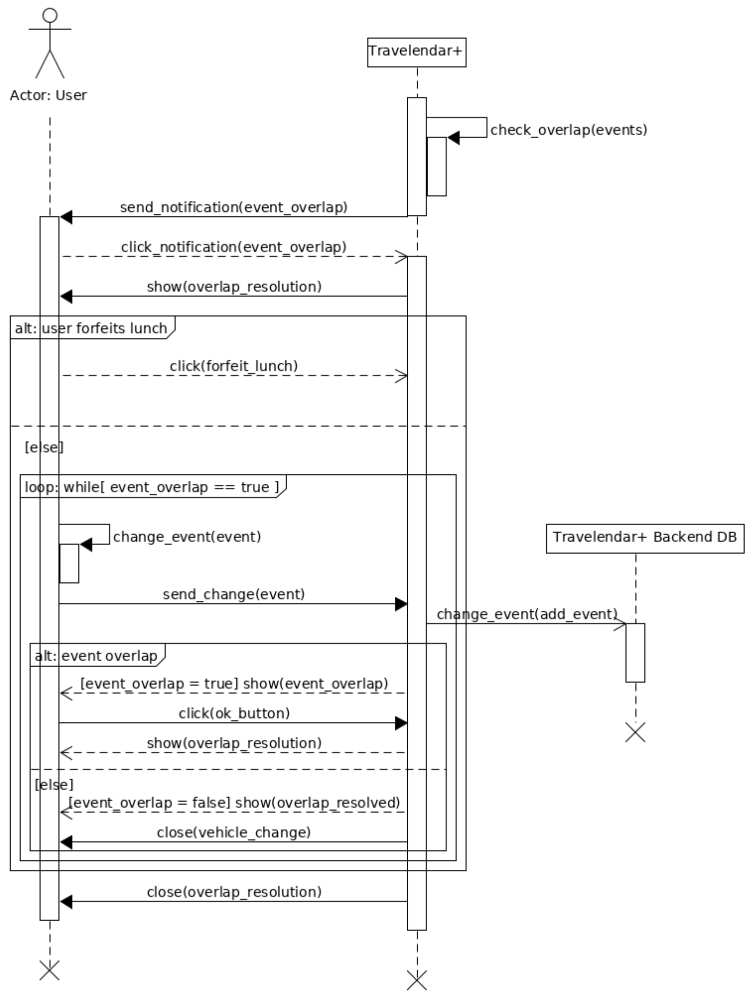
Overlap Resolution


Actor: User
Travelendar+
check_overlap(events)
send_notification(event_overlap)
click_notification(event_overlap)
show(overlap_resolution)
loop: while[event_overlap == true]
change_event(event)
Travelendar+ Backend DB
send_change(event)
change_event(event)
alt: event overlap
[event_overlap = true] show(event_overlap)
click(ok_button)
show(overlap_resolution)
[else]
[event_overlap = false] show(overlap_resolved)
close(overlap_resolution)

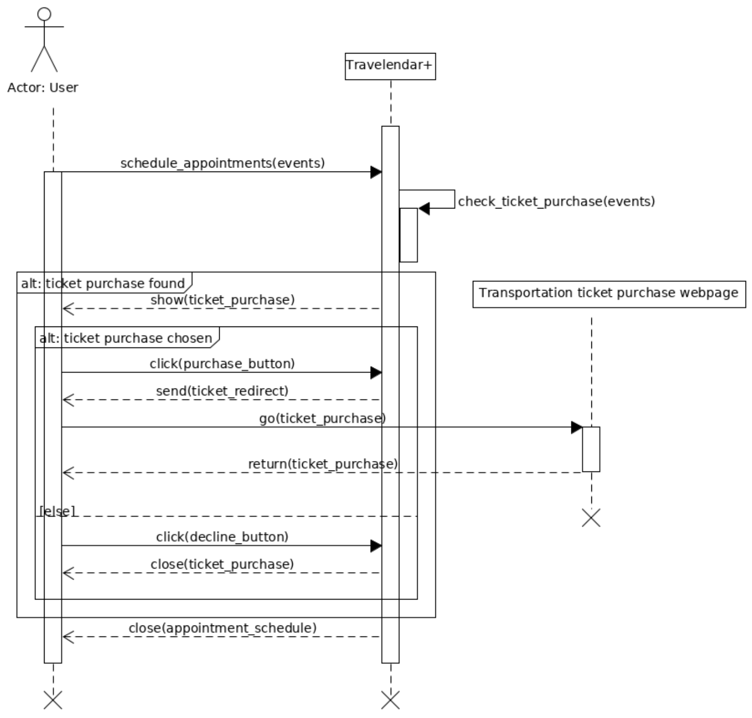
Weather Check



Lunch Conflict



Ticket Purchase



Alloy

Signatures

```
sig User{
    events: set Event,
    email:one String,
    prohibitedVehicles:set Vehicle
}

sig Vehicle{}

sig Event{
    initialTime:one Date,
    finalTime:one Date,
    position:one Position,
    warning:one Bool,
    paths:set Path
}

sig Date{
    time:one Int
}

sig Position{
    latitude:one Int,
    longitude:one Int
}

sig Path{
    startTime:one Date,
    endTime:one Date,
    vehicles:set Vehicle    ,
    startPosition:one Position,
    endPosition:one Position
}
```

Facts

```
//events must have a user
fact creatingEvents{
all e:Event | some u:User | e in u.events
}

//an event and a path can't end before start
fact timeLinearity{
    all e:Event | e.initialTime.time<e.finalTime.time
    all p:Path | p.startTime.time<p.endTime.time
}

//email are unique
fact emailUnique{
    no disjoint u1,u2:User | u1.email=u2.email
}

//date are unique
fact dateUnique{
    no disjoint d1,d2:Date | d1.time=d2.time
}

//single event not gen warning
fact warninGen1{
    all u:User| #u.events=1 implies
                u.events.warning=False
}

//event without sovrapposition mustn't gen warning
fact warningGen2{
    all disj e1,e2:Event|e1.warning=False implies
                        //there aren't sovrappositions
                        (e1.finalTime.time<=e2.initialTime.time or
                        e1.initialTime.time>=e2.finalTime.time)and
                        //also e2 not has warning
                        e2.warning=False
}

//event with sovrapposition must gen warning
fact warningGen3{
    all disj e1,e2:Event|e1.warning=True implies
                        //there are sovrappositions
                        e1.finalTime.time>e2.initialTime.time and
                        //also e2 has warning
                        e2.warning=True
}



//Events have only possible path
fact eventPath{
    all e:Event | some u:User,p:Position| choosePath[p,e,u]
}
```

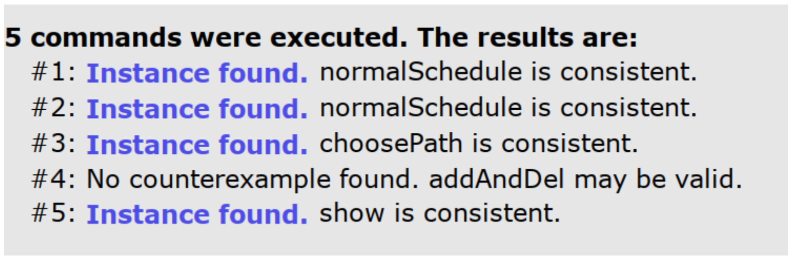
Assertion

```
//add and dell same events gen same user's set of events
assert addAndDel {
    all u1,u2,u3:User,e:Event |
    not e in u1.events and addEvent[u1,u2,e]
                    and delEvent[u2,u3,e] implies
    u1.events=u3.events
}
```

Dynamic model

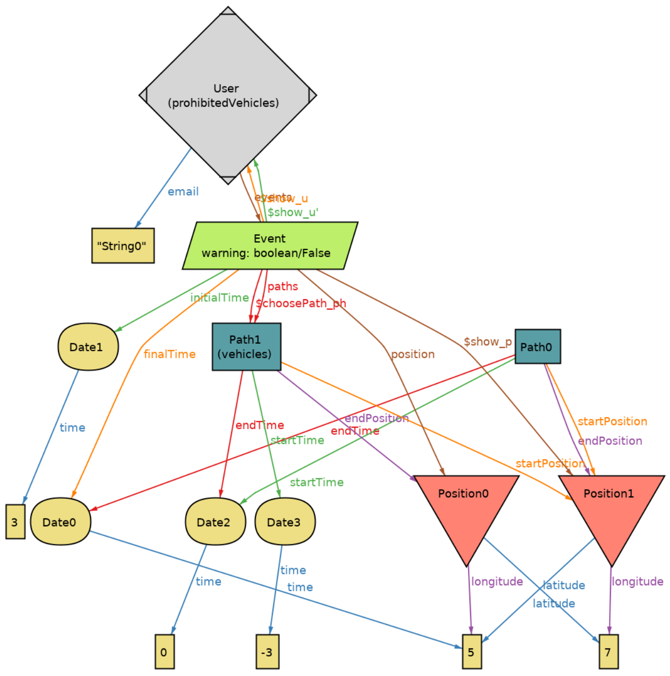
```
//the algorithm choose the paths from start position to event position
pred choosePath[p1:Position,e:Event,u:User]{
    some ph:Path |
            //start and arrive in the right position
            ph.startPosition=p1 and ph.endPosition=e.position and
            //arrive at event with no delay
            ph.endTime.time<e.initialTime.time and
            //path not has prohibited veicle
            ph.vehicles not in u.prohibitedVehicles
}

//addiction of event
pred addEvent[u1,u2:User,e:Event]{
    u2.events=u1.events+e
}

//delection of event
pred delEvent[u1,u2:User,e:Event]{
    u2.events=u1.events-e
}

pred normalSchedule{
    #User.events>=2
    all e:Event | e.warning=False
}

pred warningSchedule{
    #User.events>=2
    some e:Event | e.warning=True
}

pred show{}


run normalSchedule for 5 but exactly 1 String
run normalSchedule for 5 but exactly 1 String
run choosePath for 4 but 1 User,2 Position,2 Vehicle, exactly 1 String, 1 Event, 2 Path
check addAndDel
run show  for 4 but 2 Position, exactly 1 String,exactly 1 Event,2 Path
```

Proof of consistency



5 commands were executed. The results are:
#1: **Instance found.** normalSchedule is consistent.
#2: **Instance found.** normalSchedule is consistent.
#3: **Instance found.** choosePath is consistent.
#4: No counterexample found. addAndDel may be valid.
#5: **Instance found.** show is consistent.

Generated World



Effort Spent

Prosdocimi Marco

15/10/2017 2h

18/10/2017 2h

21/10/2017 7h

22/10/2017 4h

25/10/2017 3h

26/10/2017 1h

28/10/2017 3h

29/10/2017 2h

Ruggiano Enrico

15/10/2017 2h

16/10/2017 2h

18/10/2017 1h

20/10/2017 4h

21/10/2017 2h

22/10/2017 3h

24/10/2017 2h

25/10/2017 3h

28/10/2017 2h

29/10/2017 5h

Giacomo Vercesi

14/10/2017 5h

15/10/2017 3h

17/10/2017 1h

20/10/2017 3h

22/10/2017 2h

27/10/2017 3h

28/10/2017 5h

29/10/2017 7h

References

Picture

All the picture used for the UI are desiged by yanalya / Freepik.

Those were free downloded from http://www.freepik.com with a Free Licence.

For more informations just read the "Licence free.txt" file in the repository or visit •
http://www.freepik.com/terms_of_use

Icon Credits

- Position, Setting Icon:

  Icon made by CC 3.0 BY from www.flaticon.com

- Calendar, Upload, Menu, Submit Event, Cancel, Accept, Exit, Like, Dislike, Partial like, Hour, Destination Icons:

  Icon made by Flaticon Basic License BY from www.flaticon.com

- Vehicle Icons:

  Icon made by Flaticon Basic License BY from www.flaticon.com

Standard IEEE

standard ISO/IEC/IEEE 29148