MeteoCal Project

Requirements Analysis and Specification Document 2.0

Author:

Francesco Angelo {francesco.angelo@mail.polimi.it}

Valentina Ceriani {valentina.ceriani@mail.polimi.it}

Umberto Di Fabrizio {umberto.difabrizio@mail.polimi.it}

Prof: Elisabetta Di Nitto

Milan, January 25, 2015

Contents

1 Product Constraints

1.1 Description of the Problem

We are asked to design and implement a weather based online calendar to help people scheduling their personal events avoiding bad weather conditions in case of outdoor activities.

Users, once registered, should be able to create, delete and update events. An event should contain information about when and where the event will take place, whether the event will be indoor or outdoor. Any number of registered users can be invited to an event by the organizer will be able to update or delete the event. Invited users can only accept or decline the invitation.

Whenever an event is saved, the system must manage time consistency by avoiding conflicts with existing events and it should enrich the event with weather forecast information (if available). Moreover, it should check the forecast periodically and notify all event participants one day before the event in case of bad weather conditions for outdoor events.

In the case of bad weather, three days before the event the system will suggest the first available sunny day to the organizer, to allow rescheduling the event.

The system will notify the user when: he receives an invitation, in case the forecast for the event has changed, or if the event administrator has updated the event.

Users can select a privacy setting (public/private) for the events and for the calendars. In case of a public calendar, every user can see the details of a public event, but only busy slots if the event is private. Nobody can see a private calendar independently of the events added to it.

Notifications are received by the users when they log into the system, and also by email (invitations and cloudy outdoor events alerts).

Users can manage their own calendars: add a new calendar, edit an existing one, delete it, import and export. The administrator of an event can manage the event, invite other registered users to participate or change the privacy of it.

1.2 The Purpose of the Product

MeteoCal purpose is to remind appointments and give a clear global vision about personal schedules, in order to prevent outdoor event to be spoiled by adverse weather conditions. Moreover, the goal of this product is to avoid time conflicts between events. Both these reasons led us to design a notification system that warns the user about any of these conditions, but it lets the user decides whether reschedule the event or not. Therefore the intent is to create a system that helps and supports every user with alerts and warnings, but does not constrict their decisions.

Besides the most common private single event, MeteoCal user experience is enriched with public and shared events. Every user can create a public event, just changing the privacy settings of it, or they can decide to invite some other to their event. These features have the purpose to improve usability and organization of big events, as well as offering a better model of reality.

Calendars are the main feature of the system, so we designed them as multiple entities that can be private or public. Giving the possibility to have multiple calendar, we allow users to better categorize their events, and keep their calendars neat and clearer: a “business” calendar can be used to schedule work appointments, while the “personal” can be filled with any other events. Although we let the user split his events into multiple calendars, the system take care of checking conflicts in every user’s calendars.

1.3 Client and other Stakeholders

Our stakeholder is our professor, who gave us the project description. The request is to have a full working product that fulfill the requirements within the scheduled deadlines. What concerns our professor is to give us the know-how needed to follow every stage of the development process.

Our clients can be identified within two main groups. In the first one we have mostly calendar-oriented customers: they are interested in using our product to schedule their own private events, they have multiple calendars for a clearer view of their appointments, and they occasionally are invited to shared events. They have almost no interest in public events, and their usage of the product will be easily comparable with a planner or and agenda.

The second kind of customer is mostly event-oriented: they are using our product to invite people to their own events, join shared events and search for interesting public events. They probably have only one calendar and they own few personal events, and they are strictly related to the user community.

1.4 Domain Properties

- Weather forecast are reliable.
- Weather forecast are not always available
- Weather forecast are always available within 3 days
- Intention of the user is to use the system for its purpose

1.5 Naming Conventions and Definitions

Update an event Modify information about the event, concerning date and time, place, description and privacy settings.

Admin/ administrator/ creator/ owner Who creates the event.

User Whoever has been registered into the system

Event Independent entity created by a user, with at least date, place, duration, privacy information.

Calendar It is a calendar owned by a single user, displaying the days of each month and week through years, including a collection of events, represented in the exact schedule.

Schedule Planning the day by day events without creating unknown (to the user) conflicts.

Invitation Request to take part of an event, sent by the owner of the event.

Participate Accept an invitation.

Decline Reject an invitation

Bad conditions/ adverse weather conditions Collection of weather conditions not suitable for outdoor events: drizzly, tempestuous, showery, stormy, snowy, rainy.

Personal event Private event without guests

Shared event Private event with guests

Share a calendar Set privacy to ‘Public’

1.6 Assumptions

- System checks weather condition (if available) of an event as soon as date and time infos are selected, to promptly warn the owner in case of adverse forecast. This avoids creating an event that need to be reschedule immediately after it has been created, causing plenty of useless notifications to every invitee.
- Since there are public events, whose details are shown to all the registered users, there is a web page that collects all these events so that an user can easily browse through them and, if he wants, he can participate without receiving an explicit invitation.
- System notifies every change in an event such as change of location, time, weather forecast or in case of deleting.
- The owner of an event can invite users at any time, not only during creation
- The system allows user to schedule an event which is conflicting with another (in any of his calendar), but only after the user has been warned.

2 Functional Requirements

Unregistered users can only

- Sign up into the system
- Sign in into the system

Registered Users can

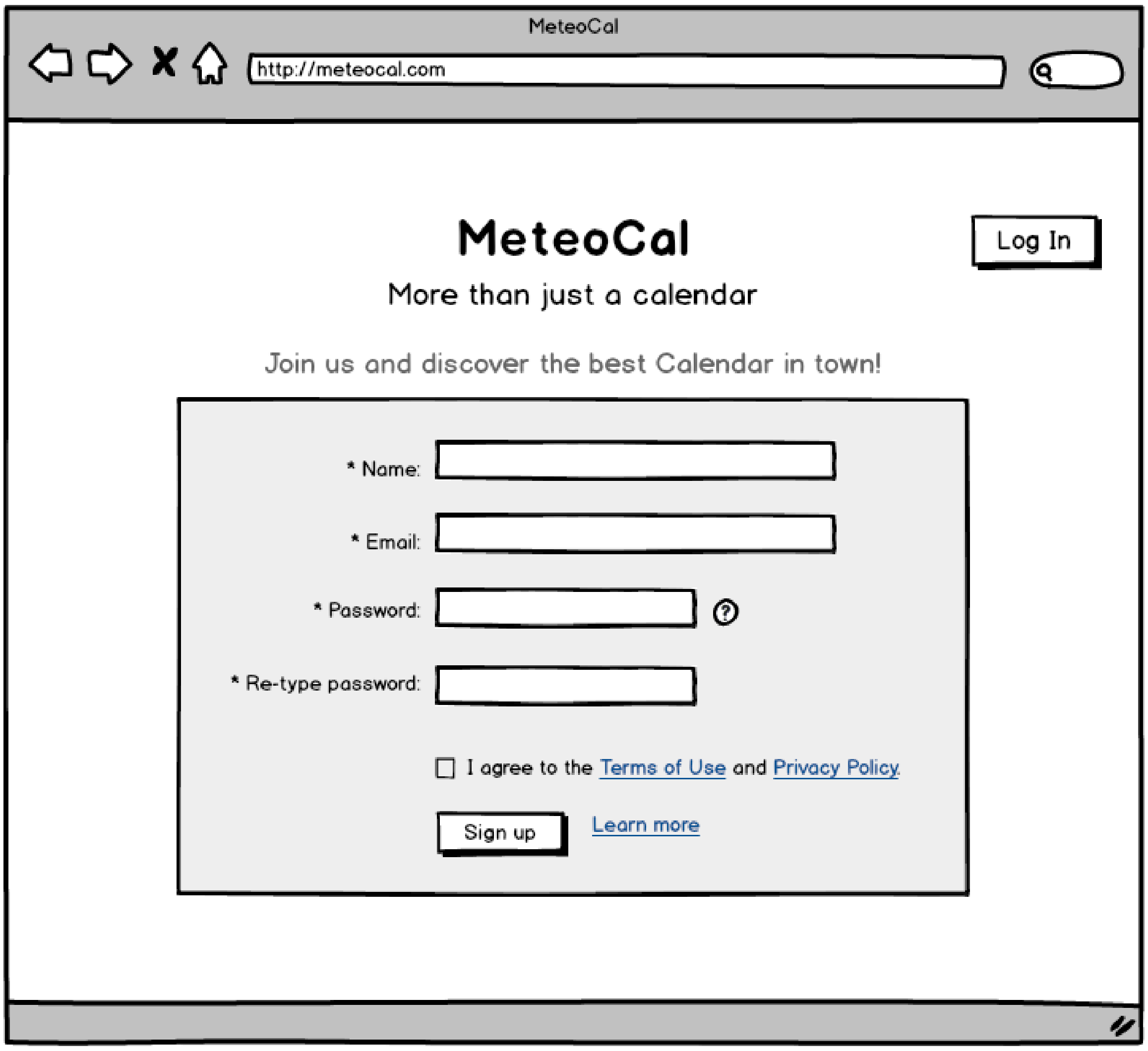
- Create and manage (edit or delete) an event
- Add an event to a calendar
- Invite people to an event
- Delete account
- Answer to an invitation
- Join public event
- Create and manage (edit or delete) a calendar
- Search for a registered user and view his calendar
- Search for and view an event
- Import and export an existing calendar
- Log out

3 Non-functional requirements

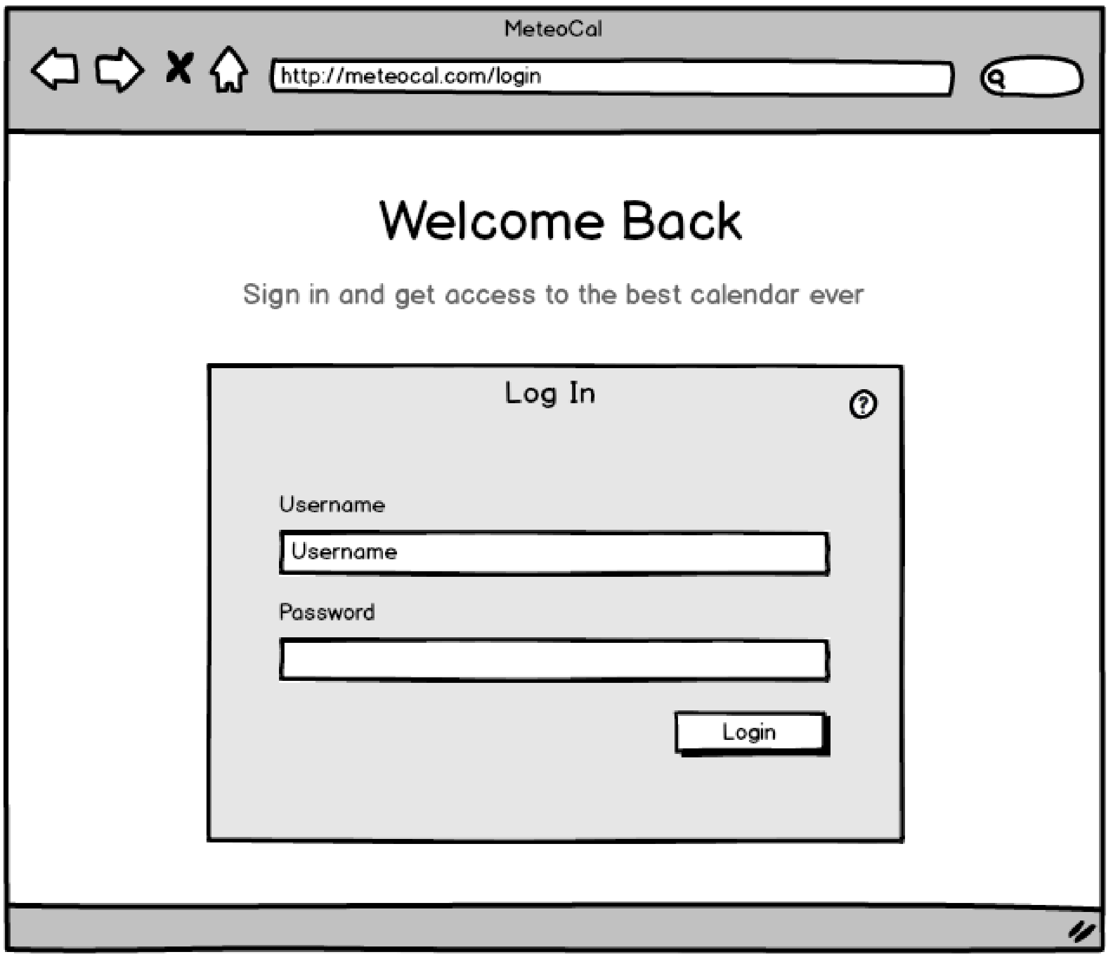
3.1 Look and Feel Requirements

The system will be implemented as a web service so that anyone with an internet connection and a web browser can manage his own calendar anywhere and whenever they need to.

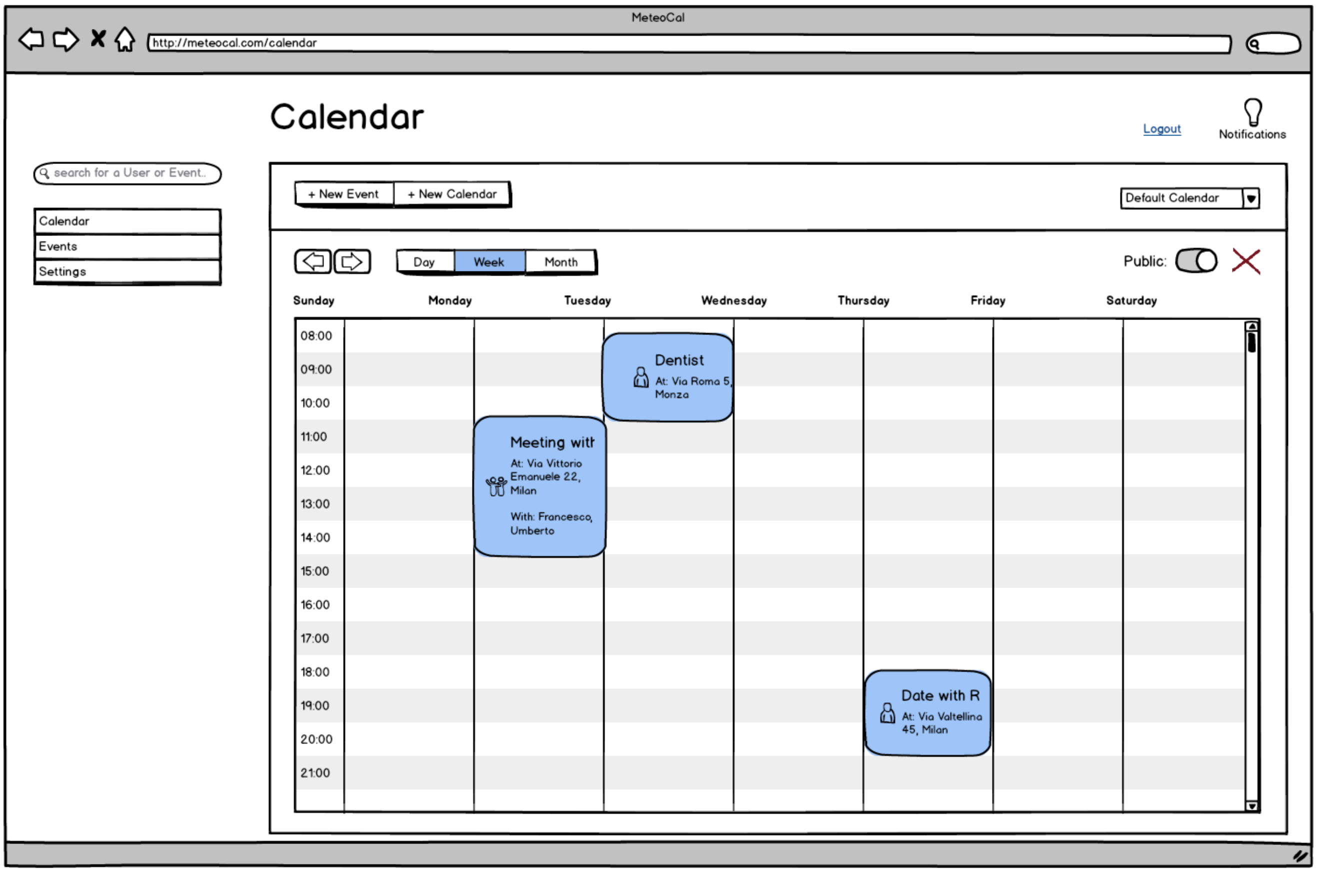
Inspired by the most popular websites style, we will design our homepage as Sign Up page with attractive look with graphic effects; the Log in function will be available clicking on the "Log In" button.


MeteoCal
http://meteocal.com
MeteoCal
More than just a calendar
Log In
Join us and discover the best Calendar in town!
* Name:
* Email:
* Password:
* Re-type password:
I agree to the Terms of Use and Privacy Policy
Sign up
Learn more

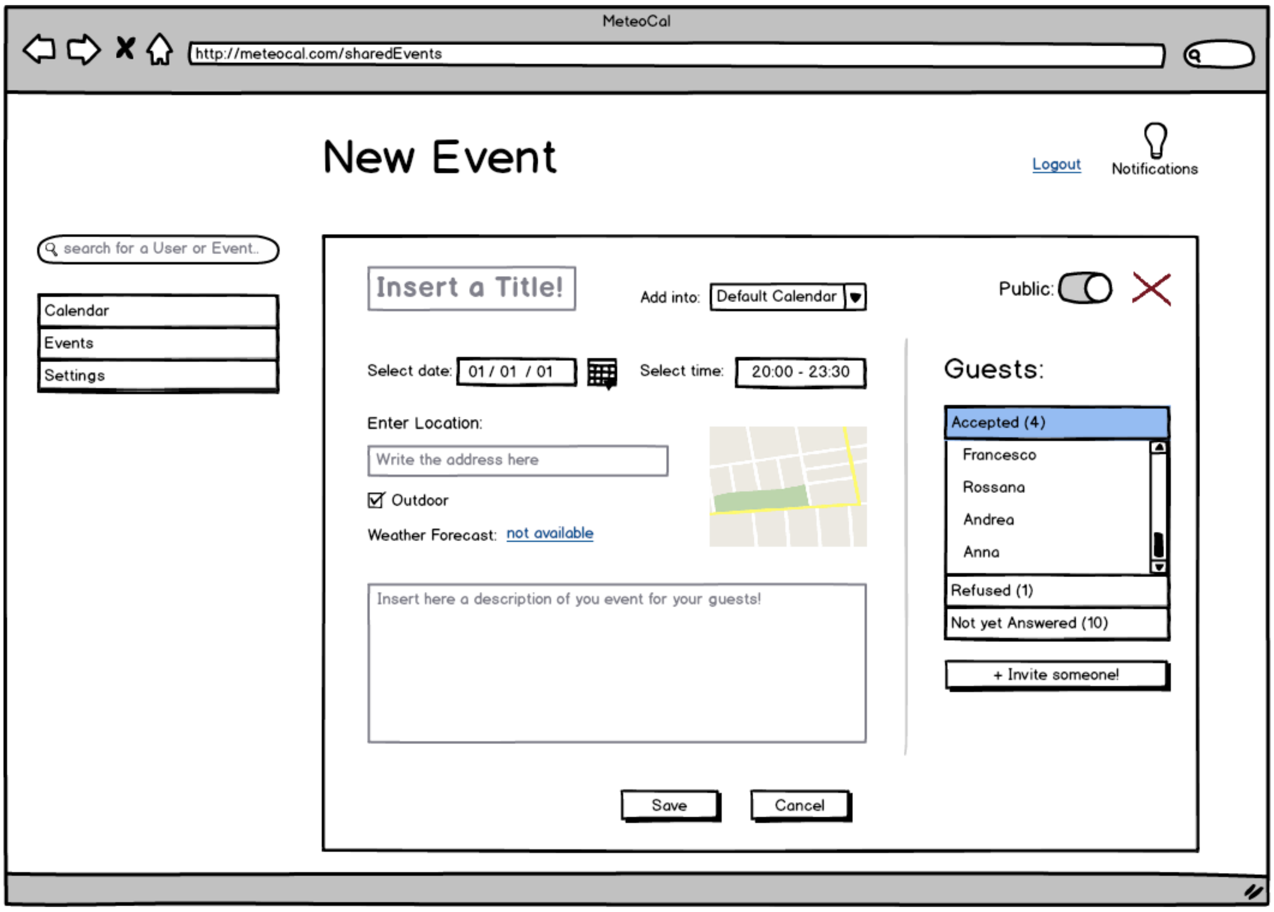
Log in page will have a easier and minimal design.


MeteoCal
http://meteocal.com/login
Welcome Back
Sign in and get access to the best calendar ever
Log In
Username
Username
Password
Login

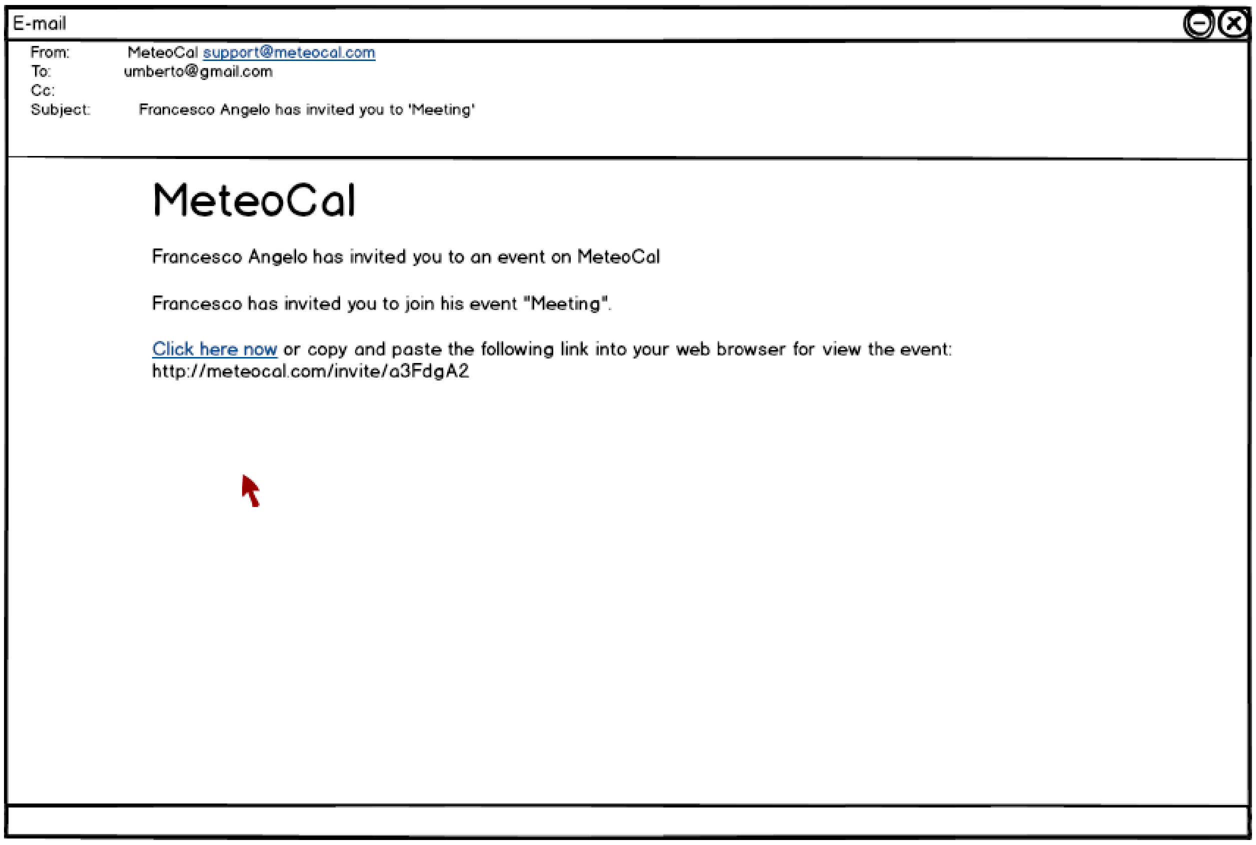
After Login or signing up, the system will display the user calendar with editing options: change the current displayed calendar, delete the current calendar, change privacy settings. Moreover, the user can customize the view, switching within 3 possible choice: Daily, Weekly and Monthly view. From this page user will be also able to create a new event or a calendar. Events will be displayed as full slots containing short information about it.



The "New Event" page will be a template of an event, with some text field to fill and a checkbox for outdoor option. Invitation will be sent from this page. Event infos will be enriched with weather forecast (if available). Event creator can choose in which calendar add the event; he can also decide not to add the event to any calendar: in this way, since the event is not scheduled in the calendar, will not be evaluated for conflicts.



Since an user is invited to an event, he can reach the event page selecting the “Invitations” tab and clicking on the desired event, or clicking on the link attached to the email notification or in-app notifications.



From: MeteoCal support@meteocal.com
To: umberto@gmail.com
Cc:
Subject: Francesco Angelo has invited you to 'Meeting'

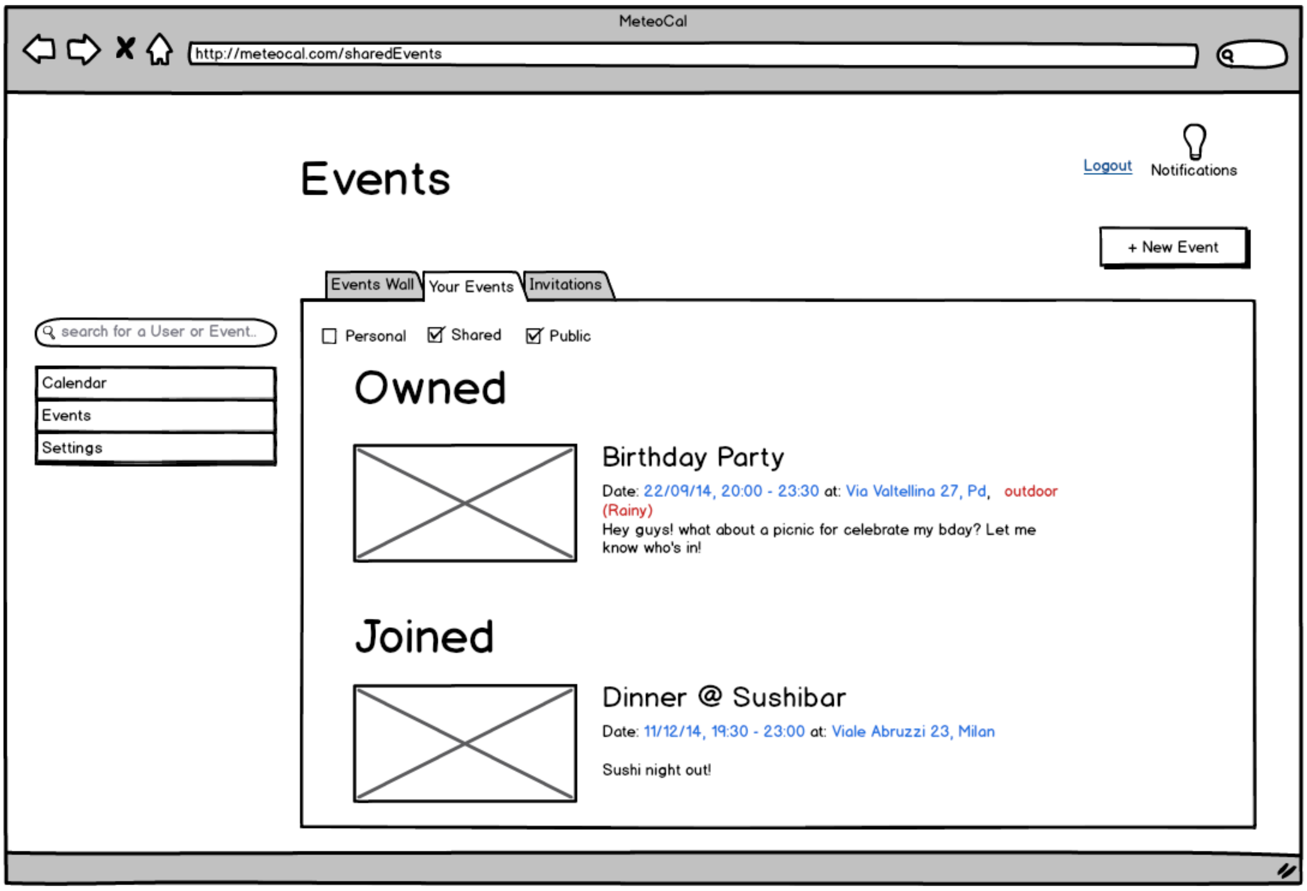
MeteoCal

Francesco Angelo has invited you to an event on MeteoCal

Francesco has invited you to join his event "Meeting".

Click here now or copy and paste the following link into your web browser for view the event:
http://meteocal.com/invite/a3FdgA2

Once created, the event will be available on the Event page, under "Your Events" tab. From this page the user will have a quick overview of all the events he is involved into, both owned events or joined ones. With the checkbox on the top, user can choose which event to put on show. From the "Invitation" tab user can have an overview of every event he has been invited to, while from "Events wall" he can discover every public event scheduled in the system.



Birthday Party

Date: 22/09/14, 20:00 - 23:30 at: Via Valtellina 27, Pd, outdoor (Rainy)

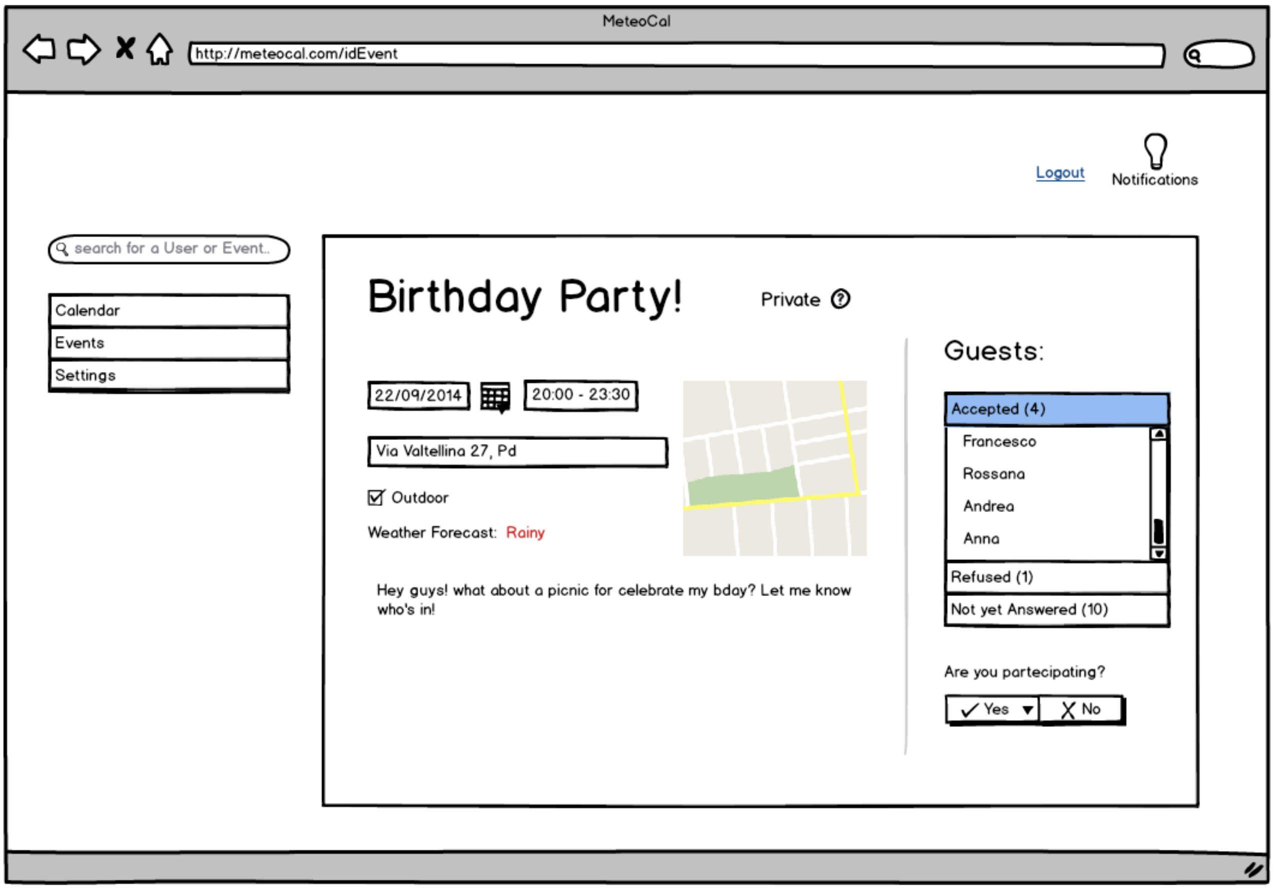
Hey guys! what about a picnic for celebrate my bday? Let me know who's in!

Dinner @ Sushibar

Date: 11/12/14, 19:30 - 23:00 at: Viale Abruzzi 23, Milan

Sushi night out!

The event page is pretty much the same of the event creation page, with every text field non editable and filled with the available information. Guests can see small map preview for the defined location, as well as the complete guests list. They can also confirm their presence on the event or reject the invitation.



http://meteocal.com/idEvent

Birthday Party!

Private

22/09/2014 | 20:00 - 23:30

Via Valtellina 27, Pd

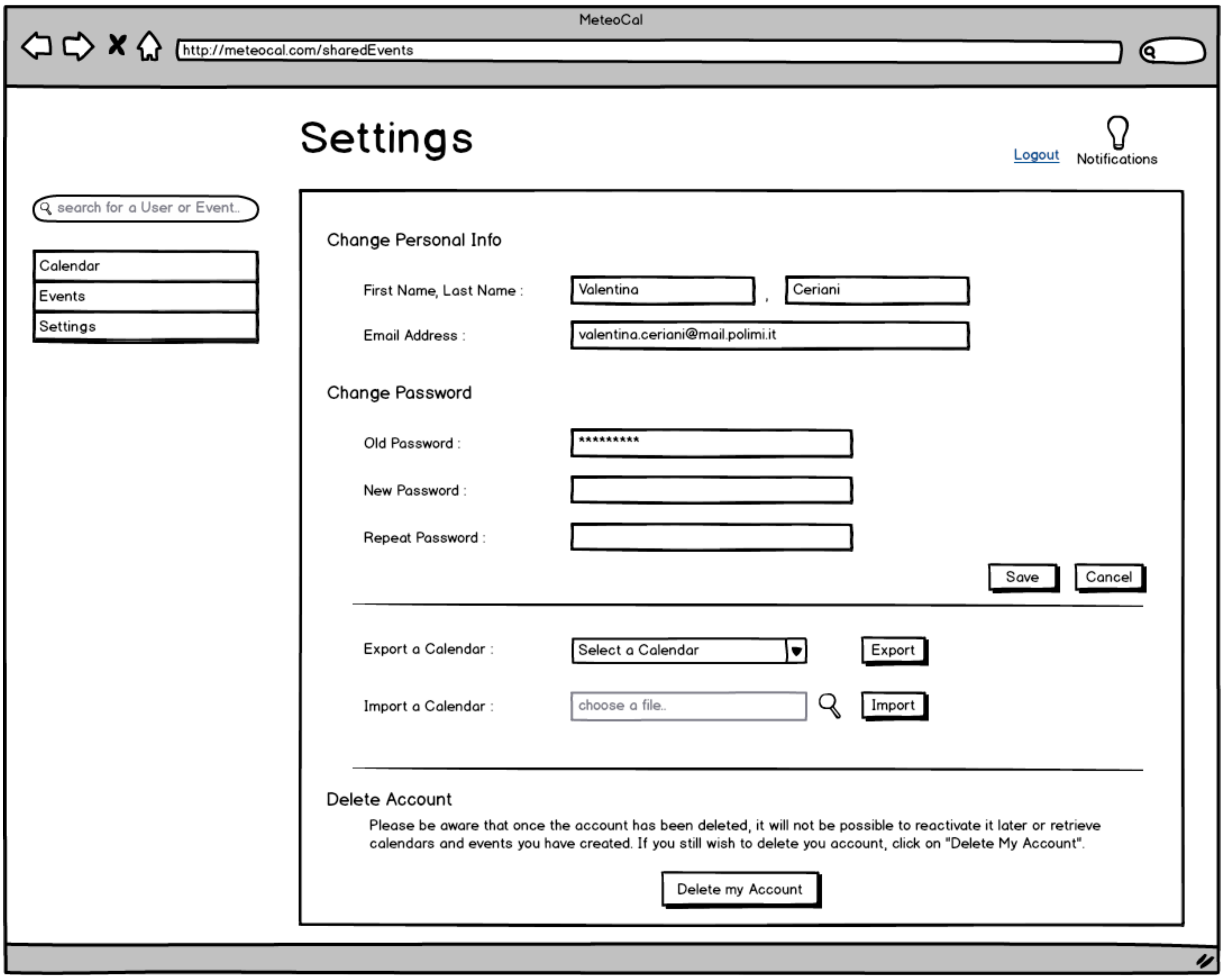
☑ Outdoor

Weather Forecast: Rainy

Hey guys! what about a picnic for celebrate my bday? Let me know who's in!

Francesco
Rossana
Andrea
Anna

On the left menu is possible to reach the settings page. It includes profile info in editable text fields, Import/Export options and the possibility to delete the account.



MeteoCal

http://meteocal.com/sharedEvents

Settings

Logout Notifications

search for a User or Event..

Calendar
Events
Settings

Change Personal Info

First Name, Last Name : Valentina , Ceriani

Email Address : valentina.ceriani@mail.polimi.it

Change Password

Old Password : *********

New Password :

Repeat Password :

Save Cancel

Export a Calendar : Select a Calendar Export

Import a Calendar : choose a file.. Import

Delete Account

Please be aware that once the account has been deleted, it will not be possible to reactivate it later or retrieve calendars and events you have created. If you still wish to delete you account, click on "Delete My Account".

Delete my Account

3.2 Ease of use and learning

Anyone with a basic understanding of web sites should be able to be at ease using the system. The interface that we want to develop must be minimal and user friendly so that the user is not overloaded by links, but his attention is focused on the main functionalities.

To this aim we also want to build a quick introductory video so that the user can quickly grasp all the functionalities of the system.

The web system should guide the user in each and every functionality avoiding mistakes by restricting the possible user choice in advance, using the system knowledge of the user's intentions and privileges.

3.3 Security Requirements

Only registered user can login into the system, in this way we provide a first layer of security.

Managing the privacy each user can protect their personal data, and the system authentication will make sure that only the users with the required privileges will access the event details.

3.4 Capacity requirements

The system must be able to collect and store all the necessary information without any sensible delay and it should be able to scale.

(If possible we will use the Google automatic services for websites scaling)

4 Specification

We deeply analyzed the problem description and the functional requirements and here we state the formal specifications that the system must fulfill.

Event specifications

- An event is created and updated by exactly one user, who is the administrator of it.
- Each event that a user participate to, must be at most in one calendar Events which are not inserted into any calendar, will not be considered as scheduled
- Event conflicts will be evaluated only among scheduled events
- It is possible to delete or edit past events.
- When the user delete an event:
- If the user is also the owner of the event, it is cancelled.
- If the event is not owned by the user the participation to the event is changed to 'Decline' (thus it disappear from the user's calendar).
- Cancel an event means to remove it from every participant's calendar and delete the invitations corresponding to that event.
- The answer to an invitation can be changed at anytime by the user.
- When the event privacy changes from public to private, the system gives the owner the possibility to send invitations to all the users who joined the event without an invitation or to close the event to only the invited users.
- The creator of an event necessarily participate to it.
- An event is created and updated by exactly one user, who is the administrator of it.
- No one can create a past event.

Calendar specifications

- A user owns at least one calendar
- A calendar is owned by exactly one user
- Every calendar has is own privacy settings.
- It's possible to have conflicts in a schedule (see assumptions)
- When a calendar is deleted the user can choose to:
  - Move the events to another calendar before deleting it.
  - Delete the calendar
  - Delete the calendar and the events created in it.
- Calendars of an user have only the events to whom he participate to.
- When a calendar is imported, the system adds to it only events that still exist.
- Search for a registered user and display their calendar:

Calendar privacy setting	*Event privacy setting*	*Calendar Visibility from the outside*
Public	Public	Details of the event are visible to all users
Public	Private	Only busy slots are visible with no infos
Private	Public/Private	Not Visible

- When a calendar is imported a new one is created in the user profile automatically
- When an event is imported the system checks that there isn't another user's calendar with that event, in that case the system will not import the event and will notify the user that some events haven't been imported because they are already in another calendar. (In a later implementation the system will ask if the user wants to move those events to the new calendar)
- Calendar has no beginning nor end.

Notification specifications

- If an event is cancelled by its owner than every participants will receive a notification.
- No user will receive notifications about an event once he has declined it

Invitations specifications

- the owner of an event is the only one who can send invitations and cannot invite himsel
- if the event is public, everyone can join it, even without an invitation
- Each user can receive at most one invitation for each event.

Other

- Events and Calendars are completely independent entities.
- If a user deletes his account than all his calendars are deleted, and the events in them follow the event deletion policy.

4.1 Actors

anonymous user This is an user not recognized by the system and he can only register or log in to access the user functionalities

user He can access all the main services since we consider he already logged in into the platform.

4.2 Scenarios

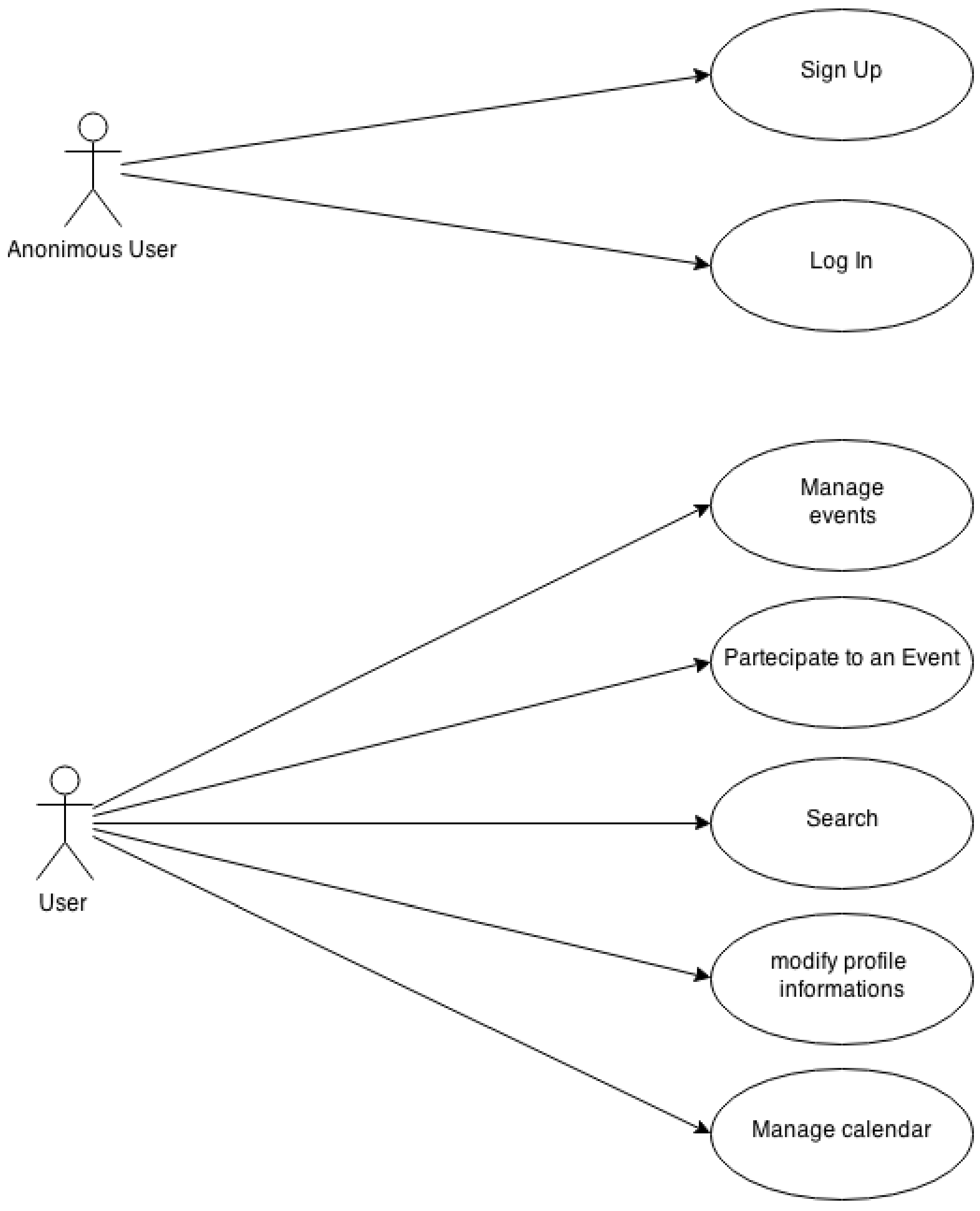
- Carlo wants to celebrate his birthday with a picnic in the park but he doesn't want the weather to spoil his day. He surf on the web to find a solution to automatically check the weather forecast for the desired day and to share all the details with his friends.
  He register to the MeteoCal platform and create the event adding his friends.
  The day before the event the system notify Carlo as well as his friends about bad weather conditions and it suggest to reschedule the event for the day after. Carlo reschedule the event and the system automatically notify his friends.

- Marta has been using the MeteoCal service for the last two months and she decides that it isn't useful anymore. She wants to delete his profile but the system alerts her that she will lose all her calendars permanently and suggest her to export them to the local pc.
  She exports the calendar as suggested and then deletes her account.
  A week later she changes her mind and wants to register again to MeteoCal to better schedule her appointments. After registering she imports the calendar back into the system and finally she can have a complete overview of her events.

- Luca wants to join a Cryptography conference in Milan but he doesn't know where it will be hosted and who created the event on MeteoCal. He login into meteoCal and browse to the public events wall, he then searches the event he is looking for. He find the conference, join the event and add it to his 'Conferences' calendar. The system notify him that there is a scheduling conflict and he cannot attend the cryptography event because in the same day he has scheduled a football match with his friends. Since Luca doesn't know which event he will attend he ignore the conflict warning and postpone the decision to the following days.

4.3 Use case diagram

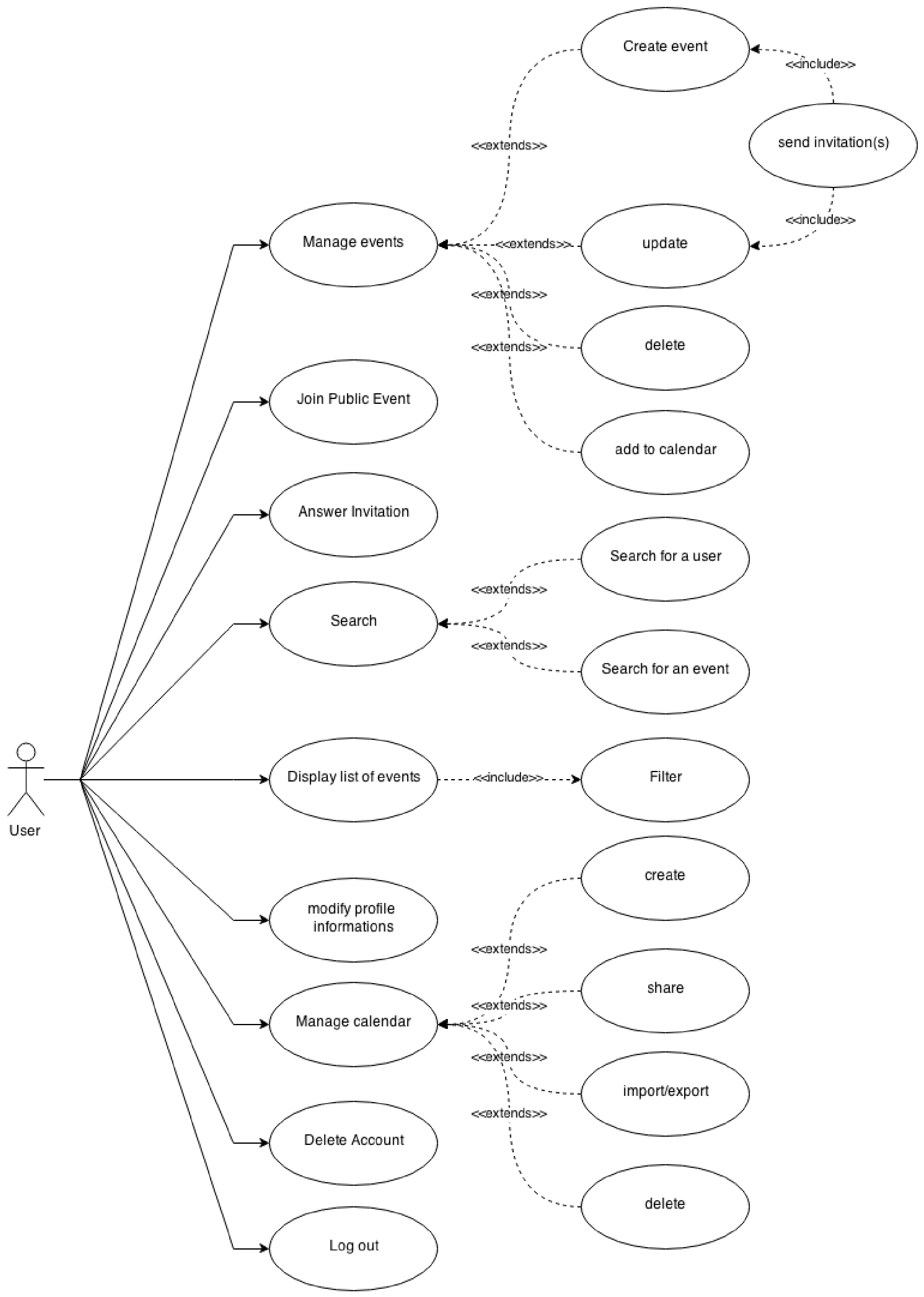
We can derive a compact and meaningful use case diagram from the above scenarios. We do not include the login in any user activity because since for security reasons no one can access the web service without being logged in.



The use cases and the scenarios are formally described in the following tables:

Name	Login
Actors	Anonymous user
Entry conditions	The user browse to meteoCal webpage
Flow of Events	• The welcome page is shown to the user. • The user click on 'Log in' • The user fill in the 'Log in' form with his email and password • The user clicks on 'Log in'
Exit Conditions	The system authenticates the user and redirect him to the home page
Exceptions	There's no user corresponding to the login credential provided by the user so the systems prompts an error message and asks to try again.

Name	Sign Up
Actors	Anonymous user
Entry conditions	The user surf to the meteoCal webpage
Flow of Events	• The welcome page is shown • The user fills in the registration form accept the Terms and Conditions and clicks on 'Submit'
Exit Conditions	• The system creates a profile for the new user. • The user is redirected to his home-page(Calendar) • The system sends a confirmation email to the user
Exceptions	The user already exist in the database so the systems prompts an error message and asks to try again


User
Manage events
Create event
send invitation(s)
update
delete
add to calendar
<<extends>>
<<include>>
Join Public Event
Answer Invitation
Search
Search for a user
Search for an event
Display list of events
Filter
modify profile informations
Manage calendar
create
share
import/export
delete
Delete Account
Log out

Name	Create an event
Actors	User
Entry conditions	The user is logged in and he's on the calendar page or in the events page
Flow of Events	• The user clicks on new event • The user is redirected to the new event creation page • The user fills in all the required informations(privacy.date and time,place) • The user may invite people to the event • The user may add the event to one of his calendar • The user saves the event
Exit Conditions	• The system creates a new event and enriches it with weather forecast • The system send notifications to all the invited users • The user is redirected to event page
Exceptions	If the user specified a calendar and there is a conflict the systems asks whether to ignore it or reschedule the event. If the event is scheduled within the next three days and there's bad weather the system asks if to ignore the situation or reschedule the event

Name	Join a public event
Actors	User
Entry conditions	User is logged in and on the Events page under the Events Wall tab
Flow of Events	• The user browse the public events in the page • Selects one event and clicks on it • The system redirect the user to the event web page • The user change the participation field to ‘Participate’
Exit Conditions	The system updates the event participants list
Exceptions	• The owner of the event changes the privacy of the event from public to private just before the user can join it. • The system doesn’t let the user to express the participation to the event and display an error message ‘You have not the privileges to see the event’ / ‘You have no invitation for this private event’

Name	Answer to an invite for a private event from notifications panel
Actors	User
Entry conditions	The user is logged in and surfing on the website
Flow of Events	• The user clicks on the notification icon • The system displays the last notifications • The user selects an invitation for an event • The user is redirected to the event page • The user chooses to Participate (and add the event to his calendar) or Decline
Exit Conditions	The system updated the participants list of the event. In case the user partecipate to the event and selected a calendar to add the event to the system also add the event to that calendar. In case the user declined the invitation the event is still in the tab Invitations but under the section 'Declined'.
Exceptions	If the event is deleted just before the user clicks on 'Participate' the system cannot perform the answer to the invitation action and informs the user that the event has been deleted / doesn't exist.

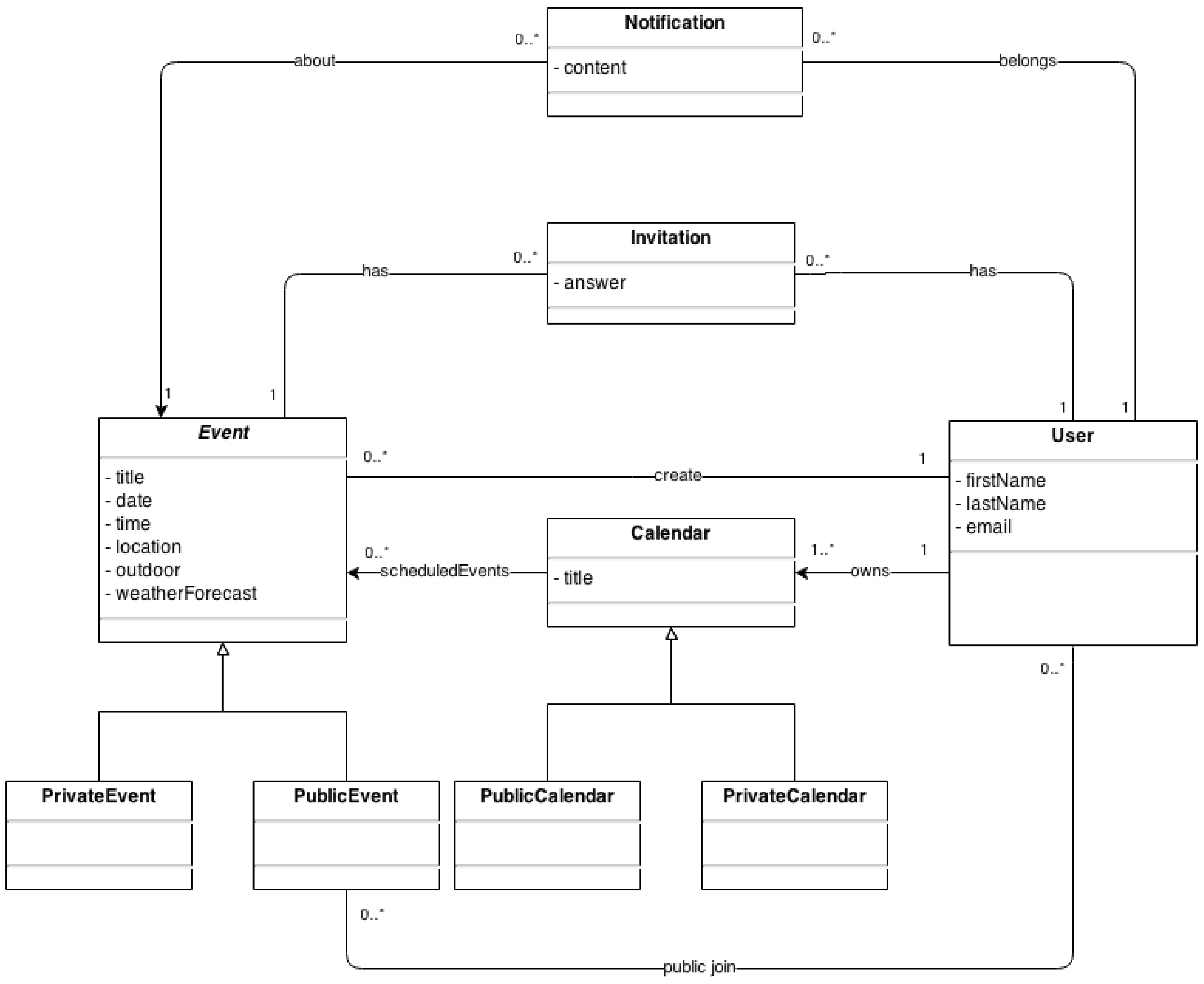
Name	Answer to an invite for a private event from email
Actors	User
Entry conditions	User is not logged in and receives an email for an invite
Flow of Events	• The user receives an email with the link of the event he was invited and click on it • The user is redirected to meteoCal welcome page • The user logs in • The user is redirected to the event page • The user chooses to Participate (and add the event to his calendar) or Decline
Exit Conditions	The system updated the participants list of the event. In case the user partecipate to the event and selected a calendar to add the event to the system also add the event to that calendar. In case the user declined the invitation the event is still in the tab Invitations but under the section 'Declined'.
Exceptions	If the event is deleted just before the user clicks on 'Participate' the system cannot perform the answer to the invitation action and informs the user that the event has been deleted / doesn't exist.

Name	Display other user public calendar (includes search for the user)
Actors	User
Entry conditions	The user is logged in and he's in any page
Flow of Events	• The user clicks on the search field and insert the name/email of the user he wants to find, then he hits Enter • The system reload the page and displays the results • The user chooses the user he was looking for and clicks on it
Exit Conditions	• The system displays the user public calendar • The user can click on the drop-down menù on the page and select an other public calendar to visualize
Exceptions	The system find no results compatible with the user's search (no target user found or no public calendar available from that user), so it displays an error message

Name	Create a calendar
Actors	User
Entry conditions	The user is logged in and in the Calendar page
Flow of Events	• The user clicks on the button 'New calendar • The user is redirected to the new calendar creation page • The user fills in all the required fields • The user clicks on 'Create'
Exit Conditions	• The system creates the new calendar • The user is redirected to the Calendar page
Exceptions	The calendar cannot be created (system checks fail like 'already existing name') and the systems displays the error message and asks the user to modify the wrong fields

Name	Delete a calendar
Actors	User
Entry conditions	The user is logged in and in the Calendar page
Flow of Events	• The user clicks the Delete icon • The system warns the user that the deletion will be permanent and asks if the user wants to export the calendar before deleting it • The user rejects the advice
Exit Conditions	• The selected calendar is deleted by the system • The events in it are cancelled(if the user was owner) and deleted (see specifications).
Exceptions	The calendar is the only one owned by the user, the system cannot delete it but advise the user to use the clean function.

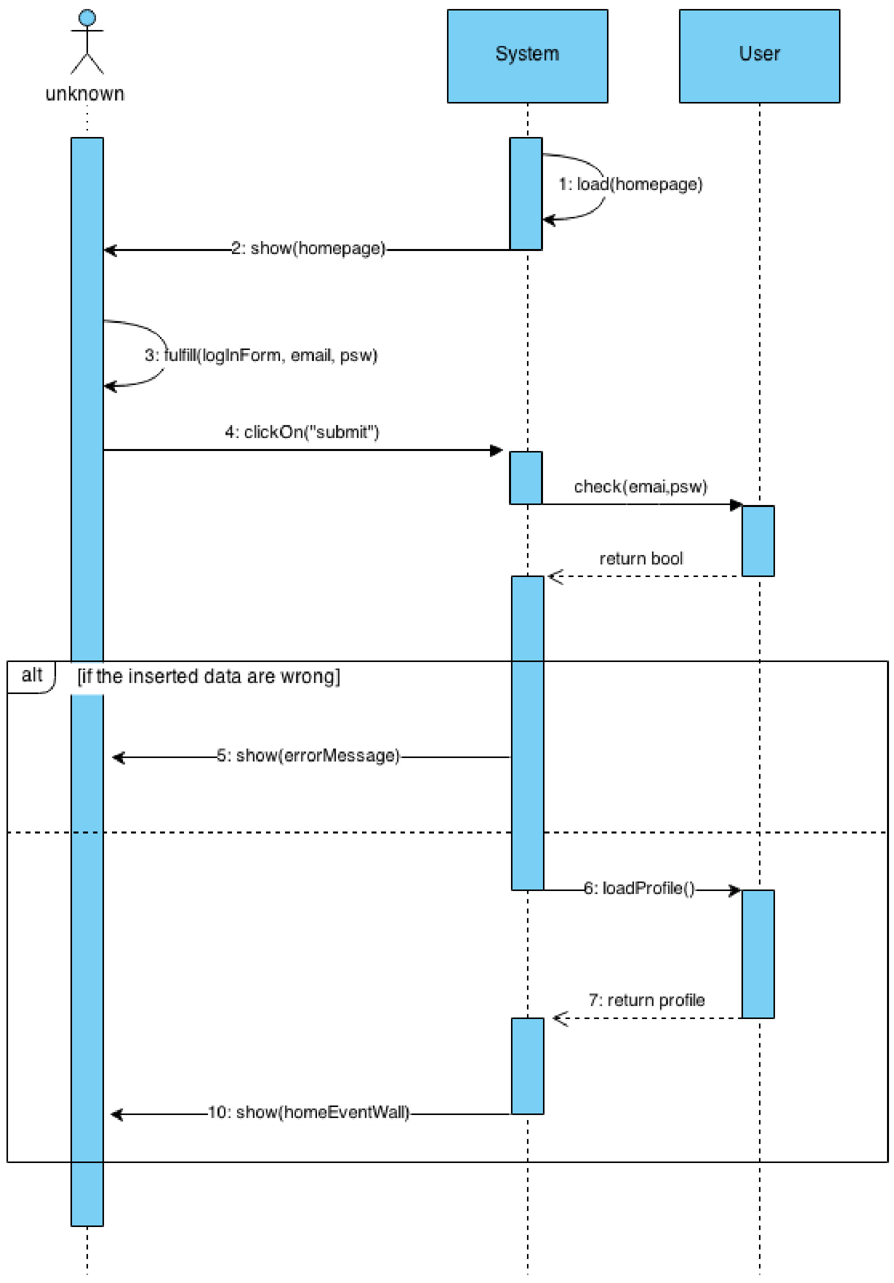
4.4 Class Diagram



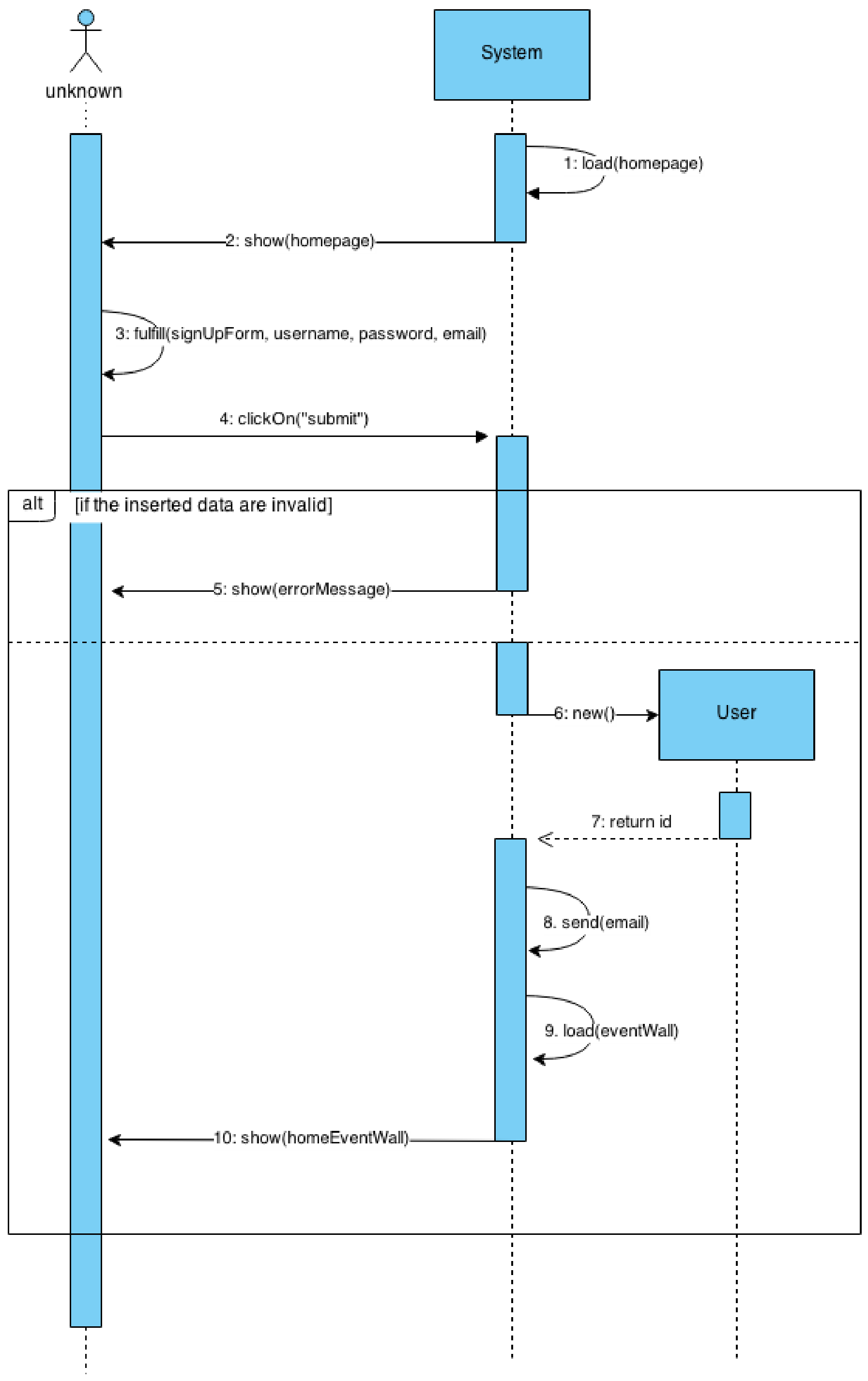
4.5 Sequence Diagram

To better explain the behavior of the system we insert the sequence diagrams of the main and more complex interactions between the actors and the system.

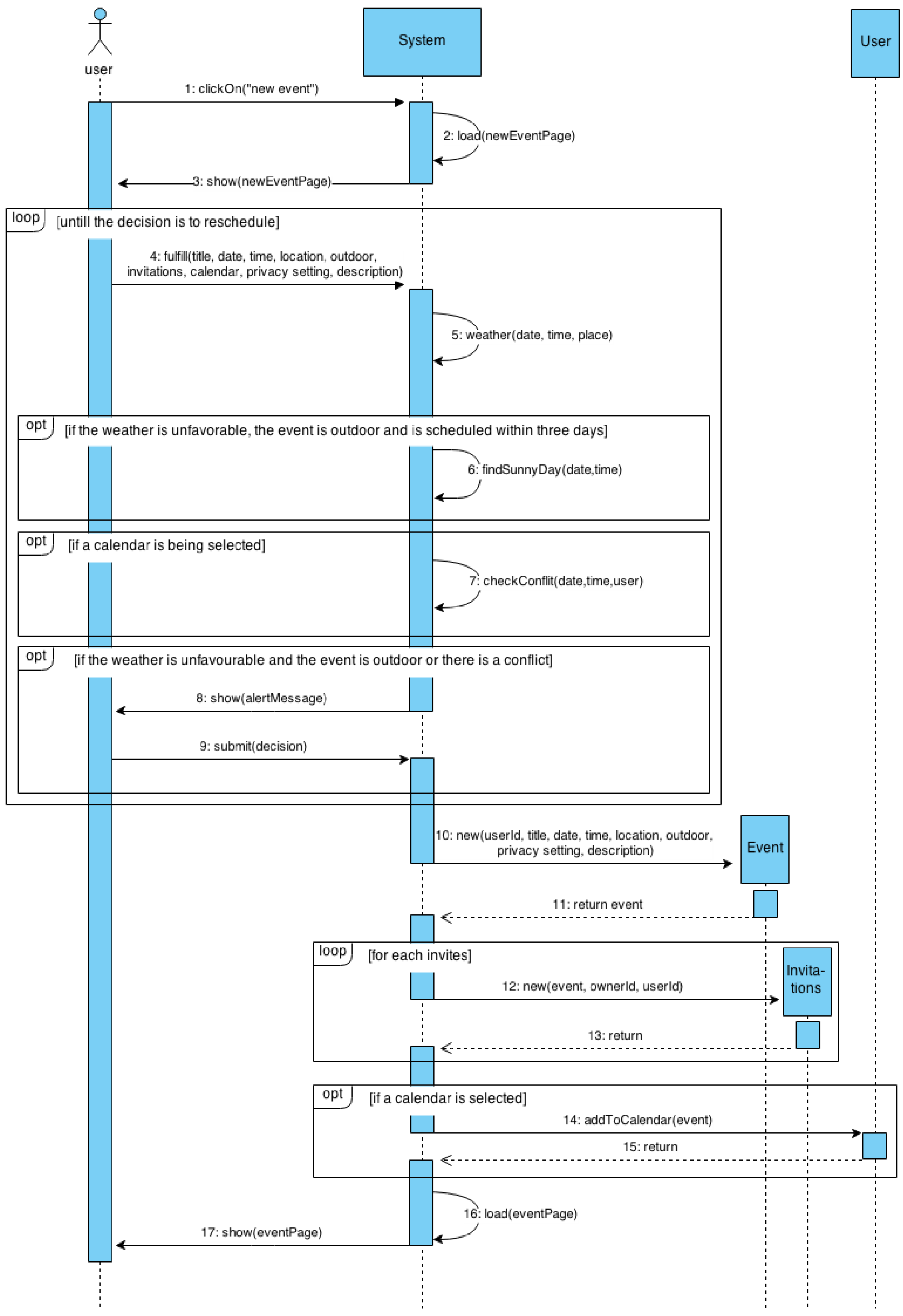
Login



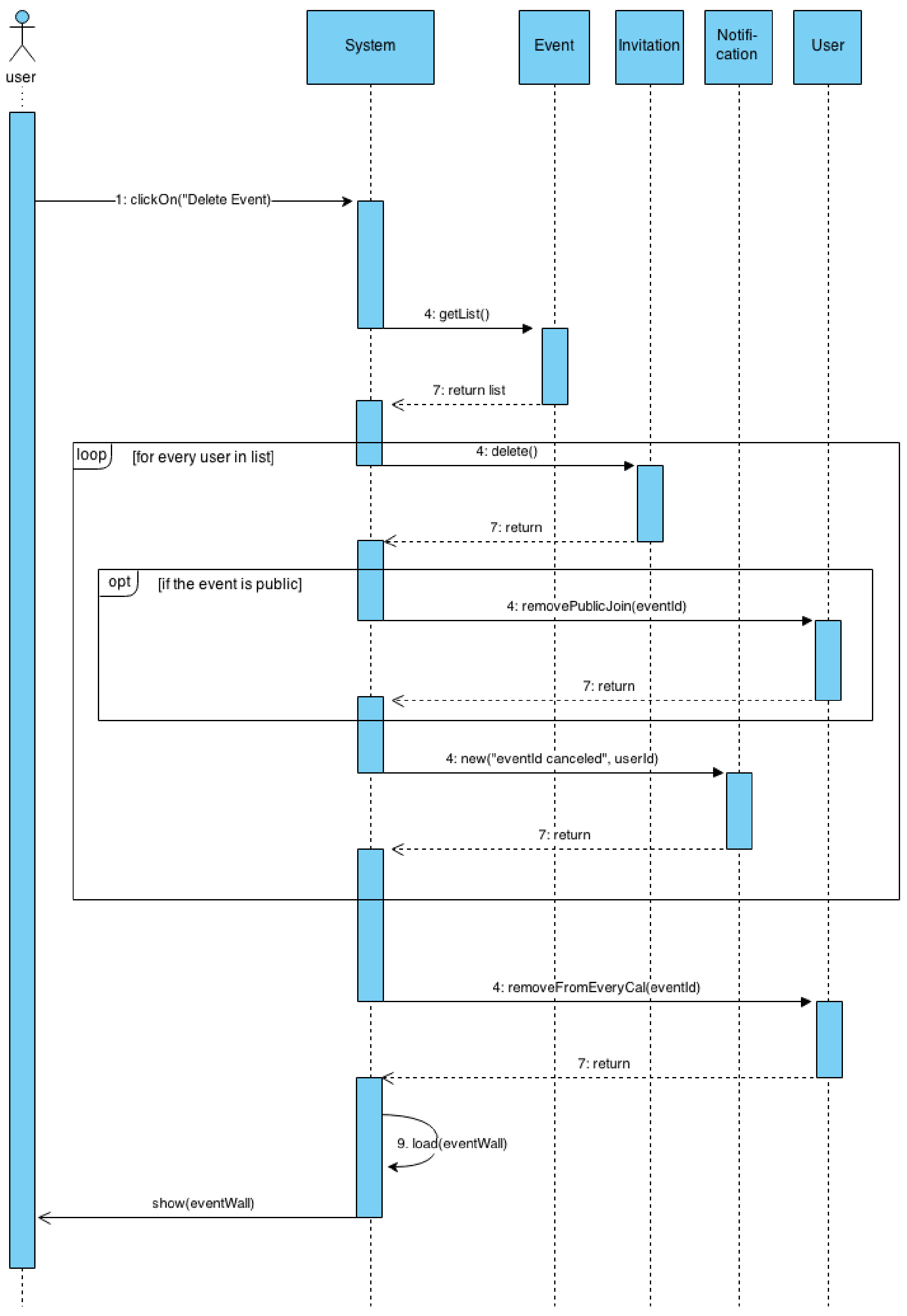
SignUp


unknown
System
1: load(homepage)
2: show(homepage)
3: fulfill(signUpForm, username, password, email)
4: clickOn("submit")
alt
[if the inserted data are invalid]
5: show(errorMessage)
6: new()
User
7: return id
8. send(email)
9. load(eventWall)
10: show(homeEventWall)

New Event

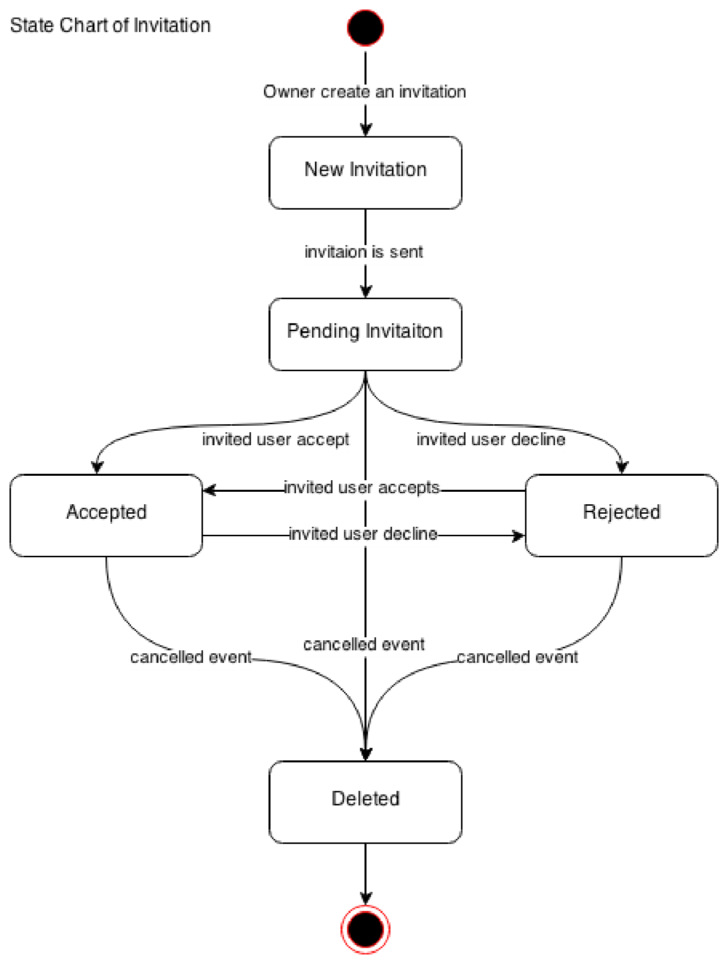

user
System
User
1: clickOn("new event")
2: load(newEventPage)
3: show(newEventPage)
loop
[untill the decision is to reschedule]
4: fulfill(title, date, time, location, outdoor, invitations, calendar, privacy setting, description)
5: weather(date, time, place)
opt
[if the weather is unfavorable, the event is outdoor and is scheduled within three days]
6: findSunnyDay(date,time)
opt
[if a calendar is being selected]
7: checkConflit(date,time,user)
opt
[if the weather is unfavourable and the event is outdoor or there is a conflict]
8: show(alertMessage)
9: submit(decision)
10: new(userId, title, date, time, location, outdoor, privacy setting, description)
Event
11: return event
loop
[for each invites]
Invita-tions
12: new(event, ownerId, userId)
13: return
opt
[if a calendar is selected]
14: addToCalendar(event)
15: return
16: load(eventPage)
17: show(eventPage)

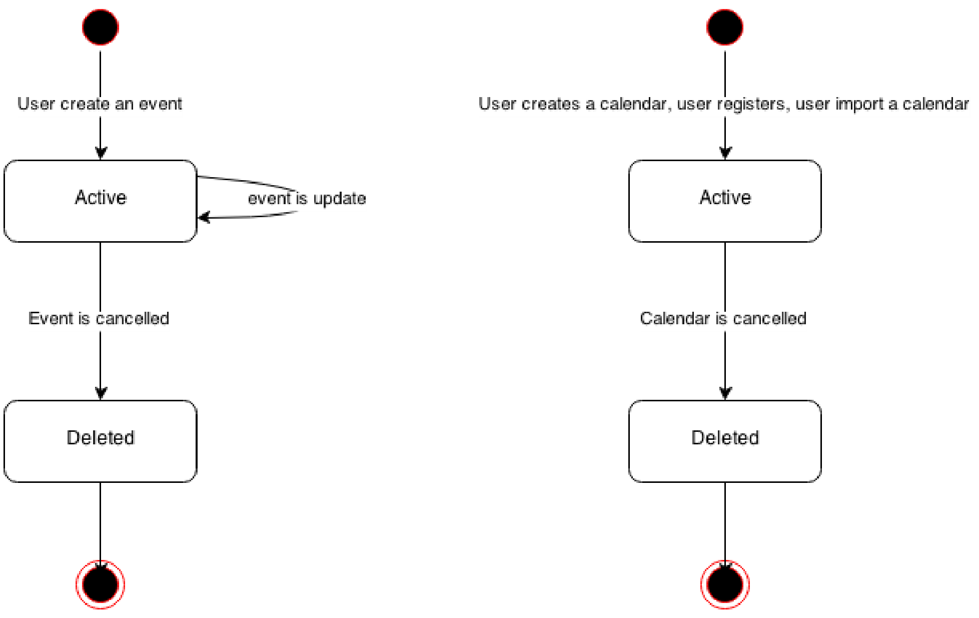
Delete Event



4.6 State chart Diagram

In this section we present the state chart of the Invitation, the Event and the Calendar.

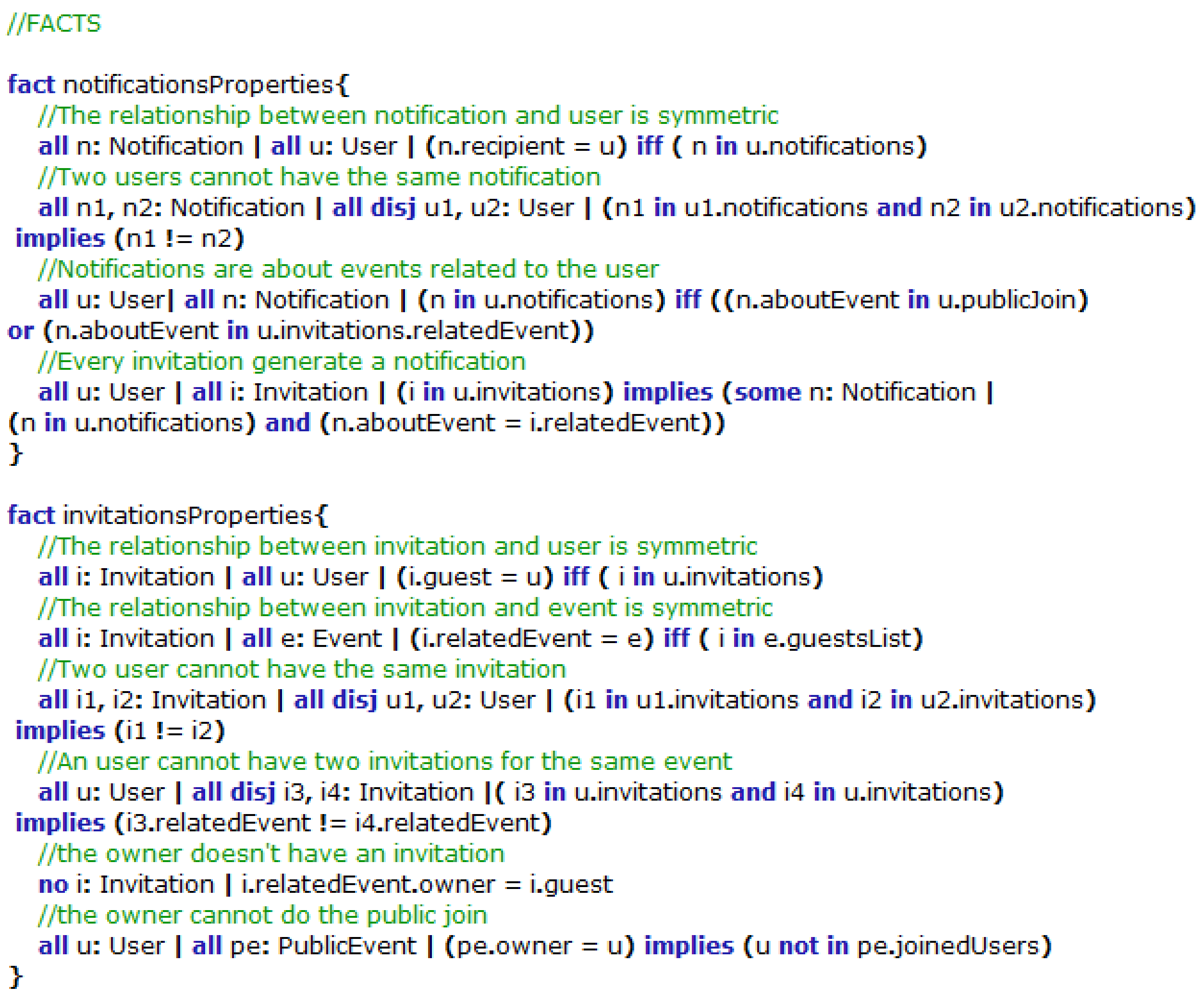




4.7 Alloy

4.7.1 Code of the model

```
module MeteoCal

//SIGNATURES

abstract sig Answer {}

one sig Accept extends Answer{}
one sig Decline extends Answer{}

abstract sig Event{
    owner: one User,
    guestsList: set Invitation
}

sig PrivateEvent extends Event{
}

sig PublicEvent extends Event{
    joinedUsers: some User
}

sig User{
    calendars: set Calendar,
    invitations: set Invitation,
    ownedEvents: set Event,
    publicJoin: set PublicEvent,
    notifications: set Notification
}{
    #calendars >0
}

sig Calendar{
    scheduledEvents: set Event
}

sig Invitation{
    guest: one User,
    relatedEvent: one Event,
    answer: lone Answer
}

sig Notification{
    recipient: one User,
    aboutEvent: one Event
}
```

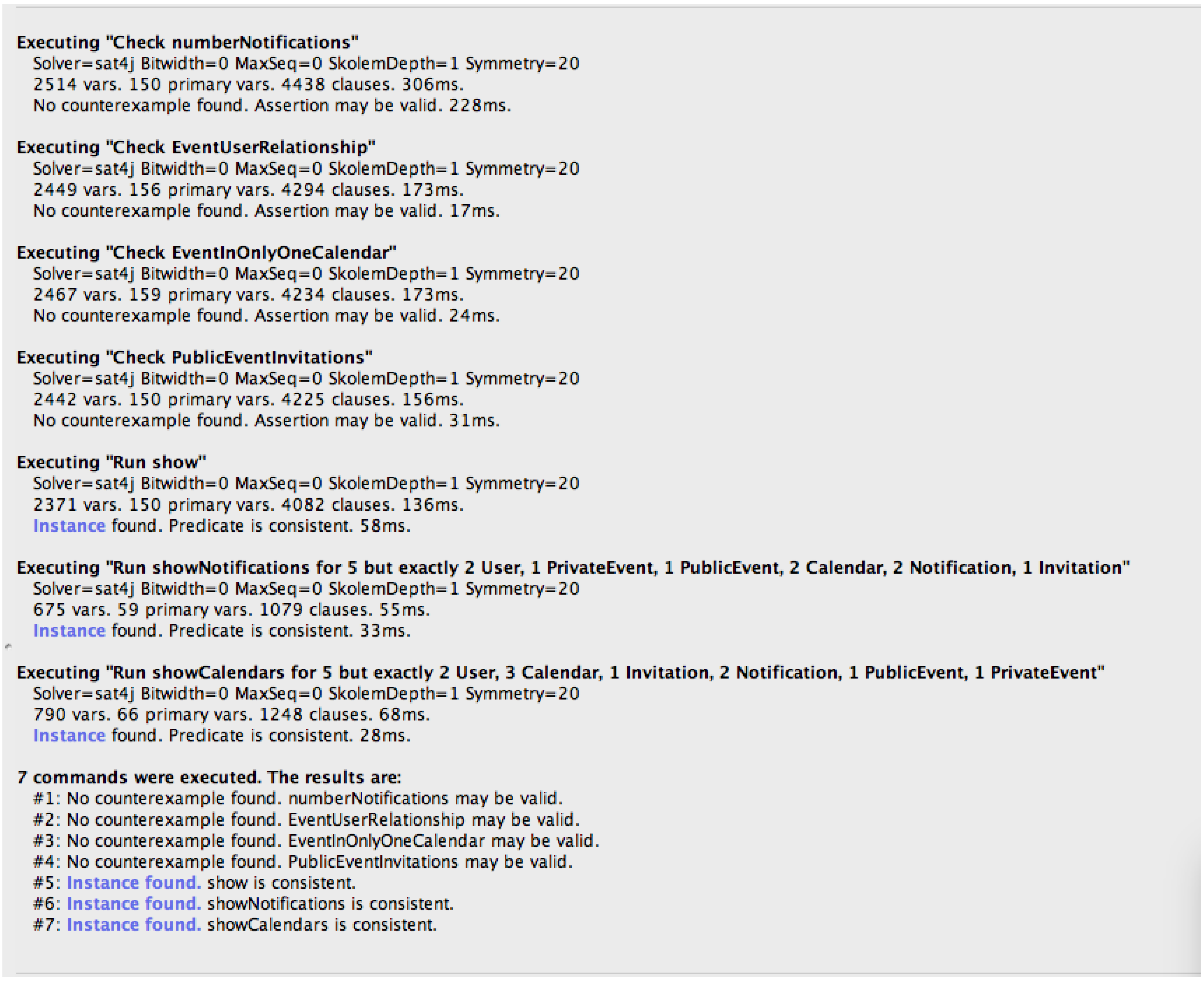


```
//FACTS

fact notificationsProperties{
   //The relationship between notification and user is symmetric
   all n: User | all u: User | (n.recipient = u) iff ( n in u.notifications)
   //Two users cannot have the same notification
   all n1, n2: Notification | all disj u1, u2: User | (n1 in u1.notifications and n2 in u2.notifications)
 implies (n1 != n2)
   //Notifications are about events related to the user
   all u: User| all n: Notification | (n in u.notifications) iff ((n.aboutEvent in u.publicJoin)
or (n.aboutEvent in u.invitations.relatedEvent))
   //Every invitation generate a notification
   all u: User | all i: Invitation | (i in u.invitations) implies (some n: Notification |
(n in u.notifications) and (n.aboutEvent = i.relatedEvent))
}

fact invitationsProperties{
   //The relationship between invitation and user is symmetric
   all i: Invitation | all u: User | (i.guest = u) iff ( i in u.invitations)
   //The relationship between invitation and event is symmetric
   all i: Invitation | all e: Event | (i.relatedEvent = e) iff ( i in e.guestsList)
   //Two user cannot have the same invitation
   all i1, i2: Invitation | all disj u1, u2: User | (i1 in u1.invitations and i2 in u2.invitations)
 implies (i1 != i2)
   //An user cannot have two invitations for the same event
   all u: User | all disj i3, i4: Invitation |( i3 in u.invitations and i4 in u.invitations)
 implies (i3.relatedEvent != i4.relatedEvent)
   //the owner doesn't have an invitation
   no i: Invitation | i.relatedEvent.owner = i.guest
   //the owner cannot do the public join
   all u: User | all pe: PublicEvent | (pe.owner = u) implies (u not in pe.joinedUsers)
}
```

```
fact eventsProperties{
   //The relationship between user and event is symmetric
   all e: Event |all u: User | (e.owner = u) iff (e in u.ownedEvents)
   //The relationship between User and PublicEvent is symmetric
   all pe: PublicEvent | all u: User | (u in pe.joinedUsers) iff (pe in u.publicJoin)
   //If a user joined a public event that user has not an invitation
   all pe: PublicEvent | all u: User | (u in pe.joinedUsers) implies
( not (pe in u.invitations.relatedEvent))
}

fact calendarProperties {
   //Each event that a user participate to, must be at most in one calendar
   all e: Event | all u: User | partecipate[u,e] implies
 (lone c: Calendar | c in u.calendars and e in c.scheduledEvents)
   //If an event is in a calendar, the user must be partecipating at it
   all e: Event | all u: User | all c: Calendar |(c in u.calendars and e in c.scheduledEvents)
 implies partecipate[u,e]
   //A calendar is owned by exactly one user
   all u1, u2: User | all c: Calendar | (c in u1.calendars and c in u2. calendars)
 implies u1 = u2
   //All calendar has an owner
   all c: Calendar | some u: User | c in u.calendars
}

//SUPPORTING PREDICATES

pred partecipate[u: User, e: Event]{
   //the creator of an event necessarily partecipate to it etc..
   (some i:Invitation  | i in e.guestsList and (i.answer=Accept) and i.guest = u)
 or (e in u.publicJoin) or (u = e.owner)
}
```

```
//ASSERTIONS

assert numberNotifications {
   //for an event, the number of notification is always greater or equal to the number of guest
   all n: Notification | all e: Event | (n.aboutEvent = e) implies ( #n >= #e.guestsList)
   //for an user, the number of notification is greater or equal to the number of invitation
   all u: User | #u.notifications >= #u.invitations
}
check numberNotifications


assert EventUserRelationship {
   //an user have an invitation if and only if he is in the guestList of the event
   all u: User | all e: Event | (e in u.invitations.relatedEvent) iff ( u in e.guestsList.guest)
}
check EventUserRelationship

assert EventInOnlyOneCalendar {
   //two calendar of the same user cannot have same events
   all u: User | all disj c1,c2: Calendar | (c1 in u.calendars and c2 in u.calendars)
 implies (c1.scheduledEvents&c2.scheduledEvents = none)
}
check EventInOnlyOneCalendar

assert PublicEventInvitations {
   //if the user is invitated to a public event he cannot do public join to it
   all u: User | all e: PublicEvent | (e in u.invitations.relatedEvent) implies (not u in e.joinedUsers)
   //intersection between guests (users with invitation) and joined user is empty
   all pe: PublicEvent | pe.guestsList.guest&pe.joinedUsers = none
}
check PublicEventInvitations



pred show(){}
run show
//run show for 4 but exactly 4 Invitation,4 PublicEvent
//PREDICATES
pred showNotifications (){}
   //Notifications are about events related to the user
run showNotifications for 5 but exactly 2 User, 1 PrivateEvent,
1 PublicEvent, 2 Calendar, 2 Notification, 1 Invitation

pred showCalendars() {}
   //If an event is in a calendar, the user must be partecipating at it
run showCalendars for 5 but exactly 2 User, 3 Calendar, 1 Invitation,
 2 Notification, 1 PublicEvent, 1 PrivateEvent
```

4.7.2 Results



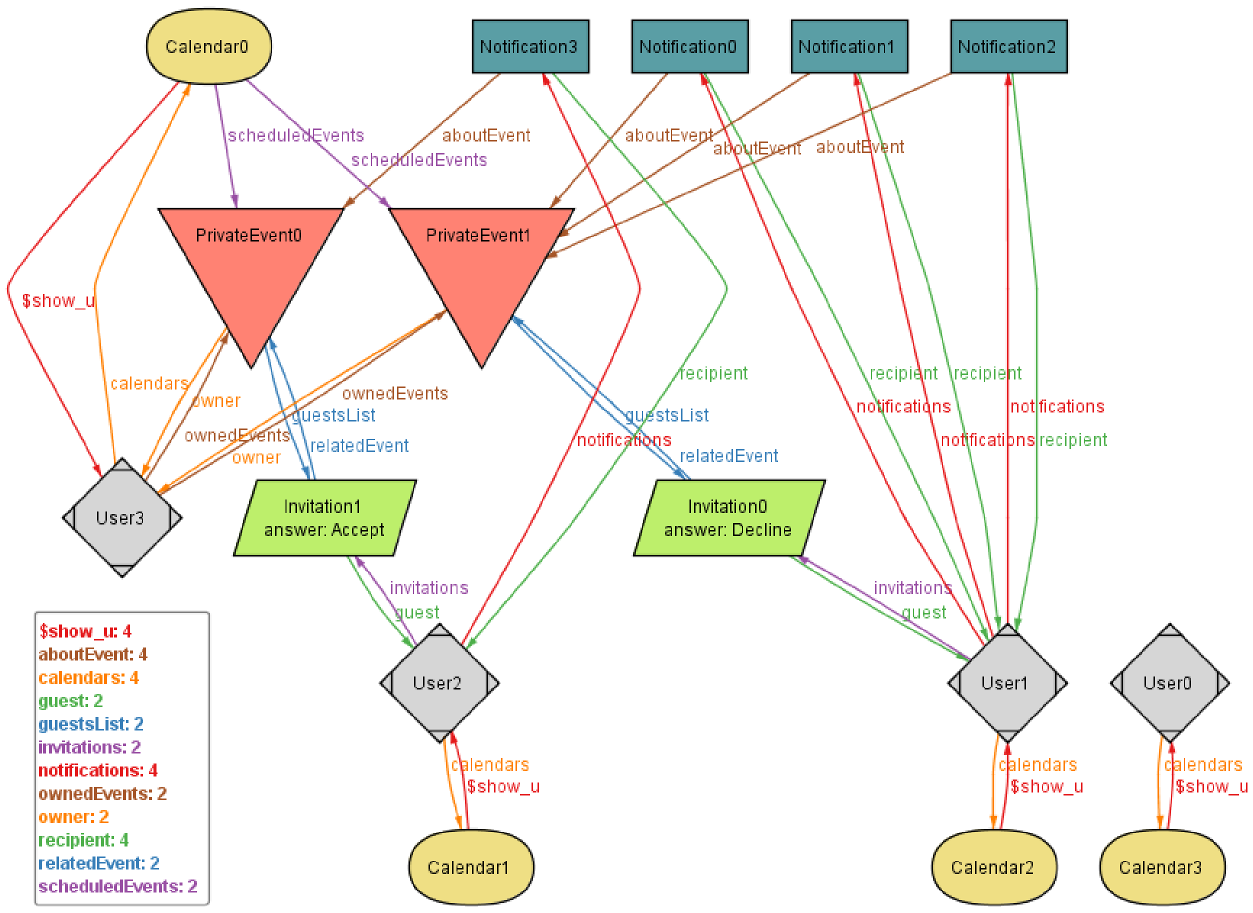
```
Executing "Check numberNotifications"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   2514 vars. 150 primary vars. 4438 clauses. 306ms.
   No counterexample found. Assertion may be valid. 228ms.

Executing "Check EventUserRelationship"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   2449 vars. 156 primary vars. 4294 clauses. 173ms.
   No counterexample found. Assertion may be valid. 17ms.

Executing "Check EventInOnlyOneCalendar"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   2467 vars. 159 primary vars. 4234 clauses. 173ms.
   No counterexample found. Assertion may be valid. 24ms.

Executing "Check PublicEventInvitations"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   2442 vars. 150 primary vars. 4225 clauses. 156ms.
   No counterexample found. Assertion may be valid. 31ms.

Executing "Run show"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   2371 vars. 150 primary vars. 4082 clauses. 136ms.
   Instance found. Predicate is consistent. 58ms.

Executing "Run showNotifications for 5 but exactly 2 User, 1 PrivateEvent, 1 PublicEvent, 2 Calendar, 2 Notification, 1 Invitation"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   675 vars. 59 primary vars. 1079 clauses. 55ms.
   Instance found. Predicate is consistent. 33ms.

Executing "Run showCalendars for 5 but exactly 2 User, 3 Calendar, 1 Invitation, 2 Notification, 1 PublicEvent, 1 PrivateEvent"
   Solver=sat4j Bitwidth=0 MaxSeq=0 SkolemDepth=1 Symmetry=20
   790 vars. 66 primary vars. 1248 clauses. 68ms.
   Instance found. Predicate is consistent. 28ms.

7 commands were executed. The results are:
   #1: No counterexample found. numberNotifications may be valid.
   #2: No counterexample found. EventUserRelationship may be valid.
   #3: No counterexample found. EventInOnlyOneCalendar may be valid.
   #4: No counterexample found. PublicEventInvitations may be valid.
   #5: Instance found. show is consistent.
   #6: Instance found. showNotifications is consistent.
   #7: Instance found. showCalendars is consistent.
```

4.7.3 World generated

The alloy analyzer has allowed us to improve our model, reasoning about constraints and behaviors upon entities and relationships. Generating possible worlds with the tool, we have studied and checked the consistency of the class diagram.

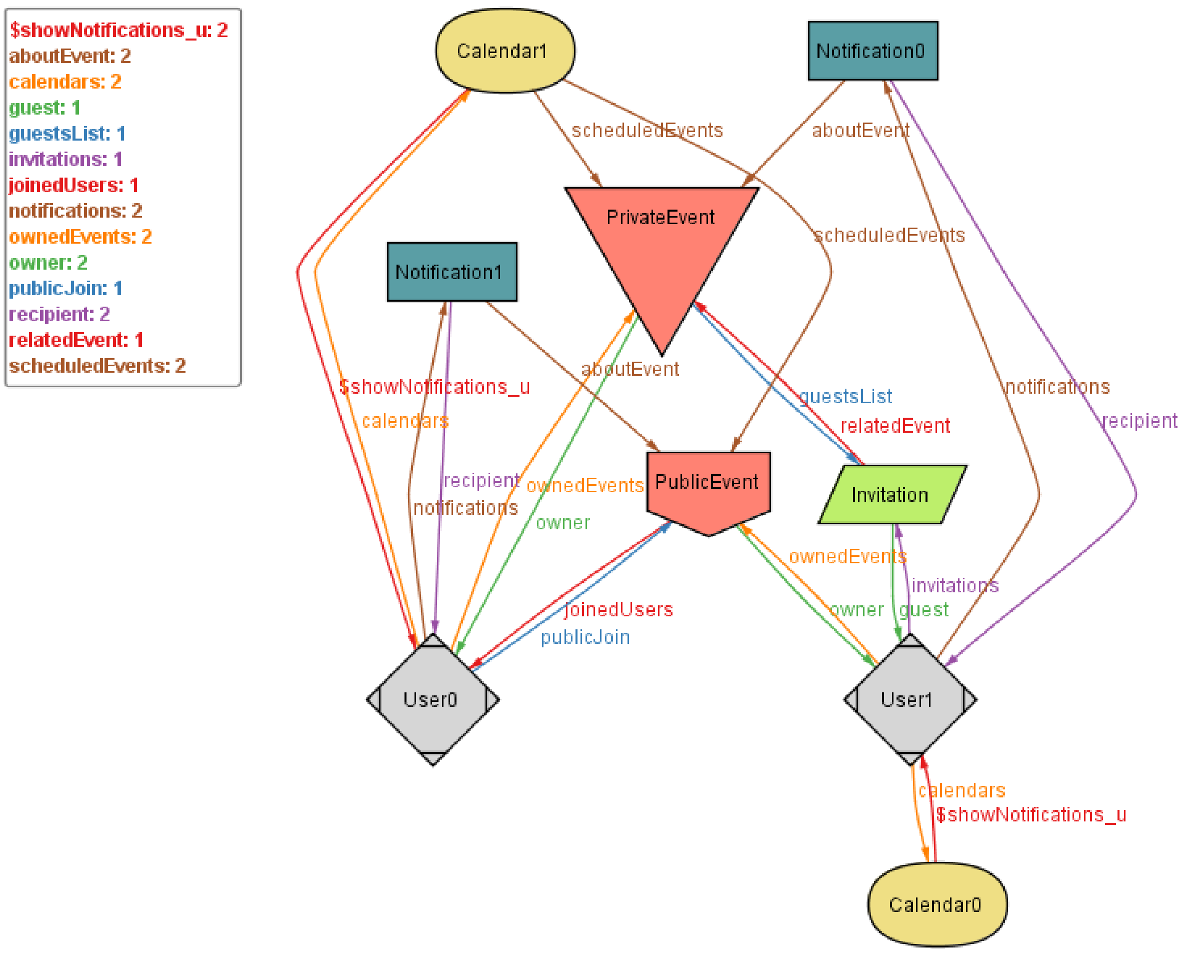
At first glance, in a general example of a world, like the one in the picture below, it can be noticed, for instance, that:

- To each users correspond at least one calendar
- For each invitations to a user correspond one notification for the same user
- Each user has at maximum one invitation for the same event
- The owner of an event can't do the public join to it or has an invitation for it
- Only events with accepted invitation or owned are in the user's calendar



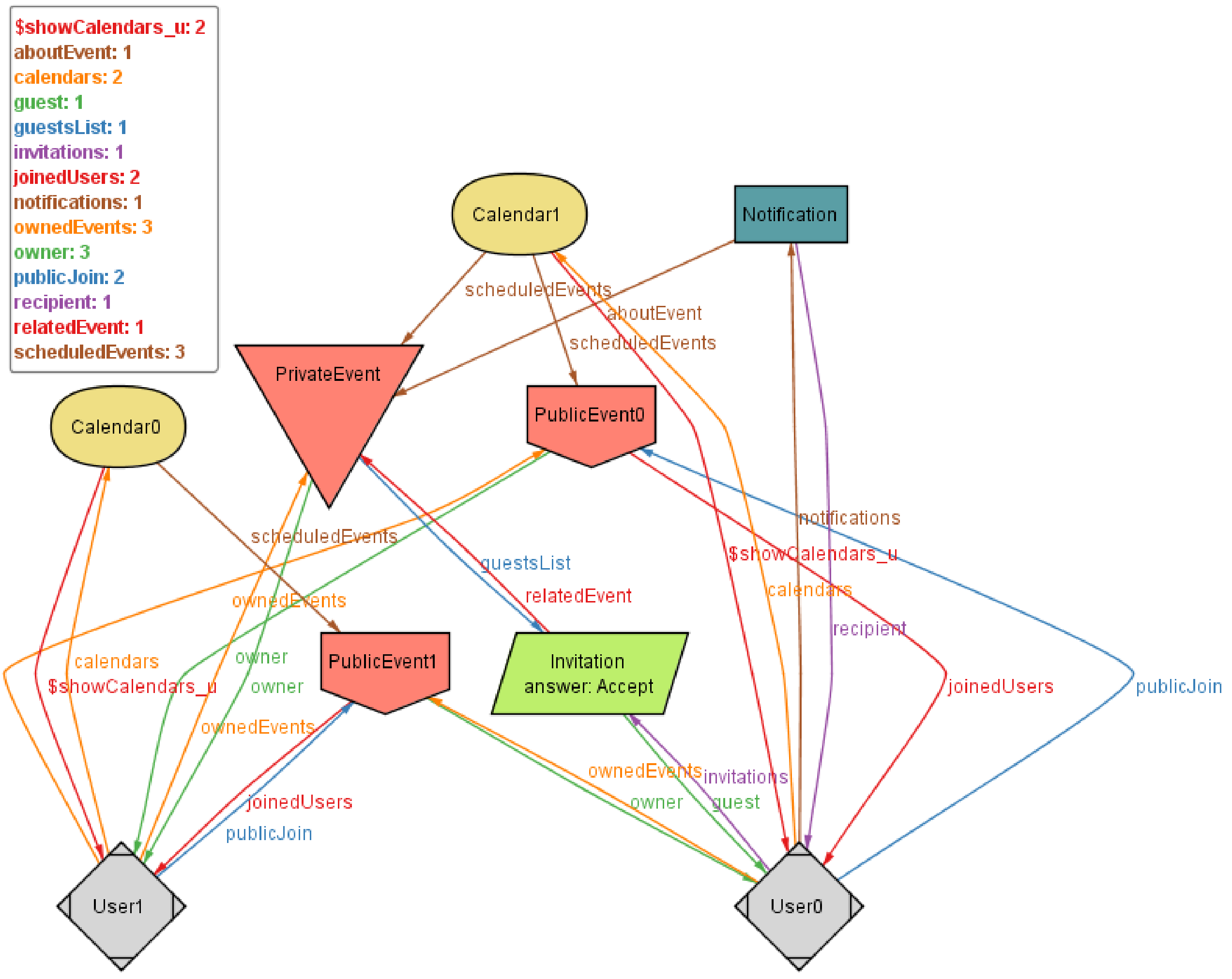
Only meaningful notification for the user

More interesting observations can be done on specific examples. In the following picture it's more evident that in our model if a user has a notification of an event, he either has an invitation for that event or has done public join for it



Only interesting event into a calendar

At last, another possible observation about the model is that in a calendar there are only user's events or events to whom he wants to participate (answering to an invitation or public joining)



4.8 Tools

- Draw.io: used for graphs editing
- Alloy tool version 4.2: to create the alloy model and the worlds
- Balsamiq Mockups: to create mockups of the web pages
- TexMaker Latex editor: to edit this pdf file

5 Time spent working on this document

Group Member	Total Hours
Francesco Angelo	43
Valentina Ceriani	45
Umberto Di Fabrizio	41

6 Final notes

In the implementation phase we decided to omit the functionality that lets the user change the privacy of his event. Although the study is corrected and the specifications about this functionality are still valid the privacy change was not a direct requirement for this project, so we left it out and planned to release this improvement in future versions of our system.