Politecnico di Milano

Software Engineering 2

MeteoCal

RASD Document

Authors:

Andrea Enrico Turri

Andrea Salmoiraghi

Fabiano Riccardi

16/11/2014

Contents

1 Introduction

1.1 Description of the given problem

We will project and implement MeteoCal, a weather based online calendar for helping people scheduling their personal events avoiding bad weather conditions in case of outdoor activities.
The system that will be developed has to allow the registration of new users and allow them to create, update, delete events, with the possibility to invite other users that can accept or decline the invitation. Events should contain information about when and where the event will take place, whether the event will be indoor or outdoor.
The system has also some other features such as notifications to participants in case of bad weather, import / export of calendars, possibility to make one's calendar or single events public, purpose closest sunny day when registering a new event, avoiding conflicts with existing events.

1.2 Scope

The software-to-be will be developed as a web based application that will let the users create, update, delete their events, with notifications about the weather and inviting other people.

1.3 Glossary: definitions

In this section we define some words that we will use in the documentation of the project.

- **Event**
  An event is a plan in the user's calendar for something that will occur in a certain place during a particular interval of time. An event is characterized by a place, a starting and end time, a flag for outdoor / indoor, a title, a list of participating users.

- **Calendar**
  A calendar is a sort of wall, a notice-board where there are dates and events where the user will participate.

- **Invitation**
  An invitation is a request sent from the user that has organized the event to another user. It can be accepted by the receiver, stay pending until the time of the event.

- **User**
  For user we mean a person who is registered in the system, a user has a calendar, can create events and can use all the functionalities described in the appropriate section of this document.

- **Organizer**
  With reference to an event, the organizer is the user that created it.
- **Invited user**
  With reference to an event, a user that received an invitation to join it.
- **Participant user**
  With reference to an event, an invited user that accepted the invitation.
- **Notification**
  A notification is something, such as an highlighted box in the system, by which notice is given.
- **Email notification**
  A notification sent as an email.
- **Username**
  The identification of a user, used to log into the system and retrieve password.
- **Wall**
  An area where notifications appear, where there is the user's calendar and it is possible to browse other users' calendars.
- **Profile**
  Every user has a profile, an area where are shown user's public events and public calendar.
- **Contact**
  For a user A, a contact is another user B that is in A's address book.
- **Address Book**
  The address book is an area where are listed all the user's contacts.

1.4 Stakeholders

Our ”financial” stakeholder is the professor who gave us this didactical project. Our professor needs are to have, within the end of the semester, a product that at least works and we think the main thing that interests the professor is to show that we have understood the development process in all its parts and that we can carry out a project from the beginning to the end passing from all its development phases. So we want to show that we can identify requirements and specifications, design our web application, implement it and then test it, providing all the documents that developers use in developing real software.

The software house X is the company that ”virtually” gave us this project to propose a new kind of service. X needs to have a product that works fine and covers all the specification given to us.

We think that our typical users can be:

- occasional users that sometimes can use this tool to plan outdoor events to share with other people
- habitual users that might find this tool useful for example for work purposes.

2 Overall description

2.1 Goals

MeteoCal should provide these main features:

- Registration of a person to the system
- Creation of a new event with the possibility to choose the best place / time w.r.t. the weather conditions
- Browse other users' public calendars and events
- Import and export of calendars
- Making their calendar public / private
- Making their events public / private
- Accept / decline other users' events
- To be notified whenever there will be bad weather conditions
- Update their events
- Delete their events
- Invite other users to join their events

2.2 Domain properties

We suppose that these conditions hold in the analyzed world:

- The weather forecast is supposed to be reliable enough to indicate the right weather in the selected place and time
- The organizer of the event will participate to the event
- If a user accepts an invitation to join an event, it is supposed to participate
- If a user declines an invitation to join an event, it is supposed not to participate

2.3 Assumptions

There are few points that could result not very clear in the specification document, so we will have to assume some facts:

- By default, when a user creates an event, this is private. The user can make it public.

- By default, a user has only one calendar which is private. The user can make it public, so that other users can see only available and 'busy' time slots, but not the detail of the single events, unless they are public.
- If a user doesn't answer to the invitation, this will remain suspended until the end time of the event. After the end time of the event, the user will not be able neither to accept nor to decline.
- When the organizer is inviting someone, it is possible to insert the email address of the users to invite which is taken as username by the system.
- Users that forgot their password, can recover it asking the system to send it on their email address.
- The calendar exported from the system is in a custom format and it is possible to import a calendar only if it is in the system's custom format. It is possible to export only the calendar of the user, not the ones of others.
- When the organizer updates an event or deletes it, all participant users will be notified.
- A user has associated a profile, which can be searched through a search field.
- As there is the possibility to browse other users' calendars, for us it wasn't so clear if there was a wall in which a user can see calendar of some other random user or only of some "friends", so we decided to define a way to see other users's calendar in the wall and to create an Address Book:
  - *A user can see in the Wall the public calendars and public events only of their contacts*
  - *A user can browse the public calendars and public events of other users that are not contact, through a search field*
  - *A contact can be added by typing its email address in the Address Book or it is automatically added when a user which is not a contact is invited to join an event*

It is important to say that the points in italic are advanced functionalities that we would like to implement to have a more complex system, but, before we want to carry out all the basic set of functionalities in order to have a complete product as requested.
Just to be ready for a possible implementation in the project, we'll carry on these functionalities in all the project's phases. We hope to have enough time to reach this goal.

2.4 Proposed system

We propose a web based platform that will give to registered people the services described below.
Users will be able to have their own calendar, to create events, include weather

details in case of outdoor events, receive notifications about bad weather forecasts, browse other's public calendars, import or export calendars, decide to make their events and/or calendar public, accept/decline invitations. Organizers can also update, delete their events and invite other users to join it.
Server will generate different user experiences and so different pages, basing on the actor that is currently using the platform.

2.5 Other considerations about the system

We want to add also other considerations because they are about our thoughts about how MeteoCal should be once developed:

- Easy to use: we will make every effort to create a user interface that allows also new users to be able to use it without using any manual, also the first time. In addition we want that users don't spend too much time to create / update / delete events or join them.

- Nearly stable: we want to guarantee basic functionalities in a stable way, we don't want do have any error or fault in every phase of the processes carried out by the system.

- Have a nice "look and feel": we will concentrate also on this point which we hardly believe that is very important, because, as said by Steve Jobs, *"Design is not just what it looks like and feels like. Design is how it works"*. We will try to make a well designed application, nice to see and that can fit any user, keeping everything as much simple as possible.

3 Actors Identifying

Registered users, organizers: their goal is to create events, both private or public, eventually invite other people, update them or remove them, also on the basis of the weather forecasts.

Registered users, invited: they can decide to accept or decline an invitation sent by some organizer user. If they accept, they will have the event on their calendar and they can see updates and receive notifications.

Registered users: they can browse the public calendars of users that decided to make their calendar public, they can create events (they become organizers) or they can be invited to join other?s events (they become invited).

4 Requirements

Analyzing goals and assumptions described in the previous chapters, we can derive the requirements, that is what our system will be able to perform.

1. **Registration of a person into the system**
   - The system will provide the classical registration form and login through username and password.
   - The system will allow also to recover forgotten password.
2. **Events management**
   - The system will allow a user to create an event, suggesting also the first sunny day for the specified location, if it will take place outdoor. Then the system enables only this user to edit or remove the event. The organizer can also invite other users to the event and set the visibility to other users, public or private.
   - The system will provide also the possibility to search public events, then their details.
   - The system will allow the users to join the events for which they received an invitation. The user will also be able to refuse the invitation.
   - The system will apply the following privacy policy for events:

     Private event with user's calendar private The event can be accessed only by participating users.

     Private event with user's calendar public The event can be accessed only by participating users: they can see details and participating users.

     Public event The event can be seen by every user: they can see details and participating users.
3. **Calendar management**
   - The system will provide to every user one calendar. A calendar can be public or private. In the last case can be seen by other users the periods in which the user is free or busy, event details are hidden, unless the single events are public.
   - The system will also allow to import or export the user's calendar, which will be in a custom format of file.
4. **Users management**
   - The system will allow the users to search for other users, add a user to the Address Book, visualize users? profiles and send invitations.
   - The system will provide a search field to find users by email.
5. **Notifications**

- The system will send notifications to users participating to an event if this is updated or deleted by the organizer, or if weather forecasts say that there it will rain in case of outdoor events. These notifications are visualized in the system when the user logs in and they are also sent as an email.
- The system will send email notifications to users invited to join an event.

4.1 Functional requirements

After analyzing the features that the system will provide, we can define functional requirements for each actor.

- **Guest**: he can only access the basic functionalities:
  - Sign up
  - Browse homepage
- **Registered user**:
  - Log in
  - Recover password
  - Create an event
  - Search public events
  - Search public calendars (users' profiles)
  - Make the calendar public or private
  - Import a calendar
  - Export the calendar
  - Add users to Address Book
  - Receive notifications
- **Organizer**:
  - Update the event
  - Delete the event
  - Invite other users to the event
  - Make the event public or private
- **Invited user**
  - Accept the invitation
  - Refuse the invitation
  - See event's details

4.2 Nonfunctional requirements

4.2.1 User Interface

The application is thought to be visualizable by the user in every moment, hence we want that it can be used also from smartphones, tablets and any browser.
One of our goals is to build a very user friendly interface, minimal and easy to use. We want that the button ”Create event” would be always visible, so as a bar where it will be possible to search (events, calendars, users) and buttons also to go to settings and logout pages.
Then we will prevent events and calendars that are private from being visualized from other users than the owners. The same for all reserved information.

...

4.2.2 Documentation

With the purpose to explain our project, we wrote the following documentation:

- **RASD**: Requirement Analysis and Specification Document, to understand better and clarify the whole project, useful also for every other stakeholder.
- **DD**: Design Document, to define the real structure of our web application and its tiers.
- **JavaDOC**: documentation to explain the code, useful for software maintenance or system updates.
- **User Manual**: for the users, it will be composed mainly by pictures, but it will also include some text in a tutorial form.
- **Installation manual**: guide for the people that will install the system on new machines, for the developers and the server administrators. The users shouldn't have any problem, since it is a Web Application which is accessible from any modern browser, without requiring any additional software.

4.2.3 Architecture

The system will need one or more dedicated servers, accessible from remote connections and running JavaEE. It will also be necessary a Database to store the users' information.
What is this information will be clarified below, looking at the class diagram, also if we will be more precise in the Design Document.
An internet connection is required to access the application.