POLITECNICO
DI MILANO

Acceptance testing document

meteocal-bcc

January 28, 2015

Manuel Grillo
Francesco Fermi
Jerry M. Jude

0. Index

Table of Contents of Acceptance testing document v1.1

The opinions expressed in this document are those of the authors, and do not necessarily reflect the opinions of the mebers of the Politecnico di Milano university.

1. Introduction

1.1. meteocal-bcc acceptance testing

This document describes the test cases that have been done during the acceptance testing of the MeteoCal project "meteocal-bcc", developed by:

- Lorenzo Bisi;
- Eric Camellini;
- Tommaso Carpi.

Please refer to *Chapter 3 – Conclusions* for the outcomes of this acceptance testing.

1.2. Overview

With the test cases (see *Chapter 2 – Test cases*) of this acceptance testing document, we are going to apprehend if the requirements provided in the Requirement Analysis and Specification Document "`RASDFinal_v2.0.pdf`" are coherently met (eventually considering the efficiency, the reliability and the usability of the overall web application).

If the outcome of this document is positive (i.e. the requirements completely fits or the differences are objectively due improvements in efficiency, reliability, or usability), it will be possible to proceed with the process that will lead to the delivery of the product to the costumer. Otherwise – if the outcome of this document is negative (i.e. the differences with respect to the requirements does not lead to concrete and objective improvements in efficiency, reliability, or usability) – it is encouraged to continue to develop and fix the problematic cases.

Obviously, the outcome of this document is valid if the assumptions described id *Chapter 2 – Assumptions* holds.

In this document the test cases will be defined as follows:

- Goal: The case that we are actually going to test (i.e. the aim of the test).
- Functionality: The functionality derived from the RASD, in its scope (i.e. the general goal that we want to satisfy and the specific requirement that – partially – aids in achieving it).
- Environment: Informally, the perceived page the test happens.
- URL: The actual URL where the test starts.
- Input: The parameters that have been set in the page during the test (i.e. the data set used for the validation of the test).
- Expected output: The output that should appear accordingly with the specifications.
- Obtained output: The actual output.
- Final output: The dynamic behavior of the web application at the end of the test.
- Possible errors: The errors that may eventually appear in the test case.
- Comments: The outcome of the test.

2. Assumptions

2.1. RASD assumptions

We assume, from the interpretation of the Requirement Analysis and Specification Document "`RASDFinal_v2.0.pdf`" that:

- In *Subsection 1.2.1 – Goals*: the goal `G2`, which states that "the owner of a calendar should be able to create, update and delete events" means that the owner of the events (i.e. the creator) should be able to create, update, or delete them.
- In *Section 3.2 – Functional requirements*: the requirement that states that an event creator can "set an event that he created as private/public and outdoor/indoor" and said to be derived from the goal `G5` in *Subsection 1.2.1 – Goals*, which states that MeteoCal will have to "allow users to manage the privacy of their calendars and events" is intended to only assert that an event creator can set an event that he created as private or public. The property of an event of being outdoor or indoor is assumed to be included in the functional requirement that states that an event creator can "update an event that he created, so changing its details and sending more invitation", which is in fact said to be derived from the goal `G2`.

2.2. Implicit requirements assumptions:

We also assume that – since MeteoCal is a web application – those requirements must hold:

- A registered user can log out;
- MeteoCal must be able to manage multiple user's sessions (of different users);
- sharing proper information (e.g. time slots must be available also when the event is private).

In addition, to fit the informal specification of the software costumer, we derive these requirements:

- Correctly schedule events by managing time consistency.

3. Test cases

3.1. Unregistered user perspective

Goal	***Registration to the system***
Functionality	To allow people to register to the system, and then to log-in, an unregistered user can register to the system.
Environment	*Registration page*
URL	`../register.xhtml`
Input	- Random but valid first name; - random but valid last name; - email; - valid password.
Expected output	*Login page*
Obtained output	Same as above
Final output	Same as above
Possible errors	- Empty required fields: The page shows error messages, no previously entered data is lost. - Invalid email: the page shows error messages, no previously entered data is lost. - Password miss-match: the page shows error messages, no previously entered data is lost.
Comments	OK.

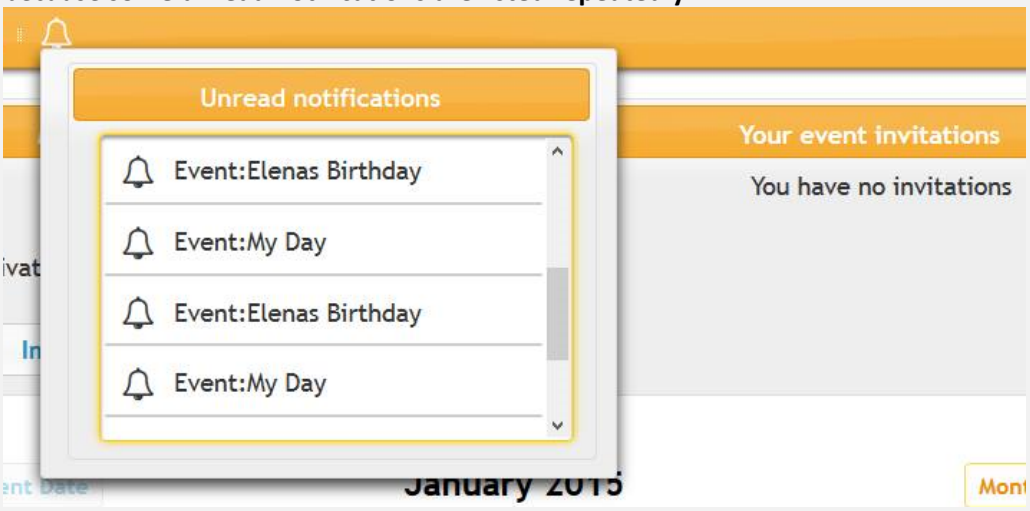
3.2. User (registered) perspective

Goal	***Log into the system***
Functionality	To allow people to register to the system, and then to log-in, a user can log into the system.
Environment	*Login page*
URL	`../user/home.xhtml`
Input	- Valid email; - valid password.
Expected output	*My Calendar Page*
Obtained output	Same as above
Final output	Same as above
Possible errors	- Empty required fields: The page shows error messages, no previously entered data is lost. - Wrong email or password: The page shows error messages, no previously entered data is lost.
Comments	OK.

Goal	*Load the calendar with the scheduled event*
Functionality	To allow users to view their calendar other user's calendars, and the contained events, a user can see his calendar with the scheduled events.
Environment	*My calendar page*
URL	`../user/home.xhtml`
Input	Log in
Expected output	Calendar with scheduled events
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*See the details of the event*
Functionality	To allow users to view their calendar other user's calendars, and the contained events, a user can see event's details.
Environment	*Event details page*
URL	`../user/event_details.xhtml`
Input	Click on the event box
Expected output	Even details with attendees. For the creator there should be the update and the delete option.
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*See weather forecasts for an event*
Functionality	To allow users to see the weather forecast for the events in their calendar.
Environment	Event details
URL	`../user/event_details.xhtml`
Input	Click on the event box to see its details
Expected output	Show weather forecast – if any – in event details page.
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*View notifications* [WARNING]
Functionality	To receive notifications about the weather forecasts for outdoor events in the calendar, a user can see the received notifications which are still unread.
Environment	*Tool panel* (i.e. toolbar, in every page).
URL	`../user/*.xhtml`
Input	Click on the notification icon
Expected output	Open a dropdown menu with a list of notifications, if any.
Obtained output	Same as above
Final output	Shows same notification repeatedly
Possible errors	No possible errors
Comments	It actually works accordingly with the requested functionality, but it's confusing because **some unread notifications are listed repeatedly**. 

Goal	*Manage calendar privacy*
Functionality	To allow users to manage the privacy of their calendar and events, a user can set his calendar as public or private.
Environment	Manage calendar screen compartment in *My calendar page*.
URL	`../user/home.xhtml`
Input	Click privacy toggle button
Expected output	Value of toggle button changes and the page refreshes. In addition, other users other users should be able to view this user's calendar properly.
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Create an event*
Functionality	To create, update and delete own events, a user can create an event and set its details (location, date, outdoor/indoor, public/private etc…)
Environment	*Event creation page*
URL	`../user/home.xhtml`
Input	- Name; - city; - date and time; - location; - duration; - outdoor or indoor event property; - private or public property; - invited users.
Expected output	Displays *My calendar page* with the user's calendar updated accordingly with the new event.
Obtained output	Same as above
Final output	Same as above
Possible errors	- Empty required fields: the page shows error messages, no previously entered data is lost. - Invalid city: the page shows error messages, no previously entered data is lost.
Comments	OK.

Goal	*Import a calendar*
Functionality	To allow users to import or export calendars, a user can import a calendar from an external source file.
Environment	Manage calendar screen compartment in *My calendar page*
URL	`/user/home.xhtml`
Input	Click on the *Import button*
Expected output	A provision to choose a file of .cal format and upload it
Obtained output	Same as above
Final output	The user's calendar was updated accordingly to the events from the imported calendar.
Possible errors	- No file selected: an error message is displayed and the page is refreshed. - The selected file is not in `.cal` format: an error message is displayed and the page is refreshed.
Comments	OK.

Goal	*Export a calendar*
Functionality	To allow users to import or export calendars, a user can export his calendar as a properly formatted file.
Environment	Manage calendar screen compartment in *My calendar page*
URL	`../user/home.xhtml`
Input	Click on the *Export button*
Expected output	Ask for the location for the downloaded file to be stored in
Obtained output	Same as above
Final output	Downloads file with a `.cal` extension and stores it in the specified location
Possible errors	No possible errors
Comments	OK.

Goal	*Receive notifications about bad weather forecast*
Functionality	To receive notifications about the weather forecasts for outdoor events in the calendar, a user can, one day before an outdoor event in his calendar, in case of bad weather forecast, receive a notification.
Environment	*Notifications menu*
URL	`../user/*.xhtml`
Input	Click on the *Notifications icon*
Expected output	To be prompted of bad weather forecast for an outdoor event
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Receive notifications about weather forecast changes*
Functionality	To receive notifications about the weather forecasts for outdoor events in the calendar, a user can, in case of weather forecast changes for an event in his calendar, receive a notification.
Environment	*Notifications menu*
URL	`../user/*.xhtml`
Input	Click on the *Notifications icon*
Expected output	To view the updated weather forecast
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Search for another user and eventually see his/her profile*
Functionality	To allow users to view their calendar, other user's calendar and the contained events, a user can search for another user (by name) and see his calendar and events, according to their privacy settings.
Environment	Search form in the toolbar and *Search results page*
URL	`../user/search_results.xhtml`
Input	Name of a registered user
Expected output	If the user is found, the searcher should be able to select him/her and view his/her calendar, if public. ~~In addition, we should also be able to add the user to our favorite list.~~
Obtained output	If the user is found, the searcher is able to select him/her and view his/her calendar, if public.
Final output	Same as above.
Possible errors	No possible errors
Comments	OK.

Goal	*Receive email notifications* **[CRITICAL]**
Functionality	Receive e-mail notifications each time a notification is sent by the system.
Environment	User's mail box
URL	n/a
Input	No input
Expected output	Email from the system to the email provided by the user
Obtained output	No email received
Final output	Same as above
Possible errors	No possible errors
Comments	**The user does not receive email notifications** from the system each time a notification is sent by it.

3.2.1. Event creator perspective

Goal	*Invite users during the event creation*
Functionality	To allow the creator of an event to invite other registered users to it, an event creator can invite other users to an event that he created, during the event creation.
Environment	*Event creation page*
URL	`../user/event_creation.xhtml`
Input	- Name; - city, - date & time; - location; - duration; - description; - invited user.
Expected output	User B has been invited by user A (i.e. the creator).
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Update an event* `[WARNING]`
Functionality	To allow the ~~owner of a calendar~~ *creator* to create, update and delete events, an event creator can update an event that he created, so changing its details and sending more invitations.
Environment	*Event update page*
URL	`../user/event_update.xhtml`
Input	Changed some field values
Expected output	Event details changed. Redirect the creator to the *Calendar page*.
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	If the invited user (in this case an attendee) is in the *Event details page* and – in the meantime – the creator updates the event, **the invited user will not be able to see the updates**, neither **by refreshing the page**. The only way to see them is to go back in the page with the calendar and re-open the *Event details page*.

Goal	*Delete an event*
Functionality	To allow the ~~owner of a calendar~~ *creator* to create, update and delete events, an event creator can delete an event that he created.
Environment	*Event details page*
URL	`../user/event_details.xhtml`
Input	Click on the *Delete button*
Expected output	The event has been deleted
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Change privacy of an event*
Functionality	To allow users to manage the privacy of their calendar and events, an event creator can set an event that he create as private/public ~~and outdoor/indoor~~.
Environment	*Event update page*
URL	`../user/event_update.xhtml`
Input	Click on the *Private button*
Expected output	Change privacy of that event
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Closest sunny day hint in case of bad weather* ~~`[CRITICAL]`~~
Functionality	To receive notifications about the weather forecasts for outdoor events in the calendar, an event creator can, three days before an outdoor event created by him, in case of bad weather, receive a notification with a suggestion of the closest sunny day.
Environment	MeteoCal or email box
URL	`../user/*.xhtml`
Input	No input
Expected output	Suggestion of the closest sunny day
Obtained output	n/a
Final output	~~No hint~~
Possible errors	No possible errors
Comments	~~**The system** notifies the user of bad weather conditions but **does not suggest the closest sunny day**.~~ The test is not valid since it is necessary to wait 3 hours from the server start-up before receiving the weather notification. This could be an issue when updating the application (thus, the costumer should be informed about it), but it cannot be considered as an actual bug.

3.2.2. Invited user perspective

Goal	*Accept an invitation*
Functionality	To allow the creator of an event to invite other registered users to it, an invited user can accept an invitation
Environment	*Event details page*
URL	`../user/event_details.xhtml`
Input	Click on the *Accept button*
Expected output	The event has been added to the invited user's calendar.
Obtained output	Same as above
Final output	We are redirected to the calendar page.
Possible errors	No possible errors
Comments	OK.

Goal	*Decline an invitation*
Functionality	To allow the creator of an event to invite other registered users to it, an invited user can decline an invitation
Environment	*Event details page*
URL	`../user/event_details.xhtml`
Input	Click on the *Decline button*
Expected output	The event has not been added to the user's calendar.
Obtained output	Same as above.
Final output	We are redirected to the calendar page.
Possible errors	No possible errors
Comments	OK.

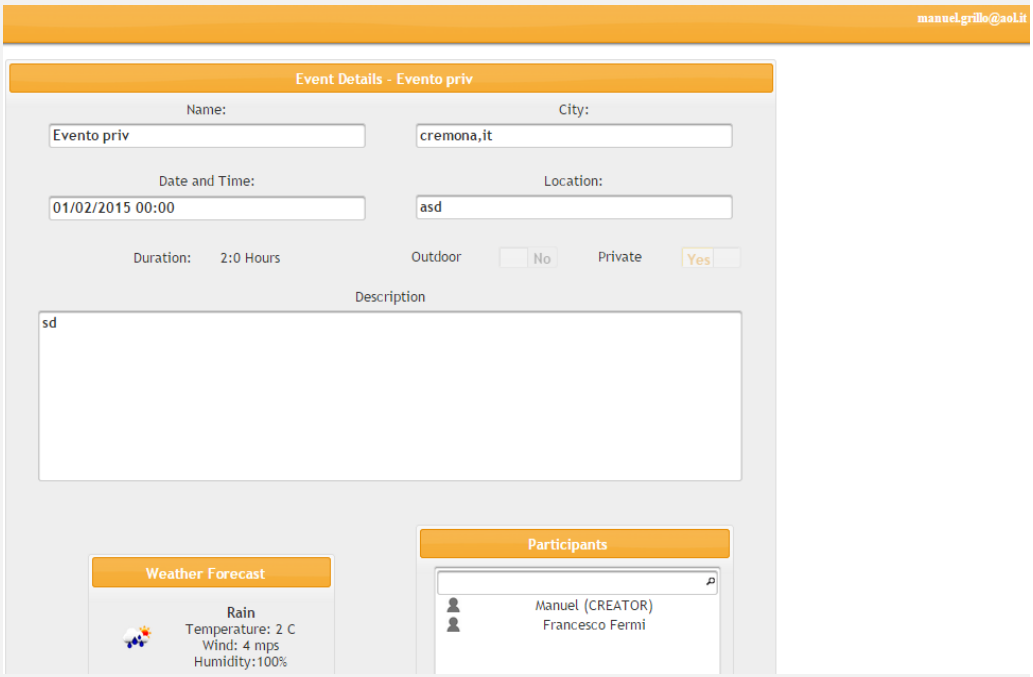
3.3. Implicit requirements

Goal	*Sign out*
Functionality	Logs the user out of the system
Environment	*Home page*
URL	`../user/home.xhtml`
Input	Click on the *Log out button*
Expected output	? to the *Login page*.
Obtained output	The session has been invalidated. Redirect to the *Login page*.
Final output	Same as above
Possible errors	No errors
Comments	OK.

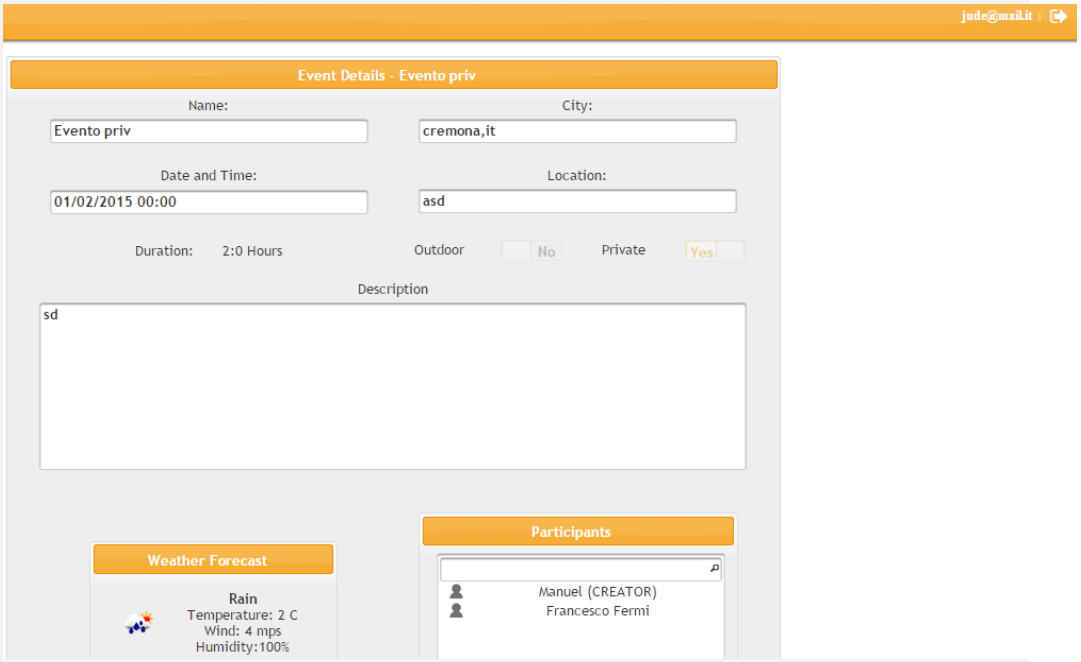
Goal	*Multi session capability*
Functionality	Multiple users can use access MeteoCal w/o conflicts
Environment	Two separate browser and user sessions
URL	n/a
Input	Simultaneous use of the web application, with two different accounts
Expected output	n/a
Obtained output	n/a
Final output	n/a
Possible errors	n/a
Comments	URL injection (see *Sharing proper information*).

Goal	*Correctly schedule events by managing time consistency*
Functionality	The overlaps between events must be somehow managed: *manage time consistency when creating an event by avoiding conflicts with existing events.*
Environment	*Create event page*, *Event update page*, *Event details page*.
URL	`../user/event_*.xhtml`
Input	- Name; - city; - date and time; - location; - duration; - outdoor or indoor event property; - private or public property; - invited users.
Expected output	n/a
Obtained output	Same as above
Final output	Same as above
Possible errors	No possible errors
Comments	OK.

Goal	*Sharing proper information* **[CRITICAL]**
Functionality	- Private calendars must be somehow managed (public ones should be visible to any registered user, while private ones should be hidden. - Private events must be somehow managed (time slots must be available also when the event is private).
Environment	*MeteoCal webapp*
URL	`../user/*.xhtml`
Input	n/a
Expected output	n/a
Obtained output	n/a
Final output	n/a
Possible errors	n/a
Comments	Private calendar are properly managed, on the other hand **the pages that belongs to private events are not managed properly**. In some scenarios a user that has not been invited to the event is able to see the details trough a URL injection: This is the event page showed at a user related to the event:



This is the same page, showed to a user that should not be able to see it:



Basically, writing the URL `../user/event_details.xhtml` it is possible to load the last *Event details page* that has been loaded by any user in the web application.

4. Conclusions

On the basis of the observations made during the test cases in *Chapter 3 – Test cases*, the overall quality of the web application is high, but some issues has been found.

Minor bugs

- In the *View notifications* test (*page 6*), **some unread notifications are listed repeatedly**.
- In the *Update an event* test (*page 10*), **the invited user is not able to see the updates refreshing the page**.

Major bugs

- In the *Receive email notifications* test (*page 9*), **the user does not receive email notifications**. [In contradiction of the functional requirement "A user can receive email notifications each time a notification is sent by the system" written in Section 3.2 – Functional requirements in `RASDFinal_v2.0.pdf`]
- ~~In the *Closest sunny day hint in case of bad weather* test (*page 11*), **the system does not suggest the closest sunny day**. [In contradiction of the functional requirement "An event creator can, three days before an outdoor event created by him, in case of bad weather, receive a notification with a suggestion of the closest sunny day" written in Section 3.2 – Functional requirements in `RASDFinal_v2.0.pdf`]~~
- In the *Sharing proper information* test (*page 14*), **the pages that belongs to private events are not managed properly**. [In contradiction of the functional requirement "A user can create an event and set its detail", which includes "location, date, outdoor/indoor, public/private, etc." written in Section 3.2 – Functional requirements in `RASDFinal_v2.0.pdf`). In addition the `RASDFinal_v2.0.pdf` specifies that "If a user sets an event as private it means that, even though his calendar is public, other users cannot see the event's details but only that he is busy" (*Chapter 2.1 – Specification*)]

The major bugs are directly related to functional requirements. For this reason we recommend to continue the development and improvement of the web application to solve at least the major bugs. When the major bugs will be solved, the application could be released in beta version (with only minor bugs).

Appendix – Revision history

Initial release:	v1.0
Current release:	V1.1
Date of the last review:	2015/02/08

v1.0

2015/01/29 – Initial document.

v1.1

2014/02/08 – Changed the status of the *Closest sunny day hint in case of bad weather* test from "CRITICAL" to "OK": the test wasn't valid because it was necessary to wait 3 hours from the server start-up before receiving the weather notification. This could be an issue when updating the application (thus, the costumer should be informed about it), but it cannot be considered as an actual bug.