Department of

Computer Science and Engineering

MeteoCal

Project development for the 2014 Software Engineering 2 Course

Supervisor: **Prof. Di Nitto**

Authors

Andrea Bignoli
Matr. 837493
andrea.bignoli@gmail.com

Leonardo Cella
Matr. 838074
leonardocella@gmail.com

Acceptance

Testing

Digest

An Acceptance Testing Document is produced to provide feedback for the implementation of a project, to ensure that the problem requirements have been satisfied and that the system functionalities don't produce unexpected behaviours.

Table of Contents

1. Introduction

This Acceptance Testing Document is produced to provide feedback for the implementation of project MeteoCal by Salvatore Agnese and Gabriele Giganti. The point of view adopted will be the one of the customer the system would be presented to. As such, particular attention will be dedicated to ensuring that the implementation provided complies with the problem requirements. The system test cases provided by the authors will be also a fundamental reference in this process, since we are going to verify the outcome of those tests and eventual non managed issues.

2. Software Verification

In this phase we are going to make sure that the software produced satisfy the conditions imposed by the problem description.

2.1. RASD Verification

The first step has been the validation of the RASD document in respect to the initial problem specification.

This allows us to:

- Certify that functionalities promised in RASD satisfy the requirements described in the problem description
- Check that the implementation of the system adheres to the guidelines of the RASD.

The first of the two constraints has been verified during the analysis of the RASD with positive result.

2.2. Implementation Verification

It is possible to verify that the RASD premises have been actually satisfied with the aid of the User Manual provided by the author of the project.

The actual process can be resumed by the following steps:

- Verify that the functionalities declared in the RASD correspond to the ones declared in the User Manual. It should also been taken in account that the User Manual may not cover deeply some functionalities that the implementation may offer.
- Follow the steps reported in the User Manual to verify that the implementation respects these guidelines.

3. Software Validation

In this phase we are going to make sure that the software produced satisfies the implementation requirements. This will be mainly related to testing the various functionalities and verify that they work correctly without unexpected results.

3.1. System Test Cases

In this phase the analysis will be based on the Test Cases Document produced by the authors of the project to test, and the test cases presented here are the ones provided by them. Our intents are:

- Verify the outcomes of the test cases where pre-conditions are met
- Verify the behaviour of the software in response to managed errors mentioned in the documentation
- Look for unmanaged errors, not found by the authors, to discover possible flaws in the software

The possible outcomes of our analysis are:

- SUCCESS
  If the main goals of the test have been accomplished and no highly relevant objection can be made to the management of any particular input condition.
- PARTIAL SUCCESS
  If the main goals of the test have been accomplished but there is one or more highly relevant objection can be made to the management of a particular input condition stated or not stated in the test case documentation, but related to it.
- FAIL
  If the main goals of the test have not been accomplished.
- UNKNOWN
  If the occurrence of some particular conditions made the test unavailable

The issues discovered in this project will be listed with an associated importance value of 1,2 or 3 with a value of 3 representing the most relevant problems.

Additional notes will be added to describe issues of different significance and explain the outcome of our acceptance process.

3.1.1. Sign Up [FR1]

Test Cases Document entry:

Objective	Create a new user
Page	Registration page
Input	Data in a proper format: Email, First Name, Last Name, Password, Sex, Birth Date, telephone, address, city, public calendar visibility
Expected Output	The system create a new user with the inserted data in The database
Obtained Output	The same of the Expected Output
Result	The system return the index page
Observations	
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
	The system shouldn't accept birth dates in the future.	1
Notes	• [UI] If telephone number isn't considered valid, the field is erased after losing focus. This is acceptable but a probably more stardarized approach would be showing an error message to the side of the input box and disable the registration button until the field is correct or empty. It should be noted that even like this the user is already able to only input numbers in the field and the correct length for the telephone number is clearly shown, so this solution is already user friendly.	

3.1.2. Log In [FR2]

Test Cases Document entry:

Objective	Login the system as User
Page	Index Page
Input	Email and Password of the User
Expected Output	The system allows User authentication
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page
Observations	
Managed Errors	Possible errors are missing data or not consistent so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	PARTIAL SUCCESS	
Issues	*Description*	*Value*
	The issue can be replicated by following these steps: • Login using valid data in the login page at http://localhost:8080/meteo/ • Login succeeds, the home page is shown. (URL doesn't change, the request has probably just been dispatched) • Refresh the page or go to http://localhost:8080/meteo/ • The login page previously mentioned is shown even though since no log out was performed, the current session is still active • Try to login using the previous data or even an other account. • The login fails. This is probably due to the fact that the session is still open. The only way for the end user to get back into the system to his pages is actually manually changing the URL to any of its personal pages, which is probably not intended.	2
	User authentication isn't used for managing page access. More information can be found in section 3.2.1.	3
Notes		

3.1.3. Create Event [FR3]

Test Cases Document entry:

Objective	Create a new event
Page	Create event page and Create event page 2
Input	Data in a proper format: Name of the event, EventDate, Duration, Address, City, Description, Type of event, public visibility
Expected Output	The system saves the data of the new Event in The database
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page, with the event just saved visible on the calendar, clickable to get a preview and expandable with the button "open event" to get event details.
Observations	To complete the save procedure, after pressing next, press "create" in the Create Event page 2, otherwise the event will not be saved.
Managed Errors	Possible errors are missing data, not consistent data and range already busy by another event so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.4. Update Event [FR4]

Test Cases Document entry:

Objective	Update the selected event, already present in the calendar
Page	Selected event Page, Update event Page 1, Update event Page 2
Input	Event to be updated (clicking it on calendar, expanding it with the button "open event" and updating clicking “Update Event”), Data in a proper format: Name of the event, EventDate, Duration, Address, City, Description, Type of event, public visibility
Expected Output	The system updates the data of the selected event in The database, removes each invite related to it and creates the new Invite (if there are no changes in event page 2,remain the same invitations of the event before the update, but with different id, so all the previously invited Users have to accept or decline again the invite)
Obtained Output	The same of the Expected Output
Result	The system returns the user’s home page, with the event just updated visible on the calendar with updated position, clickable to get a preview and expandable with the button "open event" to get updated event details. Users which have already accepted the invitation cannot view the updated event on their own calendar, before accepting the invitation again, that will contain the updated information.
Observations	To complete the update procedure, after pressing next, press "save" in the Update Event page 2, otherwise the event will be not updated.
Managed Errors	Possible errors are missing data, not consistent data and range already busy by another event so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [POTENTIAL IMPROVEMENT] Participants should be notified about the change by the system instead of removing the participation in their place. In fact it would be probably better if the user could remove his participation by himself after being notified about a change. Moreover the invitation doesn't show in any way that the previous participation has been removed due to a change. Anyway the presented solution is acceptable.	

3.1.5. Delete Event [FR5]

Test Cases Document entry:

Objective	Delete the selected event, already present in the calendar
Page	Selected event Page
Input	Event to be deleted (clicking it on calendar, expanding it with the button "open event" and deleting clicking “Delete Event”)
Expected Output	The system delete the data of the selected event in The database and removes each invite related to it
Obtained Output	The same of the Expected Output
Result	The system returns the user’s home page, without the event just deleted.
Observations	Each previously invited person will no longer see the event on their own calendar or in the user creator calendar.
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [POTENTIAL IMPROVEMENT] Participants should be notified explicitly about the event deletion.	

3.1.6. Invite User To Event [FR6]

Test Cases Document entry:

Objective	Invite User to a new event or to a selected event, already present in the calendar
Page	Create Event Page, Create Event Page2,Selected event Page, Update event Page 1, Update event Page 2
Input	Selecting next into “event creation page” (new event) or selecting next into “update event page 1”(event already exist), the already invited users (if event already exists), the selected users in the contact list, the selected data into search panel.
Expected Output	The system create the new event in The database related to the selected Event and the inserted users to invite. In case of event updating, removes each invite related to it and creates the new Invite (if there are no changes in event page 2,remain the same invitations of the event before the update, but with different id, so all the previously invited Users have to accept or decline again the invite)
Obtained Output	The same of the Expected Output
Result	The system returns the user’s home page, with the event of the invite just updated visible on the calendar with eventually updated position, clickable to get a preview and expandable with the button "open event" to get updated event details and visualize the new Invitation List. Each invited User, will receive a new Invite Notifications.
Observations	To complete the invitation procedure, press "save" (if into update event page 2) or “create” (if into create event 2), otherwise the invitation will be not created/updated.
Managed Errors	Possible errors are not consistent data (such as empty bar, not existent users or himself in the search panel, checking users form the contact list already present in the invite list, so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• We found some problematic aspects in the way participation are managed, more information can be found in section 3.3.1.	

3.1.7. Accept Invite [FR7]

Test Cases Document entry:

Objective	Accept an invite to an event created by another User
Page	Invite Notification Page
Input	The notification opened, the answer by clicking accept or decline button.
Expected Output	The system update the selected invitation in the database setting answer to true, and NotifiedAnswer to true.
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page, with the event related to the invite just accepted visible on the calendar, clickable to get a preview and expandable with the button "open event" to get event details and visualize the Invitation List and his answer into accepted list. Notification is no more visible in the notifications panel
Observations	
Managed Errors	Possible errors is the range already busy by another event so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.8. Decline Invite [FR8]

Test Cases Document entry:

Objective	Decline an invite to an event created by another User
Page	Invite Notification Page
Input	The notification opened, the answer by clicking accept or decline button.
Expected Output	The system update the selected invitation in the database setting answer to false, and NotifiedAnswer to true.
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page. Notification is no more visible in the notifications panel
Observations	
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.9. Enrich Event With Forecast [FR9]

Test Cases Document entry:

Objective	Enrich each event with Forecast Information
Page	Home Page, Selected Event Page, Invite Notifications Page, forecast Notification Page, Update for bad weather page
Input	City and Date of the corresponding event
Expected Output	The system update the selected event page, invite notification page, forecast notification page, update for bad weather page with forecast information. Moreover, it provide forecast information to Notifications Manager for provide bad weather and proposal update notifications to user.
Obtained Output	The same of the Expected Output
Result	Notifications are received or the page mentioned above are correctly opened.
Observations	
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.10. Send Notification Of Bad Weather [FR10]

Test Cases Document entry:

Objective	Send notification of bad weather
Page	Home Page
Input	Event invitation accepted by the user and the forecast details (bad forecast about the event that takes place not more than 1 day before the event) about the event (outdoor) related to the accepted invitation.
Expected Output	The system generate the bad weather notification
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page. Notifications are received by the user and visible in the notifications panel of the homepage
Observations	
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
	• Only participants are notified about the bad weather conditions but not the creator.	1
Notes		

3.1.11. Retrieve Notification [FR11]

Test Cases Document entry:

Objective	Retrieve notification
Page	Home Page
Input	Notification list generated by the system
Expected Output	The system allow user to open notification into notifications by clicking the “open” link of the notification
Obtained Output	The same of the Expected Output
Result	The same of the Expected Output
Observations	
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.12. Make Calendar Public [FR12]

Test Cases Document entry:

Objective	Make user calendar visibility public
Page	Registration Page, Profile Page
Input	Checking “public calendar”
Expected Output	The system set true the publicVisibility field of the user in the database
Obtained Output	The same of the Expected Output
Result	The system returns the index page(during registration) or homepage (during profile modification)
Observations	When another user open the user profile, the relative calendar will be visible in the "profileofuserxPage"
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.13. Make Calendar Private [FR13]

Test Cases Document entry:

Objective	Make user calendar visibility private
Page	Registration Page, Profile Page
Input	Unchecking “public calendar”
Expected Output	The system set false the publicVisibility field of the user in the database
Obtained Output	The same of the Expected Output
Result	The system returns the index page(during registration) or homepage (during profile modification)
Observations	When another user open the user profile, the relative calendar will be invisible in "profileofuserXPage"
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [UI] It would be clearer to explicitly state that the calendar is private instead of just not showing it.	

3.1.14. Make Event Public [FR14]

Test Cases Document entry:

Objective	Make event visibility public
Page	Create event page, Create event page 2, Update event Page 1, Update event Page 2
Input	Checking “public visibility”
Expected Output	The system set true the publicVisibility field of the event in the database
Obtained Output	The same of the Expected Output
Result	The same of create event/update event
Observations	To complete the procedure, press "save" (if into update event page 2) or “create” (if into create event 2), otherwise the publicVisibility will be not created/updated. When another user open the user profile, the relative event will be openable and expandable in the "profileofuserxPage" calendar.
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• Some page navigation issues have been highlighted in section 3.3.2. • Some flaws regarding event privacy have been highlighted in section 3.3.3.	

3.1.15. Make Event Private [FR15]

Test Cases Document entry:

Objective	Make event visibility private
Page	Create event page, Create event page 2, , Update event Page 1, Update event Page 2
Input	Unchecking “public visibility”
Expected Output	The system set false the publicVisibility field of the event in the database
Obtained Output	The same of the Expected Output
Result	The same of create event/update event
Observations	To complete the procedure, press "save" (if into update event page 2) or “create” (if into create event 2), otherwise the publicVisibility will be not created/updated. When another user open the user profile, the corresponding event will appear as busy, clickable but not expandable
Managed Errors	
Unmanaged Errors	

Validation output:

Outcome	PARTIAL SUCCESS	
Issues	*Description*	*Value*
	Some flaws regarding event privacy have been highlighted in section 3.3.3.	3
Notes		

3.1.16. Propose Change Date [FR16]

Test Cases Document entry:

Objective	The system propose a change date for bad weather to the eventCreator
Page	Update for bad weather page
Input	Event created by the user and the forecast details (bad forecast about the event that takes place at least 2 day and not more than 3 day before the event), the selected date.
Expected Output	Event created by the user and the forecast details (bad forecast about the event that takes place at least 2 day and not more than 3 day before the event), the selected date.
Obtained Output	The same of the Expected Output
Result	The system returns the user's home page, with the event just updated visible on the calendar with updated position, clickable to get a preview and expandable with the button "open event" to get updated event details. Users which have already accepted the invitation cannot view the updated event on their own calendar, before accepting the invitation again, that will contain the updated information.
Observations	
Managed Errors	Possible errors is the range already busy by another event so the system does not return the user home page, but remains on the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes		

3.1.17. Add User To Contact List [FR17]

Test Cases Document entry:

Objective	The system add another user into own contact list
Page	Home page, profile of user x
Input	Email into search user panel, user searched
Expected Output	The system saves the new contact into isfriend table of The database
Obtained Output	The same of the Expected Output
Result	The system remain into profileOfUserX page
Observations	The User added is now included in the contact list Page of the logged user
Managed Errors	Possible errors are not consistent data (such as empty search bar, not existent users in the search panel, adding himself, so the system remain in the same page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [UI] On the User Page there is a button to add or remove the searched result from the contact list. The button is shared for the two operations, and labelled as “Add | Remove contact”. A better UI choice would probably be representing a single button where the label represents the action actually performed, "Add" or "Remove". In this case, a button labelled as “Add contact” would be clearer.	

3.1.18. Remove User From Contact List [FR18]

Test Cases Document entry:

Objective	The system remove another user into own contact list (already present)
Page	Home page, profile of user x, Contact List Page
Input	Email into search user panel, user searched, contact list user opened
Expected Output	The system remove the contact from isfriend table of The database
Obtained Output	The same of the Expected Output
Result	The system remain into profileOfUserX page
Observations	The User removed is no more included in the contact list Page of the logged user
Managed Errors	Possible errors are not consistent data (such as empty search bar, not existent users in the search panel,so the system remain in the same page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [UI] On the User Page there is a button to add or remove the searched result from the contact list. The button is shared for the two operations, and labelled as “Add | Remove contact”. A better UI choice would probably be representing a single button where the label represents the action actually performed, "Add" or "Remove". In this case, a button labelled as “Remove contact” would be clearer.	

3.1.19. Change Personal Information [FR19]

Test Cases Document entry:

Objective	Change personal information with inserted data
Page	Profile Page
Input	Data in a proper format: New Password, Repeat new Password, telephone, address, city, public calendar visibility
Expected Output	The system update the data of the logged user in The database
Obtained Output	The same of the Expected Output
Result	The system remain in the Profile Page and return an added message
Observations	
Managed Errors	Possible errors are missing data, not consistent data (e.g. new password and repeat new password different), so the system remains in the current page, where these errors are reported
Unmanaged Errors	

Validation output:

Outcome	SUCCESS	
Issues	*Description*	*Value*
Notes	• [SECURITY] To modify the password the system should probably ask the current one, to adhere to the most common security standards.	

3.2 Additional System Tests

The following tests have been done to complement the ones proposed by the authors of the project. This has been done to cover some aspects that we considered relevant to the application usage. Unlike the test cases listed above, the ones listed below have been written by the acceptance testing team.

3.2.1 Page Access Resrictions

Objective	User pages access should be restricted to logged users
Page	Any of the pages that should be accessed only by logged user (e.g. Home Page)
Input	A user tries to access the page by manually changing the URL in his browser
Expected Output	Pages should be correctly shown if the user is logged in the system. If no session is access, the pages shouldn't be shown without any system errors. Showing the login page instead is an acceptable solution.
Obtained Output	Pages are shown if the user is logged. If the user isn't logged unexpected and potentially behaviours have been discovered.

Validation output:

Outcome	FAIL	
Issues	*Description*	*Value*
	Page access is not ruled by security constraints. Different pages that should require login are accessible (by changing manually the URL) when no session is active. This causes NullPointer exception when the page references properties relative to the current session (that does not exist). This happens for example at: http://localhost:8080/meteo/faces/user/homepage.xhtml Other pages that show non intended behaviours are: http://localhost:8080/meteo/faces/user/profilePage.xhtml http://localhost:8080/meteo/faces/user/createEvent.xhtml Being able to access the last one is actually really dangerous since it is possible, using this method, to create events without an active session! A consequence of this fact is that the newly created event has emailCreator equal to null. Since sessions expire due to limited duration, this may happen unexpectedly when a user is logged for some time. Moreover the occurrence of this situations can be dangerous to the execution of the application itself that can crash, making a fresh re-start necessary.	3
Notes		

3.3 Additional Notes

We considered some aspects of our analysis as better representable by a textual explanation, rather than a test case, since the latter approach wouldn't be appropriate.

3.3.1 Participation Management

Participation to the events are considered though as no more than a positive answer to an invitation. This causes some behaviours that may be considered rather unpleasant for the end user. An example of this is shown here:

The creator of an event can invite any other user and remove the invitations at any time he wants.

- The creator invites user X to event E.
- X accepts the invitation and event E is added to X's calendar.
- Creator of event E decides to remove the invitation sent to X. Consequence: the participation of X to E is completely removed, and E doesn't appear anymore on X calendar, while X doesn't receive any notice of this.

This behaviour doesn't necessarily go against the initial specification, but it's quite unexpected by the end user. If the creator of the event is able to remove invited users and their participation, they should get notified about this.

In fact, we think that participations should be managed by the participants themselves. This obviously isn't possible if the event is deleted, but even then, the participants should be notified.

At the current state of the system, there's no way for a user to cancel his participation to an event.

Since this isn't part of the problem requirements and the presented implementation isn't in contrast with the RASD document, the approach used can still be accepted.

3.3.2 Page Navigation URLs

There is no way to get a link to the page of an event or of a user on the system, since the pages dedicated to showing these contents don't use the URLs to identify the resource requested by the system. While this doesn't represent an obstacle to the system in providing its services to the users there are also some negative consequences.

For example there's no way for a person to show an other a particular event since there is no way to link it, nor a search functionality.

Another important consequence is correlated to the security protection used in managing event privacy level and will be analysed in the next section.

3.3.3 Event Page Access

At the current state of the system the only way for a user to access the page of an event he's not invited to is clicking on that event on the calendar of a participant that has set his calendar as public. Even then, only invited users can actually participate to the event since there is no “Participate” button.

The only protection that a private event page has is actually the absence of a link to that page visible to user non invited to the event. This by itself, could be considered a rather unsafe approach to this resource management problem.

This approach in fact has a flaw in the current state of the system, even if its occurrence is rather hidden. The steps to observe this flaw are the following:

- Public event E is created
- Since the event is public, it will be visible on the calendar page of the creator or a participant to the event if the owner set his calendar as public. User U, not invited to the event, uses the search functionality to get on such page, where there is a link to E, since E is public and stops here
- Creator of E modifies E writing secret information in the description of the event and setting the event as private
- User U which still has available a link to E, since he got to a calendar containing it when E was public. User U can then use this link and access the page of E and read all sensible information contained in its description

3.4 Code Quality And Maintainability

The provided code has no documentation for the most part, since there is no JavaDoc and comments are almost totally absent.

On the bright side the structure of the code is very modular, and the operations contained in each module are usually not complex. So, regardless of the lack of documentation, the code should be maintainable with very little effort.

4 Conclusions And Global Evaluation

As we stated before we can affirm that the RASD document and the implementation provided comply with the initial problem description.

In particular the set of operations performed in the system test cases fully fully covers the requirements extracted from the initial description.

As for the implementation, the majority of operations tested showed no highly relevant issues. There are anyway some that needs to be fixed since they can represent a danger for the execution of the system or problematic for the end user. The most important issue is probably the one regarding the fact that page access should be restricted to logged users, since it can lead to application failures.

On the other hand, participation management should be improved. Even though this is not explicitly stated in the problem description, the ability to cancel a participation to an event should be present in the final product.

Moreover the flaw regarding unauthorized access to private event pages should also be fixed since it can be considered as a security breach of moderate entity.

Finally, the ability to link event pages should also be granted to make access to the resources more adherent to the common standards and to improve the overall usability of the system.

To recap, the product answers fully the requests stated in the problem descriptions, but has some flaws that, for the most part, should be easily fixable.