Test Case Document
<MeteoCal>

MATTEO GAZZETTA *837853*

ALESSANDRO FATO *838218*

January 20, 2015

Status: Final

Version: 1.0

Contents

1 Introduction

This document shows the tests that are been executed on the application. We executed two topologies of test:

- Test with manual verification of the system
- Automatic Unit test with JUnit and Mockito

2 Manual Test

This type of test was performed manually creating the operations that the application users may perform

2.1 [TC1]: The User logs into the system

Goal	Allow a User to access in the personal calendar page
Preconditions	• The User's information are in the database • The System is installed correctly
Operating Environment	MC's home page
Input	1. Press the button "Login" 2. Insert User's email and password 3. Press the button "Login"
Expected Result	The User logs into the system after inserting the correct User's code and password.
Postconditions	The User logs in and he is redirected at the User's calendar page
Possible Errors	• The access data inserted are not correct • The system is not correctly installed and configured

2.2 [TC2]: The User logs into the system with Facebook

Goal	Allow a User to access in the personal calendar page
Preconditions	• The User's information are in the database • The System is installed correctly
Operating Environment	MC's home page
Input	1. Press the button "Login" 2. Insert User's email and password 3. Press the button "Login"
Expected Result	The User logs into the system after inserting the correct User's code and password.
Postconditions	The User logs in and he is redirected at the User's calendar page
Possible Errors	• The access data inserted are not correct • The system is not correctly installed and configured

2.3 [TC3]: The User logs out from the system

Goal	Allow an User to exit from <*MeteoCal*>
Preconditions	• The User is already authenticated
Operating Environment	Every User's page
Input	1. Press the button "Logout"
Expected Result	The Employee logs out
Postconditions	The Employee logs out and he is redirected at to the home page
Possible Errors	

2.4 [TC4]: The Visitor registers into the system

Goal	Create a new User in the system
Preconditions	• The System is installed correctly
Operating Environment	MC's home page
Input	1. Press the button "Registration" 2. Insert new User's email, password, first name, last name and other personal information 3. Press the button "Register"
Expected Result	The Visitor is registered into the system after inserting valid information.
Postconditions	The Visitor is registered and he views the home page
Possible Errors	• The email already present in the system • The password violate the constraint

2.5 [TC5]: The User add a new event

Goal	Create a new Event in the system
Preconditions	• The System is installed correctly
Operating Environment	User's calendar page
Input	1. Press on the desired date 2. Insert new Event's title ("Event Title"), location from the list ("Milano, IT") and adjust the start date ("17/01/2015 12:00") and end date ("17/01/2015 15:00") and the other event's information ("IN-DOOR","PUBLIC"). 3. Press the button "Save"
Expected Result	The Event is successfully saved into the system and displayed to the user's calendar page.
Postconditions	The Event is insert and User views his calendar page
Possible Errors	

2.6 [TC6]: The user search public user's calendar

Goal	View a user's public calendar
Preconditions	• At least a public calendar present in the system
Operating Environment	User's calendar page
Input	1. Press on the search bar 2. Insert the name of the user with the public calendar. 3. Select the desired user from the result clicking on it
Expected Result	The System successfully shows the selected user's calendar with his events.
Postconditions	The User views the selected user's calendar with all his events
Possible Errors	

2.7 [TC7]: The user search public event

Goal	View a public event in the system
Preconditions	• At least a public event present in the system
Operating Environment	User's calendar page
Input	1. Press on the search bar 2. Insert the name of the public event. 3. Select the desired event from the result clicking on it
Expected Result	The Event is successfully displayed to the user's calendar page.
Postconditions	The User views the event information
Possible Errors	

2.8 [TC8]: The user search an owned event

Goal	View an owned event in the system
Preconditions	• At least an owned event present in the system
Operating Environment	User's calendar page
Input	1. Press on the search bar 2. Insert the name of the owned event. 3. Select the desired event from the result clicking on it
Expected Result	The Event is successfully displayed to the user's calendar page.
Postconditions	The User views the event information
Possible Errors	

2.9 [TC9]: The user search a participating event

Goal	View a participating event in the system
Preconditions	• The user is participating at least to one event.
Operating Environment	User's calendar page
Input	1. Press on the search bar 2. Insert the name of the participating event. 3. Select the desired event from the result clicking on it
Expected Result	The Event is successfully displayed to the user's calendar page.
Postconditions	The User views the event information
Possible Errors	

2.10 [TC10]: The user modify personal information

Goal	Successfully change the user information
Preconditions	• The System is installed correctly
Operating Environment	User's setting page
Input	1. Change the desired information 2. Insert the current user password. 3. Press the button "Save"
Expected Result	The information is successfully updated into the system and the system shows to the user a confirm message.
Postconditions	The user information is changed
Possible Errors	• Wrong user password • Wrong modified email • Modified email already present • The new password violate the constraint • Missing required information

2.11 [TC11]: The user modify system settings

Goal	Successfully change the system settings
Preconditions	• The System is installed correctly
Operating Environment	User's setting page
Input	1. Change the desired settings 2. Insert the current user password. 3. Press the button "Save"
Expected Result	The settings is successfully updated into the system and the system shows to the user a confirm message.
Postconditions	The system settings is changed
Possible Errors	• Wrong user password • Missing required information

2.12 [TC12]: The user add a public calendar to preferred

Goal	Add a public calendar to the user's preferred list
Preconditions	• At least a public calendar present in the system
Operating Environment	Public calendar page
Input	1. Press on the star add button
Expected Result	The Calendar is successfully added to the user's preferred list and the system shows to the user a confirm message.
Postconditions	The calendar is added to user's preferred and User still views the public calendar page
Possible Errors	

2.13 [TC13]: The user remove a public calendar from preferred

Goal	Remove a public calendar from the user's preferred list
Preconditions	• At least a public calendar present in the user's preferred
Operating Environment	Public calendar page
Input	1. Press on the star button
Expected Result	The Calendar is successfully removed from the user's preferred list and the system shows to the user a confirm message.
Postconditions	The calendar is removed from user's preferred and User still views the public calendar page
Possible Errors	

2.14 [TC14]: The EO modify an event

Goal	Modify an owned Event in the system
Preconditions	• The user have organised at least one event
Operating Environment	User's calendar page
Input	1. Select the organised event from the calendar 2. Changed the information 3. Press the button "Save"
Expected Result	The Event is successfully modified into the system and displayed to the user's calendar page and the system shows to the user a confirm message.
Postconditions	The Event is modified and User views his calendar page
Possible Errors	• Missing required information

2.15 [TC15]: The EO delete an event

Goal	Delete an Event from the system
Preconditions	• The user have organised at least one event
Operating Environment	User's calendar page
Input	1. Select the organised event 2. Press the "Delete Event" button
Expected Result	The Event is successfully removed from the system,the calendar and the user receive a confirmation message.
Postconditions	The Event is removed and User views his calendar page
Possible Errors	

2.16 [TC16]: The EO send invitation to an owned event

Goal	Send invitation to a user for an owned event
Preconditions	• The user have organised at least one event • At least another user in the system
Operating Environment	User's calendar page
Input	1. Press on the organised event 2. Press the "Add" button 3. Insert the invited user information (First Name, Last Name, email) 4. Select the correct user from the result list 5. Press the button "Add"
Expected Result	The Invited User is successfully added in the list of participants in pending state and the user receive a confirmation message.
Postconditions	The Invited User receive a notification and is added to the event invited users and EO views the event detail page
Possible Errors	• Not valid invited user information

2.17 [TC17]: The EO accept reschedule event for bad weather condition

Goal	Accept the suggestion from the system to reschedule the event for bad weather condition
Preconditions	• There is an OUTDOOR event with bad weather condition within three days
Operating Environment	User's calendar page
Input	1. Press on the detail button of the reschedule event notification 2. Press the button "Accept"
Expected Result	The Event is successfully reschedule into the system and displayed accordingly in the user's calendar page.
Postconditions	The Event is moved and have a good weather condition. If present all the participants receive a new invite for the rescheduled event
Possible Errors	• Not a valid reschedule is possible

2.18 [TC18]: The EO decline reschedule event for bad weather condition

Goal	Decline the suggestion from the system to reschedule the event for bad weather condition
Preconditions	• There is an OUTDOOR event with bad weather condition within three days
Operating Environment	User's calendar page
Input	1. Press on the detail button of the reschedule event notification 2. Press the button "Decline"
Expected Result	The Event is maintained schedule into the system like before and the notification is removed.
Postconditions	The Event is not moved and the notification is cancelled
Possible Errors	

2.19 [TC19]: The EP accept invitation

Goal	The user accepts the event notification invitation
Preconditions	• At least an event in the system not owned by the user
Operating Environment	User's calendar page
Input	1. Press on the detail button of the event notification 2. On the dialog showed press the "Accept" button.
Expected Result	The User participate to the event and the notification is cancelled from the system.
Postconditions	The user is in the event participants list
Possible Errors	

2.20 [TC20]: The EP decline invitation

Goal	The user decline the event notification invitation
Preconditions	• At least an event in the system not owned by the user
Operating Environment	User's calendar page
Input	1. Press on the detail button of the event notification 2. On the dialog showed press the "Decline" button.
Expected Result	The User doesn't participate to the event and the notification is cancelled from the system.
Postconditions	The user isn't in the event participants list
Possible Errors	

2.21 [TC21]: The EP cancel participation

Goal	Cancel a participation from an Event in the system
Preconditions	• The user participates at least one event
Operating Environment	User's calendar page
Input	1. Select the participated event 2. Press the "Cancel Participation" button
Expected Result	The Event is successfully removed from the user participated lists and the user receive a confirmation message.
Postconditions	The Event is removed from the user participated list
Possible Errors	

2.22 [TC22]: The EP accept invitation to a Reschedule Event

Goal	Accept the invitation from the system to a rescheduled event for bad weather condition
Preconditions	• There is an OUTDOOR event with bad weather condition within three days reschedule by the EO
Operating Environment	User's calendar page
Input	1. Press on the detail button of the reschedule event notification 2. Press the button "Accept"
Expected Result	The User participate to the event and the notification is cancelled from the system.
Postconditions	The user is in the event participants list
Possible Errors	• Not a valid reschedule is possible

2.23 [TC23]: The EP decline invitation to a Reschedule Event

Goal	Decline the invitation from the system to a rescheduled event for bad weather condition
Preconditions	• There is an OUTDOOR event with bad weather condition within three days reschedule by the EO
Operating Environment	User's calendar page
Input	1. Press on the detail button of the reschedule event notification 2. Press the button "Decline"
Expected Result	The User doesn't participate to the event and the notification is cancelled from the system.
Postconditions	The user isn't in the event participants list
Possible Errors	

3 Automatic Test

3.1 Unit Test

3.1.1 HandleUserImplTest

TestClass	HandleUserImplTest
TestMethod	testMergeOldUserNewUser()
Preconditions	User used two different social account
Tested Methods	HandleUserImpl.mergeOldUserNewUser(handleUser.em, newUser, oldUser);
Test Flow	• Create new user and old user • Add an event to old user • Merge the two users • check if the event is in new user calendar
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testMergeUserAccount()
Preconditions	User used two different social account
Tested Methods	User HandleUserImpl.mergeUserAccount(newUser, oldUser)
Test Flow	• Create new user and old user • Add account Facebook account information to oldUser • Merge the two users • Check if the returned user have the Facebook account information of old user
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testGetUser()
Preconditions	At least one User in the system
Tested Methods	UserDTO handleUser.getUser(0L)
Test Flow	• Create a user with id 0 and an email • Set the mocked query to return the created user • Check if the returned UserDTO have the same email of the created User.
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testCheckAccessCredential()
Preconditions	At least one User in the system at login or change info/settings
Tested Methods	long handleUser.checkAccessCredential("0", "password")
Test Flow	• Create a user with id 0 and password "password" • Set the mocked query to return the created user • Check if the returned id is 0.
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testAddUser()
Preconditions	
Tested Methods	handleUser.addUser(userInsert)
Test Flow	• Create a UserDTO to insert • Check if the persisted user have the same information of the insert user.
Result	Test passed with doAnswer()

TestClass	HandleUserImplTest
TestMethod	testDoLogin()
Preconditions	At least one User in the system
Tested Methods	boolean handleUser.doLogin(userInsert);
Test Flow	• Create a UserDTO to login • Mock the query to return the user representing the UserDTO • Assert that the method return true
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testGetOwner()
Preconditions	At least one User in the system
Tested Methods	UserDTO = handleUser.getOwner(calendarId)
Test Flow	• Create a User • Mock the query to return the user • Call the method with the user's calendar id • Assert that the UserDTO is equals to the user
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testSearch()
Preconditions	At least one User in the system
Tested Methods	List<ResultDTO> handleUser.search(query)
Test Flow	• Create a User • Mock the query to return the user • Assert that the name of the returned resultDTO is equals to the user first name and last name
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testChangeSettings()
Preconditions	At least one User in the system
Tested Methods	handleUser.changeSettings(loggedUser)
Test Flow	• Create a UserDTO with same id of user • Change the UsetDTO settings • Mock the query to return the user • Assert that the settings of user is equals to the UserDTO settings
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testGetCalendarVisibility()
Preconditions	At least one User in the system
Tested Methods	Visibility handleUser.getCalendarVisibility(calendarId)
Test Flow	• Create a user with public calendar • Mock the query to return the user's calendar • Call the method with the user's calendarId • Assert that the visibility returned is public
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testChangeCalendarVisibility()
Preconditions	At least one User in the system
Tested Methods	handleUser.changeCalendarVisibility(visibility)
Test Flow	• Create a user with public calendar • Mock the query to return the user's calendar • Call the method with the Visibility.PRIVATE • Assert that the visibility of the user's calendar is private.
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testAddNotification()
Preconditions	At least one User and One Event in the system
Tested Methods	handleUser.addNotification(eventNotificationDTO)
Test Flow	• Create a user, an event and eventNotificationDTO. • Mock the query to return the user and the event • Call the method with the eventNotificationDTO as param • Assert that the user have the event notification in his list.
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testSearchUser()
Preconditions	At least one User in the system
Tested Methods	List<ResultDTO> handleUser.searchUser("")
Test Flow	• Create a User • Mock the query to return the user • Assert that the name of the returned resultDTO is equals to the user first name and last name
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testAcceptNotification()
Preconditions	At least one User and One Event in the system
Tested Methods	handleUser.acceptNotification(eventNotificationDTO)
Test Flow	• Create a user, an event, an eventNotification and DTO. • Mock the query to return the eventNotification • Call the method with the eventNotificationDTO as param • Assert that the user dosen't have the notification in his list • Assert that the event have the user in the participants list
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testDeclineNotification()
Preconditions	At least one User and One Event in the system
Tested Methods	handleUser.declineNotification(eventNotificationDTO)
Test Flow	• Create a user, an event, an eventNotification and DTO • Mock the query to return the eventNotification • Call the method with the eventNotificationDTO as param • Assert that the user dosen't have the notification in his list • Assert that the event dosen't have the user in the participants list
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testAddPreferedCalendar()
Preconditions	At least two User in the system
Tested Methods	handleUser.addPreferedCalendar(1L)
Test Flow	• Create a user, and a calendar of another user • Mock the query to return the user and the calendar • Call the method with the id of the calendar • Assert that the user preferred calendars list contains the calendar
Result	Test passed

TestClass	HandleUserImplTest
TestMethod	testDelPreferedCalendar()
Preconditions	At least two User in the system
Tested Methods	handleUser.delPreferedCalendar(1L)
Test Flow	• Create a user with a preferred calendar • Mock the query to return the user and the calendar • Call the method with the id of the calendar • Assert that the user preferred calendars list doesn't contains the calendar
Result	Test passed

3.1.2 HandleEventImplTest

TestClass	HandleEventImplTest
TestMethod	testAddEvent()
Preconditions	
Tested Methods	long handleEvent.addEvent(user.getId(), eventDTO)
Test Flow	• Create a User and an eventDTO to insert • Check if the persisted event have the same information of the insert event.
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testGetEvents()
Preconditions	At least one User and One Event in the system
Tested Methods	List<EventDTO> handleEvent.getEvents(user.getId())
Test Flow	• Create a user and an event • Set the mocked query to return the created user • Check if the returned UserDTO have the same email of the created User.
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testGetEvent()
Preconditions	At least one User and One Event in the system
Tested Methods	EventDTO handleEvent.getEvent(user.getId(), eventDTO.getId())
Test Flow	• Create a user and an event • Set the mocked query to return the created event • Check if the returned EventDTO have the same id of the created Event.
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testUpdateEvent()
Preconditions	At least one User and One Event in the system
Tested Methods	long handleEvent.updateEvent(user.getId(), eventDTO)
Test Flow	• Create a user and an event and a modified eventDTO • Set the mocked query to return the created event • Check if the returned id is the same have the event have the same attribute of eventDTO.
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testRemoveEvent()
Preconditions	At least one User and One Event in the system
Tested Methods	handleEvent.removeEvent(user.getId(), eventDTO)
Test Flow	• Create a user and an event • Set the mocked query to return the created event • Check if the user doesn't have the event in his organized list
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testCancelEvent()
Preconditions	At least one User and One Event in the system
Tested Methods	handleEvent.cancelEvent(ep.getId(), eventDTO)
Test Flow	• Create a user and an event • Set the mocked query to return the created event • Check if the user doesn't have the event in his participated list
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testSearch()
Preconditions	At least One Event in the system
Tested Methods	List<ResultDTO> handleEvent.search(event.getName())
Test Flow	• Create an event and a resultDTO of that event • Set the mocked query to return the created event • Check if the returned resultDTO is equals
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testMoveEvent()
Preconditions	At least One Event in the system
Tested Methods	handleEvent.moveEvent(eventDTO.getId(), dayDelta, minuteDelta);
Test Flow	• Create an event • Set the mocked query to return the created event • Call the method with dayDelta and/or minuteDelta • Check if the event startdate and enddate is moved respecting the parameter passed
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testResizeEvent()
Preconditions	At least One Event in the system
Tested Methods	handleEvent.resizeEvent(eventDTO.getId(), dayDelta, minuteDelta)
Test Flow	• Create an event • Set the mocked query to return the created event • Call the method with dayDelta and/or minuteDelta • Check if the event enddate is moved respecting the parameter passed
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testAddParticipant()
Preconditions	At least one User and One Event in the system
Tested Methods	handleEvent.addParticipant(eventDTO.getId(), selectedResult)
Test Flow	• Create an event and a selectedResult (the user) • Set the mocked query to return the created event • Check if the user is added to the event participants
Result	Test passed

TestClass	HandleEventImplTest
TestMethod	testCheckEventWeatherCondition()
Preconditions	At least One Event with bad weather forecasted
Tested Methods	handleEvent.checkEventWeatherCondition(user.getId())
Test Flow	Setup the test need to verify
Result	Test passed but non completed

3.1.3 HandleForecastImplTest

TestClass	HandleForecastImplTest
TestMethod	testGetForecast_String_Date()
Preconditions	At least one forecast in the system
Tested Methods	ForecastDTO handleForecast.getForecast(location, date)
Test Flow	• Create a forecast • Set the mocked query to return the created forecast • Verify the forecast returned is correct
Result	Test passed

TestClass	HandleForecastImplTest
TestMethod	testGetForecasts()
Preconditions	At least one forecast in the system
Tested Methods	List<ForecastDTO> handleForecast.getForecasts(location)
Test Flow	• Create a forecast • Set the mocked query to return the created forecast • Verify the forecasts returned are correct
Result	Test passed

TestClass	HandleForecastImplTest
TestMethod	testGetForecast_long()
Preconditions	At least one forecast in the system
Tested Methods	ForecastDTO handleForecast.getForecast(idForecast)
Test Flow	• Create a forecast • Set the mocked query to return the created forecast • Verify the forecast returned is correct
Result	Test passed

TestClass	HandleForecastImplTest
TestMethod	testRemoveOldForecast()
Preconditions	At least one event with one forecast in the system
Tested Methods	handleForecast.removeOldForecast()
Test Flow	• Create a forecast related to an event • Set the mocked queries to return the created forecast and the event • Verify the forecast is removed and the event have another created forecast
Result	Test passed

TestClass	HandleForecastImplTest
TestMethod	testSearchLocation()
Preconditions	At least one location in the system
Tested Methods	List<String> handleForecast.searchLocation(queryString)
Test Flow	• Create a Location • Set the mocked query to return the created location • Assert that the returned list contains the location name
Result	Test passed

3.2 Integration Test

We only correctly setup the integration test environment using Arquillian and H2 for the database because with our entities we had problem with derby database (User entity in particular). In the source code you find the arquillian.xml and glassfish-resources.xml with the relative settings.

TestClass	HandleEventImplIT
TestMethod	UserManagerShouldBeInjected()
Test Flow	assertNotNull(handleEvent)
Result	Test passed

TestClass	HandleEventImplIT
TestMethod	EntityManagerShouldBeInjected()
Test Flow	assertNotNull(em)
Result	Test passed