POLITECNICO
MILANO 1863

MyTaxiService

Integration Test Plan Document

Davide Citterio, Lorenzo Cunial, Massimo Beccari

January 20, 2016

Contents

1 Introduction

1.1 Revision History

This is the version 1.0 of the document.

1.2 Purpose and Scope

The purpose of this document is to provide a testing plan to integrate all the components of MyTaxyService definded on the Design Document, according to the Requirements Analysis and Specification Document, in particular to verify that every interface between components works correctly. The scope is to test the behavior of the system in a fast and accurate way to reach the goals of the application, reported into the RASD. Below you can find all the single integration test goal description, the environments needed and the strategies that have to be used.

1.3 List of Definitions and Abbreviations

1.3.1 Definitions

All the definitions needed are contained in rasd12.pdf.

1.3.2 Abbreviations

Abbreviation	Definition
DBMS	Database management system
SMS	Short Message Service
API	Application programming interface
MTS	My Taxy Service

1.4 List of Reference Documents

1.4.1 Reference documents

[ESA]	ESA Software Engineering Standards (ESA PSS-05-0 Issue 2), ESA Board for Software Standardization and Control (BSSC), 1991
[SVVP]	Software Verification and Validation Plan, SPINGRID team, TU/e, Version 0.1.2, March 2006

1.4.2 Applicable documents

[RASD]	Require Analysis and Specification Document https://github.com/davidecitterio/IngSw2/blob/master/Deliveries/rasd12.pdf
[DD]	Design Document https://github.com/davidecitterio/IngSw2/blob/master/Deliveries/dd10.pdf

2 Integration Strategy

2.1 Entry Criteria

We assume that for each stand-alone component developers has to provide at least 95% of unit test coverage. Moreover we assume that GoogleMaps API, SMS service and DBMS system that are integrated and used by the components, work correctly and are in a stable version. The integration test between components can be done if each component involved are completely developed and ready to use to reach their goals. Otherwise integration test can be made the same using stubs specifically made.

2.2 Elements to be integrated

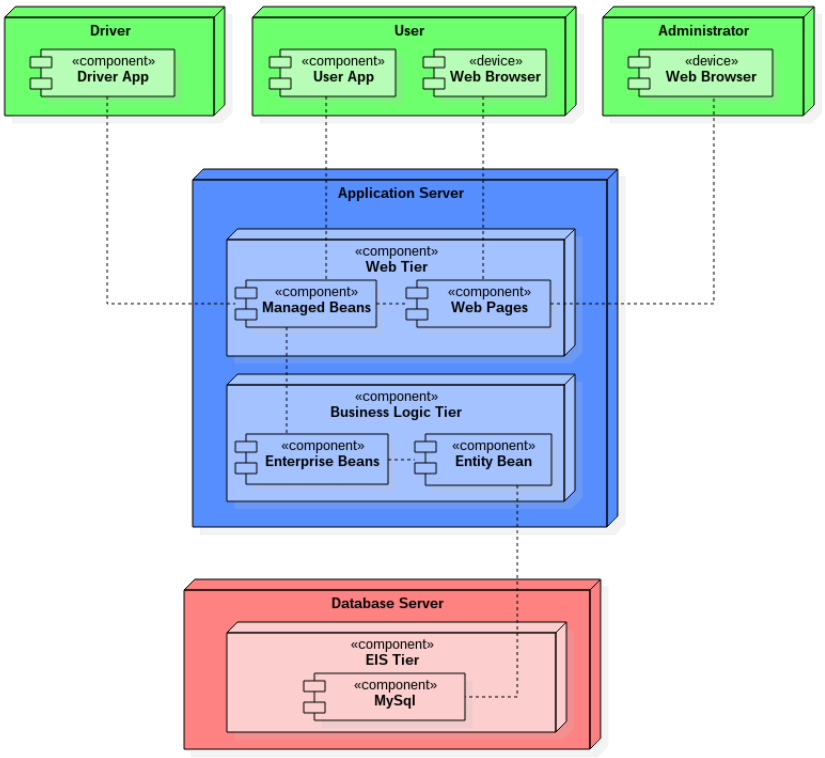

Figure 1: components overview

Referring to our design document, in particular to the "high level component and their interaction" schema, we identify the following components:

- driver application;
- user application;
- user web browser;
- administrator web browser;
- web pages and related managed beans;
- business logic;
- database.

Note that the integration between the managed beans and the business logic is implicit: they are on the same device (the application server), and so their interaction is simple: when a user/driver/administrator requests an operation, the managed bean that receives the request calls the appropriate function of the business logic subsystem (a function contained in UserOperationBean/DriverOperationBean/AdminOperationBean respectively).

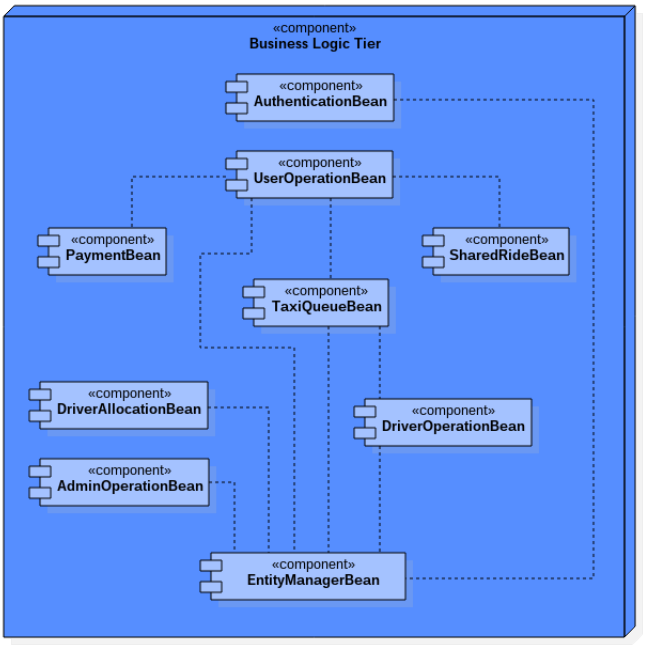

Figure 2: business logic components

About the Business Logic, we properly managed it like a subsystem. In it we can identify other sub-components:

- AuthenticationBean;
- UserOperationBean;
- TaxiQueueBean;
- PaymentBean;
- SharedDriveBean;
- DriverAllocationBean;
- DriverOperationBean;
- AdminOperationBean;
- EntityManagerBean.

2.3 Integration Testing Strategy

About the integration of the components we decided to use a bottom up approach. This choice is due to the architecture of MTS System, based on high modularity. Bottom up approach increase efficiency of entire process, and guarantee easy and faster testing phase. First of all, it has to be integrate the server side subsystem and test it, in order to let the core of MTS do its job, after that the integration with the client side of application (Components used by drivers, users and administrator) could be possible and more simple due to the correct work of the server side. In the first step, server integration, in order to allow the communication between all the external components, we use drivers that permit us to simulate the behavior (in particular the response) of real components not yet tested and integrated but that have to be connected to the server side. We adopt this two-step strategy in order to not develop more drivers for the client-side testing phase that allow to saving time. We assume that Managed Beans and Web Page components, due to the same level in the architecture, are sufficiently tested by appropriated unit tests. So the integration test is not necessary.

2.4 Sequence of Component/Function Integration

2.4.1 Software Integration Sequence

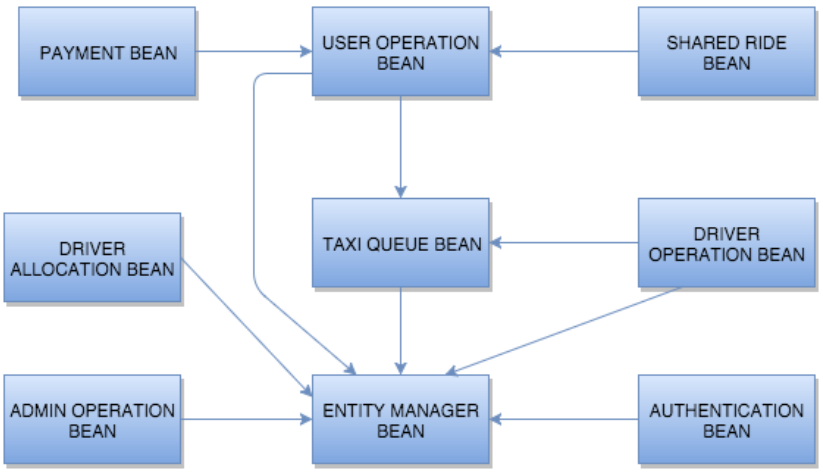

Figure 3: business logic integration

In the following two paragraphs, the integration test procedures and cases relative to the Business Logic subsystem are listed. The procedures are going to be tested in numerical order (TP1 → TP2 → TP3) and each of them tells which and in which order the single test cases are going to be executed.

Business Logic subsystem: integration test procedures

ID	Integration Test Procedure	Paragraphs
TP1	Test the User's operation beans	3.1.1
TP2	Test the Driver's operation beans	3.1.2
TP3	Test the Admin's operation beans	3.1.3

Business Logic subsystem: integration test cases

ID	Integration Test Case	Paragraphs
I1	UserOperationBean → EntityManagerBean	3.1.4
I2	AuthenticationBean → EntityManagerBean	3.1.5
I3	UserOperationBean → EntityManagerBean, TaxiQueueBean, PaymentBean	3.1.6
I4	UserOperationBean → EntityManagerBean, TaxiQueueBean, PaymentBean, SharedRideBean	3.1.7
I5	TaxiQueueBean → DriverOperationBean, EntityManagerBean	3.1.8
I6	DriverOperationBean → EntityManagerBean	3.1.9
I7	DriverAllocationBean → EntityManagerBean	3.1.10
I8	AdminOperationBean → EntityManagerBean	3.1.11

2.4.2 Subsystem Integration Sequence

In the following schemas we identify components and their interaction.

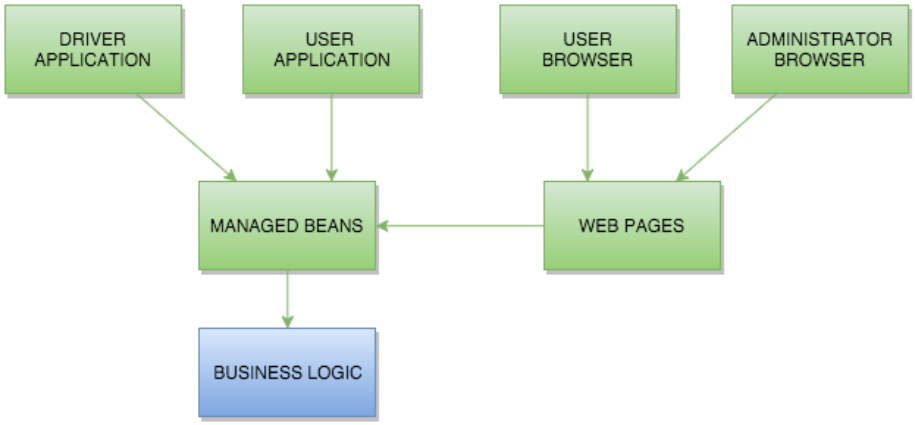

Figure 4: front-end components integration

Integration test procedures

ID	Integration Test Procedure	Paragraphs
TP4	Test the Driver's frontend part	3.2.1
TP5	Test the Admin's frontend part	3.2.2
TP6	Test the User's frontend part (Web Browser)	3.2.3
TP7	Test the User's frontend part (Web Browser)	3.2.4
TP8	Test the User's frontend part (Application)	3.2.5
TP9	Test the User's frontend part (Application)	3.2.6

Integration test cases

ID	Integration Test Case	Paragraphs
I9	Driver Application → Web Pages	3.2.7
I10	Driver Application → Web Pages	3.2.8
I11	Driver Application → Web Pages	3.2.9
I12	Driver Application → Web Pages	3.2.10
I13	Driver Application → Web Pages	3.2.11
I14	Driver Application → Web Pages	3.2.12
I15	Admin Web Browser → Managed Beans	3.2.13
I16	Admin Web Browser → Managed Beans	3.2.14
I17	Admin Web Browser → Managed Beans	3.2.15
I18	Admin Web Browser → Managed Beans	3.2.16
I19	Admin Web Browser → Managed Beans	3.2.17
I20	Admin Web Browser → Managed Beans	3.2.18
I21	User Web Browser → Web Pages	3.2.19
I22	User Web Browser → Web Pages	3.2.20
I23	User Web Browser → Web Pages	3.2.21
I24	User Web Browser → Web Pages	3.2.22
I25	User Web Browser → Web Pages	3.2.23
I26	User Web Browser → Web Pages	3.2.24
I27	User Web Browser → Web Pages	3.2.25
I28	User Web Browser → Web Pages	3.2.26
I29	User Web Browser → Web Pages	3.2.27
I30	User Mobile App → Managed Beans	3.2.28
I31	User Mobile App → Managed Beans	3.2.29
I32	User Mobile App → Managed Beans	3.2.30
I33	User Mobile App → Managed Beans	3.2.31
I34	User Mobile App → Managed Beans	3.2.32
I35	User Mobile App → Managed Beans	3.2.33
I36	User Mobile App → Managed Beans	3.2.34
I37	User Mobile App → Managed Beans	3.2.35

3 Individual Steps and Test Description

3.1 Business Logic

3.1.1 Integration Test Procedure TP1

Test Procedure Identifier	TP1
Purpose	This test procedure verifies that all the enterprise beans relative to user's operations work (for the list of user's operations see the RASD, page 12)
Procedure Steps	I1 → I2 → I6 → I5 → I3 → I4

3.1.2 Integration Test Procedure TP2

Test Procedure Identifier	TP2
Purpose	This test procedure verifies that all the enterprise beans relative to driver's operations work (for the list of driver's operations see the RASD, page 12)
Procedure Steps	(I6 →)* I7

***NOTE**: even if the I6 test case regards the driver, the user also needs it to complete his operations. So, since the procedure TP1 is executed before procedure TP2, at the time TP2 runs I6 is alredy been executed and it's not needed to run it again.

3.1.3 Integration Test Procedure TP3

Test Procedure Identifier	TP3
Purpose	This test procedure verifies that all the enterprise beans relative to administrator's operations work (for the list of admin's operations see the RASD, pages 12-13)
Procedure Steps	I8

3.1.4 Integration Test Case I1

Test Case Identifier	I1T1
Test Item(s)	UserOperationBean → EntityManagerBean
Input Specification	Create typical registration data input
Output Specification	Check that the system records the user
Environmental Needs	None

NOTE: this test case is repeated twice, because the shared ride tests needs a least two users.

3.1.5 Integration Test Case I2

Test Case Identifier	I2T1
Test Item(s)	AuthenticationBean → EntityManagerBean
Input Specification	Create typical log in data input
Output Specification	Check that the system grants the access
Environmental Needs	I1 succeeded, recorded data of a driver and an administrator

NOTE: this test case is repeated three times: once for an user, once for a driver and once for an administrator.

3.1.6 Integration Test Case I3

Test Case Identifier	I3T1
Test Item(s)	UserOperationBean → EntityManagerBean, TaxiQueueBean, PaymentBean
Input Specification	Create typical reservation data input
Output Specification	Check that the system records the reservation, requests a driver for the reservation, calls the (external) SMS service function to notify the user, calls the PaymentBean function to calculate the fee
Environmental Needs	I5 succeeded

3.1.7 Integration Test Case I4

Test Case Identifier	I4T1
Test Item(s)	UserOperationBean → EntityManagerBean, TaxiQueueBean, PaymentBean, SharedRideBean
Input Specification	Create typical reservation data input
Output Specification	Check that the system records the reservation, calls TaxiQueueBean to request a driver for the reservation, calls the (external) SMS service function to notify the user, calls PaymentBean to calculate the fee, calls SharedRideBean to check the compatibility with other rides
Environmental Needs	I5 succeeded, recorded data of another compatible reservation

3.1.8 Integration Test Case I5

Test Case Identifier	I5T1
Test Item(s)	TaxiQueueBean → DriverOperationBean, EntityManagerBean
Input Specification	Create typical data input for driver request
Output Specification	Check that the system finds an available driver for the ride
Environmental Needs	I6 succeeded, recorded data of a driver, Driver stub

3.1.9 Integration Test Case I6

Test Case Identifier	I6T1
Test Item(s)	DriverOperationBean → EntityManagerBean
Input Specification	Create typical data input for driver status
Output Specification	Check that the system update the driver status
Environmental Needs	Recorded data of a driver

Test Case Identifier	I6T2
Test Item(s)	DriverOperationBean → EntityManagerBean
Input Specification	Create typical data input for driver answer
Output Specification	Check that the system manage the driver answer
Environmental Needs	Recorded data of two drivers and a request

3.1.10 Integration Test Case I7

Test Case Identifier	I7T1
Test Item(s)	DriverAllocationBean → EntityManagerBean
Input Specification	Create typical data input for allocation request
Output Specification	Check that the system finds a city zone for the driver
Environmental Needs	None

3.1.11 Integration Test Case I8

Test Case Identifier	I8T1
Test Item(s)	AdminOperationBean → EntityManagerBean
Input Specification	Create typical system settings data input
Output Specification	Check that the system update the system settings
Environmental Needs	Recorded data of system settings

Test Case Identifier	I8T2
Test Item(s)	AdminOperationBean → EntityManagerBean
Input Specification	Create typical data input for admin request
Output Specification	Check that the system finds and returns the requested data
Environmental Needs	Recorded data of system settings and drivers

3.2 Front-end components

3.2.1 Integration Test Procedure TP4

Test Procedure Identifier	TP4
Purpose	This test procedure verifies that the admin's frontend part works correctly (for the list of admin's operations see the RASD, page 12-13)
Procedure Steps	I9 → I12 → I13 → I14 → I11 → I10

3.2.2 Integration Test Procedure TP5

Test Procedure Identifier	TP5
Purpose	This test procedure verifies that the driver's frontend part works correctly (for the list of driver's operations see the RASD, page 12)
Procedure Steps	I15 → I20 → I17 → I16 → I18 → I19

3.2.3 Integration Test Procedure TP6

Test Procedure Identifier	TP6
Purpose	This Procedure verify if the user operation of request and reservation work correctly
Procedure Steps	I22 → I24 → I25 → I27 → I23

3.2.4 Integration Test Procedure TP7

Test Procedure Identifier	TP7
Purpose	This Procedure verify if the user is correctly able to change his settings
Procedure Steps	I21 → I26 → I28 → I23

3.2.5 Integration Test Procedure TP8

Test Procedure Identifier	TP8
Purpose	This Procedure verify if the user operation of request and reservation work correctly
Procedure Steps	I31 → I33 → I34 → I36 → I32

3.2.6 Integration Test Procedure TP9

Test Procedure Identifier	TP9
Purpose	This Procedure verify if the user is correctly able to change his settings
Procedure Steps	I30 → I35 → I37 → I32

3.2.7 Integration Test Case I9

Test Case Identifier	I9T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Fulfill login form
Output Specification	Successful login
Environmental Needs	Device

Test Case Identifier	I9T2
Test Item(s)	User Web Browser → Web Pages
Input Specification	Insert wrong login-data
Output Specification	Login Not permitted
Environmental Needs	Device

3.2.8 Integration Test Case I10

Test Case Identifier	I10T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Make logout
Output Specification	Successful logout
Environmental Needs	Device,I9 succeeded

3.2.9 Integration Test Case I11

Test Case Identifier	I11T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Set ready status
Output Specification	Correct response to the request
Environmental Needs	Device,I13 succeeded

3.2.10 Integration Test Case I12

Test Case Identifier	I12T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Accept a request
Output Specification	Correct response to the request
Environmental Needs	Device,I9 succeeded

Test Case Identifier	I12T2
Test Item(s)	Driver Application → Web Pages
Input Specification	Discard a request
Output Specification	Correct response to the request
Environmental Needs	Device,I9 succeeded

3.2.11 Integration Test Case I13

Test Case Identifier	I13T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Request for showing price
Output Specification	Correct response to the request
Environmental Needs	Device,I12 succeeded

3.2.12 Integration Test Case I14

Test Case Identifier	I14T1
Test Item(s)	Driver Application → Web Pages
Input Specification	Request to proceed serving another user
Output Specification	Correct response to the request
Environmental Needs	Device,I13 succeeded

3.2.13 Integration Test Case I15

Test Case Identifier	I15T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Correct login request
Output Specification	Successful login
Environmental Needs	Web Browser

3.2.14 Integration Test Case I16

Test Case Identifier	I16T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Request for change some fees
Output Specification	Correct response to the request
Environmental Needs	Web Browser,I15 succeeded

3.2.15 Integration Test Case I17

Test Case Identifier	I17T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Request for add/delete driver
Output Specification	Correct response to the request
Environmental Needs	Web Browser,15 succeeded

3.2.16 Integration Test Case I18

Test Case Identifier	I18T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Request for disable/enable some components
Output Specification	Correct response to the request
Environmental Needs	Web Browser,15 succeeded

3.2.17 Integration Test Case I19

Test Case Identifier	I19T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Request for showing past/scheduled/current rides
Output Specification	Correct response to the request
Environmental Needs	Web Browser,15 succeeded

3.2.18 Integration Test Case I20

Test Case Identifier	I20T1
Test Item(s)	Admin Web Browser → Managed Beans
Input Specification	Request for showing the map of active taxies
Output Specification	Correct response to the request
Environmental Needs	Web Browser,15 succeeded

3.2.19 Integration Test Case I21

Test Case Identifier	I21T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Fulfill sign-in form
Output Specification	Successful registration
Environmental Needs	Web Browser

3.2.20 Integration Test Case I22

Test Case Identifier	I22T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Fulfill login form
Output Specification	Successful login
Environmental Needs	Web Browser

Test Case Identifier	I22T2
Test Item(s)	User Web Browser → Web Pages
Input Specification	Insert wrong login-data
Output Specification	Login Not permitted
Environmental Needs	Web Browser

3.2.21 Integration Test Case I23

Test Case Identifier	I23T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Make Logout
Output Specification	Successful logout
Environmental Needs	Web Browser

3.2.22 Integration Test Case I24

Test Case Identifier	I24T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Fulfill request form
Output Specification	Successful request and sms sent
Environmental Needs	Web Browser, Mobile Phone

Test Case Identifier	I24T2
Test Item(s)	User Web Browser → Web Pages
Input Specification	Compile request form with no valid data
Output Specification	Error of request
Environmental Needs	Web Browser

3.2.23 Integration Test Case I25

Test Case Identifier	I25T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Fulfill reservation form
Output Specification	Successful reservation
Environmental Needs	Web Browser

Test Case Identifier	I25T2
Test Item(s)	User Web Browser → Web Pages
Input Specification	Fulfill reservation form with sharing
Output Specification	Successful reservation and correct managing of shared option
Environmental Needs	Web Browser

3.2.24 Integration Test Case I26

Test Case Identifier	I26T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Change some user setting
Output Specification	Successful changing and save new values
Environmental Needs	Web Browser

3.2.25 Integration Test Case I27

Test Case Identifier	I27T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Change reservation already made
Output Specification	Successful changing
Environmental Needs	Web Browser

3.2.26 Integration Test Case I28

Test Case Identifier	I28T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Request chronology of action
Output Specification	Correct response for the request
Environmental Needs	Web Browser

3.2.27 Integration Test Case I29

Test Case Identifier	I29T1
Test Item(s)	User Web Browser → Web Pages
Input Specification	Request chronology of action
Output Specification	Correct response for the request
Environmental Needs	Web Browser

3.2.28 Integration Test Case I30

Test Case Identifier	I30T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Fulfill sign-in form
Output Specification	Successful registration
Environmental Needs	User Smartphone

3.2.29 Integration Test Case I31

Test Case Identifier	I31T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Fullfill login form
Output Specification	Successful login
Environmental Needs	User Smartphone

Test Case Identifier	I31T2
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Insert wrong login-data
Output Specification	Login Not permitted
Environmental Needs	User Smartphone

3.2.30 Integration Test Case I32

Test Case Identifier	I32T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Make Logout Successful logout
Output Specification	Successful logout
Environmental Needs	User Smartphone

3.2.31 Integration Test Case I33

Test Case Identifier	I33T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Fullfill request form
Output Specification	Successful request and sms sent
Environmental Needs	User Smartphone

Test Case Identifier	I33T2
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Compile request form with no valid data
Output Specification	Error of request
Environmental Needs	User Smartphone

3.2.32 Integration Test Case I34

Test Case Identifier	I34T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Fulfill reservation form
Output Specification	Successful reservation
Environmental Needs	User Smartphone

Test Case Identifier	I34T2
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Fulfill reservation form with sharing
Output Specification	Successful reservation and correct managing of shared option
Environmental Needs	User Smartphone

3.2.33 Integration Test Case I35

Test Case Identifier	I35T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Change some user setting
Output Specification	Successful changing and save new values
Environmental Needs	User Smartphone

3.2.34 Integration Test Case I36

Test Case Identifier	I36T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Change reservation already made
Output Specification	Successful changing
Environmental Needs	User Smartphone

3.2.35 Integration Test Case I37

Test Case Identifier	I37T1
Test Item(s)	User Mobile App → Managed Beans
Input Specification	Request chronology of action
Output Specification	Correct response for the request
Environmental Needs	User Smartphone

4 Tools and Test Equipment Required

We have decided to use a testing automated tool, like JUnit, for the testing of the Business Logic subsystem, and manual testing for the integration of the system because the test cases are not so big and limited.

About the tests environment needs, we assume that each test is made in the environment indicated. In particular for the test involved the user application we need that this app runs into the different mobile OS (as said in DD document). So it has to run on Android OS version 4.0, 5.0 and 6.1 to verify that each release support the app, on Apple iOS version 8 and 9, same aim as android one. About the user and administrator interaction with web pages, the application has to be tested on the most common web browser, so Google Chrome (version 46), Mozilla Firefox (version 43), Apple Safari (version 9 for Mac, version 5 for Windows PC), Opera (version 15). Driver application, according to design document, has to be tested only on android device, with same version of user application one. We suggest to test browser application on different operating system, so at least on Microsoft Windows (version 8.x and 10.x), Apple OSX (10.10 Yosemite and 10.11 El Captain), and at least on one Linux Release like Ubuntu 15.x (the most common). About the server side, it has to be tested on the cloud machine that will host that. In particular provider has to guarantee same situation of network traffic as the usual behavior of the system. Database is hosted in MySql database (version 5.7.x) running on linux server, and business logic is running on GlassFish server version 4.4.x.

5 Program Stubs and Test Data Required

As said above, the integration strategy, consisting in 2 phases, requires drivers only in the first step about the server integration, since the second step is performed by manual testing.
So, as declared in the Environmental Needs of test case I5, it's only needed a Driver stub.

The test data required are specified in the Environmental Needs of each test case.
According to Design Document, test data are stored in MySql database structured as ER diagram report.

6 Work time

To produce this document, each member have worked around 11 hours.