myTaxiService

Design Document

Belluschi Marco 791878, Cerri Stefano 849945, Di Febbo Francesco 852389

November 23, 2015

Contents

Chapter 1

Introduction

1.1 Purpose

This document contains the complete design description of myTaxiService. This includes the architectural features of the system down through details of what operations each code module will perform and the database layout. It also shows how the use cases detailed in the RASD will be implemented in the system using this design. The primary audiences of this document are the software developers.

1.2 Scope

MyTaxiService is a taxi service for a large city. The main goals of the system are:

- simplify the access of passengers to the service
- guarantee a fair management of taxi queues

The system architecture will be a three-tier architecture: client, server application and server database. It will be created by using the MVC architectural pattern.

The system will be divided into components with respect to the principles leading to good design:

- Each individual component will be smaller in order to be easier to understand
- Coupling will be reduce where possible
- Reusability and flexibility will be increase in order to make easier future implementation

The system will have efficient algorithm in order to increase its performance; in the document will be given special attention to the sharing algorithm.

1.3 Definitions, Acronyms, Abbreviations

Definitions

- User: person that uses the service applications
- Visitor: user that has not registered nor logged in
- Registered user: user that has registered to the service
- Passenger: passenger registered to the service
- Taxi driver: taxi driver registered to the service
- System: the union of software and hardware to be developed and implemented

Acronyms

- RASD: requirements analysis and specification document
- AES: Advanced Encryption Standard
- FIFO: First In First Out
- ETA: estimated time of arrival
- API: application programming interface
- GPS: Global Positioning System
- MVC: Model View Controller
- UX: User Experience

1.4 Reference Documents

- Software Engineering 2 Project AA 2015/2016: Project Description And Rules
- Software Engineering 2 Project AA 2015/2016: Assignments 1 and 2 (RASD and DD)
- Software Engineering 2 Project AA 2015/2016: Structure of the design document

1.5 Document Structure

This document is essentially structured in seven parts:

- Introduction: it gives a description of the document and some basical information about the system design and archicture.
- Architectural Design: This is the core of the document. It gives general information about the architectural design. It also describes how the system will be divided into components and how the components communicate. It also has a description of the design pattern and architectural styles that will be used.
- Algorithm Design: it gives a description of the main algorithm that will be implemented. More focus will be given in the sharing algorithm.
- User Interface Design: it gives a description of the user interfaces of the system and the flow from one interface to another.
- Requirements Traceability: this section documents the life of a requirement and provides bi-directional traceability between various associated requirements.
- References: it gives information on the guidelines used in order to redact this document.
- Appendix: it provides informations that are not considered part of the actual DD. It includes: software and tools used, project group organization.

Chapter 2

Architectural Design

2.1 Overview

This chapter describes the software system, the relationships between software components and the relationship to actors with the system. Each component is described by a specification and an interface design. The specification is a description of its purpose, its functionality, its attributes (including dependency on other components) and the constraints under which it must operate. It also describes resources, that is, any elements used by the component which are external to the design such as physical devices and software services. The interface design is the list of the services that it provides to clients. These services are methods (procedures and functions), each carefully documented.
Each component in turn may provide its services by having an internal architectural design with its own set of subordinate components. These components may be called sub-components. The decomposition of a higher-level component into subordinate component must be explicit. The algorithm that shows how each method of the larger component is performed by these components must be explicit. Any data stored in an component must be explicitly described.

2.2 High level components and their interaction

DBMS: this component provides access to all of the data contained in the database. It provides various functions that allow entry, storage and retrieval of large quantities of information and provides ways to manage how that information is organized.

TaxiQueue this component manages the queue of available taxis for each city zone.

Request this component manages the incoming requests from passengers and responses from taxi drivers. It manages all types of requests, including standard, shared and reserved rides thanks to the interaction with TaxiQueues component. It takes care of messages dispatching to users.

MapServices

UserManager

2.3 Component view

2.4 Deployment view

2.5 Runtime view

2.6 Component interfaces

2.7 Selected architectural styles and patterns

2.8 Other design decisions

Chapter 3

Algorithm Design

Chapter 4

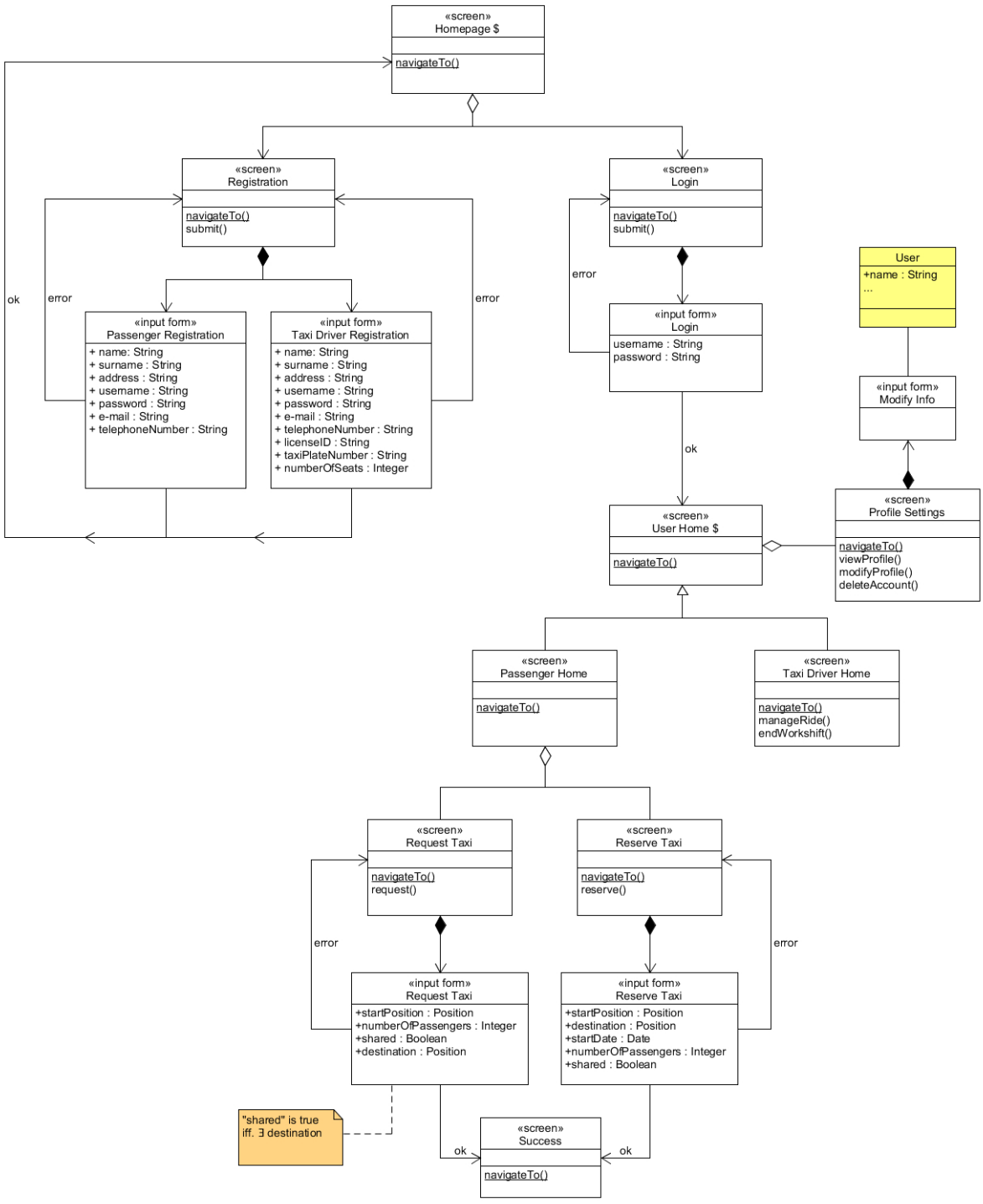
User Interface Design

4.1 Purpose

This chapter gives a description of the user interfaces of the system and the flow from one interface to another. The description includes an UX diagram and some mockups in order to understand how a user can do actions using the interface given by the system.

4.2 UX diagram

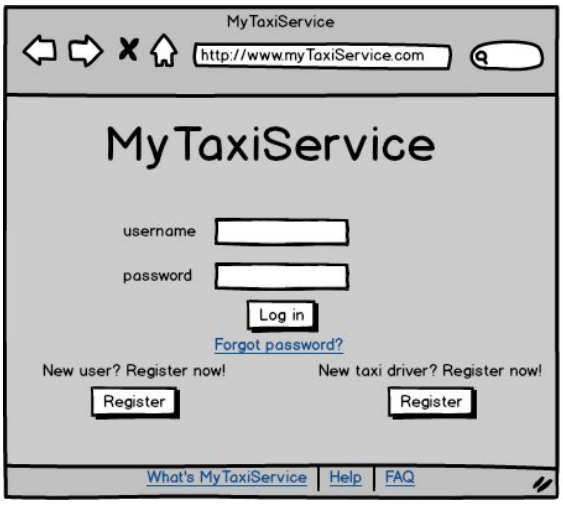
Here is the UX diagram:


«screen»
Homepage $
navigateTo()
«screen»
Registration
navigateTo()
submit()
«screen»
Login
navigateTo()
submit()
ok
error
error
error
«input form»
Passenger Registration
+ name: String
+ surname : String
+ address : String
+ username : String
+ password : String
+ e-mail : String
+ telephoneNumber : String
«input form»
Taxi Driver Registration
+ name: String
+ surname : String
+ address : String
+ username : String
+ password : String
+ e-mail : String
+ telephoneNumber : String
+ licenseID : String
+ taxiPlateNumber : String
+ numberOfSeats : Integer
«input form»
Login
username : String
password : String
ok
User
+name : String
...
«input form»
Modify Info
«screen»
User Home $
navigateTo()
«screen»
Profile Settings
navigateTo()
viewProfile()
modifyProfile()
deleteAccount()
«screen»
Passenger Home
navigateTo()
«screen»
Taxi Driver Home
navigateTo()
manageRide()
endWorkshift()
«screen»
Request Taxi
navigateTo()
request()
«screen»
Reserve Taxi
navigateTo()
reserve()
error
error
«input form»
Request Taxi
+startPosition : Position
+numberOfPassengers : Integer
+shared : Boolean
+destination : Position
«input form»
Reserve Taxi
+startPosition : Position
+destination : Position
+startDate : Date
+numberOfPassengers : Integer
+shared : Boolean
"shared" is true
iff. ∃ destination
«screen»
Success
navigateTo()
ok
ok

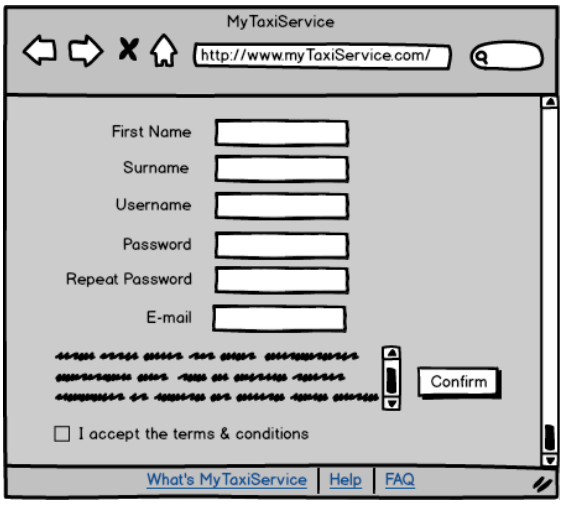
4.3 mockups

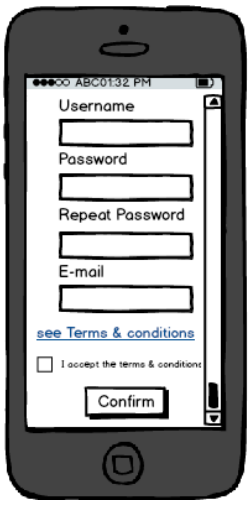
Here we will presented some mockups of MyTaxiService. Some of them referring to the RASD document Section 3.1.1 User Interface.

Log in: In the figure below is shown MyTaxiService's homepage

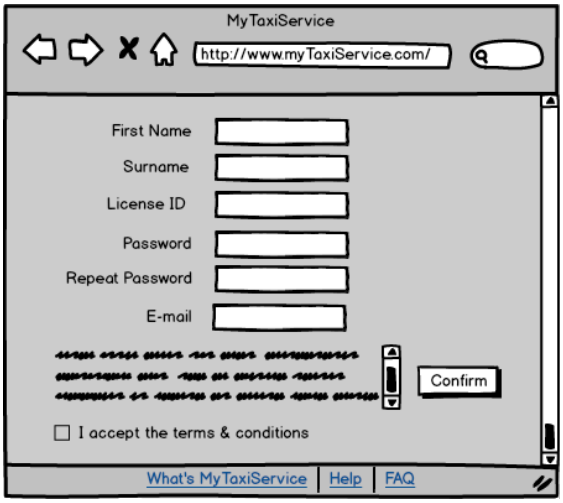


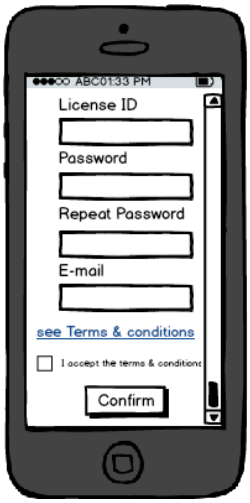
Registration passenger: View of the visitor that wants to register as a passenger



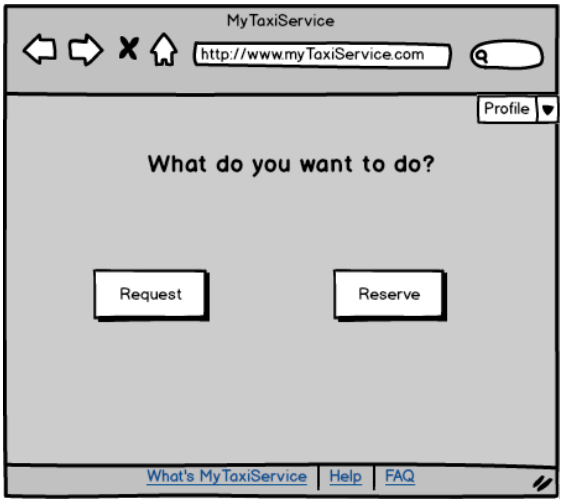


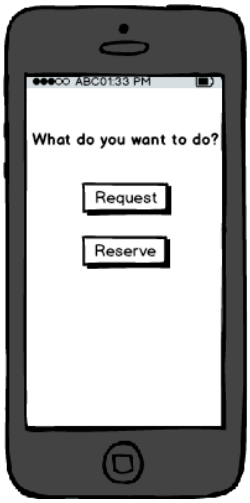
Registration taxi Driver: View of the visitor that wants to register as a taxi driver



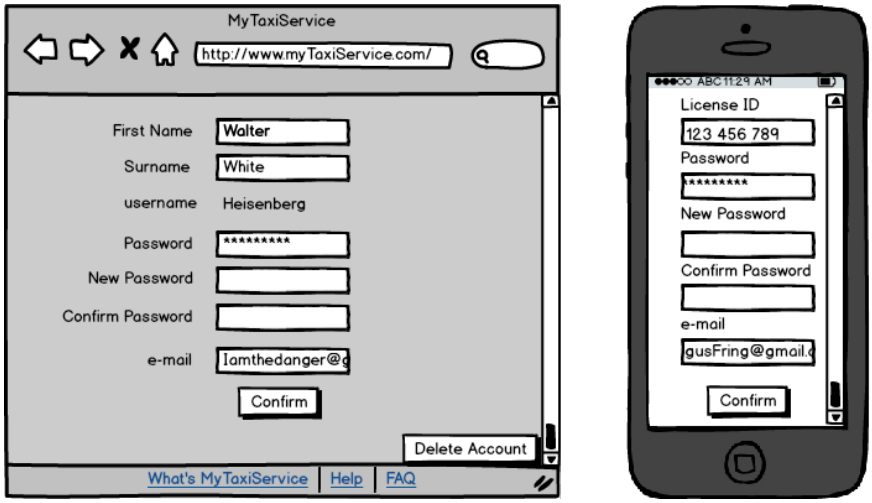


Passenger view: View of the passenger

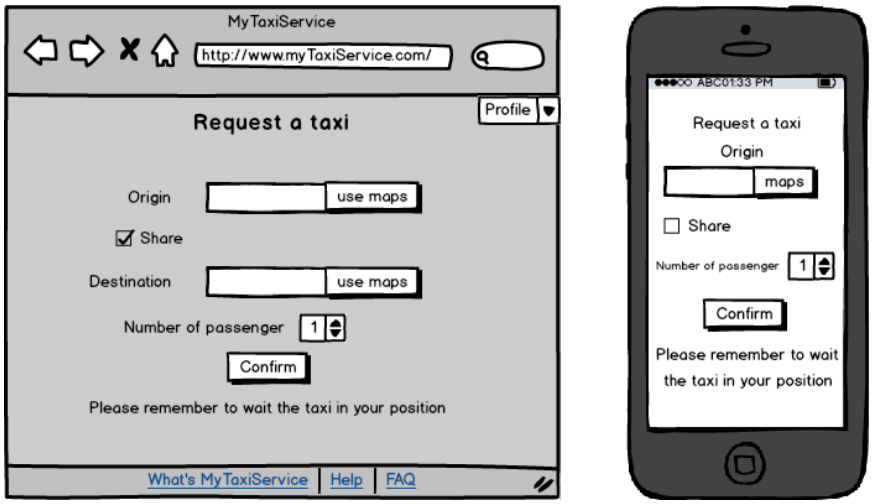




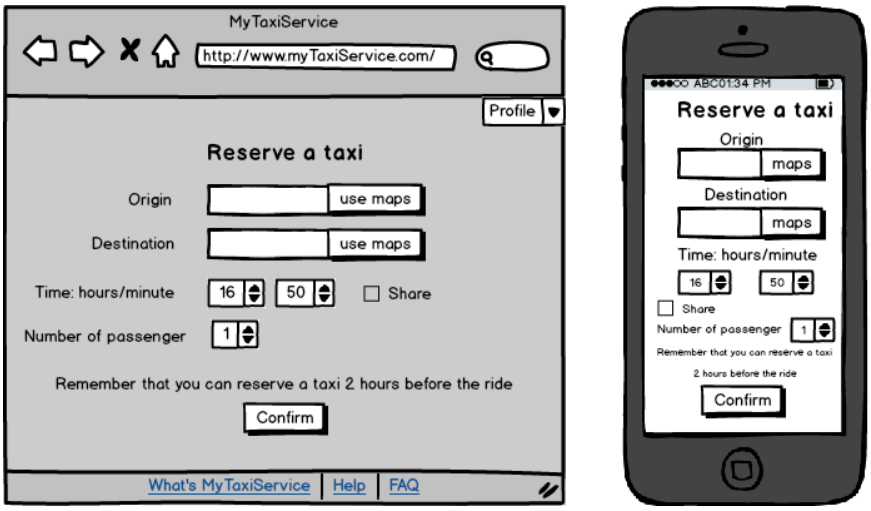
Profile: View of the profile of the user



Request a taxi: View of the passenger when he/she requests a taxi

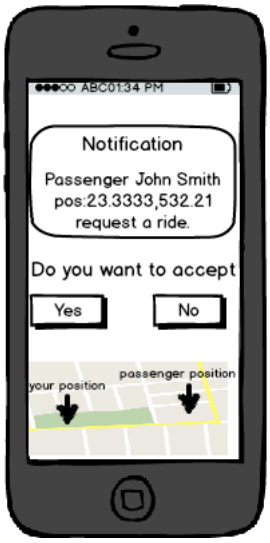


Reserve a taxi: View of the passenger when he/she reserves a taxi

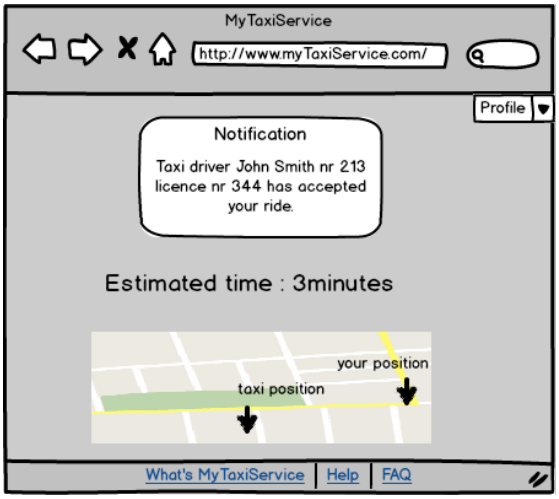


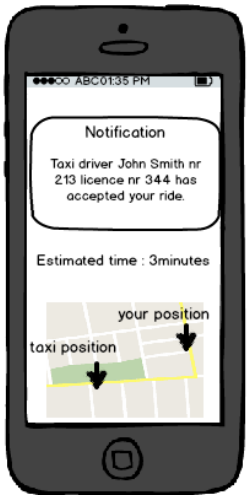
Taxi driver view: View of the taxi driver

Taxi driver notification: Notification that the taxi driver, choosen by the system, sees when a passenger request a ride.



Passenger notification : Notification that the passenger see when a taxi accept the ride





Chapter 5

Requirements Traceability

Chapter 6

References

Appendix A

Appendix

A.1 Software and tool used

- LaTeX (http://www.latex-project.org/) : to redact and to format this document
- Balsamiq Mockups (http://balsamiq.com/products/mockups/): to create mockups
- Microsoft Office Visio 2013 (https://products.office.com/it-it/Visio/flowchart-software): to create sequence diagrams and state-charts
- Eclipse Luna (https://eclipse.org/luna/): to draw global use case, class diagrams and UX diagram

A.2 Working hours

This is the time spent for redact the document

- Belluschi Marco : xx hours
- Cerri Stefano : xx hours
- Di Febbo Francesco : xx hours