POLITECNICO
MILANO 1863

Politecnico di Milano
A.A. 2015/2016
Software Engineering 2: "*myTaxyService*"

Design **D**ocument

Alessandro Baldassari (mat. 841561)
Alberto Bendin (mat. 841734)
Francesco Giarola (mat. 840554)

November 20, 2015

Contents

1 Introduction

1.1 Purpose

This document presents the architecture on which *myTaxiService* will be developed; it describes the decisions taken during the design process and justifies them. The whole design process is described including also the improvements and modifications to provide additional valuable informations in case of future changes of the architecture structure.

1.2 Scope

Accordingly to the definition of the architecture design this document will focus on the non functional requirements of *myTaxiService*. Since the system architecture defines constraints on the implementation this document will be used to provide fundamental guidelines in the development phase of *myTaxiService*.

The system architecture will be organized in three categories corresponding to the functionalities of the different users: passengers, drivers and administrator.

- **Passengers**
  The system allows the user to sign up, login, request or reserve a taxi with the customization of the ride.
- **Taxi drivers**
  The system allows the user to login, set the "availability" and accept or reject jobs.
- **Administrator**
  The system allows the user to login, manage the drivers' list and supervise the system itself.

1.3 Definitions, Acronyms, Abbreviations

The following acronyms are used in this document:

- JEE: Java Enterprise Edition
- RASD: Requirements Analysis and Specification Document
- ER: Entity Relationship
- EIS: Enterprise Information System

The following definitions are used in this document:

1.4 Reference Documents

- Specification document: myTaxiService project
- Template for the Design Document
- IEEE Std 1016-2009 - IEEE Standard on Design Descriptions
- Requirements Analysis and Specification Document for *myTaxiService*

1.5 Document Structure

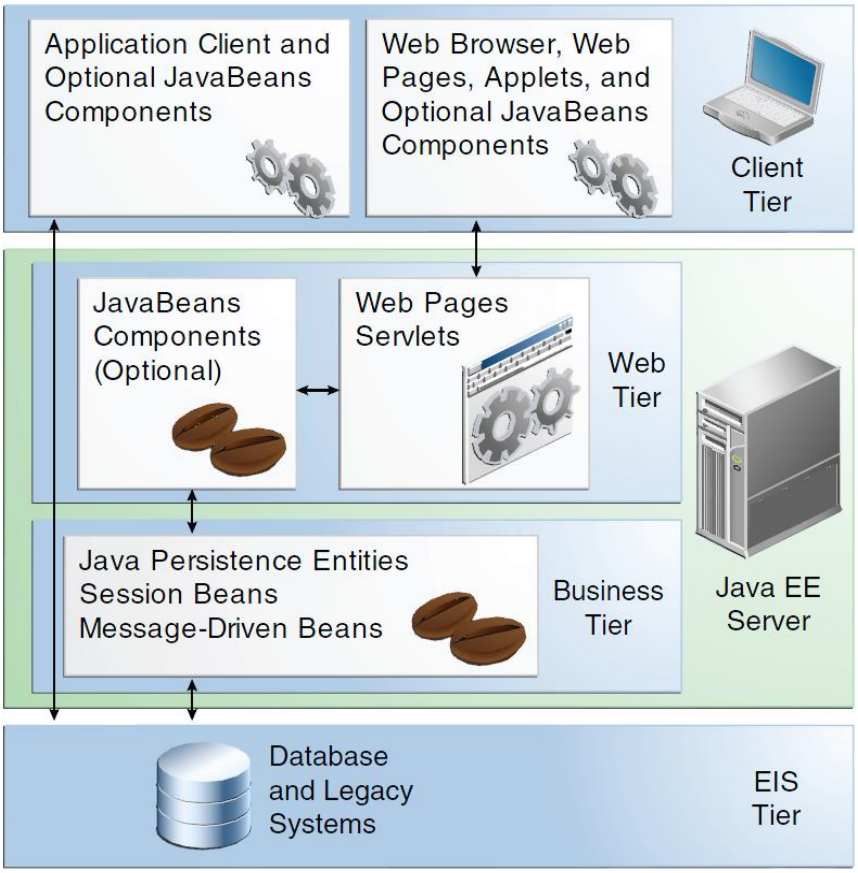
This document specifies the architecture of the system using different levels of detail. It also describes the architectural decisions and justifies them. The design was developed in a top-down way, then the document reflects this approach. The document is organized in the following sections:

1. Introduction Provides a synopsis of the architectural descriptions.
2. Architectural design
3. Algorithm design
4. User interface design
5. Requirements traceability
6. References

2 Architectural Design

2.1 Overview

This section of the design document provides a general description of the design of the system and its processes; it includes the general design context, the general approach and describes the overall design.
JEE has a four-tiered architecture divided as shown in the picture below:



1. **Client Tier**: contains Application Clients and Web Browsers and it is the layer designed to interact directly with the actors. As our project will be a web application the client will use a web browser to access pages;

2. **Web Tier**: contains the Servlets and Dynamic Web Pages that needs to be elaborated. This tier receives the requests from the client tier and forwards the pieces of data collected to the business tier waiting for processed data to be sent to the client tier, eventually formatted;

3. **Business Tier**: contains the Java Beans, which contain the business logic of the application, and Java Persistence Entities;

4. **EIS Tier**: contains the data source. In our case, it is the database allowed to store all the relevant data and to retrieve them.

We can also consider the second and third tier together, in this case the architecture becomes a three-tier one with *client tier*, *business logic tier* and *persistence tier*.
To design the system a top-down approach is used. After the identification of the main three layers, the system is decomposed in components that capture subsets of related functionalities. For each component is specified the role in the architecture and its interactions with the rest of the system.

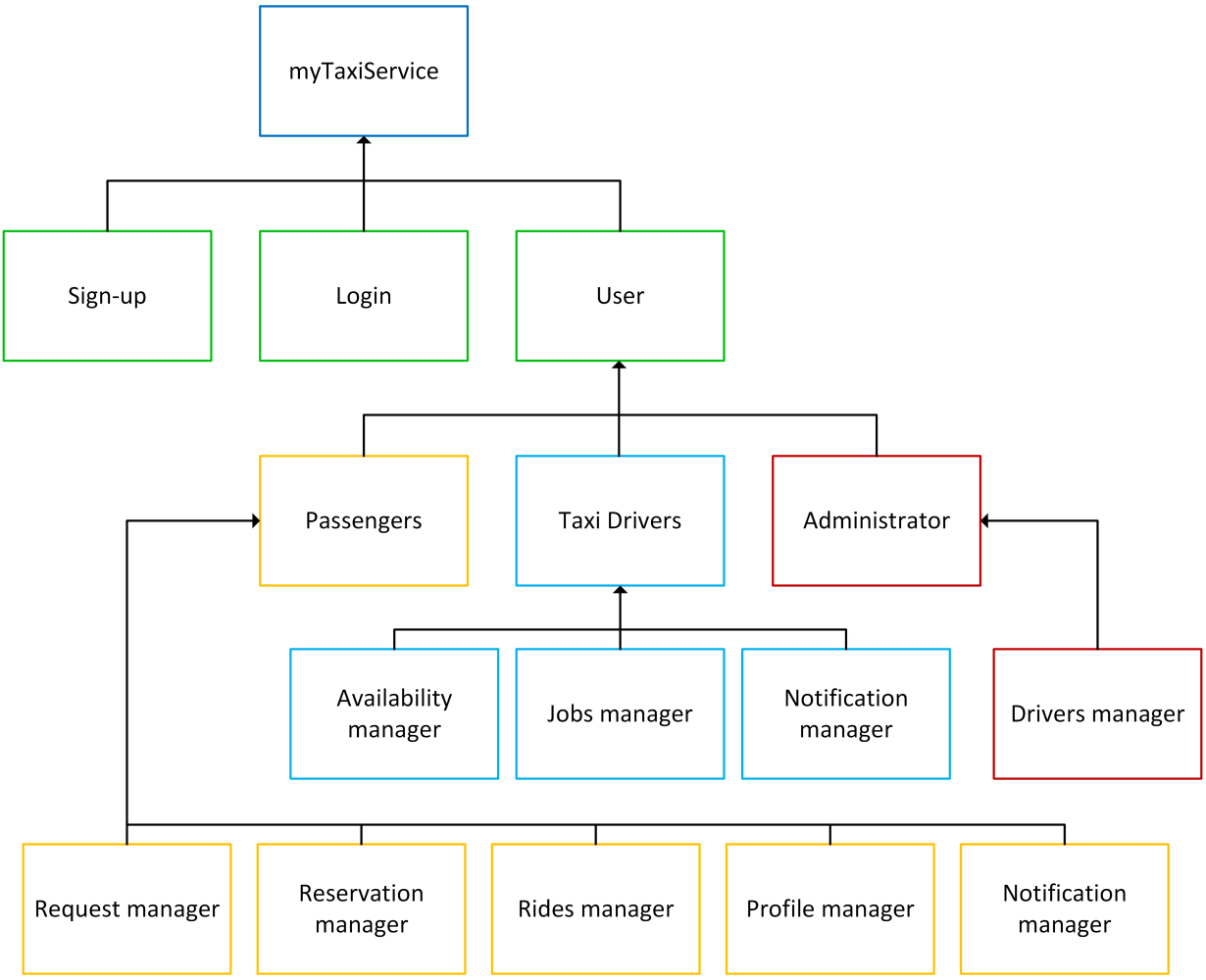
2.2 High level components and their interaction

2.2.1 Identifying sub-systems

We separate the functionalities of our system into these functional areas:

- Sign-up
- Login
- User sub-system
    - Passengers
        - New request manager
        - New reservation manager
        - Rides manager
            - Active rides
            - History
        - Profile manager
        - Notification manager
    - Taxi drivers
        - Availability manager
        - Jobs manager
        - Notification manager
    - Administrator
        - Drivers manager
            - Add driver
            - Edit driver
            - Delete driver

The following schema shows the above-mentioned division of the system.

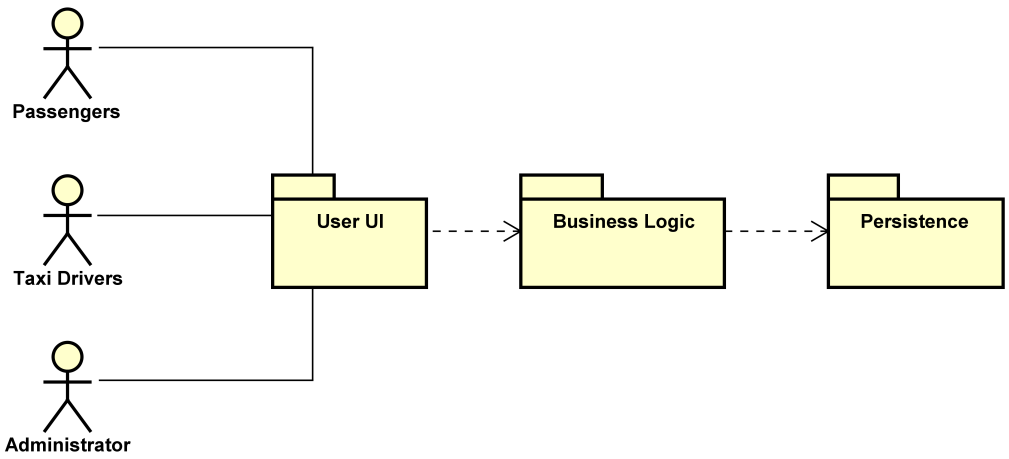


2.2.2 General Package design

In this section the general design schemas are presented specifying the basic relations between packages, use cases and users.
Each layer contains a set of functionalities satisfying the correspondent requirements. Thus we find a mapping between the use cases and the package design of the system. Considering the three-tier view of the architecture three packages are identified:

- User UI: this package is in charge of interacting with the user; it obtains the user requests, sends these to the business logic package, obtains the information needed from the latter and displays them to the user accordingly. In general the package contains the user interfaces.
- Business logic: this package is in charge of receiving and processing the User UI package requests, accessing the Persistence package when needed and sending a response accordingly.
- Persistence: this package is in charge of managing the data request from the Business logic package.

The main users: Administrator, passengers and taxi drivers directly access the User UI package but cannot see the other packages.

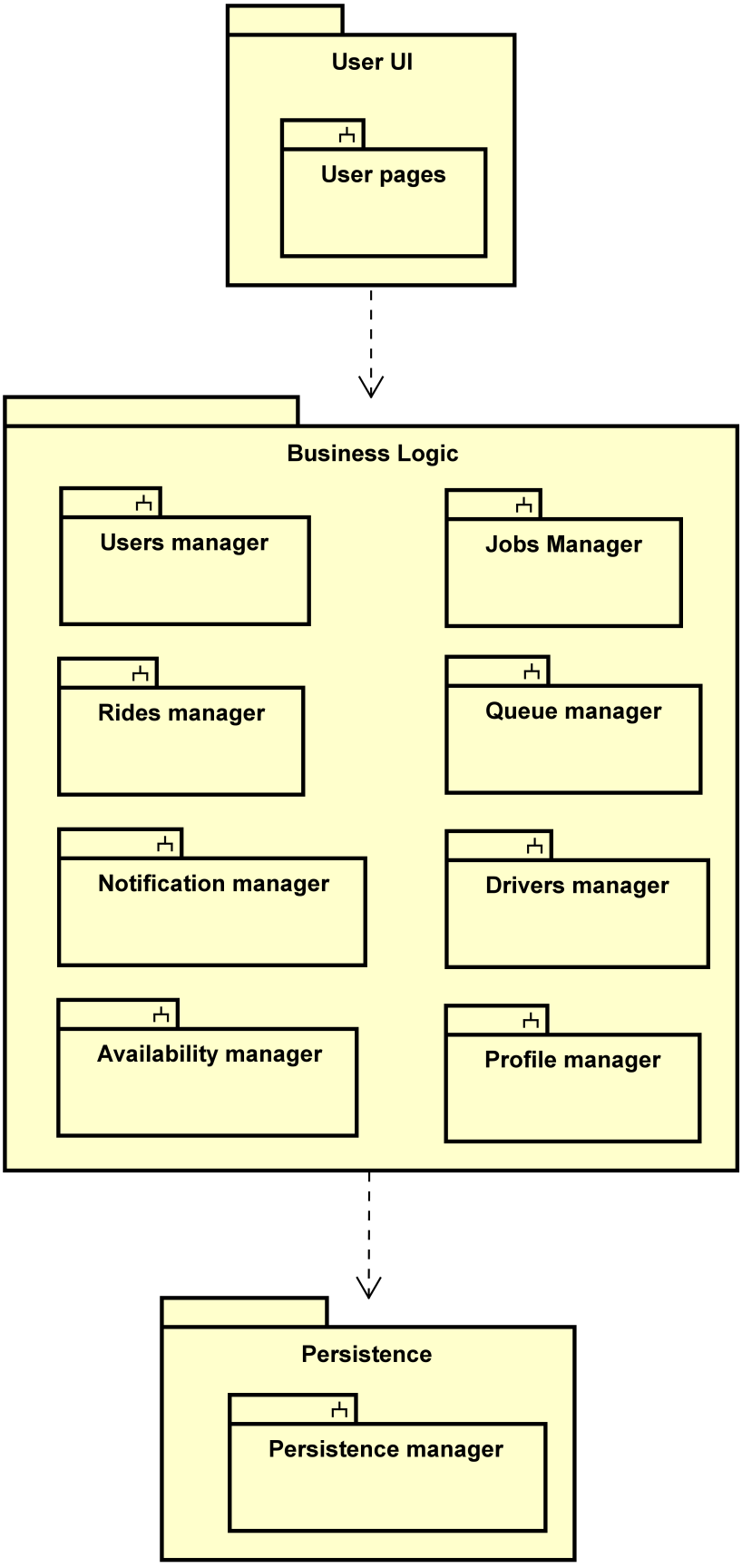


2.3 Component view

The inner packages are described as follows:

- User UI: this set of sub-packages is responsible for encapsulating the user's actions and forwarding information requests to the Business Logic sub-packages.
- Business logic: this set of sub-packages is responsible for handling requests from the User UI package, processing them and sending back a response. These packages may access the Persistence package.
- Persistence: this set of sub-packages contains the data model for the system. It accepts requests from the Business Logic package.

A more detailed view of the system:



- TeXstudio 2.10.4 (http://www.texstudio.org/) to redact and format this document.
- Astah Professional 7.0 (http://astah.net/editions/professional): to create Use Cases Diagrams, Sequence Diagrams, Class Diagrams and State Machine Diagrams.
- Microsoft Office Visio Professional 2016

7.2 Hours of work

The time spent to redact this document:

- Baldassari Alessandro: 35 hours.
- Bendin Alberto: 35 hours.
- Giarola Francesco: 35 hours.