

**D**esign **D**ocument **(DD)**

Computer Science and Engineering (CSE)

Software Engineering 2 Project

Year 2015/16

*Date: 04/12/2015 – Version: 1.0*



**STUDENTS:**

*Martino Andrea (****788701****)*

*Marchesani Francesco (****852444****)*

**PROFESSOR:**

*Mirandola Raffaela*

# Table of Contents

1. Table of Contents 2

2. Introduction 3

2.1. Purpose 3

2.2. Scope 4

2.3. Definitions, Acronyms, Abbreviations 4

2.4. Reference Documents 4

2.5. Document Structure 5

3. Architectural Design 5

3.1. Overview 5

3.2. High level components and their interaction 5

3.3. Component View 5

3.4. Deployment View 5

3.5. Runtime View 5

3.6. Component interfaces 5

3.7. Selected architectural styles and patterns 5

3.8. Other design decisions 5

4. Algorithm design 5

5. User Interface Design 6

6. Requirements Traceability 6

7. References 6

7.1. References list 6

# Introduction

## Purpose

This **Design Document** contains information about the **architecture** and other features of *myTaxiService design* such as **high-level** **algorithms** and **user interface** (with recalls from the previous **RASD**). In order to maintain a **modular** and **scalable** system for the future, we will enter in details only if strictly necessary (otherwise, we will keep a higher level of abstraction).

This **document** is coherent with *the official template* of the project on the *Beep platform*.

As we said for the **RASD**, it is important to underline that some parts of this document may evolve in the future (this may occurs for several causes).

Anyway, we will try to maintain coherence as much as possible.

Here is a resume of the steps of the project, with the related deadlines:

## Scope

The main scope of this **DD** (*Design Document*) is to give an overall guidance to the **architecture** of the **project**, which is *myTaxiDriver* (**Software Engineering 2 project** of year 2015/16 - **Politecnico di Milano**).

We described the main **goals** and **objectives** of the project in the previous *Requirements Analysis and Specification Document*.

## Definitions, Acronyms, Abbreviations

* **DD:** *Design Document*
* **RASD**: *Requirements Analysis and Specification Document*
* **mTS**: *myTaxiService*
* **Servlet**: A servlet is a program that extends the capabilities of a server.
* **Layer**: logical level of the architecture.
* **Tier**: physical level of the architecture.

## Reference Documents

The main reference document is the **Requirements Analysis and Specification Document (RASD)** of *myTaxiService*.



*Preview of the Requirements Analysis and Specification Document (RASD), our reference document.*

## Document Structure

# Architectural Design

## Overview

We start from a big consideration: *myTaxiService* is a big project. As we have seen before, in the RASD, we have a lot of potential users and a big amount of requirements.

Thus, we considered to develop our application using **Java Enterprise Edition** (**JEE**).

This will also be useful to satisfy important *Non-Functional Requirements* such as scalability, portability, availability, reliability and so on.

The applications of *myTaxiService* (the web application and the mobile one) will be *large-scale*, *multi-tiered, scalable, reliable* and the network will be *secure*.

The application developing takes as reference the standard of **Java Enterprise Edition 7** (**JEE7**), the last release available now.

We will use a *three-tier physical architecture* mapped on *four logical layers*, as the standard of JEE.

Here is the general schema of the **architecture**:



Now we will see in a deeper level of detail the meaning of each layer:

* **Client Layer**: it contains Application Clients and Web Browsers and interacts directly with the actors (Customers and Taxi Drivers). In our application, the Client can access via browser (*web application*) or via smartphone (*mobile application*).
* **Web Layer**: it contains the *Servlets* and *Dynamic Web Pages* that needs the elaboration. This tier receives the requests from the *Client* *layer* and forwards the pieces of data collected to the *Business Layer*.
* **Business Layer**: it contains the application logic (with the *Java Beans* and the *Java Persistence Entities)*. This will permit the communication between the System of mTS and the target users (Costumers and Taxi Drivers).
* **EIS (Data Layer)**: it will contain all the *data* concerning Costumers, Taxi Drivers, Calls, Reservations and other useful information. It is crucial to manage data according to strict policies about security and privacy.

It is important to underline that *myTaxiService* will also use external pre-built software products from the point of view of the business logic, such as **Google Maps** for the maps and the **online payment services API** (*e.g. PayPal*).

Here is a schema of the **three physical levels** (*tiers*) of the architecture:



## Component View

## Deployment View

## Runtime View

## Sequence Diagrams Here

## Component interfaces

## Selected architectural styles and patterns

## Other design decisions

# Algorithm design

# User Interface Design

# Requirements Traceability

# References

## References list

Here is a short list of the **references** for this **Design Document**:

* *Slides of the Software Engineering 2 course (from the* ***Beep Platform****)*
* *Design Document Template* *(by Prof. Raffaela Mirandola)*
* *Software Engineering: Principles and Practice (Hans Van Vliet)*