# List of tables

# List of figures

# Introduction

## Purpose

## Scope

## Defintions, acronyms and abbreviations

## Reference documents

## Document structure

# Archictectural design

This section presents the architectural design of the MyTaxiService software. We first present an overview of the design and a high level description of the components. These are followed by further explanations of the components from three different perspectives (component, deployment and runtime). Finally we expose the design decisions, some of which refer to the selected architectural and styles patterns.

It is worth to mention that this document refers to a logical architecture of the system, which is platform and specific technologies independent. This means that we have not made any decisions on the language or implementation frameworks (e. g. Java EE, PHP, Ruby) that shall be used; we assume that such choices will be taken in later iterations. However, some of the expected behaviors of the implementing platforms are provided.

## Overview

## High level components and their interaction

The following diagram presents the main components of the system at a high level view. It only shows their names and the other related components and actors (either users or systems). In the *Component view* we refine this model and add deeper explanations.

**DIAGRAM HERE**

The yellow actors are the users of MyTaxiService:

* **Passenger:** makes use of the MyTaxiService to make a request for a taxi service. He can communicate with the system either through the web site or the mobile application by interacting with the appropriate subcomponents of the MTSView (this will be shown in the *Component view*).
* **TaxiDriver:** makes use of the MyTaxiService to attend requests for a taxi service. He can communicate with the system through the mobile application by interacting with the appropriate subcomponent of the MTSView.

The pink boxes represent software components that define MyTaxiService. Note that the use of the MVC architectural pattern is visible:

* **MTSView:** this component is in charge of displaying graphical components to allow both the Passenger and the TaxiDriver to interact with the system. It will communicate such interactions to the MTSController. In the *Component view* we will see that this component includes subcomponents that will support the access of the users from the different devices (computer and mobile phones).
* **MTSController:** this component is in charge of receiving the events produced by the interactions between the users and the software, and performing the corresponding invocations on the MTSModel to execute the related tasks.
* **MTSModel:** this component is in charge of receiving invocations from the MTSController and execute the appropriate operations. It implements the business logic. Some operations might involve accessing to the MTS\_DB to retrieve persistent data, or to the MTSIntegration to communicate with external systems.
* **MTS\_DB:** represents the Database Management System that will store the persistent data. This document does not include a Persistent view, but we expect this component to store the following information:
  + The data of the passengers, such as name, email address, password, and whether their email address has been confirmed.
  + The data of the taxi drivers, such as name, email address, username, password, taxi code, and taxi capacity.
* **MTSIntegration:** this component is in charge of communicating with the external systems of the domain (EmailServer, MapsServer, MilanoGovernment), as consequence of requests of the MTSModel.

The green actors are the external systems that MTS will interact with (these are already implemented):

* **EmailServer:** this system will be used by the MTSModel (through the MTSIntegration) to send messages to the users as described in the *External system interfaces* section of the RASD (section 2.1.2).
* **MapsServer:** this system will be used by the MTSModel (though the MTSIntegration) to perform location analysis tasks as described in the *External system interfaces* section of the RASD (section 2.1.2).
* **MilanoGovernment:** this system will be used by the MTSModel (though the MTSIntegration) to perform data validation tasks as described in the *External system interfaces* section of the RASD (section 2.1.2).

## Component view

In this section we provide further details on the components defined in the previous section. We have identified subcomponents aiming to the *Deployment view* and the two different users, which will also be more detailed as they are explained in this subsection (by means of class diagrams).

**DIAGRAM HERE**

In the following subsections we expose the descriptions of the components in the diagram above. We only give their definition since the interfaces are defined later in the document. The components are grouped by the high level component that they refine.

### MTSView

The MTSView component has three subcomponents: MTSPassengerWebView, MTSPassengerMobileView and MTSTaxiDriverMobileView.

#### MTSPassengerWebView

This subcomponent will be used by the Passenger whenever he accesses the web site of MyTaxiService. It will communicate to MTSPassengerWebController, the corresponding subcomponent of MTSController.

The class diagram of the classes encapsulated by this subcomponent. Is analogous to the one in the MTSPassengerMobileView, so it is not presented here.

The Web Browser used by the Passenger to access the web site of MyTaxiService, so it will be the one that will communicate with the MTSPassengerWebView to get the required web pages that will be shown to the user.

#### MTSPassengerMobileView

This subcomponent will be used by the Passenger whenever he uses the MyTaxiService mobile application. It will communicate to MTSPassengerMobileController, the corresponding subcomponent of MTSController.

The following is the class diagram of the classes encapsulated by this subcomponent.

**DIAGRAM HERE**

**CLASSES DESCRIPTION HERE**

#### MTSTaxiDriverMobileView

This subcomponent will be used by the TaxiDriver whenever he uses the MyTaxiService mobile application. It will communicate to MTSTaxiDriverMobileController, the corresponding subcomponent of MTSController.

The following is the class diagram of the classes encapsulated by this subcomponent.

**DIAGRAM HERE**

**CLASSES DESCRIPTION HERE**

### MTSController

The MTSController component has three subcomponents: MTSPassengerWebController, MTSPassengerMobileController and MTSTaxiDriverMobileController.

#### MTSPassengerWebController

This subcomponent will be used by the Passenger whenever he accesses the web site of MyTaxiService. It will communicate to MTSPassengerWebController, the corresponding subcomponent of MTSController.

The class diagram of the classes encapsulated by this subcomponent. Is analogous to the one in the MTSPassengerMobileController, so it is not presented here.

#### MTSPassengerMobileController

This subcomponent will be used by the Passenger whenever he uses the MyTaxiService mobile application. It will communicate to MTSPassengerMobileController, the corresponding subcomponent of MTSController.

The following is the class diagram of the classes encapsulated by this subcomponent.

**DIAGRAM HERE**

**CLASSES DESCRIPTION HERE**

#### MTSTaxiDriverMobileController

This subcomponent will be used by the TaxiDriver whenever he uses the MyTaxiService mobile application. It will communicate to MTSTaxiDriverMobileController, the corresponding subcomponent of MTSController.

The following is the class diagram of the classes encapsulated by this subcomponent.

**DIAGRAM HERE**

**CLASSES DESCRIPTION HERE**

### MTSModel

This component has subcomponents for each user or

## Deployment view

## Runtime view

## Component interfaces

## Selected architectural styles and patterns

## Other design decision

# Algorithm design

# User interface design

# Requirements traceability

# References