Politecnico di Milano
A.A. 2015-2016
Software Engineering 2: "myTaxiService"
Design **D**ocument

Roberto Clapis (841859), Erica Stella (854443)

December 3, 2015

Contents

1 Introduction

1.1 Purpose

The purpose of the Design Document is to provide documentation in order to aid the development of myTaxiService's system by providing a description of how it should be built and how its components are expected to interact with each other.

1.2 Scope

This Design Document is intended to explain the design and architecture of myTaxiService, a new application that will provide an easy way to access the taxi service in a city. It describes the system both from a software and hardware point of view, in order to clarify the system's structure and how it accomplishes its functionalities.

1.3 Definitions, Acronyms, Abbreviations

1.3.0.1 Definitions

- *End users:* this category comprises all those who use the application[1]: administrators, taxi drivers, logged in users and guests.

1.3.0.2 Acronyms

- *UI:* user interface through which the end users can interact with the application.
- *DB:* Database.
- *DBMS:* Database Management System.

1.4 Reference Documents

- Document with the assignment for the project
- RASD for myTaxiService
- Template for the Design Document
- IEEE standard for Software Design Document
- The IEEE standard for architecture descriptions

[1] For their definition we refer to the RASD's section 1.6

1.5 Document Structure

The following parts of this document are structured in 3 sections: architectural design, user interface design and requirements traceability. The architectural design section describes the software and hardware components of the system and their interactions. The user interface design section which refers to the "User Interfaces" subsection of the RASD. The requirements traceability section that explains how the proposed design meets the requirements that have been defined in the RASD.

2 Architectural Design

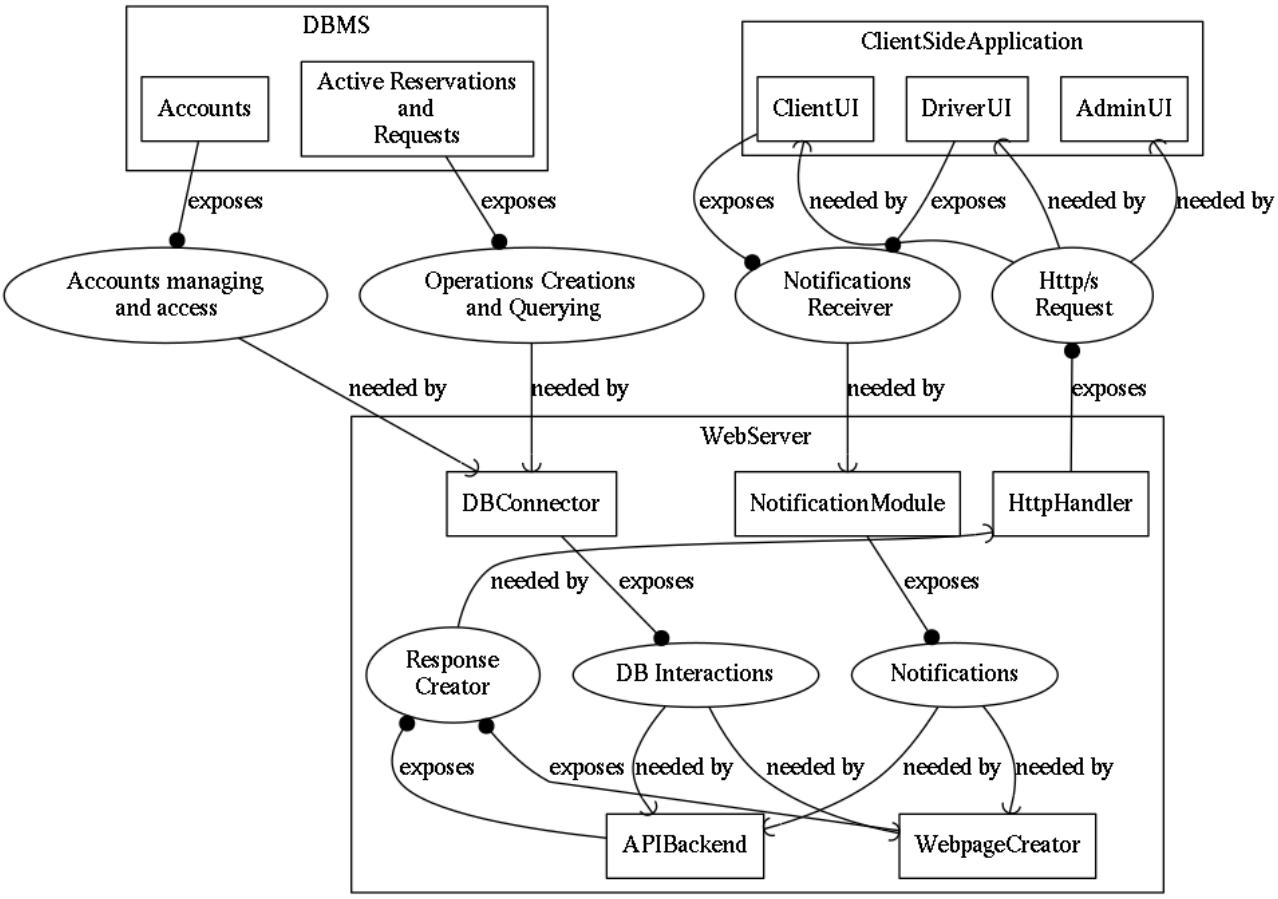
2.1 Overview

The system to be developed, as mentioned before, will be used to provide an easy access to a taxi service. Therefore, its main functionalities, that will have to be supported by the design and architecture, are: the storage of the taxi drivers' and clients' accounts, the computation of the taxi queue of each zone and the handling of requests and reservations. Furthermore, the system will have to comply to quality of service attributes as specified in the RASD.

2.2 High Level Components and Their Interaction

myTaxiService's system is composed by three main components: DBMS, Web Server and client application. The client application provides the UI through which end users can access the application's services. These requests are forwarded to the Web Server which is in charge of providing a response, eventually querying the Database in the DBMS for information. The Web Server is also responsible for answering to the API calls coming from external applications and notifying the end users for particular events, like when a request is accepted by a taxi driver. The DBMS stores all the information of the end users's accounts, the active requests and reservations.

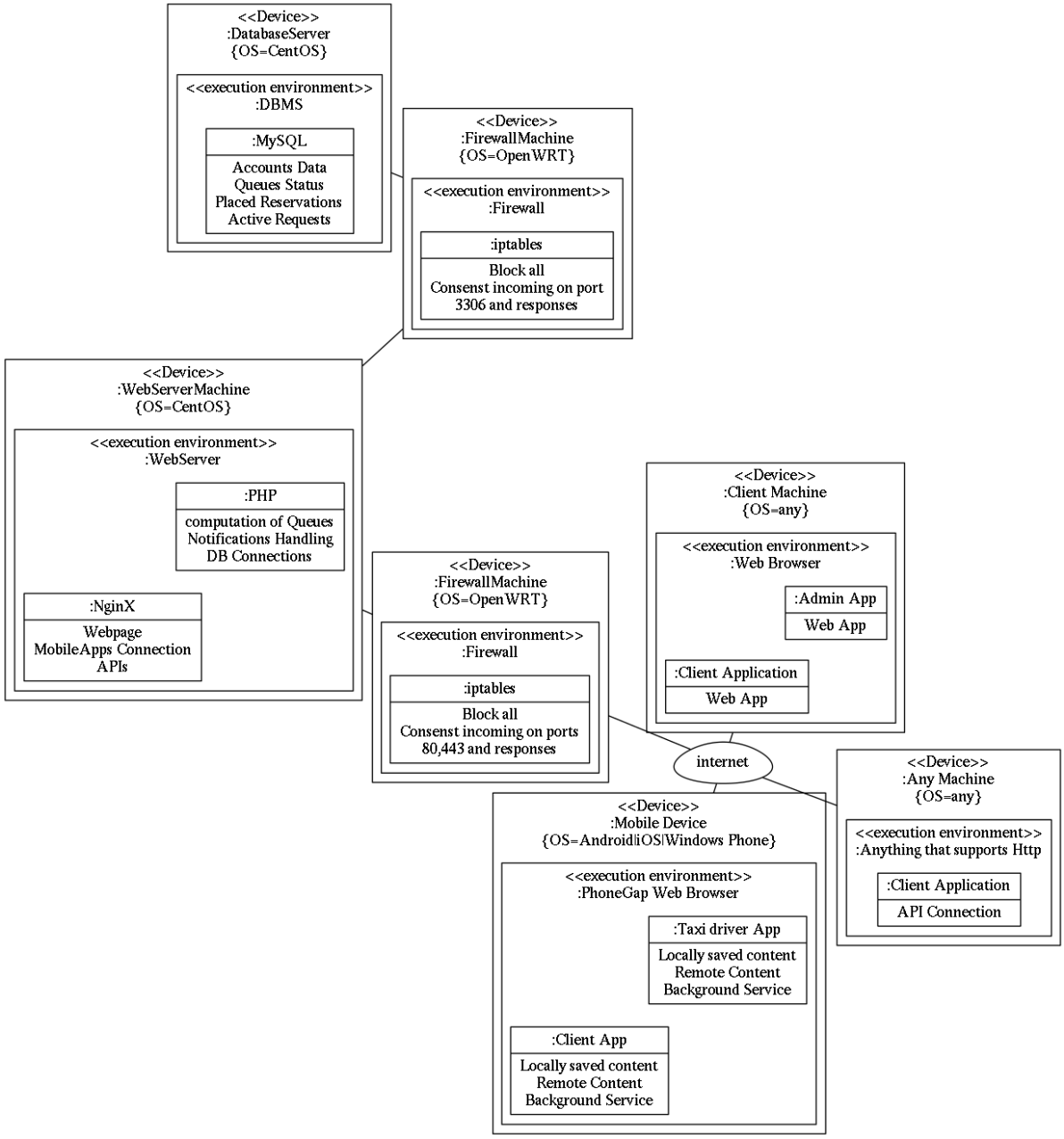
2.3 Component View



Meanwhile the DBMS and the ClientSide application are pretty much self explanatory the WebServer part requires some details:

- The Webpage Creator is the responsible for both the Mobile App and Web App responses. The httpHandler will recognise the request by the parameters and ask the Webpage Creator to either create the full Html page (in the case of the Web App) or just send a partially created page that only contains the dynamic data that the Mobile App will then include in its interface.
- The API Backend will be called by the httpHandler and will respond with Only the data requested in a JSON format

2.4 Deployment View



This is just a pretty standard configuration with DMZ, Internal network, double firewall and untrusted network, see more in section 2.8

2.5 Runtime View

2.6 Component Interfaces

2.6.0.1 DBInteractions This interface encapsulates and exposes all the operations the Web Server needs to interact with the DB. The operations are divided in 2 categories based on the DB's part they manage:

- *Accounts managing and access:* this interface exposes methods to manage the stored end users accounts.

Method	Parameters required	
addAccount	type of the account,credentials	adds an account to the DB of th
modifyAccount	user ID, attribute to modify, new value of the attribute	modifies the attribute of the
deleteAccount	user ID	deletes
getAccount	user ID	returns the account

- *Operations creations and querying:* this interface exposes methods to manage and retrieve requests and reservations.

Method	Parameters required
addRequest	starting locations's zone, destination's zone, user's ID
deleteRequest	request ID
toggleStateOfTheRequest	request ID
getRequest	request ID
getAllTheRequestsOfAnUser	user ID
addReservation	starting locations's zone, destination's zone, meeting time, user ID
deleteReservation	reservation ID
toggleStateOfTheReservation	reservation ID
getReservation	reservation ID
getAllTheReservationsOfAnUser	user ID

2.6.0.2 Http/s Request This interface exposes –¿ tables divided by type of user that uses the method

- *Functionalities exploited by guests:*

- *Functionalities exploited by logged in user:*

Method	Parameters required	Notes
requestTaxi		
cancelRequest		
showRequestDetails		
reserveTaxi		
cancelReservation		
showReservationDetails		
searchForETA		
register		
login		
logout		

- *Driver:*

Method	Parameters required	Notes
answerRequest		
answerReservation		
toggleState		

2.6.0.3 Notifications This interface exposes methods to notify end users (except for administrators) of particular events.

2.7 Selected Architectural Styles and Patterns

2.8 Other Design Decisions

3 User Interface Design

This section has been explored in RASD's section 2.1.1 "User Interfaces" so we refer to that one.

4 Requirements Traceability

5 References

6 Appendix

Appendix for Roberto Clapis
Work hours: 15

Software Used:

Task	Software
Edit LaTeX Source	Vim
Edit Graphs Sources	Vim
Edit sources for Sequence Diagrams	Vim
Convert Sequence Diagrams to images	Quick SequenceDiagramEditor
Generate and Raster directed graphs	Dot
Generate and Raster undirected graphs	Fdp
General images mangling and cropping	ImageMagick & Shotwell
Convert LaTeX source to PDF	LaTeXMK
Spell Check	Aspell
LaTeX Check	LaCheck