

# Software Engineering 2: myTaxiService

---

CASATI FABRIZIO, CASTELLI VALERIO

# Context

---

- Aim: build a taxi management system for a medium-large city
- The city area is divided in taxi zones, there must be a taxi queue for each zone
- Taxis are matched with requests depending on the zone they come from
- Interaction with the system happens through smartphones, tablets and web browser
- Support both immediate requests and reservations (for registered users only)
- The existing taxi management service of the city is fully migrated and decommissioned
- 24/7 service (ideally)

# Domain Assumptions (I)

---

- Taxis are uniquely associated to drivers and viceversa
- Passengers don't need to choose a particular taxi driver among those available for their ride
- Taxi drivers are provided with a mobile phone with an active data plan by the city council
- Passengers are not allowed to place reservations more than 15 days in advance or cancel reservations after a taxi has already been scheduled for them
- Rides can be requested by all passengers, while reservations can only be placed by registered passengers
- The only taxis eligible for fulfilling a reservation are the ones present in the queue of the zone associated with the reservation source address 10 minutes before the scheduled meeting time

# Domain Assumptions (II)

---

- In order to receive ride requests, a taxi driver must explicitly mark himself as available
- A taxi driver will not be able to receive ride requests while he's unavailable
- Taxis that are considered to be out-of-city will not be able to receive calls
- A taxi driver who is currently on a ride will not be able to receive calls
- A taxi driver must always notify the system when he terminates a ride
- After a taxi driver has been associated to a call, he must confirm or refuse the request within two minutes. After that period of time, the call is considered refused
- If no taxis are available to fulfill a reservation 10 minutes before the scheduled meeting time, attempts of rescheduling are to be made at intervals of 2 minutes for at most 20 times

# Functional Requirements

---

- The city administration must have the possibility to enter and update taxi driver data and the taxi zone division
- Taxi drivers must be able to
  - communicate their availability status
  - receive, accept, refuse and drop ride requests
  - communicate they have terminated a ride
- Passengers must be able to request rides and, if logged in, also place and manage their reservations
- The system must support third party expansion through plugins and remote services
- It must be possible to verify the identity of a passenger before taking him onboard

# Stakeholders and actors

---

## STAKEHOLDERS

- Passengers
- Taxi drivers
- City council
- Taxi drivers' union
- Mobile phone producers
- Wireless carriers
- Third party developers

## ACTORS

- Guest passenger
- Guest taxi driver
- Logged in passenger
- Logged in taxi driver
- Administrative personnel
- Mapping service
- Remote services

# UI Mockup (I)

## HOMEPAGE

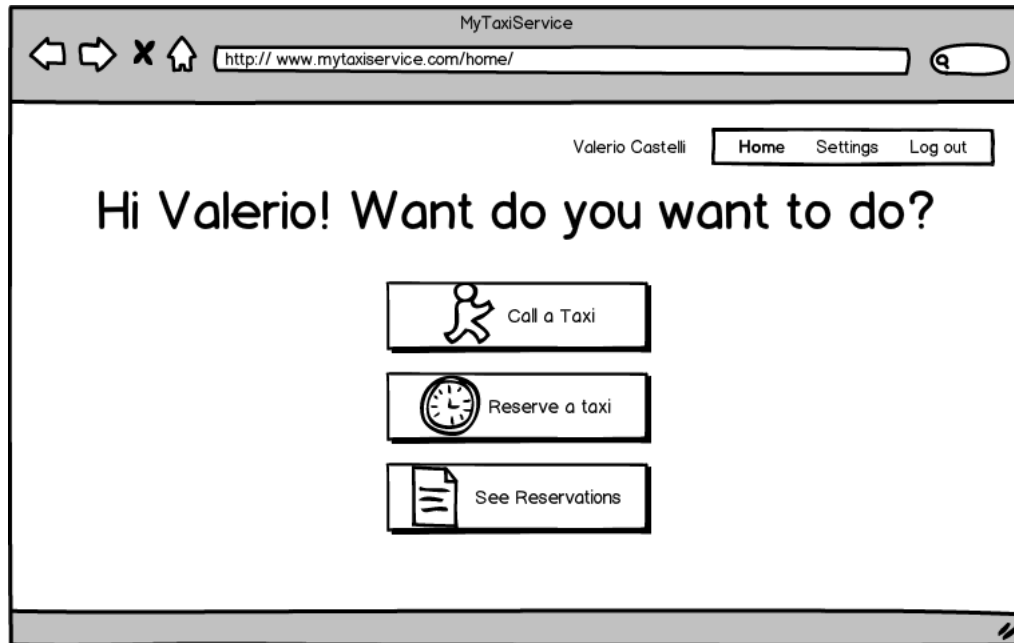
The homepage mockup is displayed within a browser window titled "MyTaxiService". The address bar shows "http:// www.mytaxiservice.com/home". The main content area features a large heading "Welcome to MyTaxiService!". Below the heading are two buttons: "Call a Taxi" with a person icon and "Log in" with a checkmark icon. At the bottom, there is a blue link labeled "Register".

## USER LOGIN

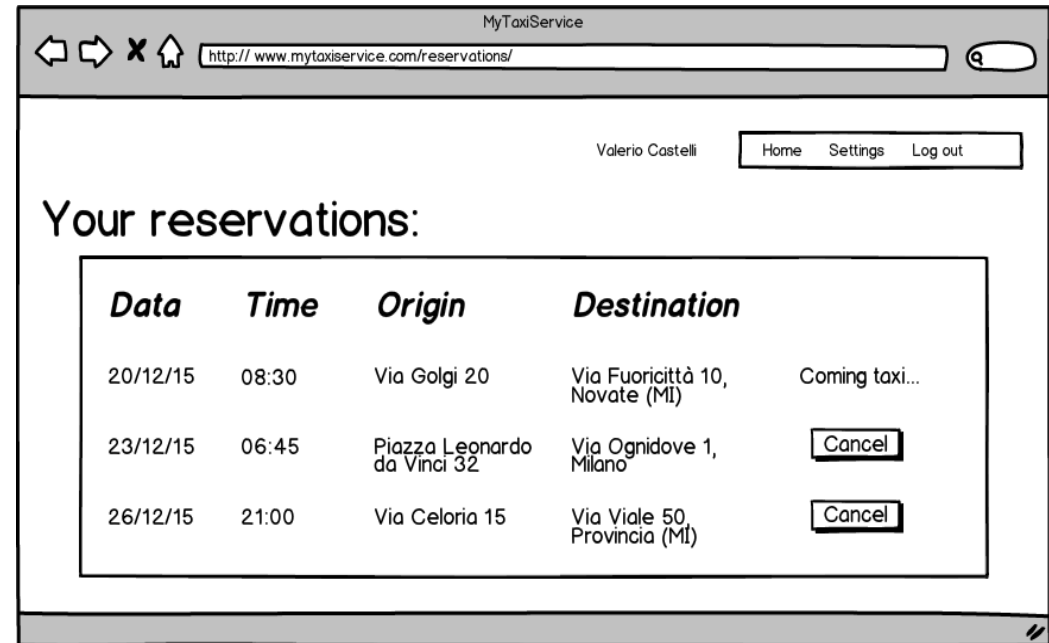
The user login mockup is displayed within a browser window titled "MyTaxiService". The address bar shows "http:// www.mytaxiservice.com/login/". The main content area features a "Log in" label above a form. The form contains two input fields: "Username" and "Password". Below the "Password" field is a "Confirm" button. At the bottom of the form area, there is a blue link labeled "Register".

# UI Mockup (II)

## PASSENGER HOMEPAGE



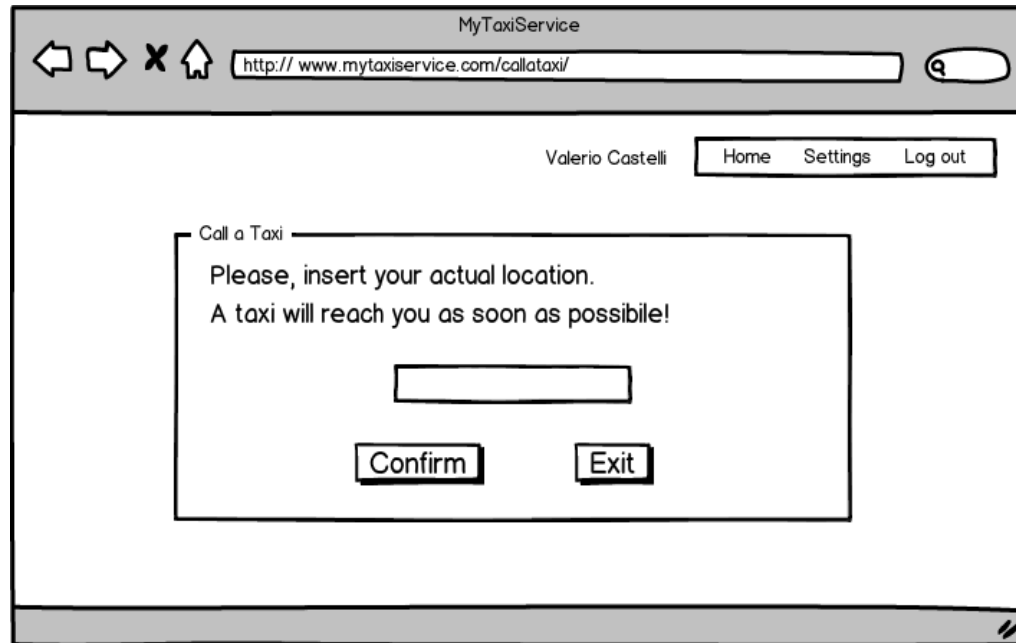
## PASSENGER RESERVATIONS





# UI Mockup (III)

## PASSENGER REQUEST



MyTaxiService

http://www.mytaxiservice.com/callataxi/

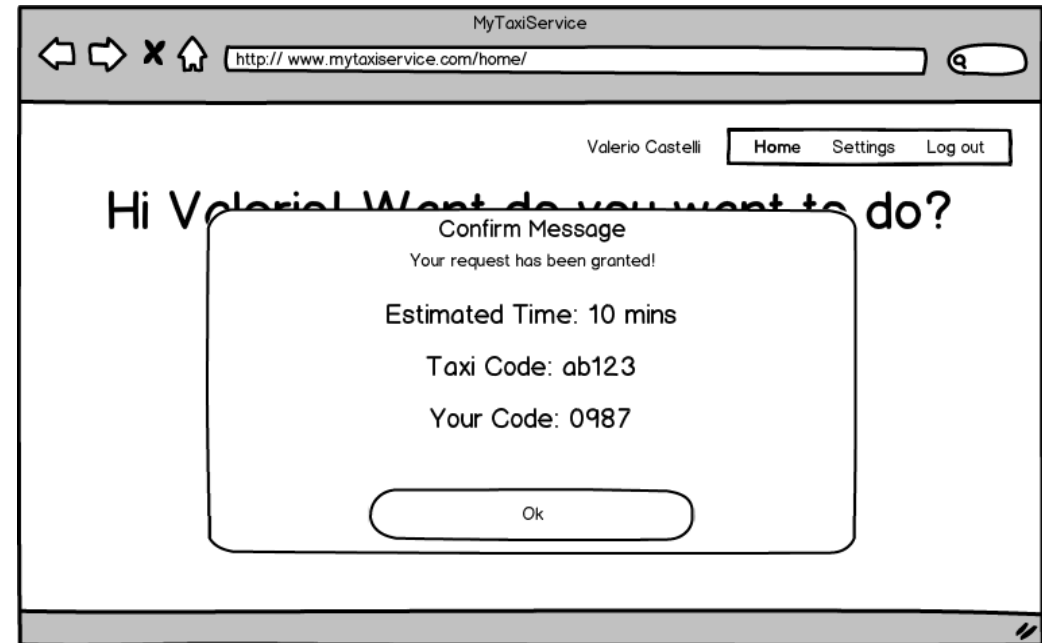
Valerio Castelli Home Settings Log out

Call a Taxi

Please, insert your actual location.  
A taxi will reach you as soon as possible!

Confirm Exit

## REQUEST CONFIRMATION



MyTaxiService

http://www.mytaxiservice.com/home/

Valerio Castelli Home Settings Log out

Hi Valerio! What do you want to do?

Confirm Message

Your request has been granted!

Estimated Time: 10 mins

Taxi Code: ab123

Your Code: 0987

Ok

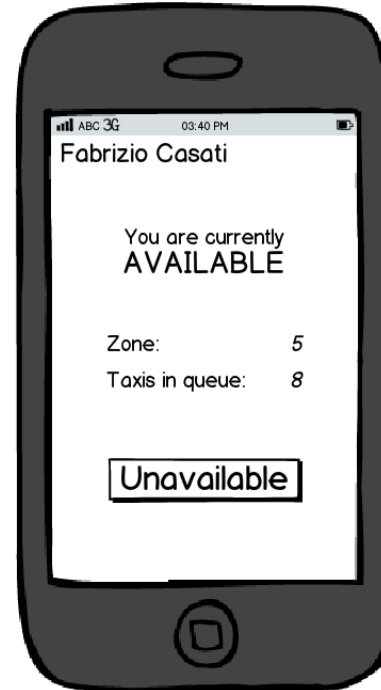
# UI Mockup (IV)

---

MOBILE LOGIN



TAXI AVAILABLE



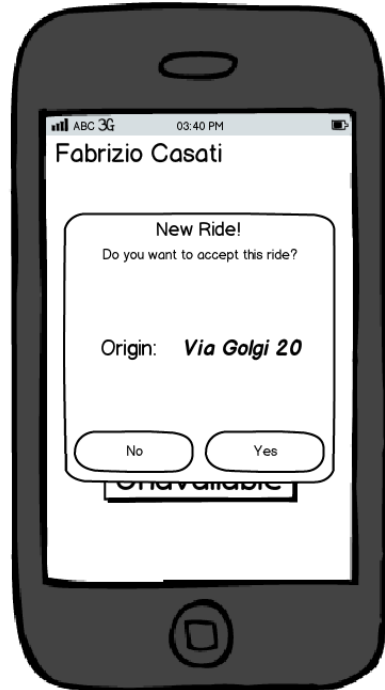
TAXI UNAVAILABLE



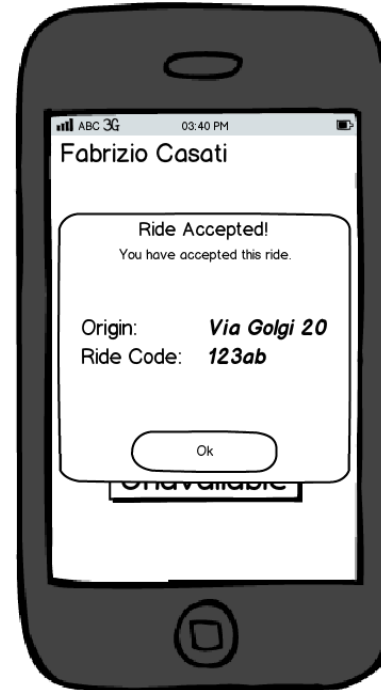
# UI Mockup (V)

---

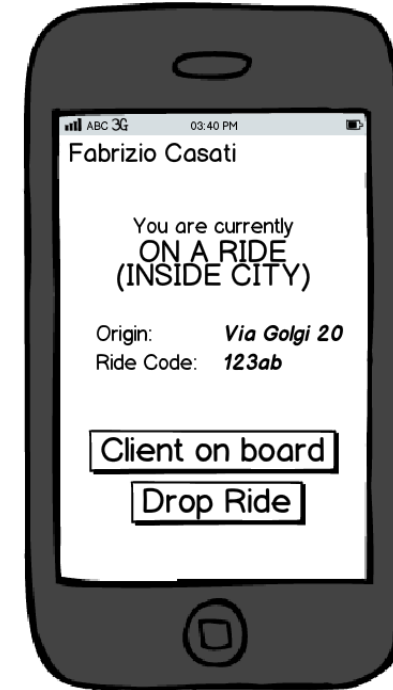
RIDE REQUEST



ACCEPTED RIDE

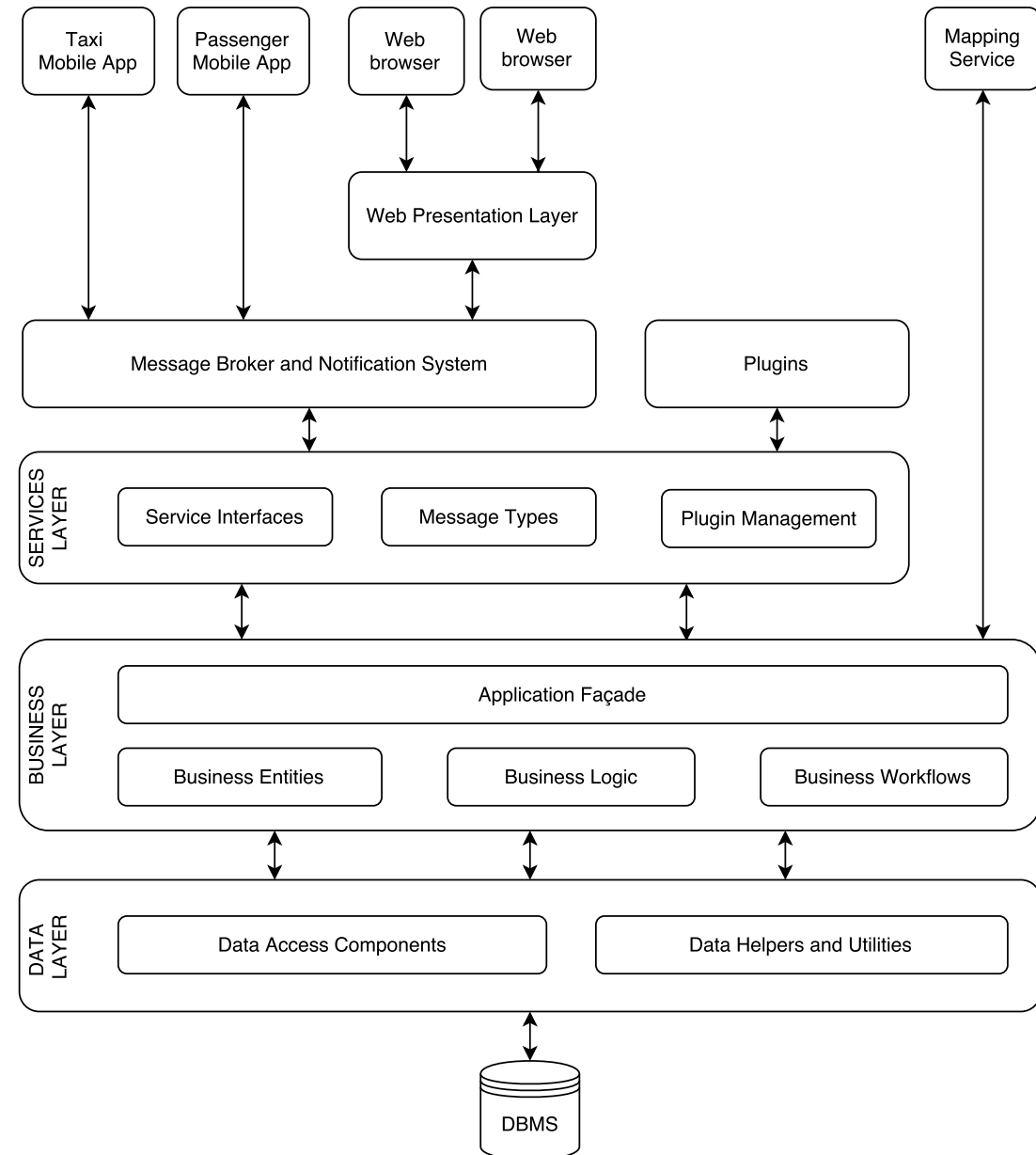


CURRENTLY RIDING



# Architectural Design

- Multi-layer and multi-tier architecture
- Access to DBMS is mediated by an intermediate abstraction (Data Layer) for flexibility
- Business Layer implements core functionalities, exposed through an Application Façade
- A subset of functionalities is made available for remote calls via a Service Layer (SOAP and JAX-WS)
- Communication between remote clients and central system happens through a message broker and notification system
- Implementation based on Java EE



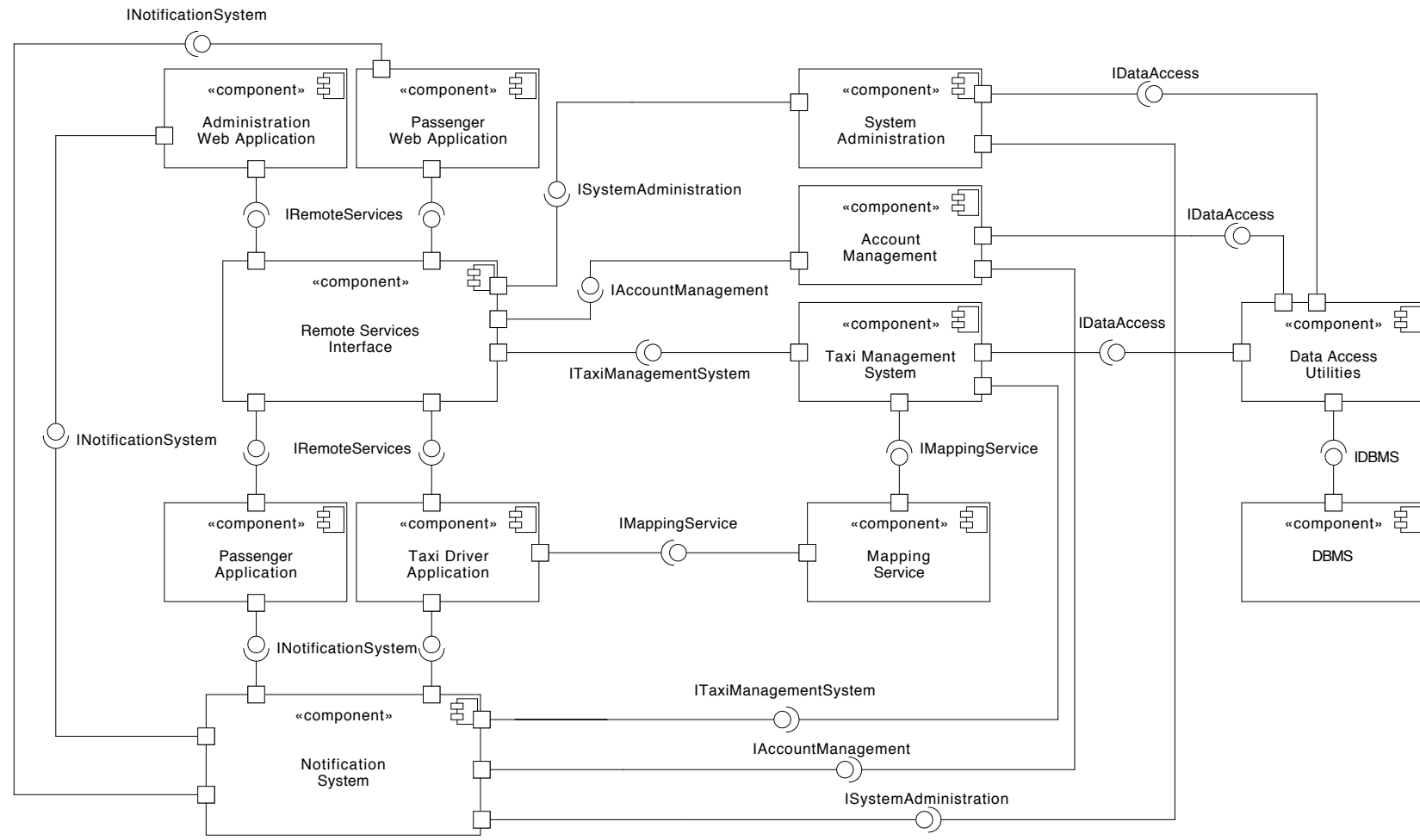
# Architecture: high level components

---

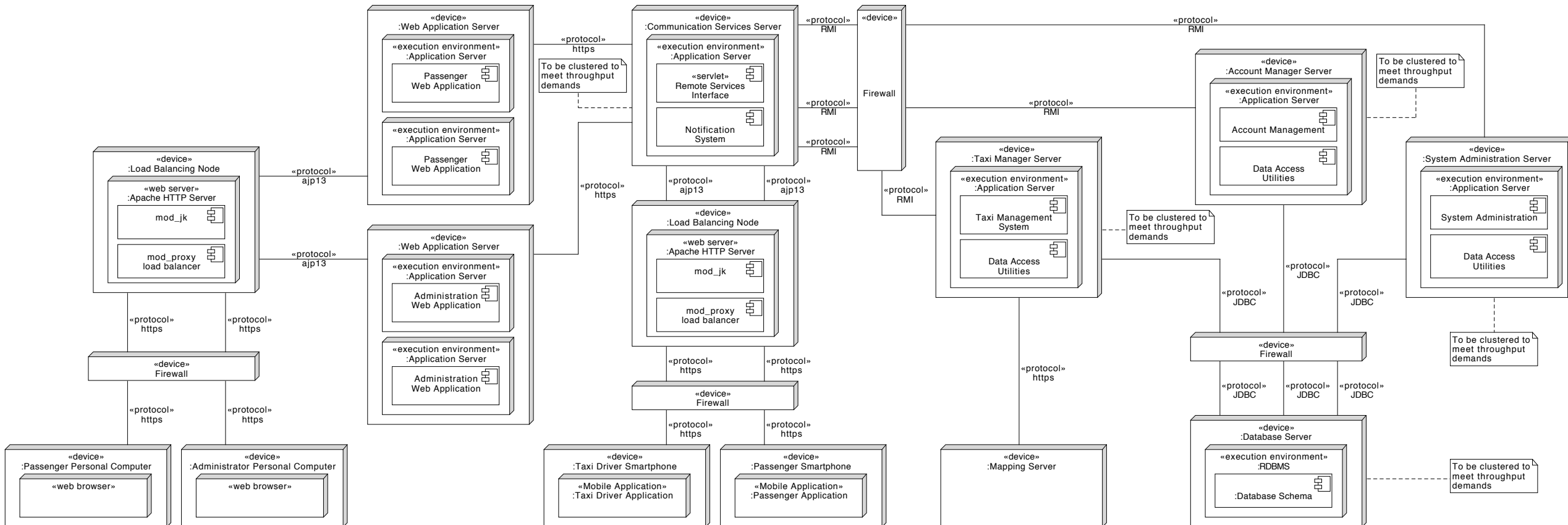
- Account Management
  - Passenger registration and login procedure
  - Settings management and password retrieval
- Taxi Management System
  - Maintains the availability status of each taxi updated
  - Manages the taxi queue associated with each zone in the city
  - Accepts and handles taxi reservations
  - Fulfills taxi requests by selecting the first available taxi in the corresponding taxi zone
- Mapping Service
  - Reverse geocoding, maps and ETA
- System Administration
  - Insert, update, delete taxis and zones
  - API permission management
  - Statistical queries and service monitoring
- Remote Services Interface
  - Exposes APIs to third party services
- Notification System
  - Sends notification to users
- Data Access Utilities
  - Mediates access to DBMS
- Database Management System (DBMS)

# High level components

- **Taxi Management System:**
  - Reservation Management
  - Request Management
  - Location Management
  - Taxi Management
- **Account Management:**
  - Passenger Registration
  - Login
  - Password Retrieval
  - Settings Management
- **System Administration:**
  - API Permission Management
  - Zone Division Management
  - Taxi Driver Management
  - Service Statistics
  - Plugin Management



# Deployment view



# Remote API

---

- Methods exposed by the core system via a web service interface
- Service Oriented Architecture (SOA)
- XML-based requests (SOAP)
- Managed by the JAX-WS API of Java EE
- Each session bean exposes a subset of the available methods
- Plugins are allowed to extend the offered API
- Remote third party services can invoke the exposed methods
- Different levels of privileges are supported
- Details discussed in the DD (Component Interfaces section)



# Architectural Styles

---

- Service Oriented Architecture (SOA)
  - Used to expose the APIs of the core system
  - Allows easily expansion of the offered functionalities
- Layered Architecture
  - Separation of concerns
  - Great clarity and flexibility
  - Different services run on different machines
- Client/Server Architecture
  - Business Logic implemented in servers
  - Clients used only for presentation purposes
- 4-tier Architecture
  - Better security and resilience
  - Critical services are isolated and protected from external attacks
- Cloud Architecture
  - Great scalability of hardware resources
  - Cost-effective
  - No necessity of maintaining a server farm
- Publisher/Subscriber Architecture
  - Used to implement the notification system
  - Flexible with respect to further expansions

# Other design decisions

---

## CORE SYSTEM IMPLEMENTATION: JAVA EE

- Supports multi-tiered applications
- Highly reliable
- Interoperable (JAX-WS)
- Automatically manages reliable DB transactions, resource allocation and secure network communications
- Supports load balancing
- Mature, well known, supported technology

## APPS: NATIVE FRAMEWORKS AND HTML5

- Native look & feel on each platform
- Supports platform-specific advancements
- Very small codebase
- Only implements presentation
- WebApps implemented in HTML5
- Available both on desktop and mobile
- Tailored to the specific form-factor