# 1 Overall Description

## 1.1 Product Perspective

The platform aims at replacing the traditional means of communication between taxi drivers and customers, and it is therefore designed as a stand-alone entity that is able to operate with a minimum set of dependencies towards external systems.

The external interfaces on which the system relies, in order to be operational, are described in the following subsections.

### 1.1.1 System Interfaces

- Google Maps API: used in both the back-end and the taxi-side applications, this free API is necessary to elaborate routes and display maps. The system interacts with the API through standard GET requests.
- GPS: this well known system is required by the mobile applications to get information about the users' position. The interaction happens through the standard GPS protocol.

### 1.1.2 User Interfaces

### 1.1.3 Hardware Interfaces

- Router: the internet connectivity of the back-end system towards the external world is managed by a physical router that automatically manages ports and resources in order to realize standard communication protocols such as NAT and DHCP.
- Firewall: access to the back-end system is protected through the use of a professional grade physical firewall which is directly linked to the router and automatically allocates ports and resources.
- Device support: the back-end system is designed to be run on usual server hardware with a high fault-tolerance degree and supports hardware scaling (such as replication); the mobile applications are designed to be run on ARM based architectures; the user-side web application does not have specific hardware requirements, since it relies on a different level of abstraction than its mobile counterpart which is not to be taken into account in the design of the hardware infrastructure.

### 1.1.4 Software Interfaces

- GNU/Linux Red Hat Enterprise Linux: operating system run on the back-end that acts as a virtualization level between the hardware and the software. Detailed documentation about the software can be found at https://access.redhat.com/documentation/en/red-hat-enterprise-linux/

- Apache WebServer: software bundle run on the back-end, which is used to allow interaction between the back-end system and the user-side application. It is used to handle standard HTTP requests from the customer-side web application and to provide CGI services (Apache Module mod_cgid - https://httpd.apache.org/docs/2.4/mod/mod_cgid.html) to all the user-side applications. Detailed documentation about the software is found at https://httpd.apache.org/docs/2.4/
- SQLite: DBMS software component run on the back-end, which is used to store user data in a non-resource-intensive fashion. Detailed documentation about this module is found at https://www.sqlite.org/docs.html

#### 1.1.5 Connection Interfaces

- EDGE, HSDPA, 3G, 4G
- HTML, SOAP/REST protocols

#### 1.1.6 Memory Constrains

- Server: 128GB vRAM required, 512GB recommended
- 2TB storage capacity + 64GB flash (32GB x2 mirroring) for applications system and server OS

#### 1.1.7 Operations

- Taxi Operations: login, signup, availability notifications, request reply, report
- User Operations: login, signup, requests operations, user profile operations (show, edit)
- System Operations: notifications (to user and taxi driver), queue management
- Backup operations (scheduled at 4AM every day)
- Restore operaions from secondary storage

#### 1.1.8 Site adaptation requirements

- UPS system
- 250V/110V Electrical system
- Ethernet connection
- Anti-thief system, safety room (safety grid on each door and eventually windows) and alarm

- Anti-fire system and alarm
- Anti-flooding system
- Air conditioning

## 1.2 Product Function

### 1.2.1 Taxi Function

- Taxi Login: the taxi driver logs into the system;
- Taxi Sign-up: the taxi driver provides his personal information, Username and Password, taxi and driving license to the Operator;
- Notification of Availability: with this operation the taxi driver notifies is presence in a specific zone of the town;
- Notification of Unavailability: the taxi driver notify the end of his service;
- Taking charge: the taxi driver positively reply to a customer request;
- Reject request: the taxi driver negatively reply to a customer request;
- Report user: the taxi driver report a problem with customer;

### 1.2.2 User Function

- User Login: the user get logged into the system providing his account information;
- User Sign-up: the user (previously Guest) request a signup to the system providing his personal information;
- Request Taxi: the user request a taxi to his location;
- Book Taxi: the user reserve a taxi to a specific location at the specific time;
- Cancel request: the user cancel the previous request or reservation;
- Report taxi: the user report a problem with the taxi driver or the service;
- Show Profile: the system provides the user information;
- Modify Profile: the user edit his information;

#### 1.2.3 System Function

- Notification to User
  1. Incoming taxi: a taxi take charge of the request;
  2. No taxi available;
- Notification to Taxi driver
  1. Taxi request: a user need a taxi;
  2. First of the queue: the taxi is the first of the queue of the current zone;
  3. Queue change: notification to the taxi of the new position of the queue;
- Queues management: change the queues after a request reply or new taxi driver in zone;

### 1.3 User Characteristic

#### 1.3.1 User:

- Low knowledge of mobile app use;
- Age: over 18;

#### 1.3.2 Taxi Driver:

- Owner of taxi license and driver license;
- Low knowledge of mobile app use;
- Good knowledge of the local language;

### 1.4 Constrains

- GPS Coverage;
- Privacy policy;
- Devices Constrains (presence of GPS and Internet connection);

### 1.5 Assumption and Dependencies

### 1.6 Apportioning of requirements