# Lab2: Requirements

For the following cases define

1 Stakeholders

Drivers / car owner

Gas station owner

Administrator

Map and navigation service, payment service

Start up partner

Start up financer (bank, venture capital)

Competitors

Play store / Apple store (security, financial part)

Cloud service for operation

2 Context diagram and interfaces

Actors : Drivers / car owner, Gas station owner , Administrator, Map and navigation service, payment service

3 Functional requirements

|  |  |
| --- | --- |
| F1 | Manage gas stations |
| F1.1 | Create a new station, define services offered (gas, fuel electric) |
| F1.2 | Show stations in a certain area, on map |
| 1,2.1 |  |
|  | Show stations in a certain area, as a listing ordered by price |
|  |  |
|  |  |
| F2 | Select a station and navigate to it |
|  |  |
|  |  |
| F3 | Select a gas station and show all its properties (price of diesel, price of gas, quality level, opening times, distance from driver location ) |
|  | Select a station and Input price of fuel for it |
|  | Select a station and Input quality level for it (by driver only) |
|  |  |
| F4 | Manage peer reviews |
| F4.1 | User enters review for a certain gas station (driver) |
| F4.2 | System computes aggregate review for a station |
| F4.3 | Select algorithm for aggregating reviews (administrator) |
| F4.4 | Recognize reviews from trolls (how?) |
| F5 | Manage user and user preferences |
|  | Create user (account, pwd) |
|  | Define user preferences (favorite station, favorite fuel or electric, car name and type) |
|  |  |

4 Non functional requirement

Response time for 4.4 should be 1sec, 4.3 should be 0.1 sec

Response time for all functions should be <0.5 seconds

Reliability

Privacy

Safety

Security and privacy

Used memory should be ..

## EZGas

EZGas is an application to help drivers find gas at lowest prices.

Gas station owners can register their gas station with prices and eventually discounts. Users look for gas stations closest to them and with best prices and quality of service.

## Robotic vacuum cleaner

Since several years robotic vacuum cleaners (RVC) are available. An RVC is capable of cleaning the floors of a house in autonomous mode.

An RVC system is composed of the robot itself and a charging station. The charging station is connected to an electric socket in the house, and allows charging the battery on board of the robot.

The robot itself is composed of mechanical and electric parts, a computer, and sensors. One infrared sensor in the frontal part recognizes obstacles, another infrared sensor always on the frontal part recognizes gaps (like a downhill staircase). A sensor on the battery reads the charge of the battery. The computer collects data from the sensors and controls the movement of four wheels. Another sensor on one of the wheels computes direction and distance traveled by the robot.

Finally on top of the robot there are three switches: on-off, start, learn.

The learn button starts a procedure that allows the robot to map the space in the house. With a certain algorithm the robot moves in all directions, until it finds obstacles or gaps, and builds an internal map of this space. By definition the robot cannot move beyond obstacles, like walls or closed doors, and beyond gaps taller than 1cm.

The starting point of the learn procedure must be the charging station. When the map is built the robot returns to the charging station and stops.

The start button starts a cleaning procedure. The robot, starting from the charging station, covers and cleans all the space in the house, as mapped in the ‘learn’ procedure.

In all cases when the charge of the battery is below a certain threshold, the robot returns to the charging station. When recharged, the robot completes the mission, then returns to the charging station and stops.