# Official Requirements Document

Authors: CHEN JINZHUO (S274294)

Date: 24/03/2020

# Contents

- [Abstract](#abstract)

- [Stakeholders](#stakeholders)

- [Context Diagram and interfaces](#context-diagram-and-interfaces)

+ [Context Diagram](#context-diagram)

+ [Interfaces](#interfaces)

- [Stories and personas](#stories-and-personas)

- [Functional and non functional requirements](#functional-and-non-functional-requirements)

+ [Functional Requirements](#functional-requirements)

+ [Non functional requirements](#non-functional-requirements)

- [Use case diagram and use cases](#use-case-diagram-and-use-cases)

+ [Use case diagram](#use-case-diagram)

+ [Use cases](#use-cases)

+ [Relevant scenarios](#relevant-scenarios)

- [Glossary](#glossary)

- [System design](#system-design)

- [Deployment diagram](#deployment-diagram)

# Abstract

Each gas station in Italy is free to define the price of fuels (regular gasoline, premium gasoline,

diesel). Drivers are interested in finding as easily as possible the closest gas station with the

cheapest price.

# Stakeholders

| Stakeholder name | Description |

| Localization service |The smartphone app shows the gas stations in the area of the driver |

| Administrator |Uses the application to manage location and price|

| User (driver) |A driver first has to sign in to access the service. |

# Context Diagram and interfaces

## Context Diagram

```plantuml

left to right direction

actor Administrator as a

a -- (La Tazza)

```

## Interfaces

| Actor | Logical Interface | Physical Interface |

| ------------- |:-------------:| -----:|

|Administrator|GUI |Screen, keyboard|

User (driver)| PC or smartphone| GUI|

Admin | PC | GUI|

Localization service| GPS in smartphone| Localization API / OS call|

# Stories and personas

The user can customize what is shown (for instance show all gas stations and all prices, or only gas

stations that have prices below average, show all prices or only gasoline / only diesel).

The user can then select a gas station and ask directions to it (navigation function). The navigation

function is external (for instance uses a Google navigation service).

The prices are collected by the drivers themselves. So a function must be available for users to

communicate to the application the prices of a certain gas station. Therefore a price has a time tag,

and an obsolescence property too. The application has to manage somehow the reliability of prices,

in terms both of obsolescence, and trust of users that communicate the prices. Trust of users is

evaluated by reports or votes of other users (for instance if a user reports that the price of gas

station X is wrong, and user U has communicated that price recently, then the trust of U lowers.

# Functional and non functional requirements

## Functional Requirements

| ID | Description |

| ------------- |:-------------:|

| FR1 | User management: CRUD User, authorize and authenticate |

| FR2 |User trust: compute trust of user using the trust of his/her price reports |

| FR3 | Gas station management: CRUD Gas station |

| FR4 | Price report management. CRUD Price report, attach price report to user |

| FR5 | Price management. CRUD price management, compute price starting from(recent) price reports of a station, compute level of trust of the price reportSignal price is correct / incorrect |

| FR6 | Navigation. Compute path from position of user to defined gas station |

| FR7 | Show gas stations and their fuel prices on a map, customizable by area, fuel type |

## Non Functional Requirements

| ID | Type (efficiency, reliability, .. see iso 9126) | Description | Refers to |

| ------------- |:-------------:| :-----:| -----:|

| NFR1 | Usability | Application should be used with no training by any colleague in the office | All FR |

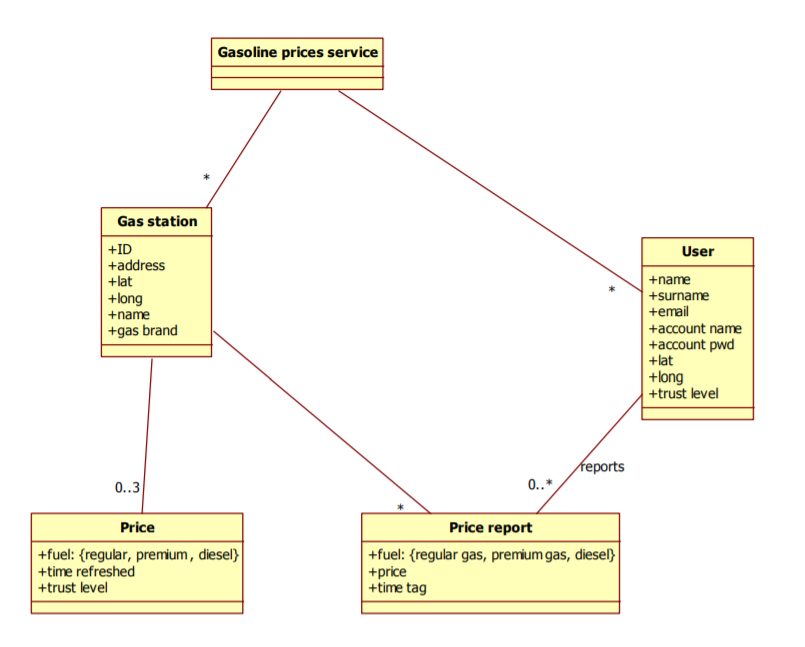
| NFR2 | Performance | All functions should complete in < 0.5 sec | All FR |

| NFR3 | Portability | The application (functions and data) should be portable from a PC to another PC in less than 5 minutes | All FR |

| NFR4 | Localisation | Decimal numbers use . (dot) as decimal separator ||

# Use case diagram and use cases

## Use case diagram



```plantuml

left to right direction

actor Administrator as a

a -- (FR1 Record price)

a -- (FR2 Record gas station)

a -- (FR3 Record price report)

a -- (FR4 Record user)

a -- (FR5 record gasoline prices service)

```

## Use Cases

### Use case 1, UC1 - FR1 Record price

| Actors Involved | Administrator |

| ------------- |:-------------:|

| Precondition | Capsule T exists, colleague C exists |

| Post condition | T.quantity\_post < T.quantity\_pre |

| | C.PersonalAccount.balance\_post < C.PersonalAccount.balance\_pre |

| Nominal Scenario | Administrator selects capsule type T, selects colleague C, Deduce quantity for capsule T, deduce price of T by account of colleague C|

| Variants | Account of colleague C has not enough money, issue warning |

### Use case 2, UC2 - FR2 Record gas station

| Actors Involved | Administrator |

| ------------- |:-------------:|

| Precondition | Capsule T exists, visitor has no account |

| Post condition | T.quantity\_post < T.quantity\_pre |

| | LaTazzaAccount.amount\_post > LaTazzaAccount.amount\_pre |

| Nominal Scenario | Administrator selects capsule type T, Deduce quantity for capsule T, add price of T on LaTazzaAccount.amount|

| Variants | |

### Use case 3, UC3 - FR3 Record price report

| Actors Involved | Administrator |

| ------------- |:-------------:|

| Precondition | Personal Account PA exists , quantity >0 |

| Post condition | PA.balance\_post = PA.balance\_pre + quantity |

| | LaTazzaAccount.balance\_post = LaTazzaAccount.balance\_pre + quantity |

| Nominal Scenario | Administrator selects account PA of colleague C, increase account of quantity, increase LaTazza account of quantity|

| Variants | |

### Use case 4, UC4 - FR4 Record user

| Actors Involved | Administrator |

| ------------- |:-------------:|

| Precondition | Capsule type CT exists, LaTazzaAccount.balance has enough money for the purchase|

| Post condition | CT.quantity\_post > CT.quantity \_pre |

| | LaTazzaAccount.balance\_post < LaTazzaAccount.balance\_pre|

| Nominal Scenario | At time of order Administrator records money spent for order. At time of reception administrator selects capsule type CT, increases its quantity by a given number|

| Variants | |

### Use case 5, FR5 record gasoline prices service

| Actors Involved | Administrator |

| ------------- |:-------------:|

| Precondition | Colleague C exists |

| Post condition | |

| Nominal Scenario | Administrator selects colleague C, defines a time range, application collects all transactions for C (recharges and capsules taken) in the time range and presents them|

| Variants | |

# Relevant scenarios

## Scenario 1

| Scenario ID: SC1 | Corresponds to UC1 |

| ------------- |:-------------|

| Description | Obtain price of fuels|

| Precondition | Driver is registered and has valid account Driver has defined preferences (fuel = diesel, criterion = lowest price within 5km radius) Driver accesses via smartphone|

| Postcondition | system shows gas stations on map |

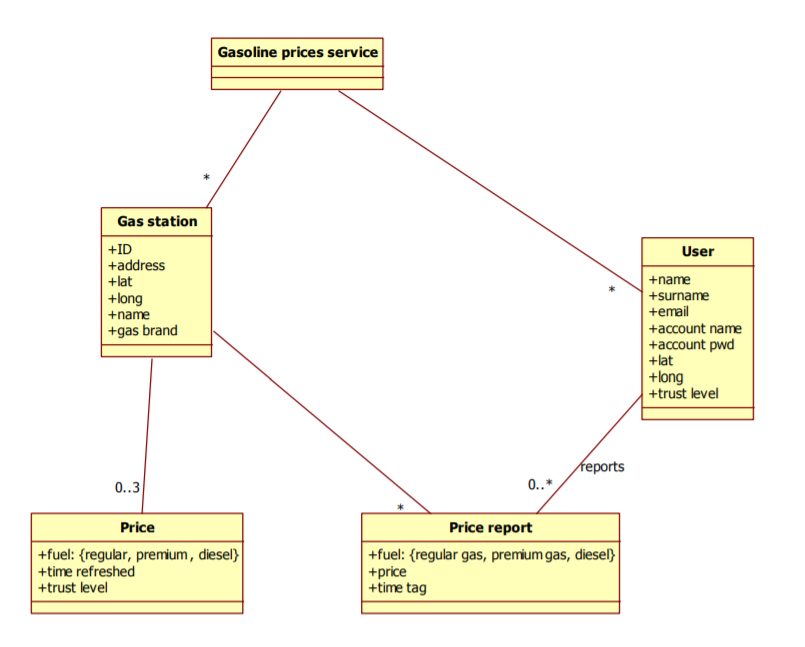
| Step# | Step description |

| 1 | login |

| 2 | system retrieves location of driver, retrieves map, retrieves gas stations and prices in 5km radius, computes average price on those gas stations |

| 3 | logout |

# Glossary



```plantuml

class Price

class Price {

+ fuel

+ time refreshed

+ trust level

}

class PriceReport {

+ fuel

+price

+time tag

}

class GasStation {

+ ID

+address

+ lat

+long

+name

+gas brand

}

class User{

+ name

+surname

+email

+account name

+account pwd

+lat

+long

+trust level

}

class GaslinePriceService {

}

# System Design

Not really meaningful in this case. Only one personal computer is needed.

```plantuml

class "Personal Computer"

```

# Deployment Diagram

As a stand-alone application only one node is needed.

```plantuml

artifact "La Tazza Application" as a

node "Personal Computer" as n

a -- n

```