**# Official Requirements Document**

Author : Giovanni Tangredi

Date: 24/03/2020

Version: 5

Changes:

2 - Added functional requirements and nonfunctional requirements

- Fixed some errors

3 - Added User Case Diagram and some User Cases

* Fixed some error, modified FRs

4 - Added others user cases, added Relevant scenarios

* Fixed some errors

5 - Fixed some errors

**#Contents**

* Abstract
* Stakeholders
* Context Diagram and Interfaces
* Context Diagram
* Interfaces
* Stories and personas
* Functional and Nonfunctional requirements
* Functional requirements
* Nonfunctional requirements
* Use case diagram and use cases
* Use case diagram
* Use cases
* Relevant scenarios

**Abstract**

EZgas‘ developers want to make it easier for thousands of people to find and compare the fuel prices of the numerous gas stations in their area.

Ezgas is a web app that locates the gas stations in the area of the user, shows them and displays the current fuel prices.

Visitors can only see the location of the gas stations and corresponding prices, Registered Users can also modify prices of the gas station so that the app has always the most updated and correct prices.

The Users can insert and update fuel prices of the gas stations .

For not wasting too many resources and time the app’s developers decided to use a pre-exist Map Services (Ex. Google Maps) without developing one on their own.

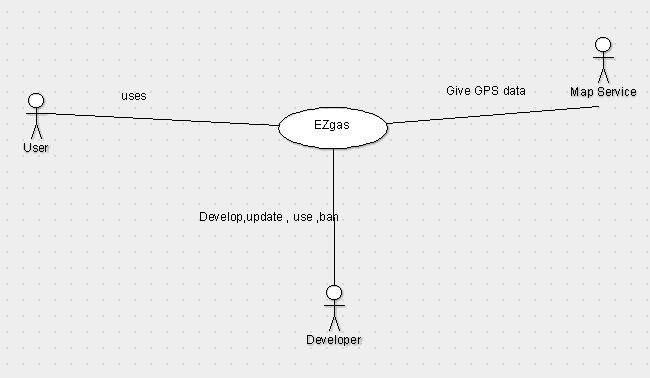
The developers update the app regularly and they take care of problems and faults in the database/app and of the communication between the app and the Map Service. They can also delete/ban users.

**Stakeholders**

|  |  |
| --- | --- |
| Stakeholder Name | Description |
| Developer | Create and update the app features , can delete/ban users. |
| User | Uses the app, can update fuel prices of a gas station |
| Map Service | Used to get a user or visitor position and to locate close gas stations. |

**Context Diagram and Interface**

**Context Diagram**



**Interfaces**

|  |  |  |
| --- | --- | --- |
| Actor | Logical Interface | Physical Interface |
| Map Service | REST API | Internet Connection |
| User/Visitor | GUI | TouchScreen,Monitor, Keyboard |
| Developer | GUI,IDE | Monitor, Keyboard |

**Stories and Personas**

Alberto is on holiday in Pescara, he’s noticed that his car is quite out of gas and he uses the app to check if there are any good gas stations nearby.

He selects a radius of 1km from his position and gets the results from the app , he then decides to go to the gas station with the lowest prices.

Sofia travels a lot so she is registered into the EZgas app , she wants to know in advance what the cheapest gas station is close to her motel where she will stay so she selects the motel address as location and searches for the most suitable answer.

Marco has just opened a new gas station with lower price than competition so to advertise his new business he creates a new EZgas user account and inserts his gas station into the app. After a month he can decrease the prices again so he updates his gas station info on the app.

**Functional and non functional requirements**

**Functional requirements**

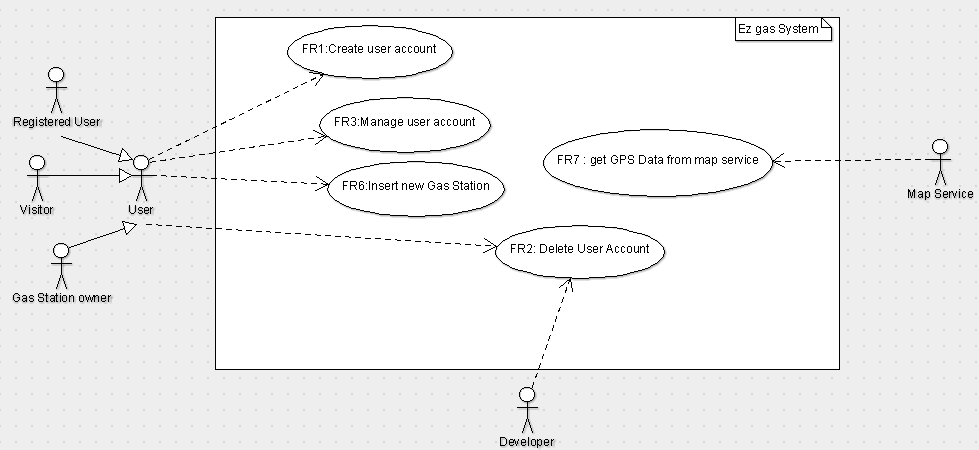
|  |  |
| --- | --- |
| ID | Description |
| FR1 | Create a new user account |
| FR2 | Delete a new user account |
| FR3 | Manage user account |
| FR4 | Find Gas Stations |
| FR5 | Show info and prices of selected gas station |
| FR6 | Insert new gas station |
| FR7 | Get GPS data from Map Service |
| FR8 | Select an arbitrary radius |
| FR9 | Update Prices |

**Non functional requirements**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Type | Description | Refer to |
| NFR1 | Usability | User/Visitor should receive a response in <1 s | All FRs |
| NFR2 | Portability | The app should run on MS Windows10 , Android and iOS | All FRs |
| NFR3 | Portability | The App should be easily ported from one device into another in less than 10 min | All FRs |
| NFR4 | Efficiency | All function should complete in <0.5s | All FRs |
| NFR5 | Localization | Km used for distances, price can be set in €,$ and £ |  |
| NFR6 | Maintainability | Database ,App and Website should be constantly updated |  |

**Use case diagram and use cases**

**Use case diagram**

****

**Use cases**

Use Case 1,UC1 - FR1 - Create user account

Actor involved: Visitor

Pre-Condition: Account does not exist

Post-Condition: Account exist

Nominal Scenario: Visitor insert Username email and password, the account is create automatically by the system

Use Case 2 UC2 - FR2 Delete user account

Actors involved: User

Pre-condition: Account exists

Post-Condition: Account does not exist

Nominal Scenario: User send elimination request, the request is accepted by the system, account and all data associated is lost

Use Case 3 UC3 - FR2 Delete user account

Actors involved: Developer

Pre-condition: Account exists

Post-Condition: Account does not exist

Nominal Scenario: Developer send manually eliminate the User account, account and all data associated is lost

Use Case 4 UC4 - FR3 Manage user account

Actors involved: User

Pre-condition: Account exists

Post-Condition: Account data modified,Account exists

Nominal Scenario: User go to his profile page and do the modification

Use Case 5 UC5- FR6 Insert new gas station

Actors involved: User

Pre-condition: Gas Station not exists

Post-Condition: Gas Station exists

Nominal Scenario: User insert the necessary information, Gas Station is created

Use Case 6 UC6 - FR7 :Get GPS data from Map service

Actors involved: Map Service

Pre-condition: Can connect to the Map service, Map service has the necessary data

Post-Condition: Get position of the user

Nominal Scenario:User want to use the app, his position is taken from the Map Service, user give a radius, the Gas station in the radius are showed

Use Case 7 UC7 - FR8 :Update Prices

Actors involved: User

Pre-condition: Gas station and User exists

Post-Condition: Prices changed

Nominal Scenario:User select a gas station , change the prices , system apply changes

**Relevant Scenarios**

Scenario 1:

Scenario ID: SC1 - Correspond to UC 5

Description: User insert new gas station

Precondition: The gas station not exists

Post condition : The gas station is inserted

Steps:

1 - User select “insert new gas station”

2 - User compile the form with Name, location address and prices

3 - System heck if Name and address are correct and not used

4 - System adds new gas station

Scenario 2:

Scenario ID: SC2 - Correspond to UC 5

Description: User insert new gas station, but gas station already exists

Precondition: The gas station exists

Post condition : Issue Error, abort operation

Steps:

1 - User select “insert new gas station”

2 - User compile the form with Name, location address and prices

3 - System check if Name and address are correct and not used

4 - System issue an error

5 - Error is reported to the user

Scenario 3:

Scenario ID: SC3 - Correspond to UC 6

Description: User search for gas station in the area

Precondition: User position can be obtained, User has the map service

Post condition : User position obtained

Steps:

1 - Connect to Map Service Server

2 - Ask for User position

3 - Map Service send user GPS info to the System

4 - System save the data