# Lab2 (cont): requirements

Goal of this lab is to practice the techniques to formalize the requirements of a software product.

Consider the EZGAs case (Lab 2)

Create in your git repository a markdown document with the following structure (requirements document) and fill it in

# Stakeholders

| Stakeholder name | Description |

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

# Context Diagram and interfaces

## Context Diagram

## Interfaces

| Actor | Logical Interface | Physical Interface |

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

# Functional and non functional requirements

## Functional Requirements

| ID | Description |

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

| FR1 | tbc (To be completed) |

## Non Functional Requirements

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

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

| NFR1 | tbc | tbc | tbc |

# Use case diagram and use cases

## Use case diagram

## Use Cases

### Use case 1, name tbc

| Actors Involved | tbc |

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

| Precondition | tbc |

| Post condition | tbc |

| | tbc |

| Nominal Scenario | tbc |

| Variants | tbc |

# Relevant scenarios

## Scenario 1

| Scenario ID: tbc | Corresponds to UC x |

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

| Description | tbc |

| Precondition | tbc |

| Postcondition | tbc |

| Step# | Step description |

| 1 | |

| 2 | |

| 3 | |

| 4 | |

For the diagrams you can use tools like Argo UML, Astah, Star UML, Plant UML.

Always consider the possible defects in a requirement document (omissions, inconsistencies, ambiguities ..).

Consider that the document must be sufficient to:

* allow another team (not you) design and code the application
* allow another team (not you) to test the application