|  |
| --- |
| Groep 1 – Cooperatio |
| System requirements specificationnnnn |
| System requirements specification Project56: Data analysis |

Cees-Jan Nolen, Steven Schenk, William de Visser, Robin Bakker, Robert Kraaijeveld - Rotterdam, 1-10-15

Version: 1.0

Table of contents

[1. Introduction 3](#_Toc431376582)

[1.1 Goal of this document 3](#_Toc431376583)

[1.2 Scope 3](#_Toc431376584)

[1.3 Sources 3](#_Toc431376585)

[1.4 Version-control 3](#_Toc431376586)

[1.5 Target audience 3](#_Toc431376587)

[2 Description 4](#_Toc431376588)

[2.1 Systemcontext 4](#_Toc431376589)

[2.2 Systemproperties 5](#_Toc431376590)

[2.3 Systemenvironment 6](#_Toc431376591)

[2.4 Restrictions to design and implementation 7](#_Toc431376592)

[2.5 Userdocumentation 7](#_Toc431376593)

[2.6 Assumptions and dependencies 7](#_Toc431376594)

[3. Functional requirements 8](#_Toc431376595)

[3.1 Business requirement 1: The system will be able to read and save CityGis-data at high speeds. 8](#_Toc431376596)

[3.2 Business requirement 2: The systems’ architecture will enable the creation and the saving of parallel aggregated reports 8](#_Toc431376597)

[3.3 Business requirement 3: The system will provide a web-interface that will display CityGis-software and hardware performance statistics for use in product development 9](#_Toc431376598)

[3.4 Business requirement 4: The system will provide a web-interface in which vehicle maintenance-data can be made into reports and downloaded. 9](#_Toc431376599)

[3.5 Business requirement 5: The system will provide a web-dashboard on which some important vehicle-maintenance data points will be shown to the vehicle administrator. 10](#_Toc431376600)

[3.6 Business requirement 6: The system will provide a web-portal on which CityGis-meta data will be up for sale to 3rd parties. 10](#_Toc431376601)

[4. Interface requirements 11](#_Toc431376602)

[4.1 Userinterfaces 11](#_Toc431376603)

[4.2 Hardware-interfaces 12](#_Toc431376604)

[4.3 Software-interfaces 12](#_Toc431376605)

[5 Non-functional requirements 14](#_Toc431376606)

[5.1 Business requirement 1: The system will be able to read and save CityGis-data at high speeds. 14](#_Toc431376607)

[5.2 Business requirement 2: The systems’ architecture will enable the creation and the saving of parallel aggregated reports. 14](#_Toc431376608)

[5.3 Business requirement 3: The system will provide a web-interface that will display CityGis software- and hardware performance statistics, for use in product development. 14](#_Toc431376609)

[5.4 Business requirement 4: The system will provide a web-interface in which vehicle maintenance-data can be made into reports and downloaded. 14](#_Toc431376610)

[5.5 Business requirement 5: The system will provide a web-dashboard on which some important vehicle-maintenance data points will be shown to the vehicle-administrator. 15](#_Toc431376611)

[5.6 Business requirement 6: The system will provide a web-portal on which CityGis-meta data will be up for sale to 3rd parties. 15](#_Toc431376612)

[6. User Requirements 16](#_Toc431376613)

[User stories: 16](#_Toc431376614)

[1.1 Use-case diagram 18](#_Toc431376615)

[1.2 Use case description 18](#_Toc431376616)

[6.3 Traceerbaarheid 23](#_Toc431376617)

# Introduction

## Goal of this document

This document specificies the requirements of Project56. The goal of this document is to provide an overview of all the functionalities that Project56 will provide.

## Scope

This SRS is a part of Project56, which will be a data-analysis system. By using a web interface, the user of this system will be able to browse through statistics and generated reports about different parts of CityGis-data.

## Sources

## Version-control

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Version | Status | Date | Authors | Remarks |
| 1.0 | Concept | 24-9-2015 | Cees-Jan Nolen, Robin Bakker, Steven Schenk, William,Robert Kraaijeveld |  |
| 2.0 | Update | 1-10-2015 | Cees-Jan Nolen, Robin Bakker, Steven Schenk, William,Robert Kraaijeveld |  |
|  |  |  |  |  |

## Target audience

This document has been made to establish a shared link between the Project Leader, the stakeholders and the development team. This document contains all the requirements that will be applied to the system so that; the users will know what to expect, the project leader will

Be able to see what requirements the software will fulfill according to the wishes of the stakeholders, and finally the development team will know what to implement.

The following stakeholders can be identified for this project:

1. The project leader
2. The potential users of the system
3. CityGis

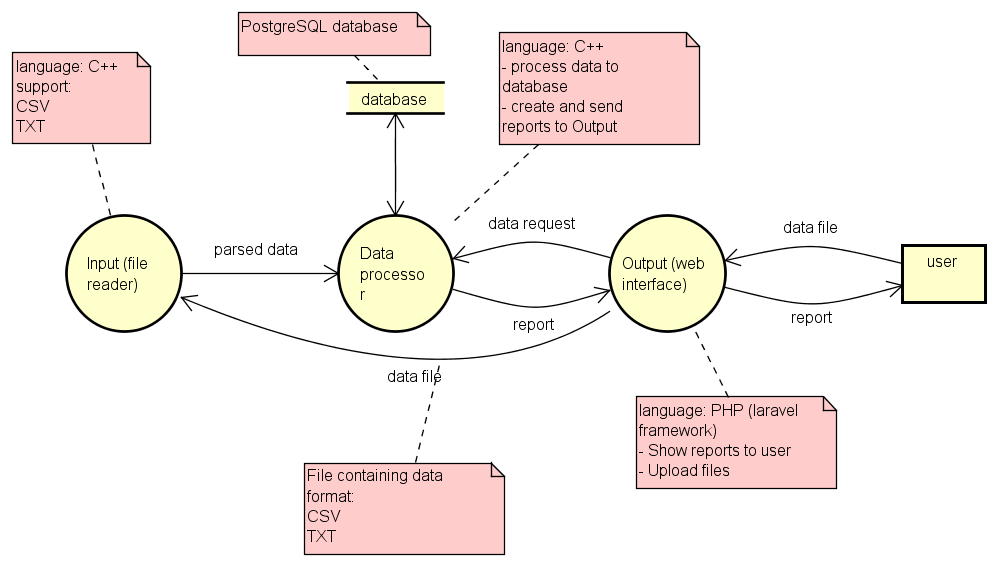
# 2 Description

## 2.1 Systemcontext

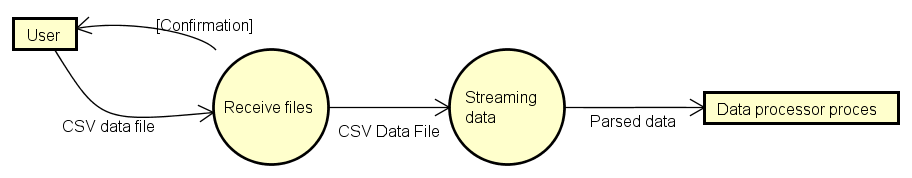
The system will be divided into three subsystems:

1. Input-system.
2. Data-processing system.
3. Output-providing system.

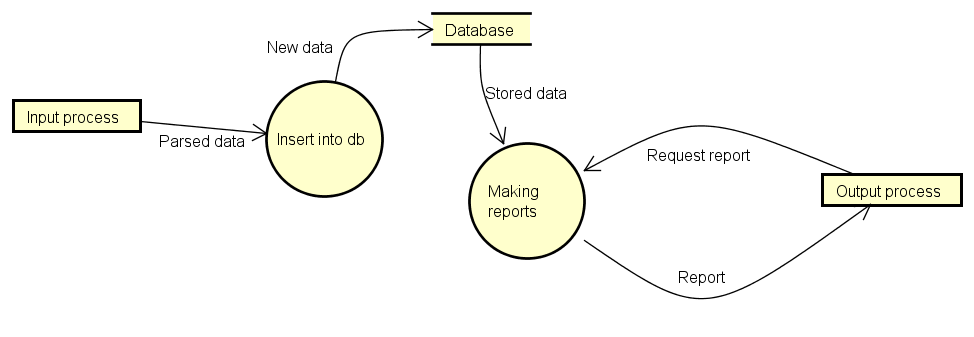
Together these three subsystems will form the complete system. The figure below will provide information about the systems’ boundary and how the system will interact with other systems and users from outside the system boundary.

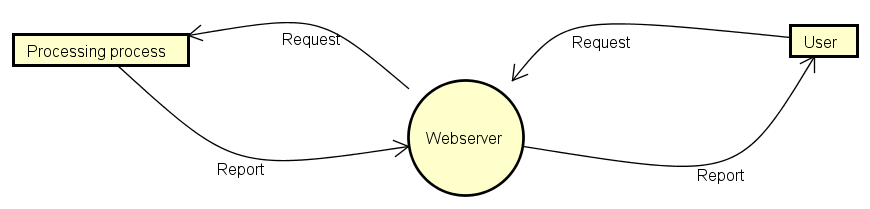


**Figuur 1: Context diagram and system boundary**

**Figuur 2: DFD Input**

*Figuur 3: DFD processing*

*Figuur 4: DFD output*



The aforementioned components are contained within the systems’ boundary. Within this boundary the 3 components interact with eachother; out of this boundary the components interact with users and files.

1. The input-component takes as input .CSV and .TXT files, either from pre-existent files or from files that are manually added by the user, and parses them so the processing component can insert the data-values into the database.
2. The processing-component takes as input the parsed data that the input-component created, and inserts these values into a PostgreSQL database. The processing component is also responsible for the creating of data-reports and aggregations, and sending these to the output-component.
3. The output-component takes as input the aggregated data-values from the processing-component, and presents these values to the user in a visual way, for example in graphs. The output-component also interacts with the user, allowing the user to add new data-files to the database and request reports.

# 2.2 Systemproperties

### 2.2.1Input-system

The input-system reads and parses the .CSV and .TXT files that it receives as input as fast as possible, and prepares them for the journey to the processing-system. This input-system has no direct link with the user, but receives files through the output-system. This system will be implemented as a REST-webservice, in order to create a more SOA-like approach throughout the system.

### 2.2.2Processing-system

The processing-system receives the parsed data that the input-system sent it, and inserts this data into our PostgreSQL database, as fast as possible of course. The processing system is also responsible for the aggregation and creation of .PDF reports (as requested from the output-system) and sending them back to the output system. This system will be implemented as a REST-webservice, in order to create a more SOA-like approach throughout the system.

### 2.2.3 Output-system

The output-system is comprised of several web-interfaces that allow the user to look at several sets of the CityGis-data in a visual way, and look up averages and measurements. The output-system also allows the user to upload his/hers own data, sending this new data to the input-system for processing. The output-system also passes any user-requests for reports to the processing-system.

## Systemenvironment

The system will be hosted on a Hogeschool Rotterdam server that will be leased by the development team. The system and the server it runs on will be using the following hardware and software during development:

##### Hardware

##### CPU: Intel Xeon e5-2650 @2.00 Ghz

##### Memory: 3953 Megabytes

##### Disk: 15 Gigabytes

##### OS: Ubuntu 14.04.2

##### Software

- C++ programming language

- PHP inc. Laravel framework

- PostgreSQL

- Git inc. Github

- Homestead\*

- Vagrant\*

- Composer\*

\*: These three software-systems are used to accommodate development in the Laravel framework.

## Restrictions to design and implementation

For this Project to succeed, it is of utmost importance that enough resources are available. It is very important that the system can be hosted on a fast server that has enough diskspace available to accommodate the large files that will be used, processed and added. A steady (internet) connection from this server to the development team is also necessary.

The development team must also use Git and Github for version control, and deposit documents and source code files in their respective git-branches.

## Userdocumentation

The system will contain a FAQ page and a customer-service-contactpage.

## Assumptions and dependencies

The user requirements were initially not provided by CityGis and were thusly created by the developers themselves. Because of this fact they may be (slightly) inaccurate and may be subject to change, according to the wishes of the Project leader.

The server that the development team has acess to at the moment of writing may not be adequate should the data-filesize grow very much (Estimated file-sizes lead up to about 10 gigabytes of data).The current setup of the server also does not allow the creation of virtual machines, which will hamper the configuration and usage of parallel processes.

All acces to the actual user-base of the system will be conducted through the Project leader, because CityGis itself could not be contacted directly. This might also make user requirements, functional requirements and non-functional requirements less accurate, as mentioned before.

For now, the system will only be compatible with data-files in the .CSV and .TXT extension, as these are two common file extensions for files of this size and the present data-files were also supplied in the .CSV format.

CityGis is not able to supply the development team with Live-feed of data-files, because of both practical- and privacyreasons. This will render the development team unable to create for instance streaming-functions in the system, as there is no real live-data.

# 3. Functional requirements

## 3.1 Business requirement 1: The system will be able to read and save CityGis-data at high speeds.

### 3.1.1 Importing CityGis files

The system will be able to import CityGis files according to the assumptions provided in chapter 2.6 of this document. *Priority: High*

### 3.1.2 Importing GUI

The system will have a graphical user interface in which the user will be able to import data-files, according to the Interface requirements of chapter 4 of this document. *Priority: High*

### 3.1.3 Importing-process error-reporting

The system will inform display any kind of errors that interrupt the import process to the user. *Priority: Medium*

## Business requirement 2: The systems’ architecture will enable the creation and the saving of parallel aggregated reports

### Selecting data

The system will enable the user to select about which data-parts he wants to create a report. *Priority: High*

### 3.2.2 Report aggregation

The system will automatically aggregate information from the chosen data-parts and enter these into the report when it is generated. *Priority: High*

### 3.2.3 Downloading of aggregated reports

The system will enable the downloading of reports about the data-parts to the PC of the user. *Priority: High*

### 3.2.4 Report Parallel downloading

The system will enable the generation and downloading of multiple reports at the same time. *Priority: High*

## Business requirement 3: The system will provide a web-interface that will display CityGis-software and hardware performance statistics for use in product development

### 3.3.1 Performance statistics web-interface

The system will provide a simple web-interface on which hardware- and software performance statistics will be displayed. *Priority: High*

### 3.3.2 Performance Measurements

The web interface will display aggregated measurements and averages of the software- and hardware-data. (which measurements the system will used will be determined at a later phase together with the Product Owner) *Priority: High*

### 3.3.3 Performance-Report downloading

The web interface will allow the generating and downloading of reports about the performance statistics. *Priority: High (See also Business requirement B2)*

## Business requirement 4: The system will provide a web-interface in which vehicle maintenance-data can be made into reports and downloaded.

### 3.4.1 Maintenance-data Web interface

The system will provide web-interface on which maintenance data statistics will be displayed. *Priority: High (See also Business requirement B3)*

### 3.4.2 Maintenance-data Aggregation

The system will automatically aggregate information from the data-parts and enter these into the report when it is generated. *Priority: High (See also Business requirement B2)*

### 3.4.3 Maintenance-data Downloading

The system will enable the downloading of reports about the data-parts. *Priority: High*

*(See also Business requirement B2)*

### 3.4.4 Maintenance-data Visual representation

The web interface will represent the measurements and averages in a visual way. *Priority: Medium*

*(See also Business requirement B3)*

## Business requirement 5: The system will provide a web-dashboard on which some important vehicle-maintenance data points will be shown to the vehicle administrator.

### 3.5.1 Alert visibility

The system will make sure that any alerts coming from the web-dashboard will be visible across al other web-interfaces. *Priority: High*

### 3.5.2 Alert information

Any alerts coming in on the web-dashboard will contain information about the problem at hand. *Priority: High*

### 3.5.3 Alert Forwarding

The web-dashboard will allow the user to forward the alert to his/her superiors.

*Priority: Medium*

## Business requirement 6: The system will provide a web-portal on which CityGis-meta data will be up for sale to 3rd parties.

### 3.6.1 Meta-data buying

The system will allow third-party buyers to buy CityGis meta data. *Priority: High*

### 3.6.2 Meta-data payment client

The system will allow third-party buyers to pay for their meta data using an external payment client. *Priority: Medium*

### 3.6.3 Meta-data Specification

The system will allow the users to specific pieces of meta-data that they want to buy. *Priority: High*

# 4. Interface requirements

## 4.1 Userinterfaces

The systems’ user interfaces will be the following:

### - Importing-interface.

In this interface the user will be able to import new .CSV and .TXT files into the system, and be notified of the importing-progress. This interface will be very minimal as far as graphics are concerned, as most users will not spend a lot of time in this interface.

### - Performance-interface.

In this interface the user will be presented various graphics and measurements about CityGis hardware- and software-performance data. What kind of graphics and measurements will be used depends on the exact wishes of the Project-leader and is therefore not specified in this document.

### - Maintenance-data interface.

In this interface the user will be presented various graphics and measurements about CityGis vehicle-maintenance data. What kind of graphics and measurements will be used depends on the exact wishes of the Project-leader and is therefore not specified in this document.

### - Alert-interface.

This interface will display alerts and notifications of important problems or impending problems that the system has seen or predicted. Alerts will contain basic information about the kind of problem the system has detected, and each alert will contain some simple buttons that will allow the user to forward the alert to superiors or co-workers, by mail for example.

### -Meta-data market interface.

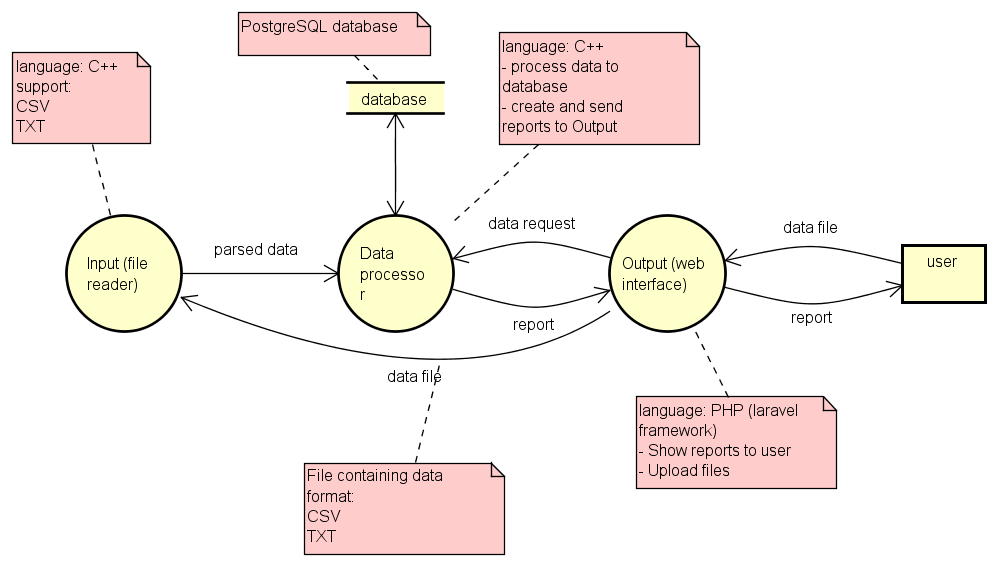
This interface will display meta-data collections that CityGis has put up for sale to the user. The meta-data collections will be provided in a simple list, with more detailed information and payment and download options for each collection if the user selects it in the list. Should a user decide to go ahead and purchase a meta-data collection, he/she will be redirected to a PayPal client.

- Help-interface

This interface will provide a basic FAQ about the rest of the web interface, and a means of contact to customer services should the user still not be able to find his/her answer.

## 4.2 Hardware-interfaces

The system will be run on the Hogeschool Rotterdam server that is currently being leased by the development team (you can find more detailed specifications in Chapter 2). This server will facilitate the 3 major system components: The input, processing and output system and it will also facilitate the PostgreSQL database.



## 4.3 Software-interfaces

As mentioned before, the system will have 3 major components and a PostgreSQL database.These will all be hosted on the Hogeschool Rotterdam server. The 3 major components (input, processing, output) will also be implemented as standalone web services, to ensure maximum portability in the future.

The 3 components input, processing and output will be implemented as REST-full web services, enabling them to portable to other servers in the future. The output, web service will send data files to the input-component, which will parse the data and pass it through to the processing web-service. This service will insert the data into the PostgreSQL database. The output-system is made up of multiple web-interfaces that display data from the processing-service in a visual way. The output-system also sends requests for report-aggregation to the processing-service.

# 5 Non-functional requirements

## 5.1 Business requirement 1: The system will be able to read and save CityGis-data at high speeds.

### 5.1.1 Importing Error reporting

The system should not report trivial errors, only those that can interrupt or otherwise damage the importing process.

### 5.1.2 Importing progress bar

The system will have a progress bar which will inform the user of the progress the importing process has made. *Priority: Medium*

### 5.1.3 Importing Speed

//The system save imported files as fast as possible.

## 5.2 Business requirement 2: The systems’ architecture will enable the creation and the saving of parallel aggregated reports.

### 5.2.1 Report Limits

The system will have a limit of 5 as to the amount of reports that can be created at the same time, so the system doesn’t crash unexpectedly.

### 5.2.2 Report Format

The system should generate the reports in .pdf format.

## 5.3 Business requirement 3: The system will provide a web-interface that will display CityGis software- and hardware performance statistics, for use in product development.

### 5.3.1 Performance report downloading Limits

The system will have a limit of 5 as to the amount of reports that can be created at the same time, so the system doesn’t crash unexpectedly.

### 5.3.2 Performance reports downloading GUI

The system will have a progress bar which will inform the user of the progress the generating process has made.

## 5.4 Business requirement 4: The system will provide a web-interface in which vehicle maintenance-data can be made into reports and downloaded.

### 5.4.1 Maintenance data limits

The system will have a limit of 5 as to the amount of reports that can be created at the same time, so the system doesn’t crash unexpectedly.

## 5.5 Business requirement 5: The system will provide a web-dashboard on which some important vehicle-maintenance data points will be shown to the vehicle-administrator.

*None.*

## 5.6 Business requirement 6: The system will provide a web-portal on which CityGis-meta data will be up for sale to 3rd parties.

### 5.6.1 Designation

The system should only put designated meta-data up for sale.

# 6. User Requirements

## User stories:

### 6.1 Business requirements 1: The system will be able to read and save CityGis-data at high speeds.

1 As a User, I want to be able to import CityGis-data in CSV format into the system, so that data can be analyzed. (**M**)

2 As a User, I want the system to be as fast as possible, so I don’t have to wait too long. **(M)**

3 As a User, I want to have a simple graphical interface for importing Data, so I can easily store my data. (**M)**

4 As a user, I want to be able to see a progress bar, so I can see how the importing process is coming along. **(S)**

5 As a User, I want the system to inform me of any possible errors in the importing process, so I know what went wrong. (**S**)

### 6.2 Business requirements 2: The systems’ architecture will enable the creation and the saving of parallely aggregated reports

1 As a User, I want to be able to download reports about the collected data and about more specific parts of that data, so I don’t have to go through all the data manually. (**M**)

2 As a User, I want the system to automatically aggregate useful information about the collected data, so I don’t have to select what I want in the report by hand. (**M**)

4 As a User, I want to be able to download these generated reports in pdf, so I can easily distribute these reports through to my colleagues. (**S**)

5 As a User, I want to be able to generate and download multiple reports at the same time, so I do not have to wait for each one of them to finish before the other. (**M**)

6 As a User, I want to be able to see progress bar of the generating/downloading process, so I know when I can safely quit the program. (**S**)

### 6.3 B3: The system will provide a web –interface that will display CityGis software- and hardware performance statistics, for use in product development.

1 As a User, I want the web-interface to be easily understood and navigated, so I do not need to be an expert-data analyst in order to use it. (**M**)

2 As a User, I want the web interface to display simple measurements and averages, so I don’t have to make these calculations myself. (**M**)

3 As a User, I want the web interface to display the statistics in a visual way, so I can easily see which components require further attention. (**S**)

4 As a User, I want to be able to generate and download reports containing the performance statistics, so I don’t have to go through all the data manually. (**S***) (See also Business requirement B2)*

### 6.4 Business requirements 4: The system will provide a web-interface in which vehicle maintenance-data can be made into reports and downloaded.

1 As a User, I want the web-interface to be easily understood and navigated, so I do not need to be an expert-data analyst in order to use it. (**M**) *(See also Business Requirement B3)*

2 As a User, I want the system to automatically aggregate useful information about the collected data, so I don’t have to select what I want in the report by hand. (**M**) *(See also Business Requirement B2)*

3 As a User, I want to be able to download reports about the collected data, so I don’t have to go through all the data manually. (**M**) *(See also Business Requirement B2)*

5 As a User, I want the system to also represent the vehicle-maintenance data in an easy-to-understand visual way, so I can easily see the trend of the data. (**S**)

### 6.4 Business requirements 5: The system will provide a web-dashboard on which some important vehicle-maintenance data points will be shown to the vehicle-administrator.

1 As a User, I want any alerts coming from the web-dashboard to be easily visible (from across all the web-interfaces), so I can address these issues quickly. (**M**)

2 As a User, I want alerts coming in the web-dashboard to contain basic information about the problem at hand, so I can quickly see what the issue is. (**M**)

4 As a User, I want the system to allow me to quickly share the information in the alerts, so I can alert my superiors. (**S**)

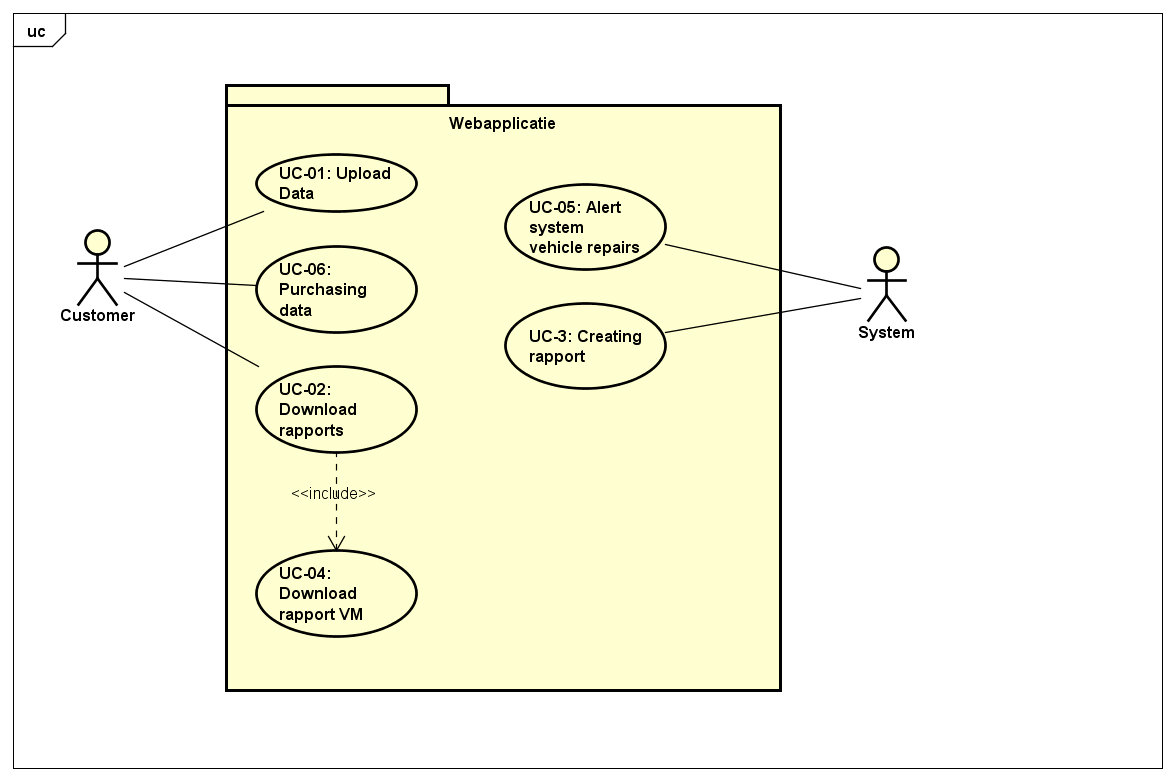
### 6.5 Business requirements 6: The system will provide a web-portal on which CityGis-meta data will be up for sale to 3rd parties.

1 As a User, I want to buy data from CityGis, so I can use it for my personal purpose. **(M)**

2 As a User, I want to be able to pay via PayPal, so I can easily transfer the money. **(M)**

3 As a User, I want to be able to select which data I want to buy, so I only have to buy the data that I need. **(S)**

## 7. Use-case diagram



## 8. Use case descriptions

|  |  |  |
| --- | --- | --- |
| **USE CASE ID** | **PRIMARY ACTOR** | **USE CASES** |
| **UC-01** | User | Upload data to the system |
| **UC-02** | User | Download rapports from the system |
| **UC-03** | System | Creating a rapport |
| **UC-04** | User | Download a rapport about vehicle maintenance |
| **UC-05** | System | Alert system for vehicle repairs |
| **UC-06** | User | Purchasing data from the system |

## Upload data to the system (UC-01)

**1 Brief Description**

This use case describes how a user can import City-gis data to the system for analyzing   
in this usecase userstories 1-5 of business requirement 1 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active network connection from the user to the system.   
   
The user has city-gis data provided in .csv-format.

**4 Basic Flow of Events**

1. The use case begins when the user selects the data on his/her PC.

2. The user uploads the selected data to the system/server

3. The system shows a progress bar while uploading the data

4. The system shows a confirmation when the upload has finished

**5 Alternative Flows**

Error

*If in step 3 of the basic flow the system gives an error, then*

The use case ends with a failure condition. And the user will know what has gone wrong.

User closes program

*If in step 1, 2 or 3 of the basic flow the user closes the program, then*

The use case ends. And the user won’t get noticed since the program has closed.   
The system will delete the file that was being uploaded.

**6 Key Scenarios**

Uploading Data to the server/system

**7 Post-conditions**

**Successful Completion**

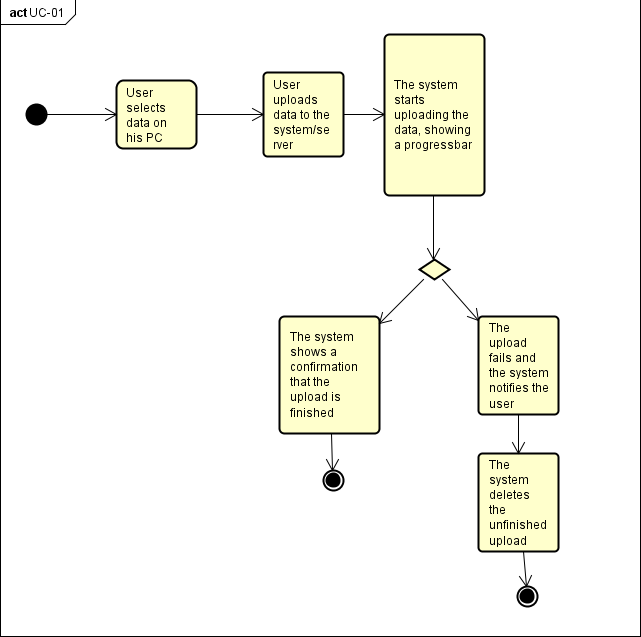
The user has received a confirmation that his/her data has been uploaded

**Failure Condition**

The data hasn’t been transferred to the system/server

**8 Special Requirements**

None



## Download rapports from the system (UC-02)

**1 Brief Description**

This use case describes how a user can download reports of city-gis data   
in this usecase userstories 1, 3-6 of business requirement 2 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active network connection from the user to the system.

**4 Basic Flow of Events**

1. The use case begins when the user requests a report of the data

2. The system returns a report

3. The browser of the user starts downloading the report

4. The system shows a progress bar of the downloading.

5. The user receives a confirmation when the download is completed.

**5 Alternative Flows**

*User closes program*

*If in step 3 of the basic flow the user closes the program, then* The use case ends with a failure condition. The download will be aborted.

*User requests multiple files*

*If in step 1 of the use case the user askes for multiple reports, then* the System will return multiple reports. Step 4 of the use case: There will be multiple progress bars to show the downloading files

**6 Key Scenarios**

Downloading reports from the server/system

**7 Post-conditions**

**Successful Completion**

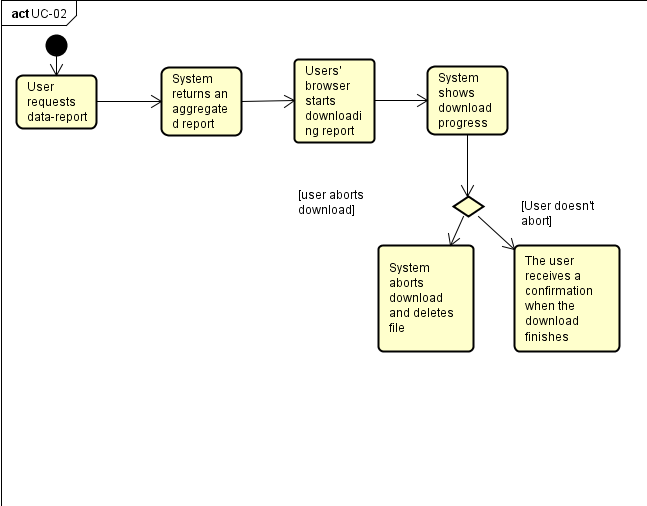
The user has received a confirmation that his/her report has been downloaded

**Failure Condition**

The report hasn’t been downloaded from the server

**8 Special Requirements**

None



## Creating a rapport (UC-03)

**1 Brief Description**

This use case describes how a the system processes data to make a report   
in this usecase userstory 2 of business requirement 2 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active server for the processing of the system

**4 Basic Flow of Events**

1. The use case begins when there is a trigger of a user that wants to download a file

2. The system searches for and selects an available server

3. The server starts processing to make a report

4. Once the processing is done the server gives a confirmation to the system

5. The system stores the file and makes it ready to download.

**5 Alternative Flows**

*All servers are busy*

*If in step 2 of the basic flow the system can’t find an available server, then* the use case ends with a failure condition. The system returns an error with error-information.

**6 Key Scenarios**

Processing reports on the server

**7 Post-conditions**

**Successful Completion**

The system stores the report

**Failure Condition**

The report hasn’t been made since there was an error

**8 Special Requirements**

None

## C:\Users\Kraaijeveld\Desktop\uc3.PNG

## Download a rapport about vehicle maintenance (UC-04)

**1 Brief Description**

This use case describes how a user can download report about vehicle maintenance   
in this usecase userstories 1-5 of business requirement 4 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active connection between the user and the server.

**4 Basic Flow of Events**

1. The use case begins when the user requests a report concerning vehicle-maintenance.

2. The system aggregates the requested data

3. The server starts processing to make a report

4. The system completes the processing

5. The system makes the report available for download for the user.

6. The browser of the user starts downloading the report

7. The system shows a progress bar to see the download progress

8. When the download is completed the user gets an confirmation

**5 Alternative Flows**

*The user quits the program*

*If in step 6 of the basic flow the user quits the program*, then the use case ends with a failure condition. The system stops the download and disconnects the connection to the user.

**6 Key Scenarios**

Downloading the report about vehicle maintenance

**7 Post-conditions**

**Successful Completion**

The user gets the report

**Failure Condition**

The report hasn’t been downloaded since there was an error

**8 Special Requirements**

None

## C:\Users\Kraaijeveld\Desktop\uc4.PNG

## Alert system for vehicle repairs (UC-05)

**1 Brief Description**

This use case describes how a vehicle office gets an alert when the car needs repair   
In this usecase userstory 1, 2 and 4 of business requirement 5 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active internet connection between the system and the user

**4 Basic Flow of Events**

1. The use case begins when the system creates a new alert in the vehicle maintenance dashboard

2. The user receives an alert icon in each web interface

3. The user can see basic information about the alert

4. The user forwards the alert to his superiors or co-workers

**5 Alternative Flows**

The user decides that the alert is not important enough to forward and deletes it.

**6 Key Scenarios**

Alerting the user

**7 Post-conditions**

**Successful Completion**

The system alerts the user

**Failure Condition**

None

**8 Special Requirements**

None

## C:\Users\Kraaijeveld\Desktop\uc5.PNG

## Purchasing data from the system (UC-06)

**1 Brief Description**

This use case describes how a user can buy city-gis meta-data   
in this usecase userstory 1- 3 of business requirement 6 are put together.

**2 Actors**

User

System

**3 Preconditions**

There is an active internet connection between the system and the user.

**4 Basic Flow of Events**

1. The use case begins when the user selects the meta-data he wants to buy

2. The user confirms his selection

3. The system provides the amount of money the user needs to pay.

4. The user pays with PayPal.

5. The system receives the purchase confirmation.

6. The system provides the data to the user for download.

7. The user downloads the data in his browser.

8. The system provides a progress bar while the data is downloading.

9. The user gets a confirmation when the download is completed.

**5 Alternative Flows**

*The user declines payment*

*If in step 4 of the basic flow the user quits the program, then*

The use case ends with a failure condition. The system shows an error message that the user didn’t pay and the system gives the option to retry the purchase.

*The user quits the program while downloading If in step 7 of the basic flow the user quits the program, then* The use case ends with a failure condition. The system stops the downloading and disconnects the connection to the user. But it keeps the payment information of the user, so he can later download again without going through the whole payment process again.

**6 Key Scenarios**

Selling data to the user

**7 Post-conditions**

**Successful Completion**

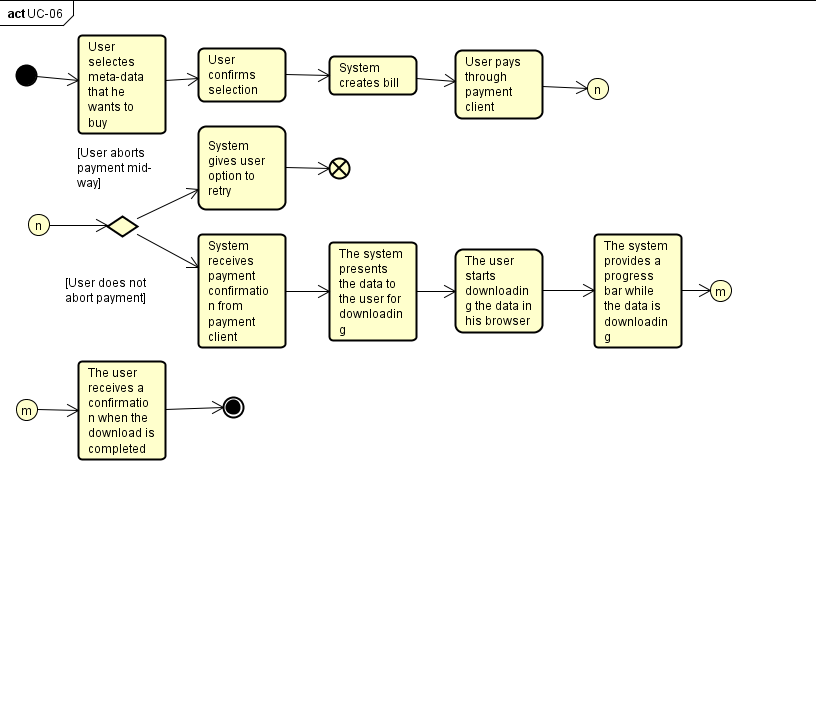
The Data is sold to the user, and the user receives the data as a download.

**Failure Condition**

The purchase is cancelled

**8 Special Requirements**

None



# 8. Extra: Project methodology

Since the beginning of year 1, we used Scrum as our software development methodology. Because Scrum worked nice for us, we want to use it again. But we decided to change the rules of Scrum a bit, because we don’t agree with all of them. So we removed some rules and edited some others. Technically, our methodology is not Scrum: It is Scrum based. Below we describe the differences of our methodology with Scrum.

**Removals:**

- Planning poker: Not only does official scrum planning poker take a lot of time, but it also does not make the development team any more productive, Because it is almost impossible for team members to actively log their time spent on each planned item; thusly, using planning poker makes little sense since you have no feedback on wether or not your prediction was correct.

- Retrospective: Just like planning poker we think the retrospective is based on a handy principle, but has been made far too time-consuming by the scrum methodology. We are all (relatively) mature student-developers and therefore we can communicate problems/feedback to eachother directly, instead of through 1.5 hours worth of sticky-note sticking.

- Burndown-chart: Our reasons for leaving the burndown chart out are the same as for the retrospective: We, being mature student developers can all ‘see’ when a sprint is going to slow, instead of acquiring this information from a graph.

- Always producing a shippable product: Sometimes you spent a lot of time on back-end work that is of great importance, but because you spent less time on the front-end, it will look like little work has been done; Even though everyone has spent a lot of time on important back-end fetaures. Therefore we would rather present our actual code/progress during Sprint Reviews and look at that with the PO, instead of constantly focusing on creating a shippable product.

**Changes:**

Our iterations will last 2 weeks. On Thursday during the project lesson we will have the sprint review. Here we will ask for some feedback and show our product/code.