**Customer Requirements Specification**

**(Lastenheft)**

(TINF19C, SWE I Praxisprojekt 2020/2021)

Project: AML NoSQL Database Management

Customer: Rentschler & Holder

Rotebühlplatz 41

70178 Stuttgart

Supplier: Team 5:

(Nils-Christopher Wiesenauer, Namid Marxen, Johannes Timter, Jonas Bihr, ~~Max Scheub~~)

Rotebühlplatz 41

70178 Stuttgart

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Author** | **Comment** |
| 0.1 | 21.10.2020 | Timter | Edit Goal |
| 0.2 | 22.10.2020 | Marxen | Edit Features |
| 0.3 | 22.10.2020 | Wiesenauer | Edit Product-enviroment |
| 0.4 | 24.10.2020 | Team | Working on everything at “Hackathon” |
|  |  |  |  |

#### ****Offene Punkte****

* Holder Fragen stellen und überprüfen lassen
* Was passiert wenn der User eine AML datei hochlädt, bzw. Eine bestehende AML Datei so verändert, dass sie keine AML Datei mehr ist? Ist das für uns ein Problem? Soll man dem User da feedback geben weil das wäre ganz schön schwer

CONTENTS

*1.* Goal 3

2. Product Environment 4

*3.* Product Usage 5

3.1. Business Processes 5

3.1.1. <BP.001>: File upload 5

3.1.2. <BP.002>: Manage saved files 6

3.1.3. <BP.003>: Search function 6

3.2. Use Cases 7

3.2.1. <UC.001> Upload AML files 8

3.2.2. <UC.002> Search for existing files 9

3.2.3. <UC.003> Download files 10

3.2.4. <UC.004> Edit files 11

3.2.5. <UC.005> Delete files 12

*3.3.* Features 13

3.3.1. /F01/Upload files 13

3.3.2. /F02/Conversion from XML to JSON 13

3.3.3. /F03/Download files 13

3.3.4. /F04/Edit files 13

3.3.5. /F05/Delete files 13

3.3.6. /F06/Search for files 13

3.3.7. /F07/List all files 13

*4.* Product Data 14

4.1. /LD10/ User-GUI 14

4.2. /LD20/ GUI-API 14

4.3. /LD/30/ API-Database 14

*5.* Other Product Characteristics 15

5.1. /NF10/Graphical User Interface 15

5.2. /NF20/Backend 15

5.3. /NF30/Database 15

5.4. /NF40/System Environment 15

6. References 16

# Goal

**A**utomation **M**arkup **L**anguage (AML) is an XML based data format to store and share engineering data about parts of production plants. These objects can consist of multiple other objects and be part of a larger assembly of objects. For example, an AML document can describe a screw, a claw, a robot or a complete manufacturing cell in different levels of detail.

It is the goal of this project to develop a web application that allows the user to perform the **CRUD** (Create, Read, Update, Delete) operations on AML files. The functions of the web application must be accessed through a graphical user interface. More precisely, the user should have the options to upload and download AML files and search for existing AML files with an ID-based search field.

Implementation: The frontend should be implemented with Angular.js. The Backend should be implemented with Express.js and Node.js. A local MongoDB instance should be used to store the AML documents.

// The mongoDB CAN be packaged as a Dockcontainer and can be configured and started through script commands. ????//

As AML is based on XML and MongoDB is only able to store JSON files, it is necessary to model and implement a conversion function for our AML documents from and to the JSON format.

The finished product should allow engineers to share and access their AML documents with others. As we are implementing the product as a web application, the product will accessible anywhere, from any web-capable end device. The process of finding existing documents and uploading your own documents should be fast and easy, even for non-engineers. Therefore, our target-group are not only engineers, but any person who whishes to inspect and understand AML documents.

Für die Verwaltung von AutomationML Dateien in einer NoSQL Datenbank soll eine CRUD

(Create,Read,Update,Delete)-Webanwendung entwickelt werden, die mithilfe des MEAN

(MongoDb, Express.js, Angular.js, Node.js)-Stack implementiert werden soll.

Da AML auf XML basiert, muss eine geeignete Möglichkeit der Konvertierung in das JSON-Format modelliert und implementiert werden.

Die GUI soll als Angular-Frontend [2] implementiert werden, das Backend basierend auf Express.js/Node.js [3]. Die MongoDb [4] soll lokal als Dockcontainer per Skriptbefehle vorkonfiguriert und gestartet werden.

Die Applikation dient zur (grafischen) Verwaltung dieser lokalen MongoDb Instanz. Um die CRUD Funktionalität zu gewährleisten sollen die Funktionen eines Dateien Up- und Downloads, und zunächst ein ID-basiertes Suchfeld bereitgestellt werden.

# Product Environment

1. CAEX (Computer Aided Engineering Exchange) to describe attributes of objects and their relations in a hierarchical structure. This is called a system topology. In this respect, CAEX forms the overarching integration framework of AutomationML. [1]
2. COLLADA to describe the geometry and 3D models of a objects
3. COLLADA also integrates motion planning. It describes the connections and relations of moveable objects, which is called Kinematics.
4. PLCopen XML describes the logic. Internal behaviour and states if objects, action-sequences and I/O connections are implemented via this format.

Angular is a TypeScript based front-end framework which is published as open source software. This framework has been around for almost 10 years and since then countless adaptations have been made. The three pillars of Angular are TypeScript, RxJS and Zone.js.

In simple terms, NodeJS is a JavaScript free and open source cross-platform for server-side programming that allows users to build network applications quickly. The runtime is intended for use outside of a browser context (i.e. running directly on a computer or server OS). As such, the environment omits browser-specific JavaScript APIs and adds support for more traditional OS APIs including HTTP and file system libraries.

ExpressJS is the most popular Node web framework and is the underlying library for several other popular Node web frameworks. It provides mechanisms to write handlers for requests with different HTTP verbs at different URL paths (routes), to integrate with “view” rendering engines in order to generate responses by inserting data into templates, to set common web application settings like the port to use for connection, and the location of template that are used for rendering the response and to add additional request processing “middleware” at any point within the request handling pipeline.

MongoDB is a document-oriented NoSQL database used for high volume data storage. Instead of using tables and rows as in the traditional relational databases, MongoDB makes use of collections and documents. Documents consist of key-value pairs which are the basic unit of data in MongoDB. Collections contain sets of documents and function which is the equivalent

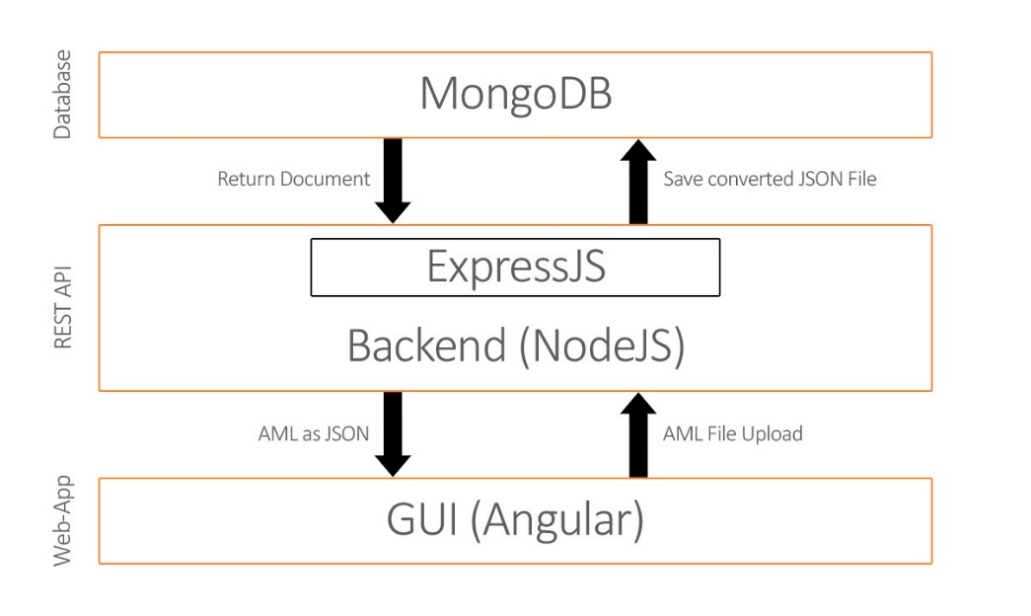


Figure 1 Product Environment

# Product Usage

The main purpose of the software will be to upload AutomationML files to a database. The uploaded files can be accessed from the user through a web-interface. In that interface the user is also able to edit, delete and download existing files. The user should also have the possibility to search for saved documents. This gives the user an easy way to upload AutomationML files into a database and conveniently handle them through a web interface.

The following features should be accessible in the GUI of a web application:

1. The user can upload AML files
2. The user can search for existing AML files through an ID-based search field
3. The user can edit existing AML documents
4. The user can delete existing AML documents

## Business Processes

This section shows the individual business processes necessary to support the system to be developed.

### <BP.001>: File upload

|  |  |
| --- | --- |
| Triggering Event: | User has an AutomationML file in XML format and wants to upload it to a database. |
| Result: | The system converts the XML file and stores the output JSON-file into the database, which can be accessed by the user. |
| Involved Roles: | User and AMLDatabase |

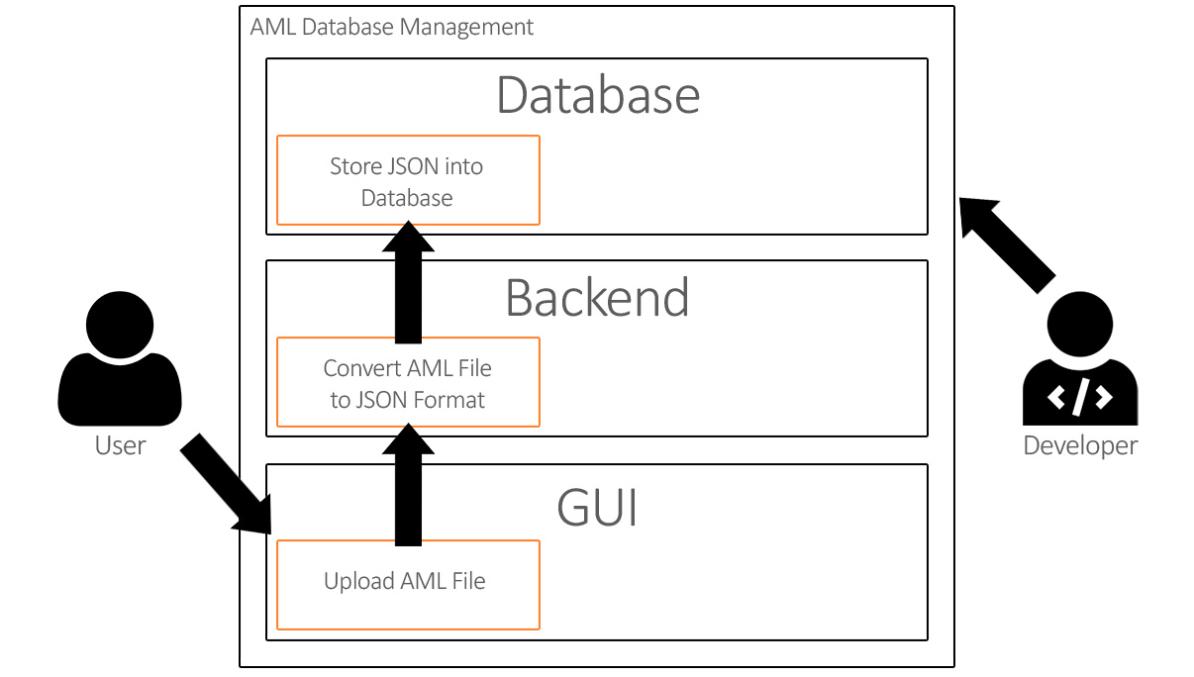


Figure 2 <BP.001> File upload

### <BP.002>: Manage saved files

|  |  |
| --- | --- |
| Triggering Event: | The user wants to read, edit or delete files in the database. |
| Result: | The system provides |
| Involved Roles: | User and AMLDatabase |

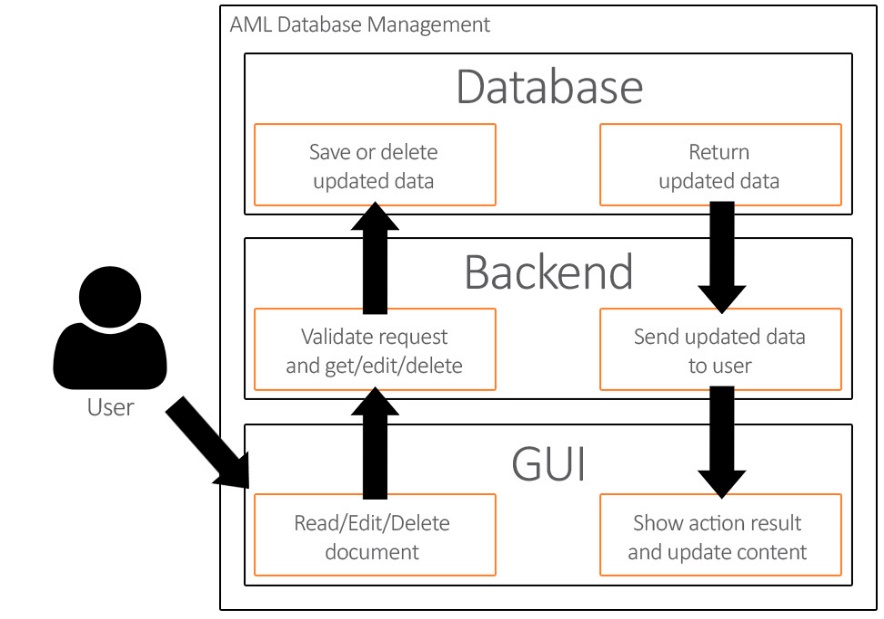


Figure 3 Manage saved files

### <BP.003>: Search function

|  |  |
| --- | --- |
| Triggering Event: | The user wants to search for existing files in the database. |
| Result: | The system gives the user the possibility to search for a document in the graphical user interface. |
| Involved Roles: | User and AML Database |

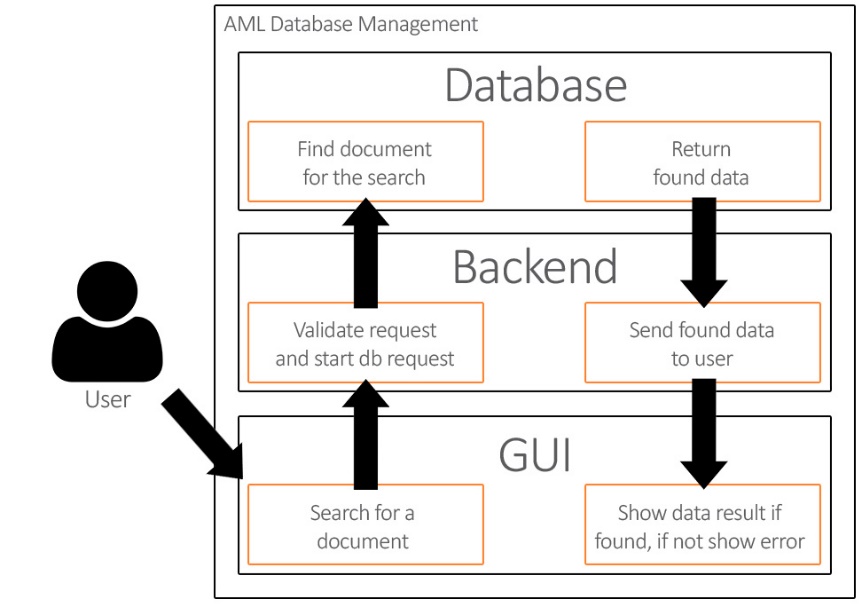


Figure 4 Use Case Overview Diagram

## Use Cases

This project will be implemented with a graphical user interface.

The user will have the opportunity to access the graphical user interface and upload AutomationML files in XML format, which will be saved into a database. He can then download, edit and delete the entries through the web-page. The user can also search for a saved file based on the id.

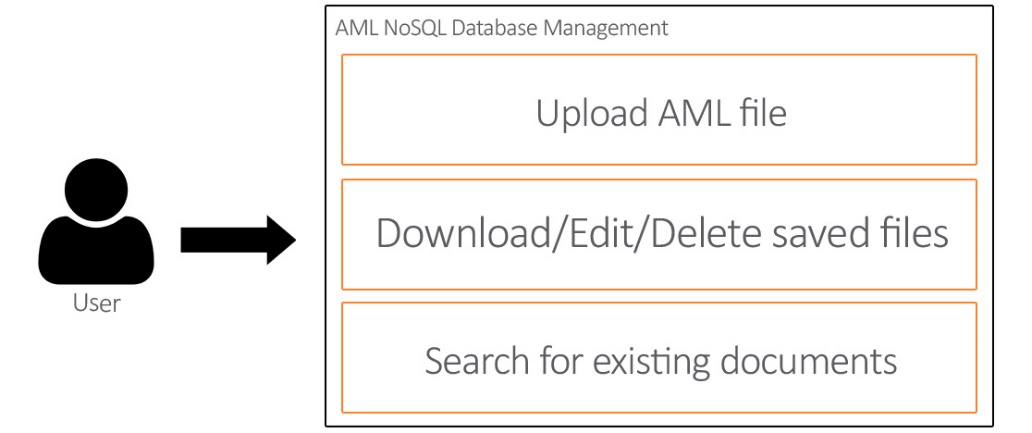


Figure 5 Use Case Overview Diagram

### <UC.001> Upload AML files

|  |  |
| --- | --- |
| **Related Business Process:** | Prozess-ID: <BP.001 >: File upload |
| **Use Cases Objective:** | User wants to upload an AML file using a web-interface |
| **System Boundary:** | The web interface is the system boundary |
| **Precondition:** | * The web interface server must be active * The file to be uploaded must be an AML file * ? The AML file must be without errors |
| **Postcondition on success:** | The page should not be closed while the user is uploading the file. |
| **Involved roles:** | User and AMLDatabase web interface |
| **Triggering Event:** | Opening the web interface in the browser and selecting the upload function and then selecting a local AML file. |

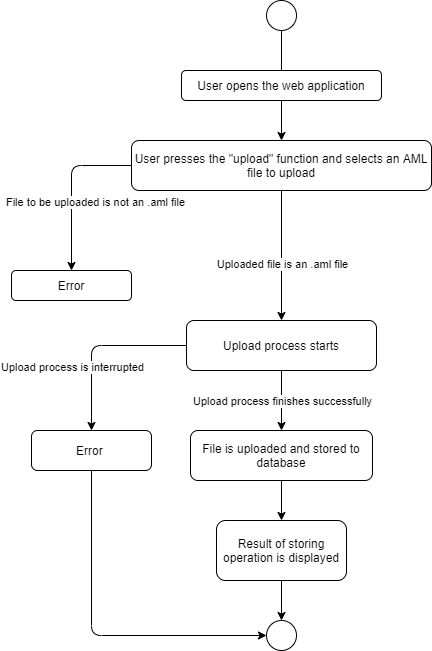


Figure 6 <UC.001> Upload AML files

### <UC.002> Search for existing files

|  |  |
| --- | --- |
| **Related Business Process:** | <BP.003>: Search function |
| **Use Cases Objective:** | User wants to search for an existing file in the web interface |
| **System Boundary:** | The web interface is the system boundary |
| **Precondition:** | * The web interface server must be active |
| **Postcondition on success:** | The page should not be closed during the search process. |
| **Involved roles:** | User and AMLDatabase web interface |
| **Triggering Event:** | Opening the web interface in the browser and selecting the search function and typing an id to be searched |

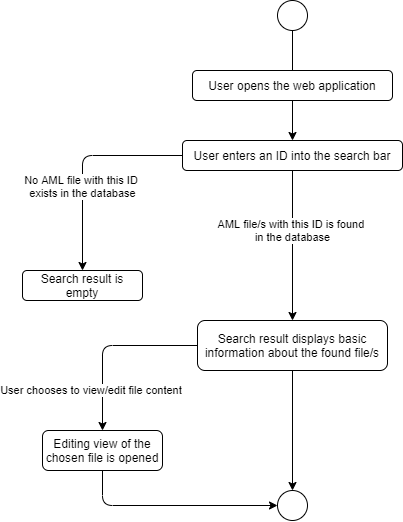


Figure 7 <UC.002> Search for existing files

### <UC.003> Download files

|  |  |
| --- | --- |
| **Related Business Process:** | <BP.002>: Manage saved files |
| **Use Cases Objective:** | User wants to download an existing file |
| **System Boundary:** | The web interface is the system boundary |
| **Precondition:** | * The web interface server must be active |
| **Postcondition on success:** | The user does not cancel the download |
| **Involved roles:** | User and AMLDatabase web interface |
| **Triggering Event:** | Opening the web interface in the browser and selecting a file and click on download. |

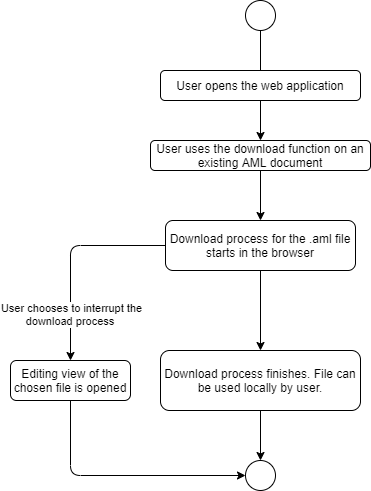


Figure 8 <UC.003> Download files

### <UC.004> Edit files

|  |  |
| --- | --- |
| **Related Business Process:** | <BP.002>: Manage saved files |
| **Use Cases Objective:** | User wants to Edit a saved file |
| **System Boundary:** | The web interface is the system boundary |
| **Precondition:** | * The web interface server must be active |
| **Postcondition on success:** | ? The edited file should be without errors |
| **Involved roles:** | User and AMLDatabase web interface |
| **Triggering Event:** | Opening the web interface in the browser and selecting a file and clicking on edit. |

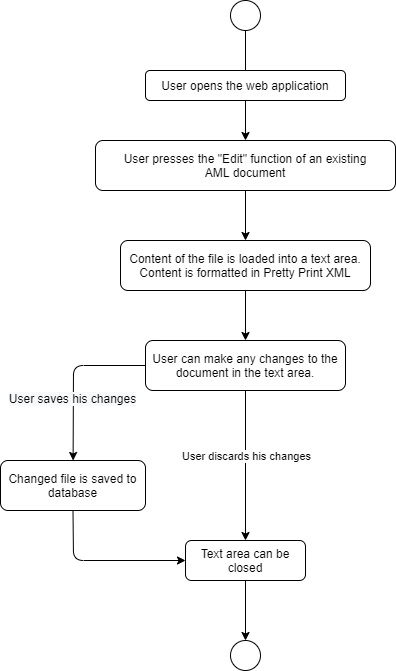


Figure 9 <UC.004> Edit files

### <UC.005> Delete files

|  |  |
| --- | --- |
| **Related Business Process:** | <BP.002>: Manage saved files |
| **Use Cases Objective:** | User wants to delete a saved file |
| **System Boundary:** | The web interface is the system boundary |
| **Precondition:** | * The web interface server must be active * ? The file is not being edited by another user |
| **Postcondition on success:** | The server does not close during the deletion process |
| **Involved roles:** | User and AMLDatabase web interface |
| **Triggering Event:** | Opening the web interface in the browser and selecting a file and clicking on delete |

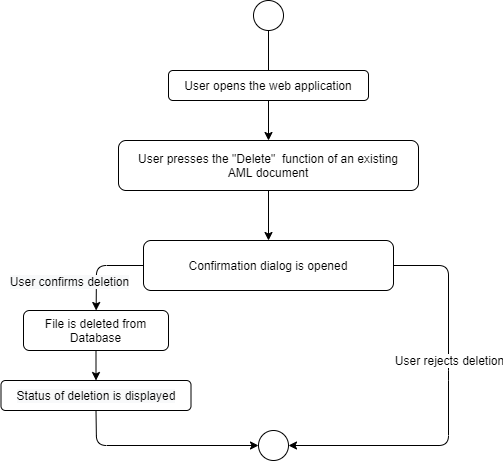


Figure 10 <UC.005> Delete files

## Features

### /F01/Upload files

The application shall be able to upload files selected locally from the user.

### /F02/Conversion from XML to JSON

The system should be able to convert an uploaded AML file to JSON and save it in the database.

### /F03/Download files

The application shall provide the user with the ability to download saved files from the database.

### /F04/Edit files

The application should provide the user with the ability to edit the documents in the database with the graphical user interface.

### /F05/Delete files

The application should provide the user with the ability to delete the documents in the database with the graphical user interface.

### /F06/Search for files

The application will be able to search for existing files based on the ID, saved in the database.

### /F07/List all files

The graphical user interface will list all files from the database with sorting and filtering functionality.

# Product Data

## /LD10/User-GUI

A User shall be able to upload AML files over an Angular based web interface. Through this web interface he shall then be able to use CRUD operations (Create, Read, Update, Delete) on the uploaded records.

## /LD20/GUI-API

The system shall have the functionality to convert AML documents from XMLto JSON format.

## /LD/30/API-Database

The system shall be able to save the JSON converted files to the database.

# Other Product Characteristics

This section describes the already known non-functional requirements for the product.

## /NF10/Graphical User Interface

The system shall support a graphical user interface, which is based on Angular.

## /NF20/Backend

The system shall support a communication between frontend and the database, based on Node.js and Express.js.

## /NF30/Database

The system should be able to save and search all files in a MongoDB database.

## /NF40/System Environment

The system should run under any operating system with a web browser installed.

# References

[1] http://angular.io

[2] http://www.mongodb.com

[3] http://www.automationml.org/o.red.c/home.html

[4] http://nodejs.org/en

[5] http://expressjs.com

[6] http://developer.mozilla.org/en-US/docs/Learn/Serverside/Express\_Nodejs/Introduction