MOD01: Server connection

TINF21C, Software Engineering I
Praxisproject 2022/23

Project

AAS-Webclient

Customer

Markus Rentschler, Christian Holder

Rotebühlplatz 41, 70178 Stuttgart

Supplier

Project Leader: Samara Dominik (inf21001@lehre.dhbw-stuttgart.de)

Product Manager: Martin Rittmann (inf21157@lehre.dhbw-stuttgart.de)

System Architect: Marcel Hintze (inf21056@lehre.dhbw-stuttgart.de)

Test-Manager: Anja Niedermeier (inf21097@lehre.dhbw-stuttgart.de)

Developer: Severin Helms (inf21047@lehre.dhbw-stuttgart.de)

Technical Documentation: Tom Engelmann (inf21010@lehre.dhbw-stuttgart.de)

Version	*Date*	*Author*	*Comment*
0.1	09.05.2023	Samara Dominik	Created, set up basic structure
0.2	09.05.2023	Marcel Hintze	Added Implementation
0.3	10.05.2023	Samara Dominik	Added Scope, Glossary
0.4	10.05.2023	Samara Dominik	Added Design
0.5	10.05.2023	Martin Rittmann	Added Module Requirements, Analysis
0.6	10.05.2023	Tom Engelmann	Added Module Tests

CONTENTS

1 Scope

Our Webclient is a surface to display Asset data coming from an AASX servers. As such, it is essential to provide the functionality for adding and building a connection to servers, as well as loading their respective data.

2 Glossary

- AAS: Asset Administration Shell
- AASX: file format to store an asset

3 Module Requirements

3.1 User View

The user can manually add a AASX server or switch between them. The user enters a url of the server in the search bar. The process of establishing a server connection runs in the background, so the user won't see much of that. There is a circular loading animation and as soon, as data is available, the user can see the assets. For usability the assets are shown before the more detailed information are fully loaded from the server, so the user can see the assets faster.

3.2 Requirements

/REQ10/ Import server

The webpage should be able to import a server by its URL and display its content.

/REQ20/ Server validation

The system shall be able to detect false server-URLs when adding a new server and throw an error to the user. and throw an error to the user.

/NF10/ Intuitive GUI

The webpage shall display a graphical user interface (GUI) to the user. This GUI must display every function provided to the user in a simple and intuitive way. It will be the only way to interact with the application.

/NF30/ Efficiency

The webpage shall add servers and apply filters in the fastest way possible as well as the user being able to find desired results with the lowest possible amount of steps.

3.3 Module Context

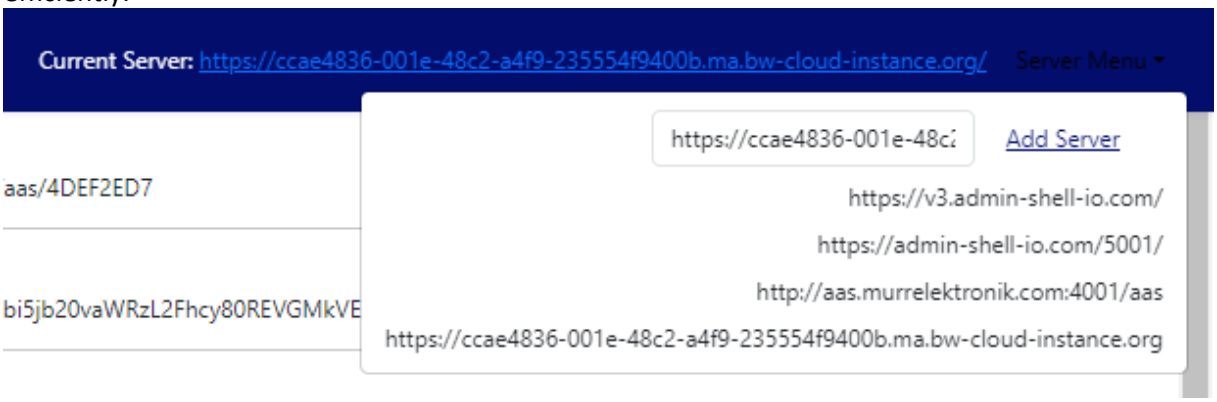
After entering a server address, this modules task is to establish a connection to the given AASX server and return the information to MOD02. Without this module, no information can be displayed on the webpage

4 Analysis

The Server connection module has the central task to send a request to the entered server url and return the data in json format so it can be displayed in the GUI. Therefore this module needs the functions to connect to the given server and then load those information if the server exists. Also, if the user switches servers, the old data is removed and the process of connecting and returning data repeats.

5 Design

To speed up the process of connecting to a server within the Webclient, we have implemented a dedicated menu at the top right corner of the webpage. This menu serves as a centralized location for users to search for and select the desired server with which they wish to establish a connection. This ensures that users can manage their server connections more efficiently.



Additional details on our Usability Concept can be found on the Usability-Concept-page.

6 Implementation

The JavaScript fetch API is used to do the requests to the AASX server when the user adds a server address. All the available assets that a server provides are requestsed from <serverAddress>/shells. This endpoint returns a list of strings like the following:


```
{
  "idShort":"Norgren_ISOLine_PRA_802032_M_100",
  "id":"https://www.norgren.com/ids/aas/4DEF2ED8",
  "idEncoded":"aHR0cHM6Ly93d3cubm9yZ3Jlbi5jb20vaWRzL2Fhcy80REVGMkVEOA==",
  "apiVersion":3,
  "submodels":
  [
```

```
    "www.example.com/ids/sm/7402_0231_5022_9880",
    "submodelIDs"
  ]
}
```

A further request is then sent to the server for each submodel of an asset. These submodels are requestsd from <serverAddress>/shells/assetID/submodels/submodelID/submodel. This request returns all data to the requested submodel. A JSON of every asset and every submodel is locally stored in the browser after all requests are finished.

7 Module Tests

The functionalities have been tested in the test suite TS-Server with the test cases TC-Server-Import-001, TC-Server-Switch-002, TC-Server-Load-003.

The import of an AAS-server has been tested in TC-Server-Import-001. It passed the test and produced the expected results.

Switching out a currently used server with another one has been tested in TC-Server-Switch-002. It passed the test and produced the expected results.

The correct display of all assets which are contained in the currently used server has been tested in TC-Server-Load-003. It passed the test and produced the expected results.