MOD02: Expert-User Mode

TINF22F, Software Engineering I
Praxisproject 2023/24

Project

AAS-Webclient

Customer

Markus Rentschler, Ivan Bogicevic

Lerchenstraße 1, 70178 Stuttgart

Supplier

Armin Taktar (Project Leader)

Lara Lorke (System Architect)

David Bauer (Product Manager)

Ümmühan Ay (Documentation)

Rafael Sancho Pernas (Developer)

Kyle Zieher (Test Manager)

Author

Rafael Sancho Pernas 06.05.2024

CONTENTS

1 Scope

According to User Case 3.1 and User Case 3.2, the AAS webclient should provide a mode with limited information for users without expertise, and a mode with detailed view for users with extensive expertise.

2 Glossary

- AAS: Asset Administration Shell

3 Module Requirements

3.1 User View

The user has the ability to switch between modes using a switch. In the User Mode, they have a view with limited information. By clicking the switch, the mode changes to Expert Mode. When the user clicks on an asset in Expert Mode, they receive a detailed view including ID, globalAssetId, and submodels.

3.2 Requirements

FR02: User can switch between “expert” an “default” mode.

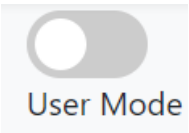
4 Analysis

The mode switch between the 2 modes must occur using a variable in the background. When the switch is pressed, the variable must be changed. Depending on which mode is selected, the React component for the assets must be filled accordingly with either a lot or a little information.

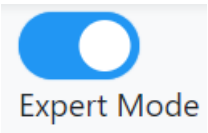
5 Design

The design of the mode switcher is a toggle button. By clicking on the toggle the toggle moves to the other side and the color switches. The text below the toggle Button shows in which mode the user is.

Design of the toggle button in user mode:



Design of the toggle button in expert mode:



6 Implementation

As in the server connection, the function getFullShellData needs to be modified. But first, the mode switch must be implemented through the toggle button. This is achieved by storing the mode as a variable named "mode" in the session storage. Pressing the toggle button will then switch the mode:

```
toggleMode() {
    const newMode = this.state.mode === 1 ? 2 : 1;
    this.setState({ mode: newMode });
    sessionStorage.setItem('mode', newMode === 1 ? 'user' : 'expert');
```

This switch of the mode is processed in the filter.js file.

In the getFullShellData function of the backend.js there are now two returns implemented for the assets. An if-statement checks if the variable “mode” in the sessionstorage is “user” or “expert”. For the case it is “expert” the function returns:

```
return {
    idShort: element.idShort,
    id: id,
    idEncoded: btoa(id),
    apiVersion: apiVersion,
    globalAssetId: element.assetInformation.globalAssetId,
    Assetsubmodels: submodelIds
};
```

If the mode is “user” the function returns:

```
return {
    idShort: element.idShort,
    id: id
};
```

The React component for the assets takes the returned JSON and fills it with the information.

7 Module Tests

The testing of this module is carried out through usability tests. The user mode is set to default, so the output is checked by clicking on the asset buttons. The expected outcome is less detailed information in the asset body.

The toggle button is tested by verifying whether a module change occurs after toggling. The expected outcome is that upon pressing the asset button again, detailed output with submodels and other information appears.

The expected outcome upon clicking again is a return to the default mode and the corresponding output.