Interface Health Monitor

Introduction

CSD, AEB, FXTA, Avaloq Adapter, Sparta Adapter and CGIX are the interfaces supported by Avaloq Interface team. Interface Health Monitor is a GUI Interface to monitor various servers under DEV/SIT/UAT environments in Avaloq interfaces to manage the delivery support. This GUI has prestored valid message requests which can confirm a success or failure response to show whether the server is up or not. This application will cater to the needs of the Testing team, the Config team and the Interface team.

Problem Statement

The steps followed currently to check the connectivity of Avaloq Interfaces takes up a lot of time. The teams involved are the Testing team, the Config team and the Interface team. The testing team has to manually test all the URLs to check if the servers are up and running. If they are not able to connect to a particular URL, that URL link is sent to the Config team. The Config team, who has access to databases, checks on the specific database/server for which URL it's pointing/connected. If base PARAM's okay, yet there are issues trying to connect to the server, the Config team reaches out to the Interface team for further checks on the required server.

The above mentioned process is not time efficient and there is a lot of repetitive work involved. The problem statement here is to design an interface which checks and updates the status of servers eliminating the need to hit individual urls manually. The application directly shows the status of various environments within Avaloq Interfaces and hence more time can be spent on resolving the issues instead of checking servers one at a time.

High Level Design and Approach

- Backend will ping the servers or environment to check whether it's up or running.
- If the response returned back is 200, it indicates that the server is up, otherwise it's down.
- This status is returned in an API.
- Then the Frontend script will fetch the data.
- And finally the result will be printed on the screen.

This approach can help the Testing Team to save time and energy by displaying the status of all the environments under that application.
It also helps the configuration team to quickly view what's the issue and accordingly can report to the interface team. Otherwise the interface team can itself view the issue and work upon it.

Technical Details

TechStack :

- Angular
- Apache Camel
- Spring Boot
- Java

Frontend Technology

The frontend technology used for building this application is Angular which is a component-based framework for building scalable web applications. It has a collection of well-integrated libraries that cover a wide variety of features, including routing, forms management, client-server communication, and more. Angular is a platform and framework for building single-page client applications using HTML and TypeScript.

Backend Technology

The backend technology used for building the application is Apache Camel and Java Springboot. Apache Camel is an open source framework for message-oriented middleware with a rule-based routing and mediation engine that provides a Java object-based implementation of the Enterprise Integration Patterns using an application programming interface (or declarative Java domain-specific language) to configure routing and mediation rules. Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications.

Working of the Application

The home page has a Dashboard with cards for CSD, AEB, FXTA, Avaloq Adapter, Sparta Adapter and CGIX Interfaces. The cards have information regarding that particular Interface. A ‘More Details’ button has been added to dashboard cards to get more details and help navigate to the respective interface page.

Navbar can also be used for navigation in Desktop view. The application has been made responsive. Sidebar is used for navigation in Mobile view.

The interface pages have information about the status of the DEV/SIT/UAT environments.
The status is shown using green ticks or red crosses. A server which is UP and running is represented by a green tick. A server which is down is represented with a red cross. Additionally, a message is displayed if required under the status.

All possible errors are handled by this application. In case the server is down, an appropriate message will be printed so that the user can understand the condition of the server. If there are

no servers present in the API then error information will be sent by the API to emphasize the problem. If the backend is up then the backend response will send an error message, if any.
If the backend is down then the frontend will manage the response that needs to be displayed on the screen.
If there is no information inside a particular application/interface or have details of another application/interface then “No data available” message will be printed for that particular interface.

The backend has a .JSON file containing all the possible environments, their servers and their corresponding URLs. The request from frontend is received, and the parameter in the URL serves as the backbone for the working. The {id} from frontend is passed on to a function, to parse the .JSON file for corresponding servers and their URLs, ping those URLs for status on their working, convert the response into an adequate JSON file, and return the JSON file back to frontend for the UI to display adequate data.

Logistics

#code repository and configuration

#Hosting and deployment (with link for main app and github (source code))

#Maintenance

Adding and changing urls

#known Issues

Time needed to complete

Duration	Work Done
13/6/2022 - 17/6/2022	Understanding project requirements and designing UI
18/6/2022 - 25/6/2022	Exploring the required technologies
26/6/2022 - 01/7/2022	Basic navigation and routing
02/7/2022 - 16/7/2022	Development of UI components, Fetching data from API and Backend work

17/7/2022 - 20/7/2022	Design and development of Dashboard, Error Handling and overall improvements
21/7/2022 - 23/7/2022	Documentation

People involved

The following people have been involved in this project:

Gaurav Gupta
Dev Manager

Savita Sharma
Avaloq Project Manager

Abhishek Joshi
Avaloq Configuration Manager

Sharath Srinivasa
Developer

Kausik Sarkar
Developer

Future Scope