Carbon Footprint Visualization

Software Architecture Document

Revision History

Version Number	Author	Date	Changes	Reviewer
1.0	Sai Varun Varanasi	25/04/2021	Initial Draft	Martin Dürsch
1.0.1	Sai Varun Varanasi	27/04/2021	Updated diagrams in 3.1 and 3.2 Updated Version numbers in 4.1	Ehsan Olyaee

Contents

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the Carbon Footprint Visualization tool to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This Software Architecture Description document provides the overview of the Carbon Footprint Visualization tool. The Carbon Footprint Visualization tool is being developed by students of Friedrich–Alexander University Erlangen–Nürnberg for Siemens Energy AG, Erlangen.

1.3 Definitions,Acronyms and Abbreviations

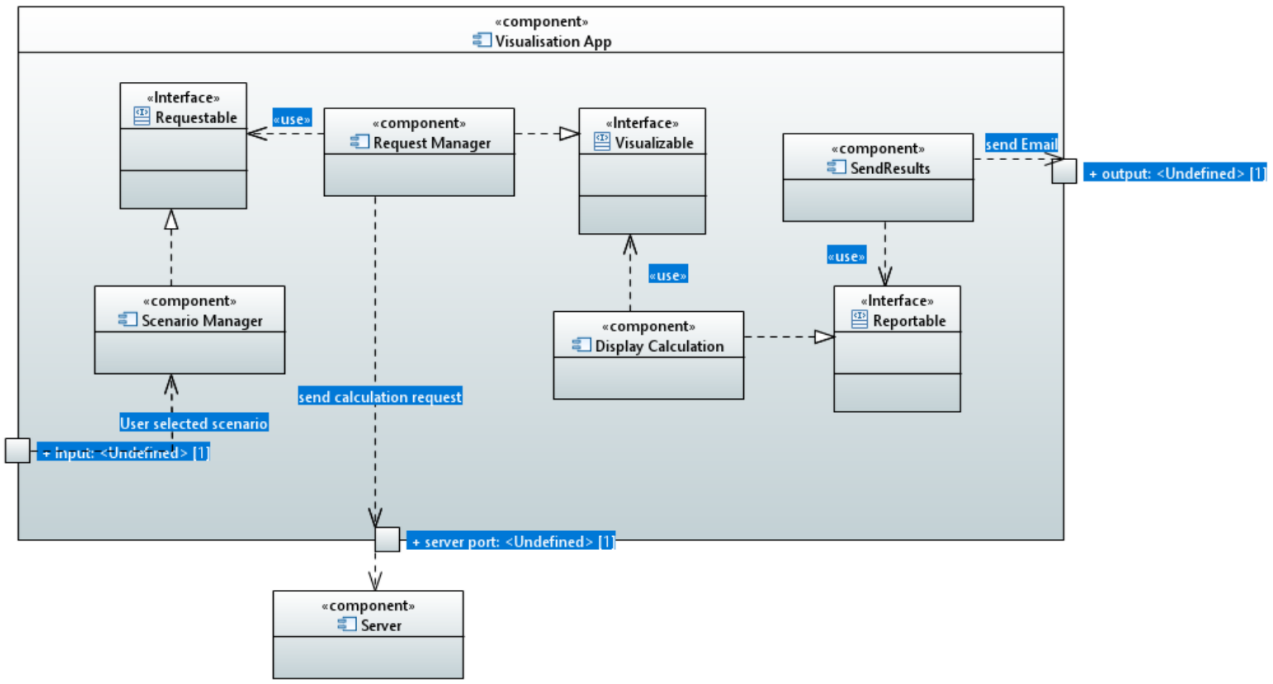
SimaPro - SimaPro application programming interface(API)
CSS - Cascading Style Sheets
HTML - HyperText Markup Language
HTTP - Hypertext Transfer Protocol

2.Project Overview

The carbon footprint visualization tool is used to visualize the carbon footprint of machinery / production environments (scenario) under different circumstances. This web application allows users to select different predefined scenarios and visualize the results. The results can then be saved, compared and shared.

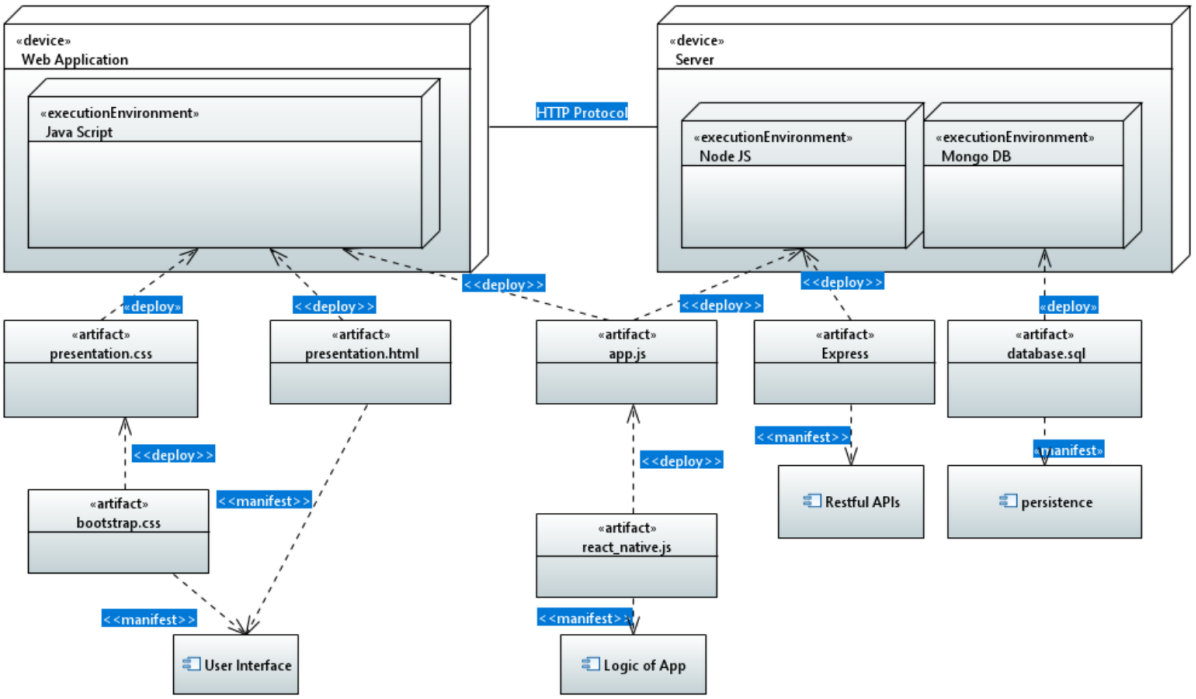
3.Architecture Overview

3.1 Runtime Components



The main component of this diagram is the “Visualization App”, which consists of some subcomponents. This main component receives as an input, a selected scenario from the user, which the component “Scenario Manager” has provided. As the user submits the scenario, the “Requestable” interface will be called in the component “Request Manger”. This component will send the scenario to the server. Once the server responds to the request, the “visualizable” interface will be called. “Display Calculation” component will implement this interface and show the result to the user. If the user wants to export the visualization results, the “Reportable” interface will be called and the “SendResult” component will send the visualization report as an email to the output port.

3.2 Code Components



This code component diagram consists of two main nodes, which are the “web application” and the “server”. The web application will interact with the server by HTTP protocol. The runtime environment of the web application is “Javascript”. CSS and HTML files will be deployed on web application to create the “user Interface”. “Bootstrap” as a framework will be also deployed on CSS. “Javascript” files will connect the web and server together and also define the logic of an app using a framework called “react native”. Node is a runtime environment for executing javascript code outside of a browser. With node we can build a web server that responds to the http requests. Express is the framework for building restful APIs. MongoDB is a database management system for storing the data.

4.Technology Overview

4.1Technology Stack

Component	**Technology/ Tools**	**Version**
Version Control(SVC)	Github	-
UI Design	Figma	-
Front end	JavaScript	ES2015
Front end Framework	React React native web HTML CSS Bootstrap	 16.13.1 0.13.12 5.0.0-alpha.8
Back end	Typescript	4.0
Back end Framework	Express Js Node Js	5.0.0-alpha.8 14.16.1
API	SimaPro	-
Database	MongoDB	4.2

4.2 Technology Summary

The UI/UX design for the project is being done in Figma. This enables the developers to come up with different design prototypes and finalize on the most attractive design before starting to work on the front end part of the website.

For the front end development React native framework is chosen as it enables the simultaneous creation of the web application and the mobile version at the same time.

Express Js is chosen for developing the back end components. This enables easy interaction between the front end and the back end components easily.