Høgskulen på Vestlandet

Fakultet for ingeniør- og naturvitskap

Institutt for datateknologi, elektroteknologi og realfag

Visualizing Smart Charging of Electrical Vehicles for Support Personnel

Vision document

Version 1.4

This document is based on Visjonsdokument from NTNU. Revision, customisations and adaptations to use at IDER, DATA-INF done by Carsten Gunnar Helgesen, Svein-Ivar Lillehaug and Per Christian Engdal. The document is also available in Norwegian.

REVISION HISTORY

Date	Version	Description	Author
26/JAN/22	1.0	Summary of product and problem	Kristin Standal
08/FEB/22	1.1	User's environment, requirements	Mads Henrik Sørbø
10/FEB/22	1.2	Functional and non-functional requirements	Kristin Standal
28/FEB/22	1.3	Resolving comments from supervisor	Mads Henrik Sørbø Roger Karlsen
20/MAY/22	1.4	Finalizing the document for hand-in	Kristin Standal

TABLE OF CONTENTS

1 INTRODUCTION 1

2 SUMMARY OF PROBLEM AND PRODUCT 2

2.1 PROBLEM SUMMARY 2

2.2 PRODUCT SUMMARY 2

3 DESCRIPTION OF STAKEHOLDERS AND USERS 3

3.1 SUMMARY OF STAKEHOLDERS 3

3.2 SUMMARY OF USERS 3

3.3 USERS' ENVIRONMENT 3

3.4 SUMMARY OF USER'S REQUIREMENTS 4

3.5 ALTERNATIVES TO OUR PRODUCT 5

4 PRODUCT OVERVIEW 6

4.1 THE PRODUCTS' ROLE IN THE USERS' ENVIRONMENT 6

4.2 PREREQUISITES AND DEPENDENCIES 7

5 PRODUCT FEATURES / FUNCTIONAL REQUIREMENTS 8

6 NON-FUNCTIONAL REQUIREMENTS 9

7 REFERENCES 10

1 INTRODUCTION

How can the different events and data streams involved in charging an electrical vehicle be visualized online, in a way that makes it easy for a non-technical support person to help customers.

Currently Tibber customer support needs to fetch, manage, and interpret data manually in various databases and systems. The types of data can be timeseries or state variables, which are fetched from the customer's electrical vehicle or their smart charger. Tibber wishes to ease this process for the support department and wishes to aggregate the data from the various logs and present it within a single view within their in-house support application named *Varys*.

2 SUMMARY OF PROBLEM AND PRODUCT

2.1 Problem summary

The problem with	The internal tool Varys does not have a single view for all the information needed for customer support to aid customers with smart charging problems and/or troubleshooting.
affects	The customer support team at Tibber and the customers that contact customer support.
as a result of this	Customer support spends unnecessary time gathering the information within Varys. Sometimes they might not find the information they need because it is hard to find.
a successful solution will	The customer support team will be more efficient and manage to help the customers in an overall better way. The new solution will make their jobs easier.

2.2 Product summary

For	Tibber
who	Needs a simple overview of the collected data
the product named	Smart Charging
which	Makes the customer support team more efficient.
Unlike	Today's system, which is complicated and confusing for a non-technical person.
our product has	A simple overview with all the necessary details to help customer support help customers with their smart charging questions/troubles.

3 DESCRIPTION OF STAKEHOLDERS AND USERS

3.1 Summary of stakeholders

Name	Description	Role during development
Tibber Customer Support	The customer support team at Tibber is the group that will reap the rewards of this project. It will make their job easier/more efficient.	Our project will be in "beta" during development and the customer support team will be able to test it while in development. Their feedback will help form the solution, as they have knowledge and experience about what would be a good solution.
Tibber		Helping us develop the solution: technical support, experience, and knowledge about the pre-existing system.

3.2 Summary of users

Name	Description	Role during development	Represented by
Tibber Customer Support	Currently customer support needs to fetch different logs and interpret results manually. They have the knowledge of common customer questions.	Assist by informing the students what the common questions are when customer calls with questions regarding smart charging.	Kai Marius, Thomas, Natalia

3.3 Users' environment

The new system is to be created in an existing application, *Varys*. Therefore, it needs to match with the existing design and layout, both visually and functionally. We do not have 100% freedom in designing this system the way we want but need to follow the limitations of the existing system. The new system is not really dependent on any hardware, as *Varys* is a web application and runs without problems in modern web browsers.

3.4 Summary of user's requirements

These requirements are obtained from the smart-charging team in Tibber. All of the requirements affect *Varys* and currently there is no existing solution for any of these requirements.

Requirement	Priority	Suggested new solution
SC 1: Overview - Smart Charging	1	New page in Varys containing the smart charging outline
SC 2: Show configuration	1	For each device a customer has related to smart charging, list the current configuration(s) of the device. Fetching information must be fast, support should not have to wait for critical information.
SC 3: Configuration checks	1	Support personnel shall get a good overview if the configuration has any known/common configuration error. If so, it shall be clearly shown.
SC 4: List scheduler events	1	Support personnel shall get a good overview of all charging schedules that have been calculated on all devices for this home.
SC 5: Visualize charging schedule	1	It shall be possible for support personnel to get a visual representation of the input and output from the scheduler algorithms (pyML execute and pyML response).
SC 6: Filter scheduler event	1	Support personnel shall be able to filter which scheduler event to see.
SC 7: Show price graph	2	Support personnel shall be able to see the price overlayed the smart charging schedule.
SC 8: Changeable values	3	Support personnel shall (maybe) be able to change values
SC 9: Load balancing	2	Support personnel shall be able to see when load balancing kicks in.
SC 10: Calculator	3	Support personnel shall be able to convert A to kW for 1 and 3 phases.

SC 11: Add visualization of logs	2	Support personnel shall be able to view the logs visualized with charts, instead of numbers/text.
SC 12: Statistics	3	We want to know how often functions are being used
SC 13: Device view	3	We have specific device views for some devices. Make the device clickable.

3.5 Alternatives to our product

As the product to be developed is unique to Tibber there are not many alternatives. Alternative products are related to visualizing charts but there is not any specific product that can be purchased and that would work “out of the box” with Varys. This means the alternative products are npm-packages for visualizing data. For example:

- Chart.js
- Highchart
- D3.js
- Recharge

4 PRODUCT OVERVIEW

4.1 The products' role in the users' environment

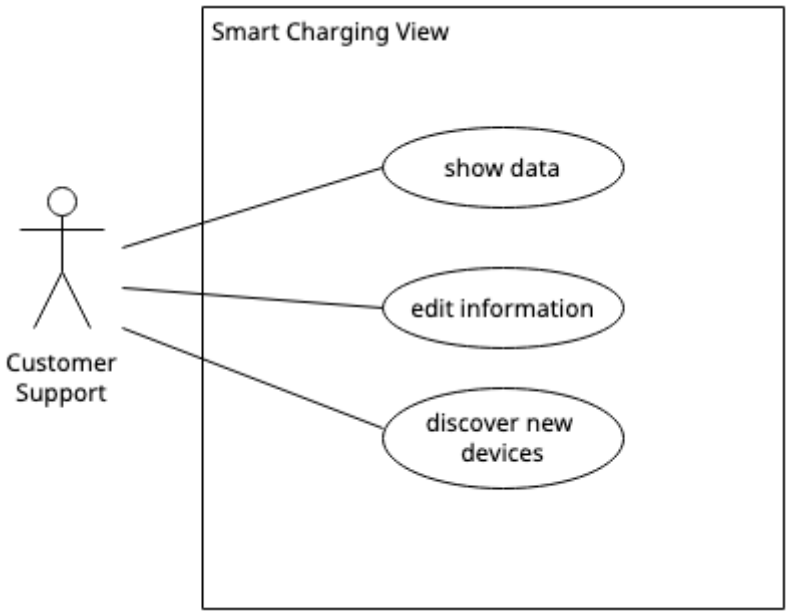

Figure 4.1 Use case diagram

The support department is equipped with personal laptop computers which are issued by Tibber. They dock their laptops when at the office such that they may have multiple monitors to work with. Varys is a browser application, so it has to support the various browsers and resolutions of customer support's monitors, which varies from 1920x1080 to 3456x2234. In some cases, support may also use their phones to access Varys, so the product must adhere to responsive web design.

Support also has access to a test environment where features that are not yet pushed to production can be tested. The data on this test environment consists of dummy data that has been manually created and can be manipulated without worrying about making changes to an actual customer.

The Smart Charging view consists of three simple use cases:

Show data	This happens when the actor clicks on the 'smart-charging'-button on the customer page. It fetches all the necessary data and displays it for the actor.
Edit information	Some of the information is editable, for example, names of devices.
Discover new devices	A button to discover new devices that are connected but not yet displayed in the view.

4.2 Prerequisites and dependencies

Varys is built with the Vue framework which is a progressive framework for building user interfaces – it is implemented as additional markup to HTML.

Varys consists of both frontend (Vue) and backend which mostly consists of authentication, authorization, and endpoints. The endpoints fetch data from a series of microservices, which together with Varys, are hosted in AWS.

Major changes in Varys could break the project, but it is unlikely that this will happen and if it happens, it is simply fixed with a patch or a rollback to an earlier version. Tibber has no plans to move away from Varys.

Varys depends on other microservices to fetch data (web-APIs). It's likely that some of these needs to change to meet the requirements. Additionally, a smart-charging centered microservice service needs to be deployed for maintainability/scalability reasons.

Some of these microservices are documented to some extent (GitHub/Notion), but mostly reduced to the minimal requirements to run them locally for testing and development. However, thanks to good coding practices they are relatively easy to read and understand.

5 PRODUCT FEATURES / FUNCTIONAL REQUIREMENTS

1. The system must show all the cars, chargers and load balancing devices for a customer, and specific parameters belong to these vehicles/devices. If a customer has several homes (addresses), devices and cars must be separated by homes in different tabs.
2. The system must show alerts if there are any errors in the data extracted from the state.
3. The system must show highlights for certain configuration checks (if a setting is on/off, etc.)
4. The system must have a tab for scheduled events and a log.
    a. The support personnel must have an overview of all charging schedules that has been calculated on all devices for this customer/home.
    b. All *execute pyML* and *pyML response* must be listed.
5. The system must visualize the charging.
    a. It should be possible for support personnel to get a visual representation of the input and output from the scheduler algorithms.
    b. Requirements:
        i. Start from when the schedule was calculated and show the next 24 hours.
        ii. Show which hours the algorithm decided to change
        iii. Show the charge input
6. The system must filter scheduler events.
    a. Support personnel must be able to filter which scheduler events to see.
    b. Requirements:
        i. Must be able to filter the list where the event pairs are shown.
        ii. Must be able to filter on time and device.

6 NON-FUNCTIONAL REQUIREMENTS

Scalability

The code must be written in such a way that it is scalable. It should not fetch data that is not needed and the code itself must be adaptable to new requirements.

Efficiency

The code must be written so it is efficient. It should fetch data in parallel where possible and it should reuse data that is unlikely to be changed during a session by caching. The CPU and memory constraints are initially set relatively conservative, and they can be lifted if needed later in development.

Flexibility / modifiability

The code must be written in such a way that it is reusable, e.g., flexible components that accept various parameters. If this is done correctly, the code itself becomes more manageable as it is easier to make changes later. There would be fewer places to make changes and it is easier to scale the software.

Testability

Testability is usually consistent with flexibility. It should be possible to create test data and test the different parts of code. When errors or incorrect input occur from the user, it should not have negative consequences, but prevent the execution of the code.

Usability

The user interface must be user-friendly and intuitive. When a customer support representative makes a mistake, e.g., invalid input in fields, they must be informed in a way that makes it obvious where the error has occurred. This can be done by marking the input with colors or descriptive error messages.

7 REFERENCES

Function requirements: https://en.wikipedia.org/wiki/Functional_requirement

Non-functional requirements: https://en.wikipedia.org/wiki/Non-functional_requirement

What is Vue.js? https://v2.vuejs.org/v2/guide/?redirect=true