Design Document

Software Design

COMP.SE.110

Version history

Version	Date	Author	Description
0.1	10.02.2022	Entire team	First draft

Contents

1	Introduction	3
	1.1 Purpose of the Document and Project	3
	1.2 Product and Environment	3
	1.3 Team Members	3
	1.4 Related Organisation	3
2	High Level Description	4
3	Boundaries and Interfaces	7
	3.1 Information Flow	7
	3.2 Components and Responsibilities	8
4	Reasonings	9

1. Introduction

1.1 Purpose of the Document and Project.

The purpose of the document is to explain how the design has happened during the project planning process and in further implementation. The goal of the project is to build a stand-alone application for monitoring real-time data on greenhouse gases and comparing current data with historical data on greenhouse gases.

1.2 Product and Environment

The product is standalone application software which can enable monitoring. The team has decided to develop the product using the Java programming language. The project also uses JavaFX library for the most part of implementation. The team has decided to use NetBeans as the IDE and Gitlab for the version control management.

1.3 Team Members

The project will be done by four team members Cecylia Borek, Gaurab Mahat, Roger Wanamo and Sai Polineni.

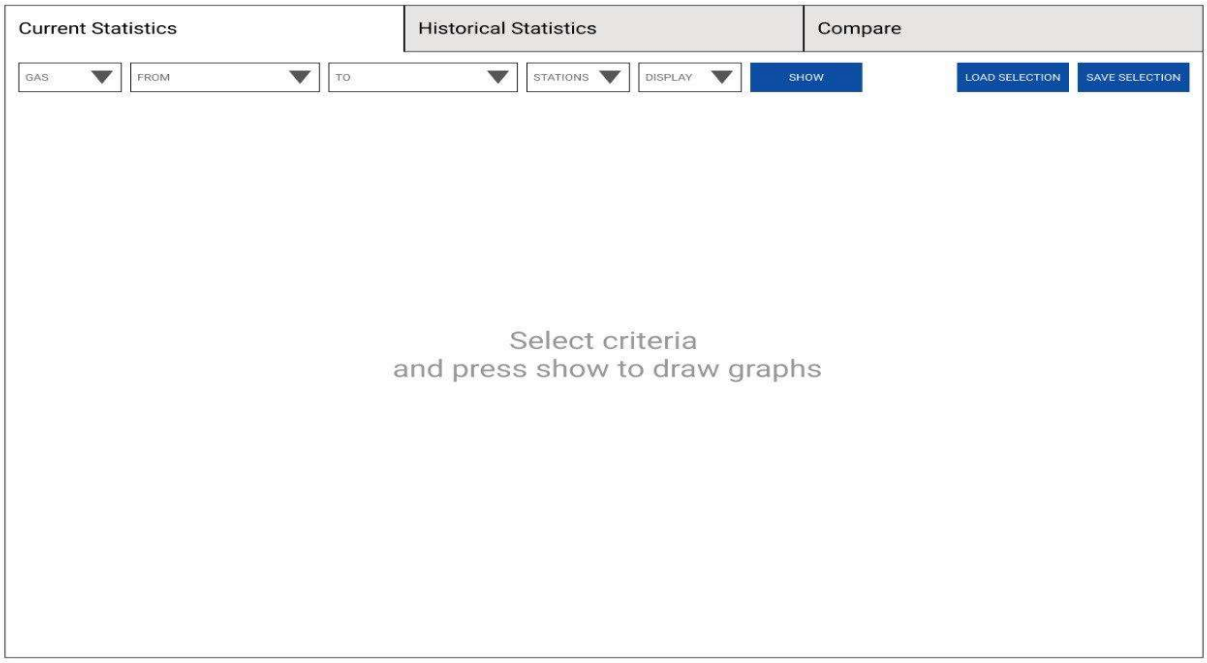
1.4 Related organization

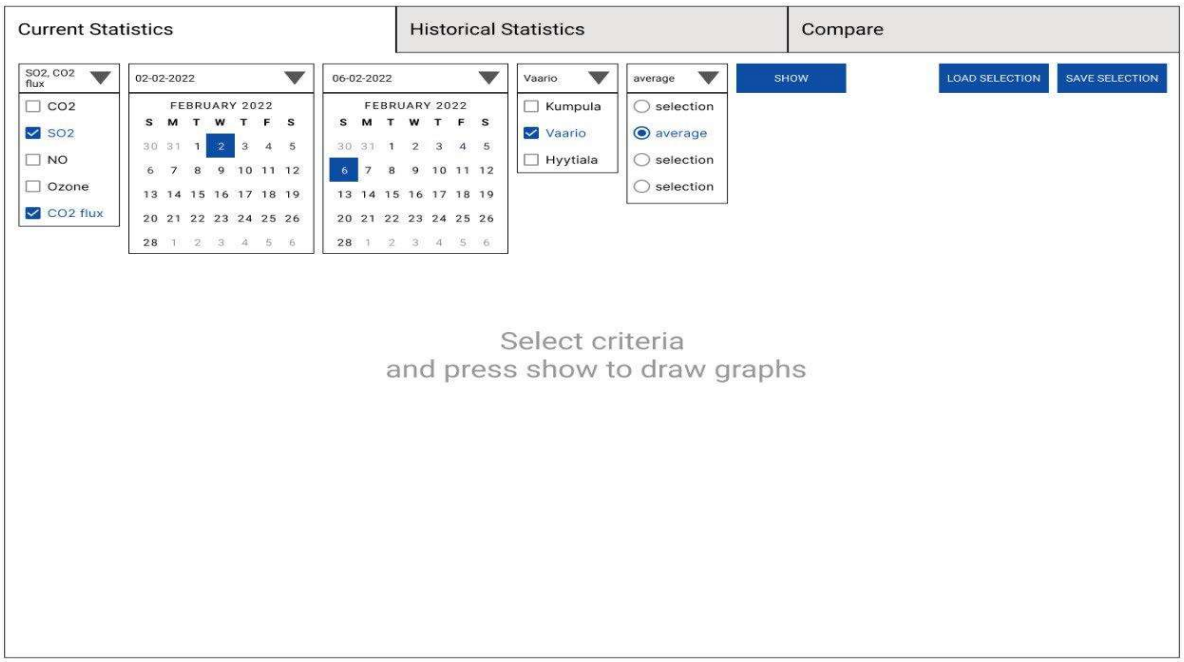
The project is assisted by Ville Heikkilä. Ville is one of the Teaching Assistants of the course. Responsible for guiding the project management towards the correct direction and providing assistance and answers when needed.

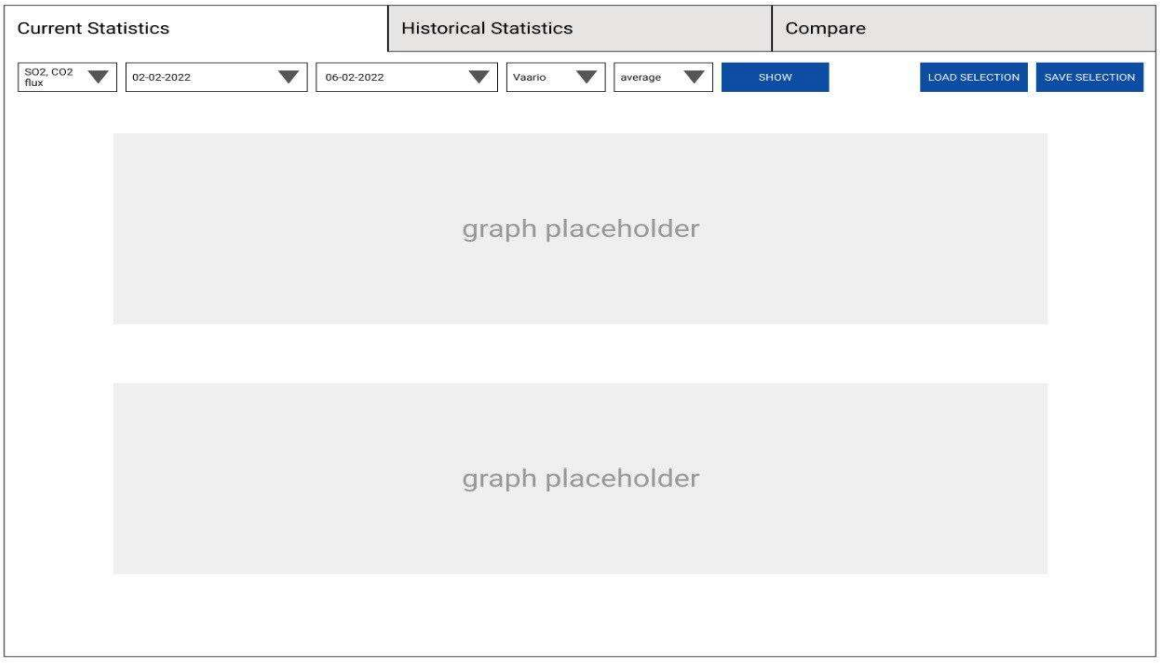
2. High Level Description

The Application will allow a user to monitor the current statistical data about the several types of greenhouse gases whose data has been taken from the SMEAR database and API. It also allows the user to watch the historical data and to compare the CO2 emissions. These operations are performed from the three tabs in the UI (User Interfaces) with the names Current statistics, Historical data, and the compare CO2.

Design Images:







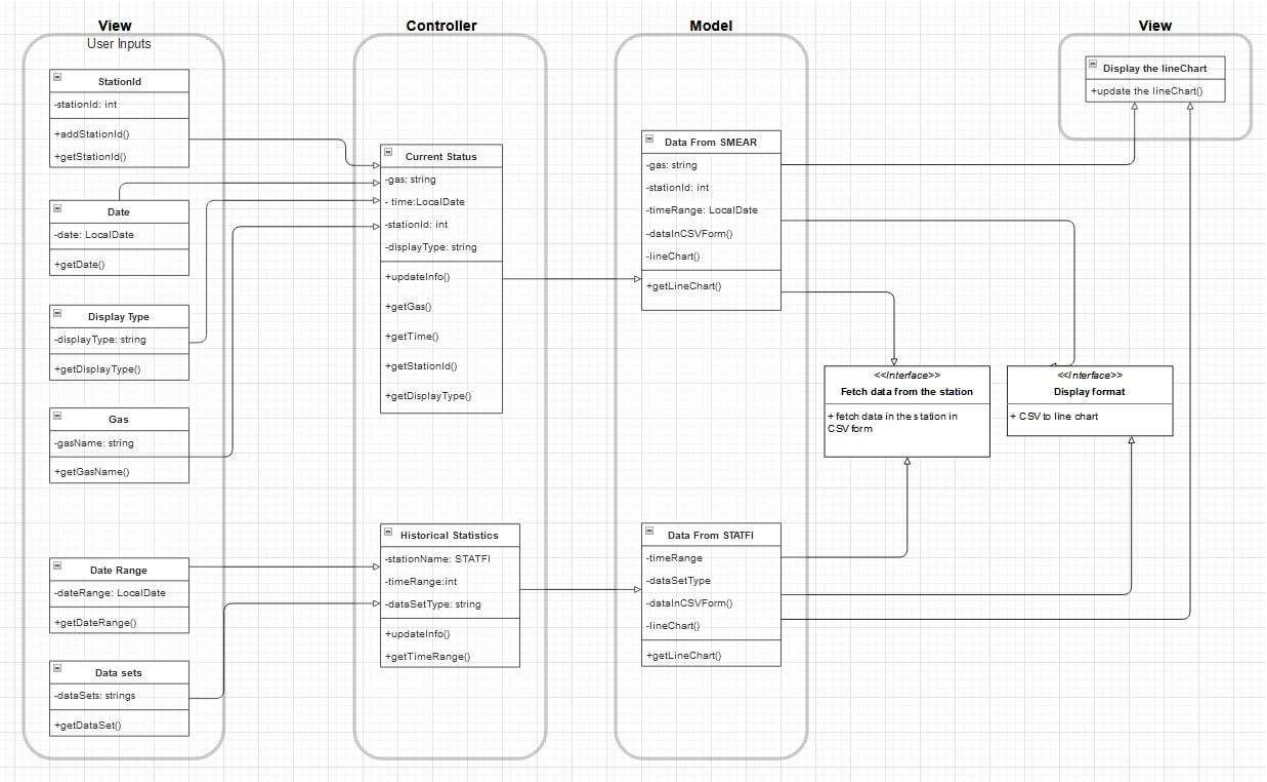
The current statistics tab will allow the user to monitor the current statistics about the different gas emissions. The Gas dropdown menu will enable the user to select a single or multiple gases at a time. The two date fields are used to select the dates with a time range in between them. The location menu contains all the current locations that can be monitored. Users can select multiple locations at a time. Users can view the graph by clicking the show button. The save

selection button can enable the user to save their selection so that they can view their saved selections next time easily.

Similarly the tabs for Historical statistics and also the compare are used to monitor the historical data from the STATFI and for the monitoring of the CO2 values.

Class Diagram:

The below class diagram will display the important classes and interfaces that are included in the project,



From the above class diagram, the View component consists of six classes. The stationid class will have the information about the station and methods to add a new station and get a stationid. The date class will contain the date attribute and a getDate method. The display type class holds the display type attribute and getDisplayType method. The Gas class will have the getName attribute and getGasName method. The Date range class will have the dateRange attribute and getDataRange method. At last the Data sets class will have datasets as an attribute and getDataSet method. These classes will serve as the user inputs in the application.

The Controller component will have mainly two classes, Current statistics and Historical statistics. The Current statistics class will draw the input from the Stationid, date, displaytype and gas

classes and it also have the methods that are available in them. In addition this the updateInfo method is present to deal with the updates that are based on the users. The Historical statistics class will draw the inputs from the Date range and Data sets classes. It has an attribute stationName with STATFI, so that it uses the STATFI data for the monitoring of the historical data in this project.

The Model component consists of two classes, Data From SMEAR and Data from STATFI. The Data From SMEAR class will draw the input from the current statistics class, and it also has additional attributes such as dataInCSVFormat, lineChart and the method getLineChart. The Data From STATFI class will draw the inputs from the Historical statistics class and it has an additional attributes such as dataInCSVFormat, lineChart and the method getLineChart. These two classes communicates with the interfaces in fetching the data from the station and in displaying the CSV information into the line chart format.

Further Display the lineChart class will gets the information from the Data From SMEAR and Data From STATFI classes and updates the latest line chart.

3. Boundaries and Interfaces

3.1 Information Flow

The information flows in the project based on the user selections. First the selections that are chosen by the user will flow as the commands to the query builder component, the query builder will convert the commands into suitable HTTP requests. The HTTP requests communicates with the backend using API interface and fetch the results. The results that are fetched in the form of JSON objects will be converted into the graphs by the data renderer component.

3.2 Components and Responsibilities

The components that are used in the project design are,

1. UI component: The UI component serves as the starting point for the whole application. It allows the user to interact and monitor the application. The UI is developed using FXML /CSS.

2. Controller Component: The Controller component interprets the commands that are given by the user, and it re-directs the commands to the logic component.

3. Application logic component: The component is used to handle the user commands that have been sent as the input.

4. Query Builder Component: The Query builder component creates the HTTP requests for the commands that are being sent from the application logic component.

5. API Interface Component: The API interface component makes the HTTP requests and communicates with the Backend. It also fetches the result data from the Backend.

6. JSON Interpreter: The JSON interpreter component is used to convert the JSON objects. These objects can be referred to the result records in this project.

7. Data Interpreter: The data interpreter component extracts the relevant data from the incoming data from the JSON interpreter component.

8. Data Renderer: The data renderer component turns the data into the graphs and those graphs can be shown.

9. System Hardware Component: The Component is used to store the graphs that the user wants to save in his device.

4. Reasonings

The decisions about the design have been made collectively by all the team members. The team members initially held a starting meeting for the project to discuss the approach and implementation of the project.