3D models for food processing systems
Marel

Operating Manual

BSc Computer Science
Spring 2018

Authors:
Daníel Freyr Sigurðsson
Egill Jónsson
Helgi Björn Bergmann
Örlygur Ólafsson

Introduction

This system is a simulation of a SENSOR-X-302 machine. It utilizes the web socket of the PLUTO system running on the machine and replicates its state. This system is for maintenance and monitoring purposes. It has a graphical interface which pinpoints where errors occur on a 3D representation of the machine, as well as aiding in diagnostics with offering detailed information about the components within the machine.

The system aims to replicate the look and feel of the current user interface that is on the machine, however some extra functionality and features have been added to manipulate the machine in 3D space and display real time diagnostics information.

Conventions in this document

Canvas - The space on screen on which the mesh will be displayed.
Mesh - The 3D replication of the machine.
Viewport - The entire visible screen in its current state.
Alarm - Indicates that a malfunction, safety switch or sensor have been triggered which hinders the machine from running.
Status indicator - small circle, color coded to match the accessibility of the machine and its components, they are as follows:

- **Green** - Indicates that the system is ready to run.
- **Yellow** - Indicates that an alarm has been triggered.
- **Red** - Indicates that the system is running and should not be manipulated in any way.

Installation

If problems persist throughout the installation process, make sure to check out the TROUBLESHOOTING section down below. The setup is described step-by-step, where step two is dependent on having finished step one, step three depends on step two and so on.

Linux

1. Prerequisites - The solution is based on a internet browser and requires a browser that can read and compile html5 documents with its corresponding CSS. This step will not be included in this guide, however we recommend Mozilla Firefox to run the solution. In addition you'll need nodejs and npm, installation of which are both included in step three running the dependency installer.

2. Acquiring privileges - In order to be able to run this setup guide, first off make sure that the production.sh script has execution privileges. This is done by navigating to this folder in a terminal and writing the following command:

```
$ chmod +x production.sh
```

3. Running the dependency installer - This script will acquire and install nodejs, npm, postgres, set up the user for the database, give the user required privileges, set up the database and tables as well as populating them with corresponding data. To run the script, make sure you are logged in as the administrator with administrator privileges. Navigate to the folder including the production.sh file (included in the repository of this solution) and in the terminal command prompt input the following:

```
./production.sh
```

4. Setting up a server - To run the solution you'll need to start a local server that receives and manipulates GET and POST requests from your browser done via node and npm. The server has a few dependencies that are required to run. These dependencies will be acquired automatically via express and npm. To do so, open a terminal window and navigate to the root folder of the solution (~/path-to-solution/3ddisplay) then typing in the following command:

```
npm install
```

The setup of the server should now be complete. However you might need to modify the config file for postgresql to allow local network traffic through to the server. This step might not always be necessary and is described in the TROUBLESHOOTING section of this document. (For advanced users the config file will be located in "/etc/postgresql/*/main/pg_hba.conf").

Windows

1. Prerequisites - first off you'll need a few programs that the solution builds on, they are as follows:
Internet Browser - The recommended browser for this is Mozilla Firefox, available at: https://www.mozilla.org/en-US/firefox/new/
Nodejs - The recommended version for this solution is (current stable) 8.11.1 available at: https://nodejs.org/en/
POSTGRES SQL - The tool for working with our database, this and PGADMIN are available at: https://www.postgresql.org/
Node Package Manager - Although this is included in the list of dependencies, this is bundled in with node. So after you've acquired node you should have npm installed as well. Documentation for npm is available with this link: https://www.npmjs.com/package/npm

2. Setting up database - For full functionality of the solution to work, the database and its tables need to be correctly modified and set-up.
To do this open a command prompt (located at: C:\Users\user\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\System Tools) and in the terminal, navigate to 'path-to-project/3ddisplay/public/BASH'. From there, input the following commands:

```
psql -d postgres -U postgres -a -f CREATEUSER.sql
```

followed by:

```
psql -d marel -U marel -a -f CREATETABLES-WIN.sql
```

In some cases psql won't be recognized by the command prompt and requires an additional setup step. This is described in the TROUBLESHOOTING section below.

Now the database and its tables should be setup.

3. Setting up a server - To setup the server a package.json file needs to be generated, however npm will compile this file for you. From the command prompt navigate to "path-to-project/3ddisplay" (where path-to-project is the path to the storage on which the solution is stored) and input the following command:

```
npm install
```

The solution should now be setup and ready to run (see: RUNNING THE SOLUTION)

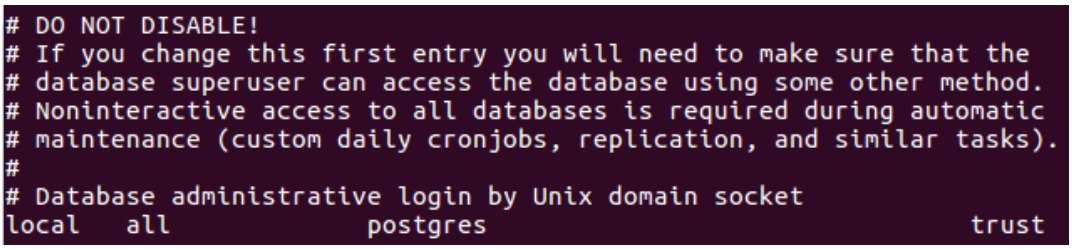
TROUBLESHOOTING:

Linux

Modifying the config file of postgresql - In some cases starting a server as localhost will receive a fatal error, this may be due to the config file being set to reject such traffic. To open postgres SQL for localhost traffic, make sure you are logged in as administrator and open a terminal window and input:

:~$ sudo nano ~/etc/postgresql/*/main/pg_hba.conf

This step might require a password for currently logged in administrator (linux user).
In this document scroll down to the section showed in the screenshot below.



The fields on the bottom line of the screenshot are as follows:
TYPE DATABASE USER ADDRESS METHOD

The server uses the database *marel* and user *marel*, in future applications creating a line which allows your machine (ip address) or local network (CIDR) to host and receive local traffic would be advisable. But for testing and trial purposes changing the last word to *trust* will allow the connection unconditionally. “This method allows anyone that can connect to the PostgreSQL database server to login as any PostgreSQL user they wish, without the need for a password or any other authentication” (https://www.postgresql.org/docs/9.1/static/auth-pg-hba-conf.html). This can be dangerous but serves the purpose of testing the system, as there is no sensitive data currently being stored in the database.

Once done modifying your *pg_hba.conf*, save the changes and close the editor.

After modifying the file, a service restart is required this is done by opening a terminal and input the command:

:~$ service postgresql restart

(you might be prompted with a password, use the currently logged in user password for linux).
You should now be able to start the server and access the webpage (See Running the solution).

WINDOWS

If the command "psql" is not unknown when running it from command prompt, you need to have Postgres SQL installed on your machine and have the file path configured. Control Panel -> System -> Advanced System Settings -> PATH -> EDIT. Here you append the following string to your path:

For PostgresSQL version 10

```
;C:\Program Files\PostgreSQL\10\bin
```

For PostgresSQL version 9.2

```
;C:\Program Files\PostgreSQL\9.2\bin
```

Note that the path “C:\Program Files\PostgresSQL\10\bin” might not always be the same if you’ve installed PostgresSQL in another location or drive. In this case find the path to the installation folder and substitute the file path with the correct path to the installation folder.

Further assistance

For further assistance in setup and configuration, contact: helgi25@simnet.is

Running the solution

Linux and windows

Starting the server - To start and open the server for local traffic on port 3000 navigate to the root folder of the solution, in a terminal/command prompt window and type in the following command:

```
node app
```

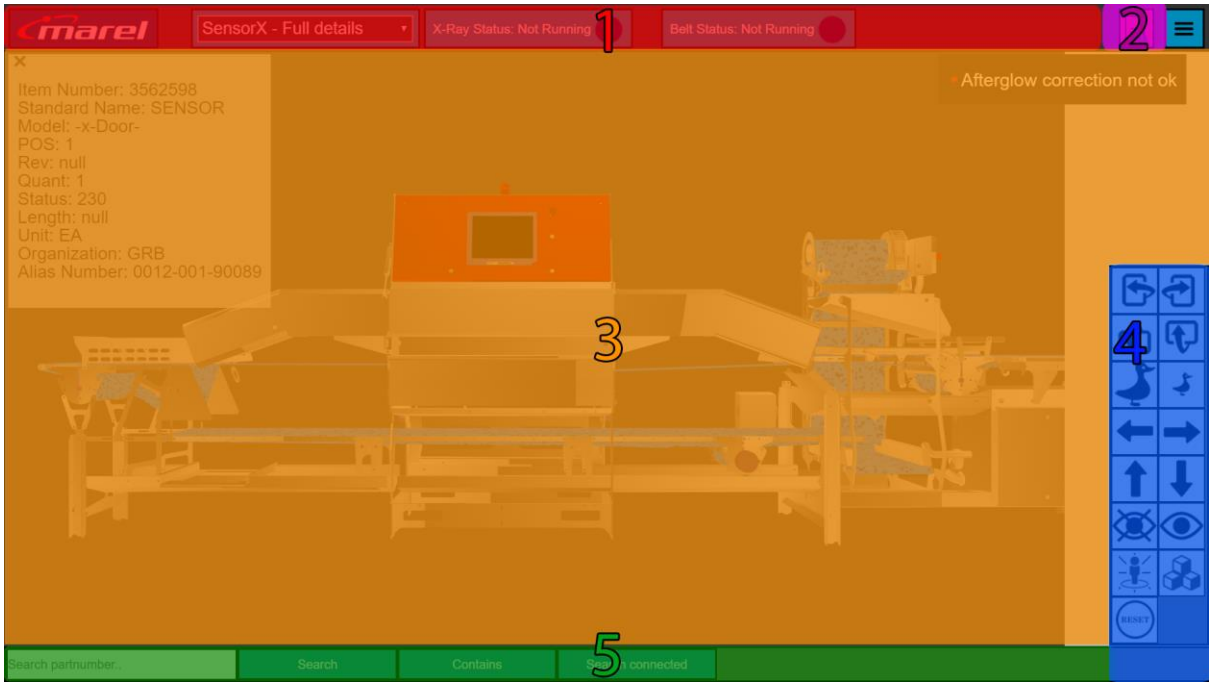
Open the solution - The solution and server should be accessible on your browser. Simply open a internet browser window and navigate to:

```
localhost:3000
```

Layout

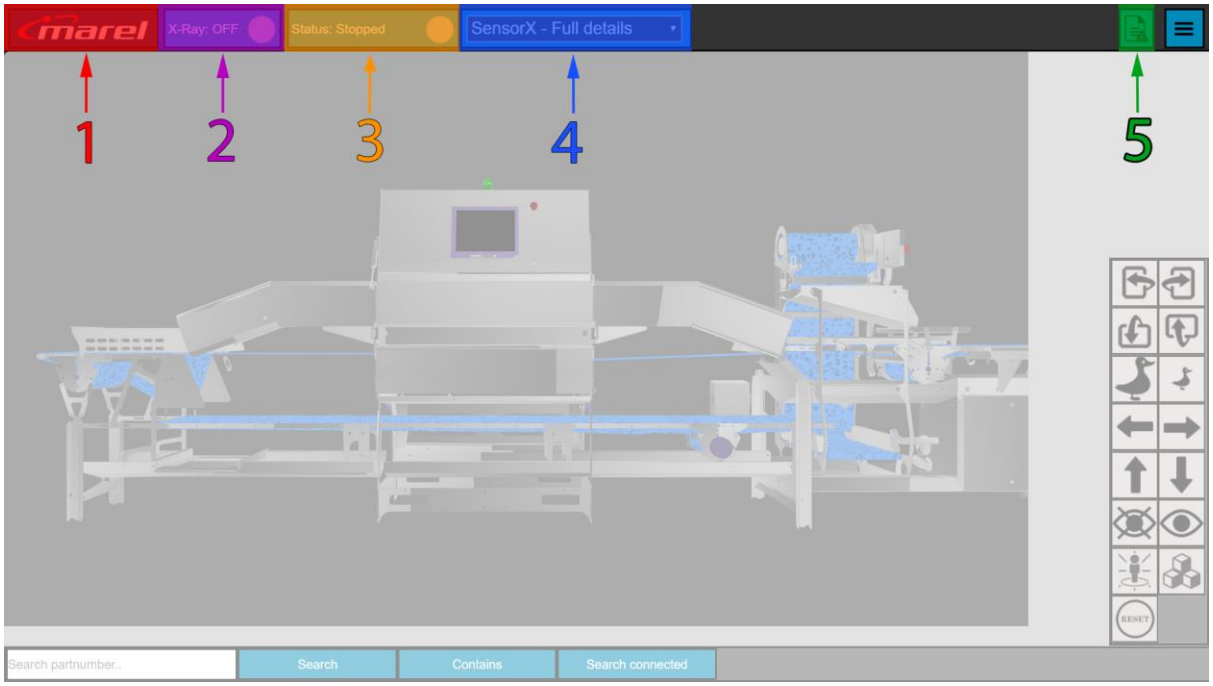
The layout of the system is built after the PLUTO software which is on the system currently running on Marels food processing units. So the controls and interface should look similar or the same in many cases. Developed with input and assistance from the software development team of Marel Iceland.

Front page



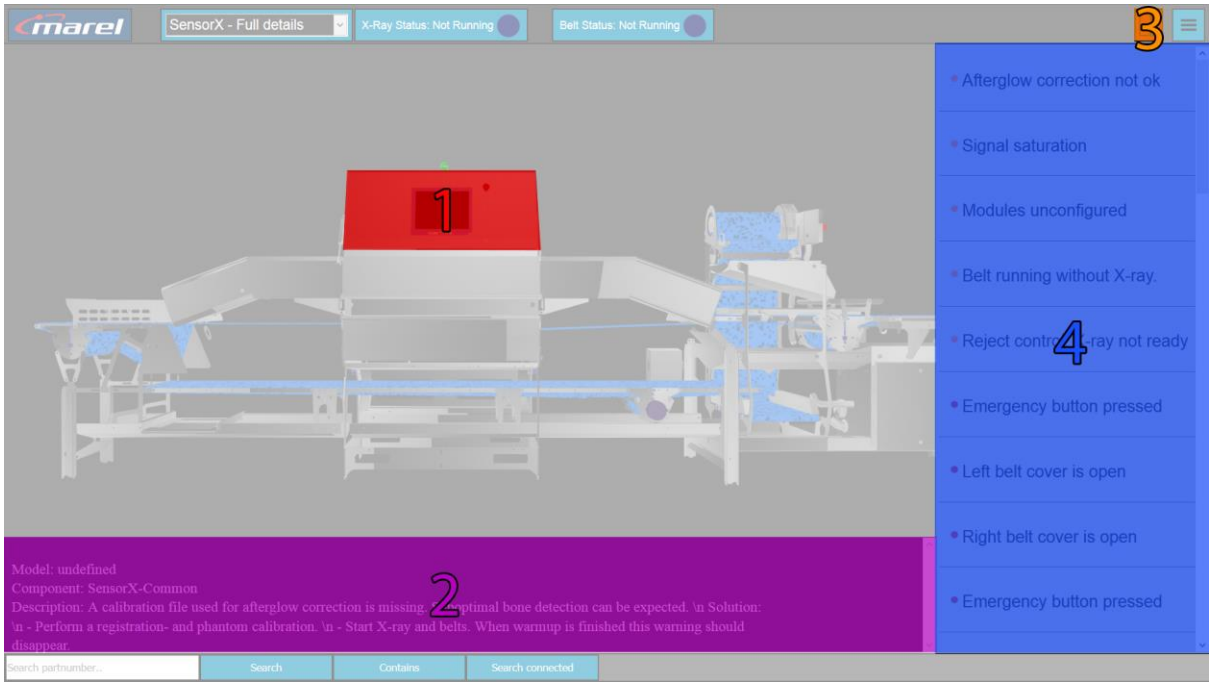
Legend:
1) **Information bar** (page 8) - Used to indicate which mesh is being viewed, the state of the system and alarm view menu.
2) **Alarm view**(page 9) - Opens a new view, where alarms and detailed alarm information can be displayed.
3) **3D canvas** (page 10) - The area that the mesh is displayed on, has a overlapping information window and small alarm indicator.
4) **Toolbar** (page 11) - The tools used to manipulate the mesh and canvas space.
5) **Search bar** (page 12) - Various helper tools and functions to aid with selection and search.

Information bar



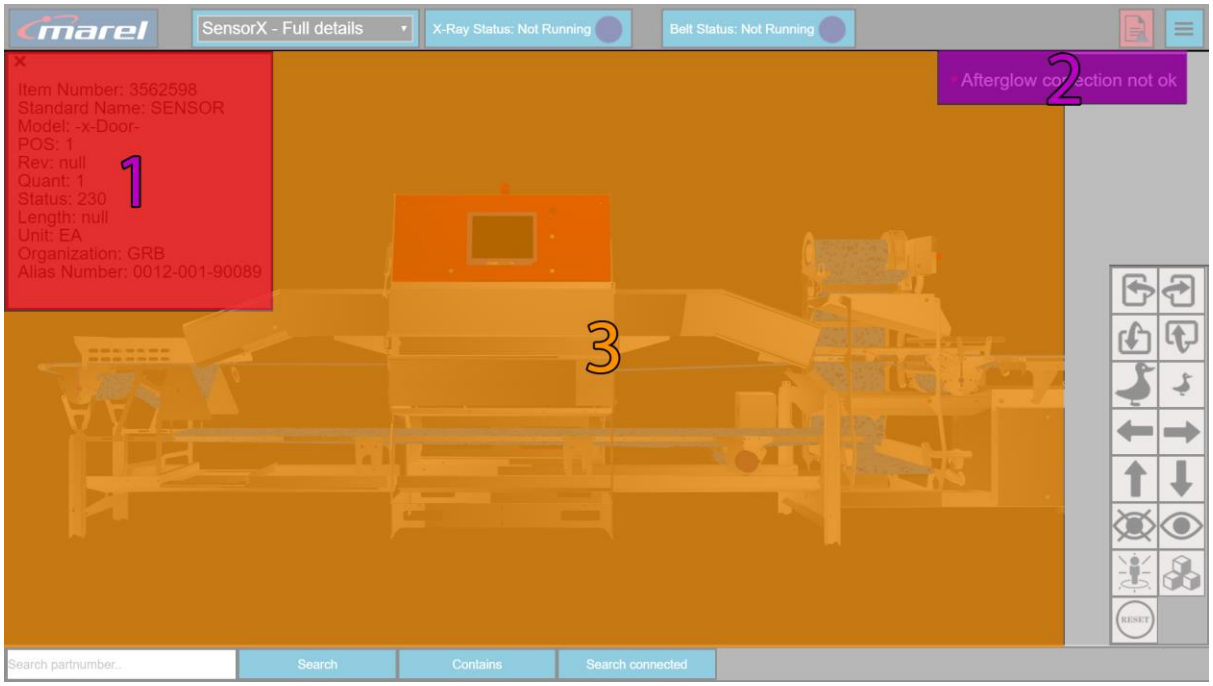
Legend:
1) **Logo**
2) **Status indicator (X-ray)** - this will change colors to: Green, Yellow or Red in correspondence with the current machine state. (see “Conventions used in this document” for more details).
3) **Status indicator (belts)** - this will change colors to: Green, Yellow or Red in correspondence with the current machine state. (see “Conventions used in this document” for more details).
4) **Model selection -** The drop down menu offers partial views of the machines components as well as a reduced version of the mesh (to save loading time).
5) **Alarm View** - Clicking this will change the viewport to an alarm information screen. Which is a list of all recent alarms triggered in the machine. For more information see: “Alarm Section” subchapter.

Alarm View



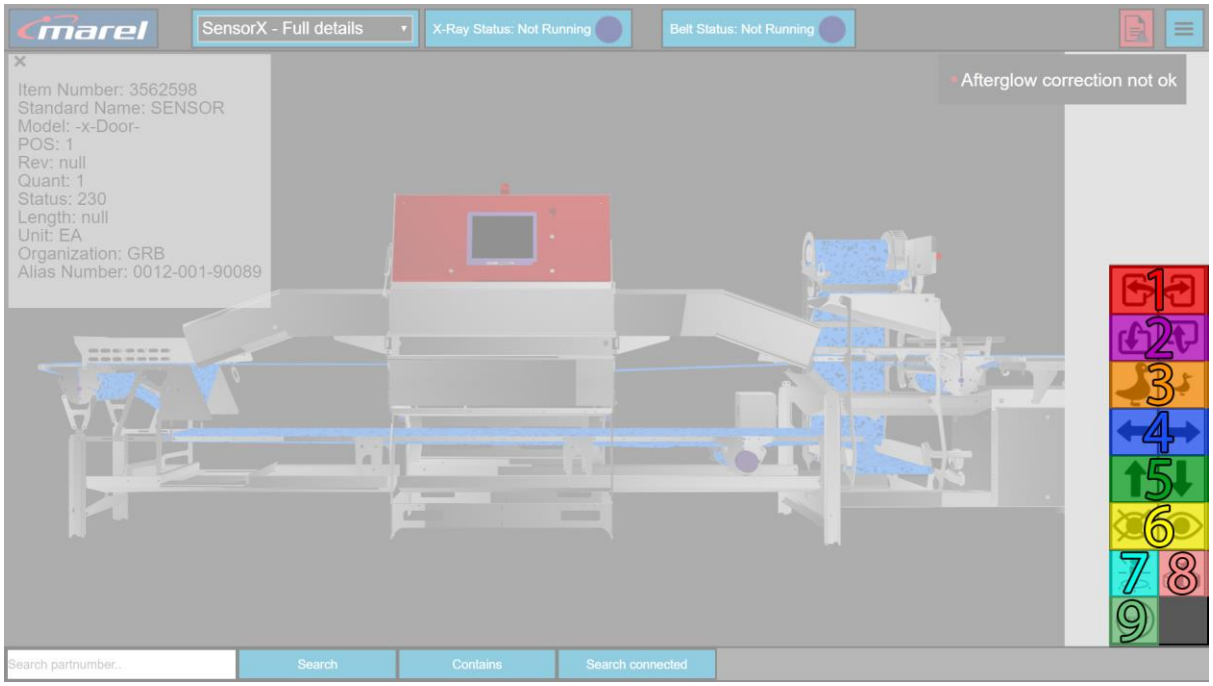
Legend:
1) **Parts involved** - When an error is selected the 3D canvas will highlight the parts involved, indicating what might be the cause of the error.
2) **Alarm details** - Detailed information window about the specific warnings (Model, component, description, Alarm ID and priority respectively).
3) **Alarm section** (close) - closes the alarm viewport and returns to the front page view.
4) **Alarm list** - Lists all warnings, the name of the alarm, the priority of the warning (color in front) and the date and time it first appeared in the system. By clicking on a specific warning in this list, addition detailed information about that specific warning as well as highlighting a component or components on the canvas. The highlighted component is the one most related to the alarm or warning.

3D canvas



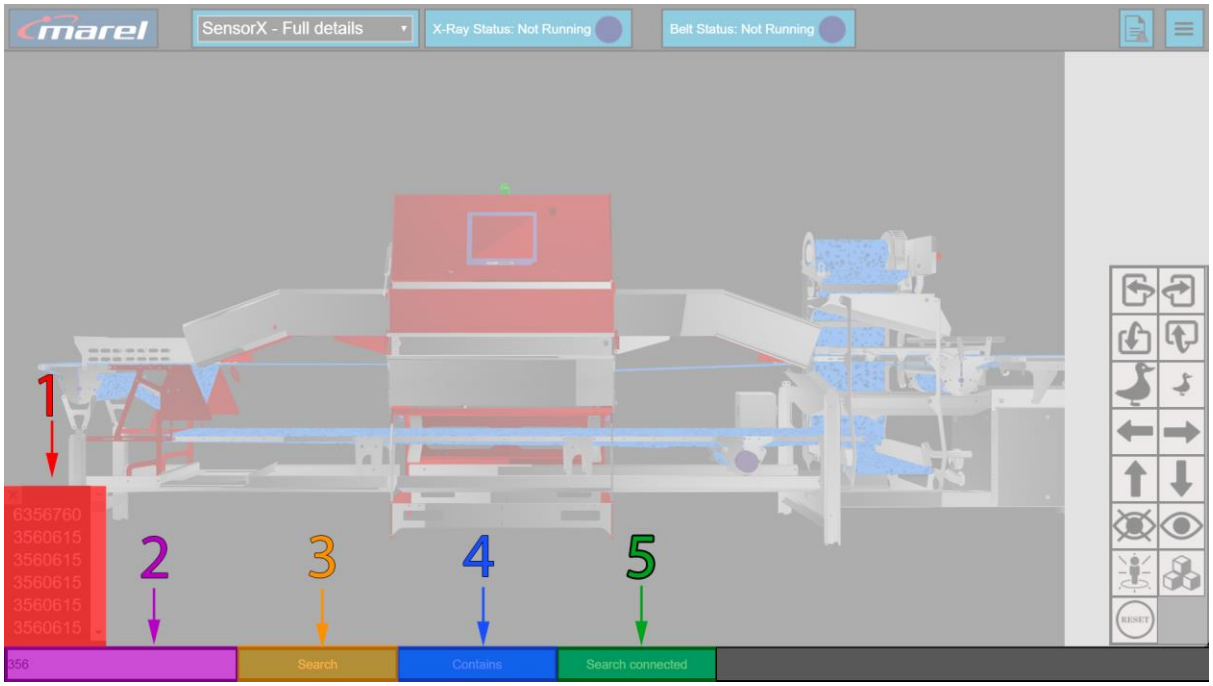
Legend:
1) **Component information window** - the window shows information about a specific component highlighted and/or selected. By default the window itself has an option to expand and show more detailed information about the part.
2) **Latest alarm window** - the window shows information about the latest alarm as well as that alarms priority.
3) **Mesh** - The 3D graphical drawing of the machine.

Toolbar



Legend:
1) Rotates the model on the canvas on the horizontal axis.
2) Rotates the model on the canvas on the vertical axis.
3) Zooms in and out on the model.
4) Moves the model on the canvas left and right.
5) Moves the model on the canvas up and down.
6) Hides and unhides selected components.
7) Isolation view, hides every other component aside from the currently selected one.
8) Select multiple, toggles the selection mode between singular and multiple select.
9) Resets the model back to original position, angle and visibility.

Search bar



Legend:

1) **Search results** - displays search results.

2) **Search input** - takes in item number and highlights the component if one is found.

3) **Search** - uses the input from the search input to send a query to the database.

4) **Contains** - performs a partial item number search. Keep in mind that if a shorter string is used to search for parts the more likely it is to have multiple components that match the search parameters.

4) **Search connected** - Shows every item on the mesh as a individual item number, this however looks up clustered items, such as items that come pre assembled from Marel. For example hatches, they contain multiple components. This type of search will look for clustered components.