INL/EXT-21-64452

Light Water Reactor Sustainability Program

INVESTIGATING APPLICATION OF LiDAR FOR NUCLEAR POWER PLANTS

September 2021

DOE Office of Nuclear Energy

INL/EXT-21-64452

Light Water Reactor Sustainability Program

INVESTIGATING APPLICATION OF LiDAR FOR NUCLEAR POWER PLANTS

Steven Prescott[1], **Michael Ziccarelli**[1], **and Eric Allen**[2]

[1]**Idaho National Laboratory**
[2]**Environmental Intellect**

September 2021

Idaho National Laboratory
Idaho Falls, Idaho 83415

http://www.inl.gov/lwrs

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Page intentionally left blank

ABSTRACT

Many evaluation, assessment, and modeling tasks at nuclear power plants require spatial information; this often requires physical visits to locations within the facility because the 2D or 3D schematics and current models do not contain enough detail or do not capture as-built and real-world conditions. These visits require extensive manual labor for not only the requesting party but also support groups, such as security. Light Detection and Ranging (LiDAR) mapping is trying to solve that problem by providing very detailed 3D models for low costs. However, the use of these models can be very limited because either component reference information is missing and too costly to add, or there is no way to extract specific spatial data needed for other tools.

This report outlines two main efforts. First, to reduce the effort of "Tagging" data in large 3D models, a general Application Programming Interface (API) was developed to import a variety of existing plant database information into a 3D visualization engine. Filters allow the user to have only zone-specific items listed; then, they can simply click and assign the information to a specific spot or component in the 3D model.

The second part of the work is the development of an interface for importing pieces from the 3D LiDAR model into other systems needed for modeling and simulation, outlined around fire modeling. This interface allows for the retrieval of item location and boundaries enabling the auto generation of models for varying tools.

ACKNOWLEDGEMENTS

This work of authorship was prepared as an account of work sponsored by Idaho National Laboratory (under Contract DE-AC07-05ID14517), an agency of the U.S. Government. Neither the U.S. Government, nor any agency thereof, nor any of their employees makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

Page intentionally left blank

CONTENTS

FIGURES

ACRONYMS

AI	Artificial Intelligence
API	Application Programming Interface
CAD	Computer-Aided Design
Ei	Environmental Intellect
FRI3D	Fire Risk Investigation in 3D
INL	Idaho National Laboratory
JSON	JavaScript Object Notation
LiDAR	Light Detection and Ranging
LWRS	Light Water Reactor Sustainability
OCR	Optical Character Recognition
P&ID	Piping and Instrument Diagram
PDMS	Plant Data Model System
PRA	Probabilistic Risk Assessment
RISA	Risk Informed Safety Analysis
UI	User Interface

Page intentionally left blank

1. INTRODUCTION

Light Detection and Ranging (LiDAR) scans and accompanying software offer many potential uses for the nuclear industry; several facilities have used them for virtual walk-downs and equipment placement planning. Since raw scan's (3D spatial representations with picture overlays) uses are limited, recently, more companies are offering tools that allow adding information to the models, so users can click on items to see names, instructional info, links to diagrams and videos, or to help maintain regulation documents.

These models enable a wide range of applications. However, adding this data to a model can be very time consuming and require someone well-versed in facility designs and operations. This is especially true with nuclear facilities due to the extremely large number of components and various safety systems. This obstacle can be overcome by using plant data already compiled for other tasks such as fire modeling.

To overcome this issue and explore additional ways to use LiDAR information and in order to reduce operation costs and maintain plant safety, Idaho National Laboratory (INL) is working with the 3D LiDAR scanning and model hosting company Environmental Intellect (Ei). Using an existing plant's Plant Data Model System (PDMS) or FRANX-fire model, we show how to simplify the process of linking the data to spatial locations or items in the 3D model. This article also outlines the software requirements and processes for retrieving specific item information from the 3D model for use in other plant analysis using the Fire Risk Investigation in 3D (FRI3D) software for design needs. This research is being done under the Risk Informed Safety Analysis (RISA) pathway of the Light Water Reactor Sustainability (LWRS) program [1].

2. LiDAR 3D VISUALIZATION FEATURES

2.1 Scan Data

LiDAR scanning captures high-resolution, spatially accurate representations of real-world conditions. Embedded photogrammetry in the LiDAR scan data captures multiple high-resolution 360-degree photos of all equipment. Combining these two data-capture techniques into a single, seamless data-capture process provides the best-possible digital representation of real-world

Figure 1. Ei mobile mapping technician with mobile mapping device.

equipment and conditions. Various hardware devices were developed to capture this data. Early versions used a tripod that was setup in different locations to capture objects from different perspectives. Newer devices, such as the one shown in Figure 1, allow the user to move and scan at the same time or in intervals. With more scanning angles, there are fewer shadow spots and better images are produced.

With new forms of mobile scanning equipment available, the cost of performing detailed scanning has gone down considerably over the years. To get an accurate cost on scanning, an assessment of the facility and areas to be scanned would need to be done and will vary, depending on the scope and type of scanning. Included in the cost of scanning is the "stitching of data," and, in the case of Ei, linking additional plant information such as Piping and Instrument Diagrams (P&IDs). It is estimated a two unit facility internal and main grounds could be scanned for $50-200K, depending on accessibility and amount of additional data to be included.

2.2 Visualization

Visualization tools for LiDAR consist of a 3D environment generated by millions to billions of points, depending on the size and resolution of the model. These points construct the shell of

Figure 2. Side-by-side comparison of photogrammetry and LiDAR data collected with mobile mapping, including measurements in the Ei viewer.

objects surrounding areas that were scanned. More recent technologies attach a picture fragment or photogrammetry data to each point that constructs a realistic view of the model.

Typical LiDAR scanning and visualization techniques require the capture and rendering of terabytes of data on a single machine, requiring end-users to have powerful computers with extremely high-capacity hard drives. The approach taken by Ei uses a modern and secure web application to render the LiDAR data using a powerful server, allowing the user to view it on a basic computer or tablet with minimal network and processing requirements. The Ei approach also seamlessly integrates photogrammetry data such that a user can easily switch between LiDAR and photo views of each piece of equipment, as seen in Figure 2.

2.3 Tagging or Pins and Linked Information

In addition to the powerful visualization, the Ei viewer enables end-users to add their own data points called "pins" in the 3D environment. They are also referred to as "tags" or "points of interest." A main use of these pins is to identify a specific component in the 3D space, as shown in Figure 3. Each can also have a bounding box that identifies the space of the component. The User

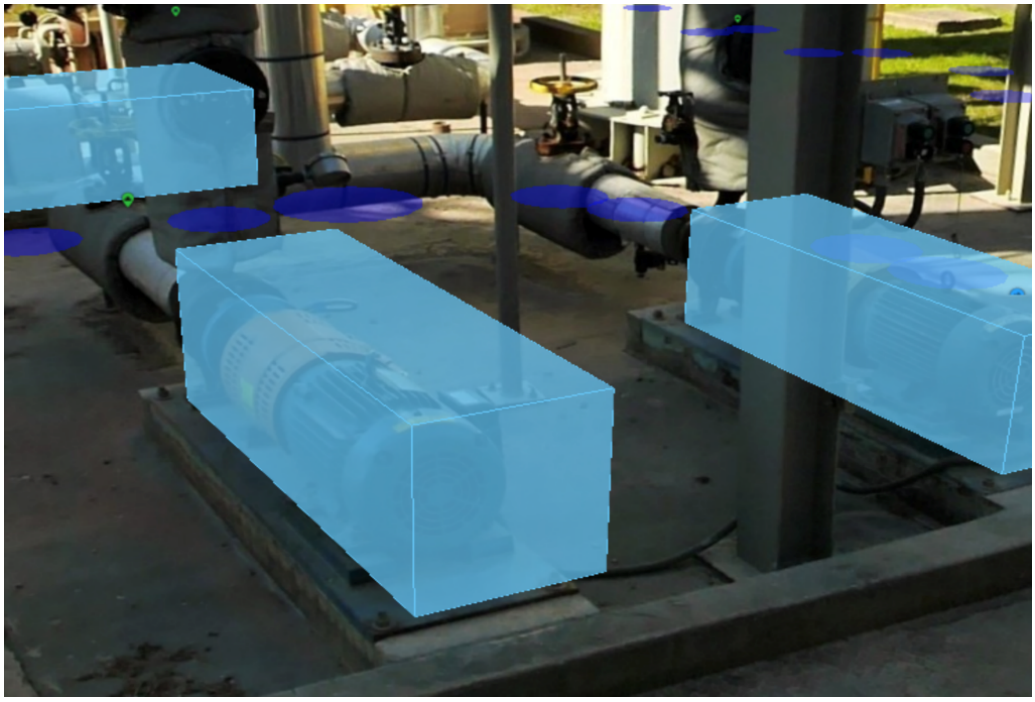

Figure 3. 3D LiDAR bounding boxes around assets of interest (pumps) in the Ei viewer.

Interface (UI) allows the pins to be added in a simple and flexible manner for the user entering data or selecting data that is preloaded into the environment.

There are also "layers" that can be created, and each pin can be placed on one or many layers. Furthermore, these can be associated with one or many "teams," which handle user permissions, manage sharing, and make it easy to add and collaborate on top of the LiDAR visualization.

While the Ei environment tries to make the tagging process easy for its general users, it can still be a very time-consuming process, and customers often have Ei do this part using P&IDs and other information. The nuclear power industry can take advantage of existing plant data in similar formats to make this process very simple, so it can be done in-house with minimal effort. Simplifying the process and being done by those familiar with the facility will reduce time and errors.

2.4 Advanced Features

While features vary amongst different point cloud viewing platforms, the Ei platform was chosen for this research because it already had many desirable features. These include the ability to import plant data, tagging filters, and initial development of an Application Programming

Interface (API) for external data retrieval. The Ei platform has many advanced features for its users [2] that may also be useful for the nuclear industry. These features include:

- Computer-Aided Design (CAD) association with data points – pins can be easily integrated with the site's existing CAD assets (both 2D, 3D, and P&IDs), enabling seamless navigation between real-world views, existing CAD and engineering drawings. This provides additional context as well as additional engineering information needed to drive various use cases.
- Communication - users can communicate and take notes very easily within the application, enabling collaboration between various stakeholders and the capture of valuable information for future users. The different teams allow for notifications to the correct people.

3. IMPORTING NUCLEAR POWER PLANT DATA

Most nuclear power facilities have databases used for risk analysis, such as fire. Examples include the FRANX database used for external hazard Probabilistic Risk Assessment (PRA) models or a PDMS. These databases contain components and event logical/physical links between items. They also can group items into physical zones. This section explains the process used to pull this data into the LiDAR visualization environment for use in tagging components. The design is intended to allow for customizing a given plant's data and the ability to be interfaced with any LiDAR visualization software that has or is willing to implement the required features.

3.1 Plant Data API and the Ei API

3.1.1 Existing plant data

The two databases being used for this research are the PDMS and FRANX databases. Both the PDMS and FRANX databases are Microsoft Access database files, consisting of all the equipment, cables, and raceways with descriptions and other information for these items or safety-related items inside the plant. The location of the items is associated with a particular physical location boundary or zone.

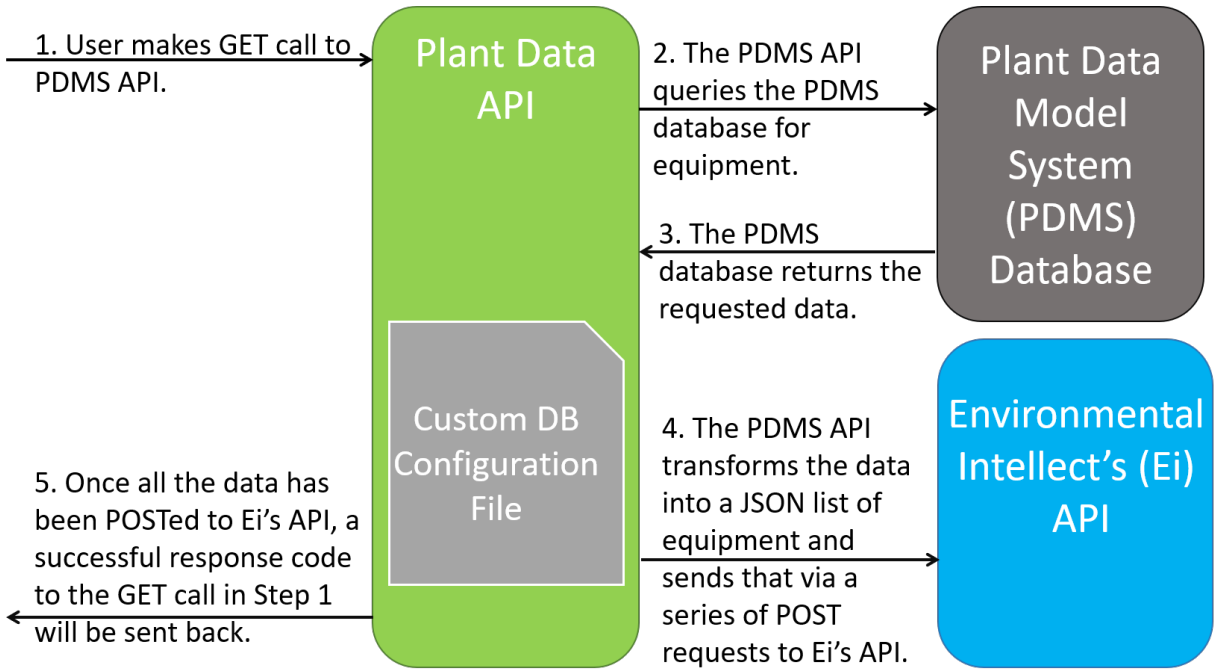

Figure 4. Data flow for plant data to Ei's LiDAR interface.

3.1.2 API design

There are two APIs under development to transfer information from a PDMS or FRANX database to Ei. The first is an API that connects directly to the database. When a "GET" request is sent to this API, it will query the database for all equipment, cables, and raceways, placing all this information into a list. Since this list can become quite large, it is broken up into smaller lists. These smaller lists are sent to Ei's system via a series of consecutive "POST" requests. After Ei has received all the data, the API sends a response code for the initial GET request.

The second API was created in Ei's system. This API allows objects, which can be displayed on Ei's web UI, to be created through a POST request. This is the POST request mentioned above, to which the lists of data will be sent, as shown in Figure 4.

Since the Plant Data API can be used at many sites that may have a variation in their database systems, the API has been constructed to allow for maximum compatibility with different types of databases. This is done by implementing the Dapper NuGet library to connect to the database and making the database connection string and the relevant table and column names configurable.

The database configuration is stored in a JavaScript Object Notation (JSON) file, which allows

for easy updates. An example JSON file is shown in Figure 5. A path to the database file is needed along with the names of the tables that contain the Equipment ID, Cable ID, Raceway ID, the respective descriptions, and Fire Zone ID. The column names for those items will also be needed. If, however, a FRANX database is being used, it will not be necessary to update the configuration file with table and column names. This is because FRANX databases have a universal format for storing this information, which has been built-in to this API.

The JSON file shown in Figure 5 outlines the customization file for attaching to the plant data. For example, if a FRANX database is used, the "DatabaseIsFRANX" value in the "Settings" group should be set to "true." It also has a place for the Ei API token, "EiApiAccessToken," which is required to send any data to Ei. All the fields and a step-by-step process will be specified in the documentation, but it is designed, so someone with knowledge of the system can set it up in less than a day's work.

3.1.3 Equipment data loader web app

In order to assist with properly modifying the JSON configuration file, a web app has been developed to walk users through the process with an easy-to-use interface. This web app, like the Plant Data API, will run and be served locally on the user's computer.

Upon opening the web app, it asks for the base URL of the Plant Data API and has a default value for this prefilled, as this will be the same in most cases. Once the URL has been provided, the user can click the "Load Settings" button to load the existing settings defined in the current JSON configuration file and activate the ability to edit them. As a visual cue to the user that steps have been activated, the relevant steps no longer appear as greyed out; Figures 6 and 7 show the web app before and after clicking the "Load Settings" button, respectively.

After the current settings have successfully loaded, the user can click the "Next" button to move on to the next step of setting the Ei API access token. Similar to above, after entering the token, the user can click "Validate Token" to test the token works properly and activate the next step. Figures 8 and 9 show screenshots of the web app before and after having the token validated.

When the Ei API access token has been successfully validated, it will make API calls to Ei to populate the next step's options for choosing the Ei "Map," "Layer," "Layer Type," and "Property" as the target destination for the equipment data load. This step is shown in Figure 10.

```
{
    "AllowedHosts": "*",
    "Database": {
        "ConnectionString": "Provider=Microsoft.ACE.OLEDB.12.0;Data
            Source=C:\\path\\to\\access\\database.accdb; Persist Security Info = False; ",
        "DatabaseIsFRANX": false,
        "UseSqlOverride": false,
        "SqlOverride": "SELECT EQ_ID AS EquipmentId, EQ_DESC AS EquipmentDescription, FZ_ID AS
            ZoneId, 'Component' AS Type FROM EQUIPMENT",
        "Tables": {
            "Equipment": {
                "Name": "EQUIPMENT",
                "Columns": {
                    "EquipmentId": "EQ_ID",
                    "EquipmentDescription": "EQ_DESC"
                }
            },
            "EquipmentFireZone": {
                "Name": "EQ_FZ",
                "Columns": {
                    "EquipmentId": "EQ_ID",
                    "FireZoneId": "FZ_ID"
                }
            },
            "Cables": {
                "Name": "CABLES",
                "Columns": {
                    "CableId": "CA_ID",
                    "CableDescription": "CA_DESC"
                }
            },
            "CableRoutes": {
                "Name": "CABLE_ROUTES",
                "Columns": {
                    "CableId": "CA_ID",
                    "RacewayId": "RW_ID"
                }
            },
            "RacewayFireZone": {
                "Name": "RW_FZ",
                "Columns": {
                    "RacewayId": "RW_ID",
                    "FireZoneId": "FZ_ID"
                }
            }
        }
    }
    "EiSettings": {
        "ApiAccessToken": "Your Key Goes here",
        "BaseUrl": "https://api.env-int.com",
        "GetMapsEndpoint": "/v1/maps",
        "GetLayersEndpoint": "/v1/layers",
        "GetLayerTypesEndpoint": "/v1/layerTypes",
        "GetPropertiesEndpoint": "/v1/properties",
        "CreateObjectEndpoint": "/v1/objects"
    }
}
```

Figure 5. JSON properties for custom database retrieval.

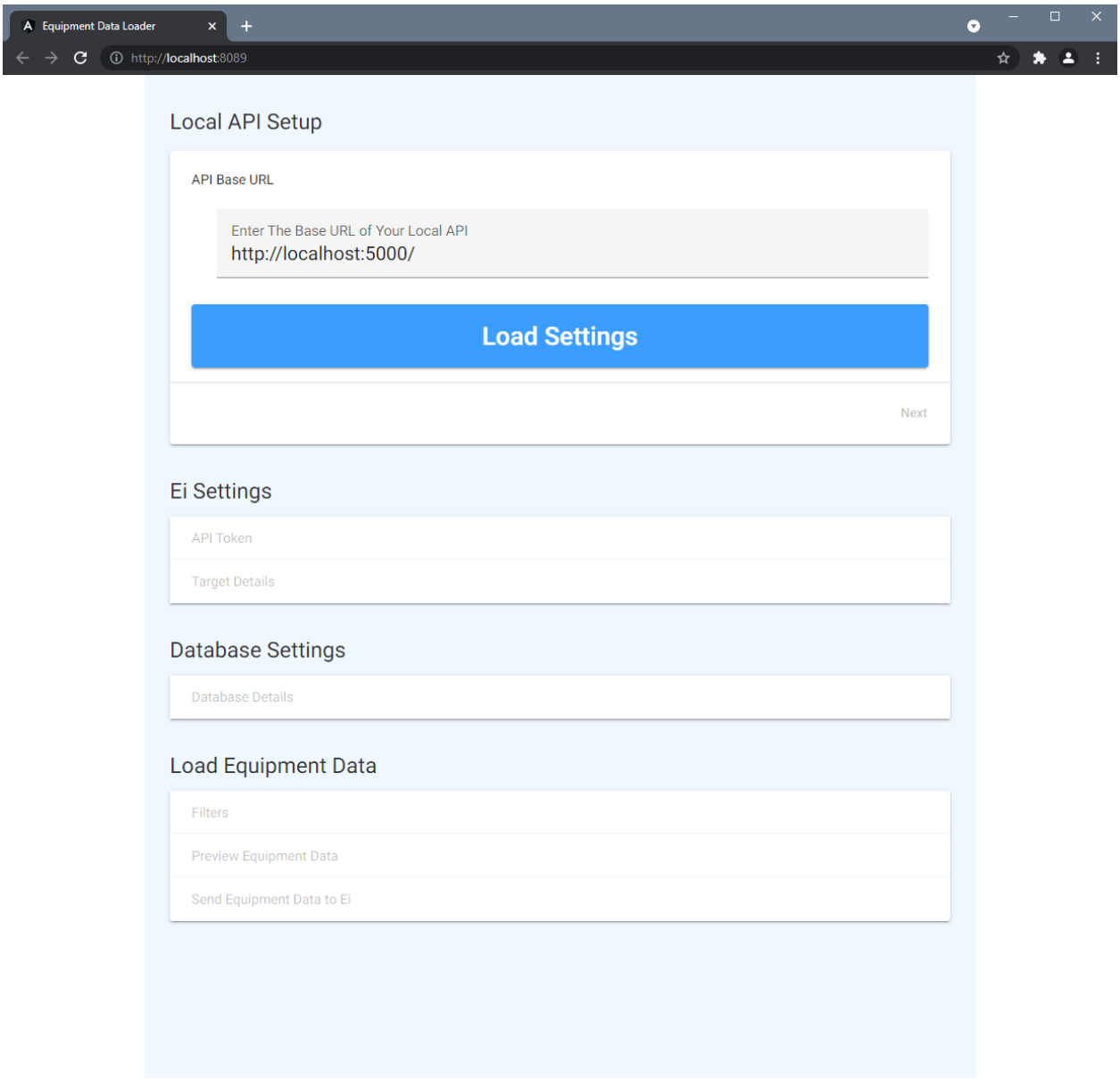

Figure 6. Initial view of the equipment data loader web app.

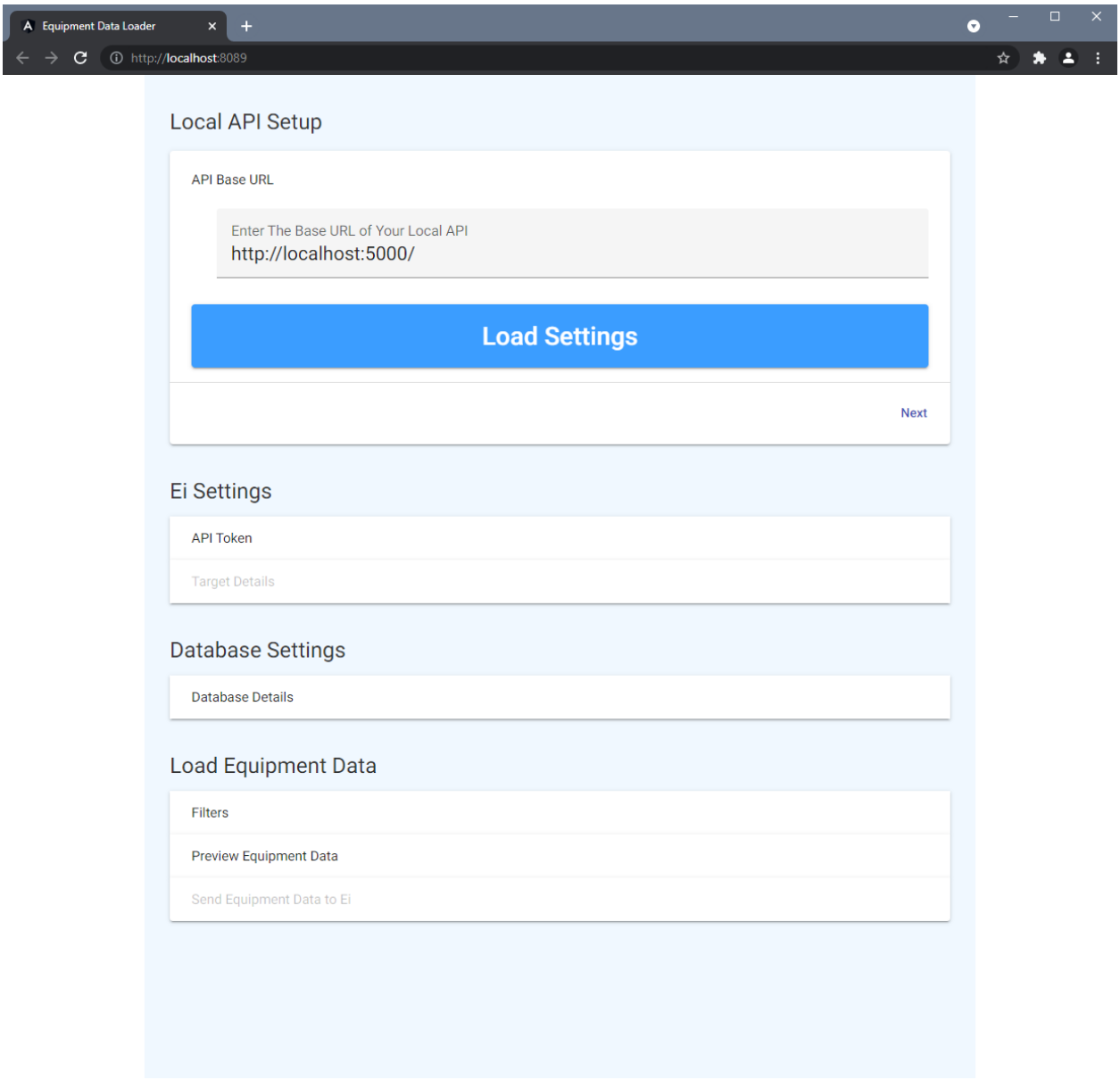

Figure 7. Screenshot of the equipment data loader web app after current settings have been loaded.

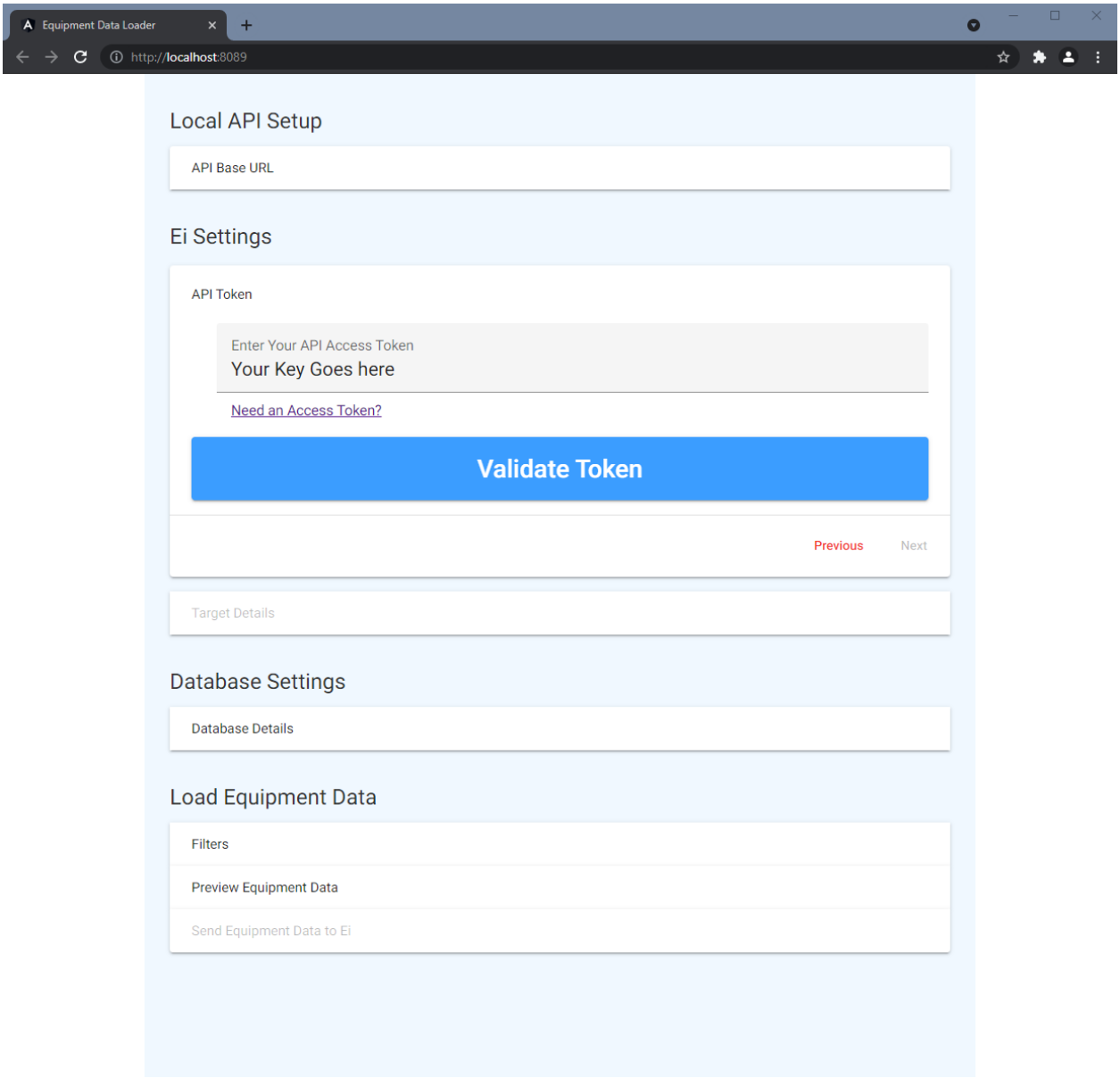

Figure 8. Screenshot of the equipment data loader web app before validating the Ei's API Access Token.

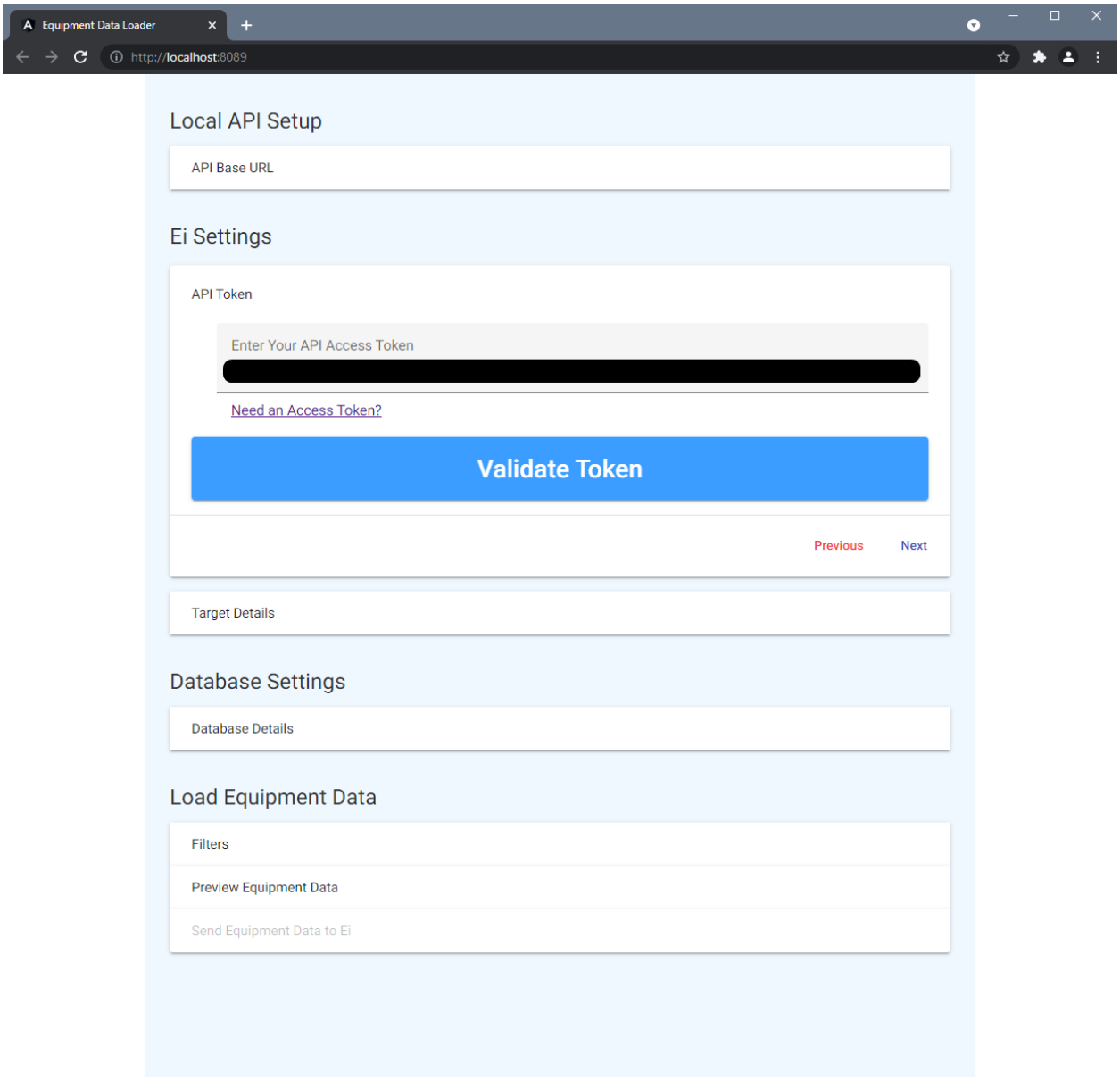

Figure 9. Screenshot of the equipment data loader web app after validating the Ei's API Access Token. For security reasons, the token has been blacked-out.

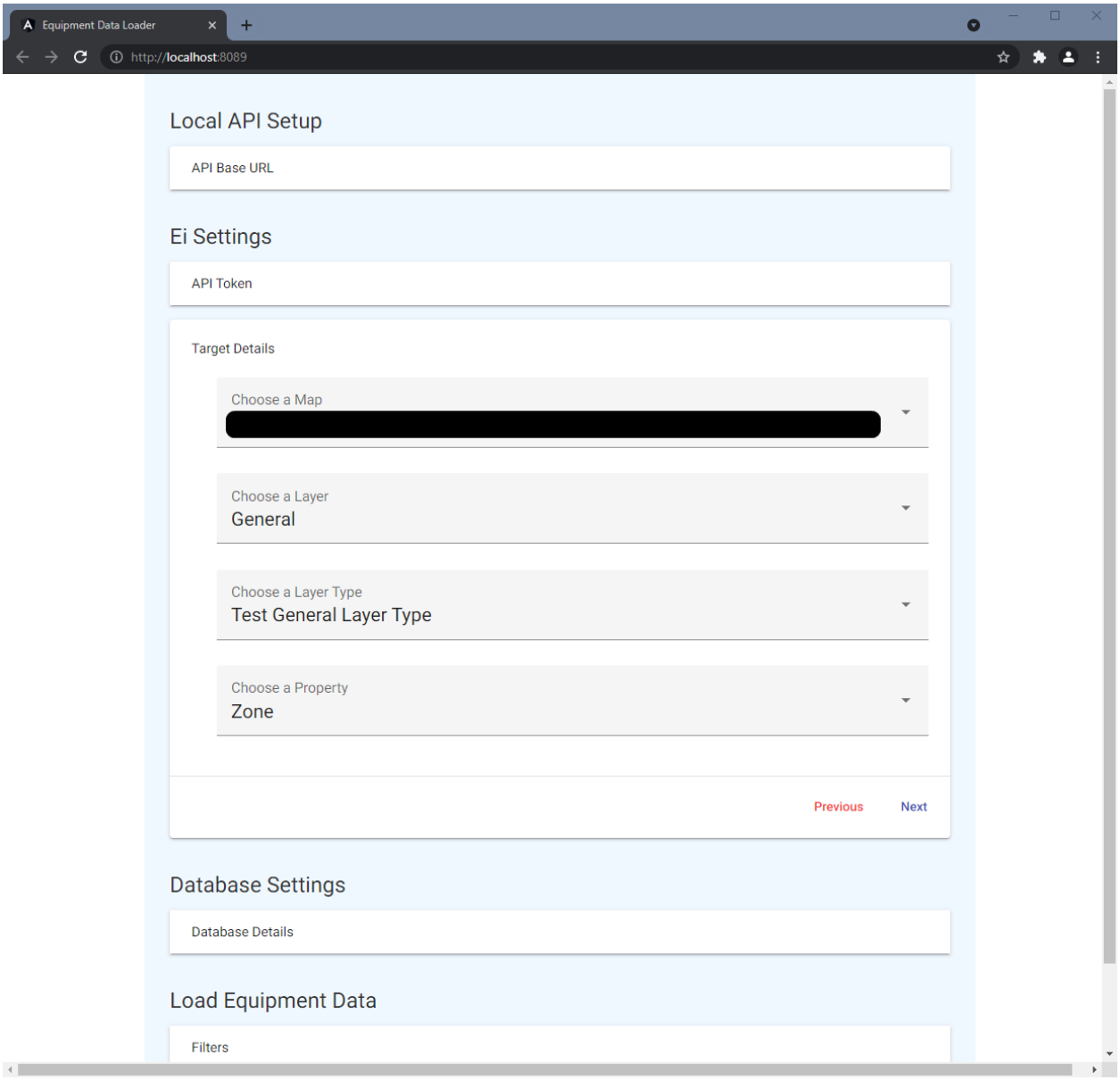

Figure 10. Screenshot of the "Target Details" section of the equipment data loader web app. For security reasons, the "Map" field has been blacked-out.

The next step in the flow is inputting the database details; these details include the path to the database file, whether the database is a FRANX database, table and column names if it is not a FRANX database, and the option to use a custom SQL query. Screenshots of this section are shown in Figures 11, 12, and 13. The custom SQL option is an advanced option available for use with non-standard databases and should be used with caution; ideally, it should only be used if the database is not a FRANX database and setting the table and column names failed to produce proper results.

When the above steps have been completed, the user is then ready to load the equipment data. Filters can be defined, as shown in Figure 14, and then, the data can be previewed before sending it to Ei, as shown in Figure 15. If the previewed data looks correct, the user can click the "Save These Settings" button shown in Figure 15 to generate the proper JSON text that can be copied and pasted into the user's JSON configuration file, "appsettings.json." A screenshot of the pop-up modal with the generated JSON text is shown in Figure 16.

After the user has verified the previewed data is correct and okay to send to Ei, they can proceed to the final section and click the "Click Here to Send The Equipment Data to Ei" button. This will send the GET request described at the beginning of Section 3.1.2, which will make calls to Ei's API to upload the data. Since this action may take some time, a loading bar will appear under the the button, as shown in Figure 17, and disappear once the data has been successfully uploaded to Ei. If an error occurs during this process, the user should see a message appear on the screen saying so.

3.2 Customization

The plant data API and the equipment data loader web app were built to specifically communicate with Ei. However, since the code is simply a framework using Ei's API, it can be customized to adapt to the APIs of other tools. This would allow for the equipment data to be extracted in any desired format and then imported into any other tool or system.

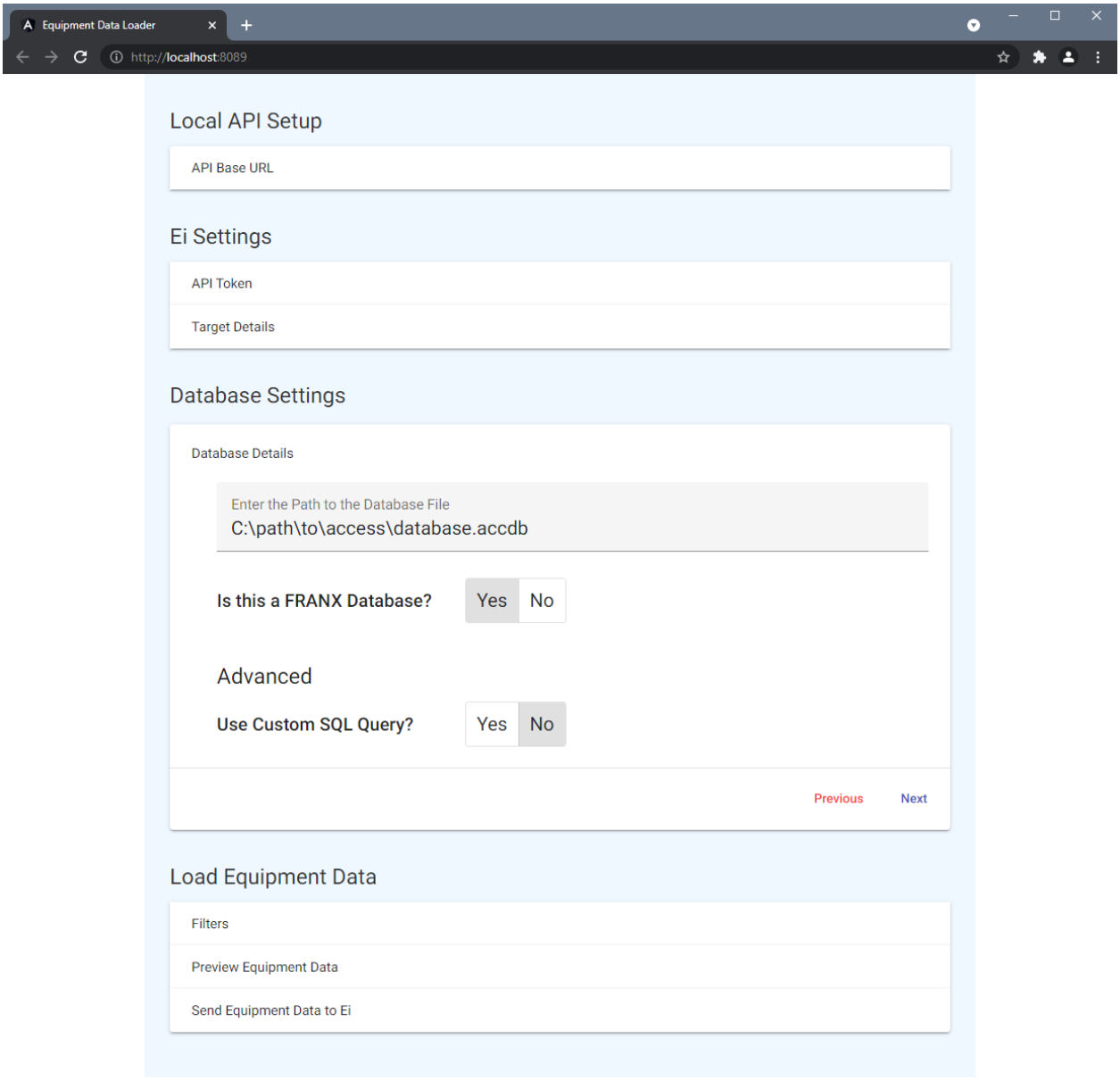

Figure 11. Screenshot of the "Database Details" section of the equipment data loader web app for a FRANX database.

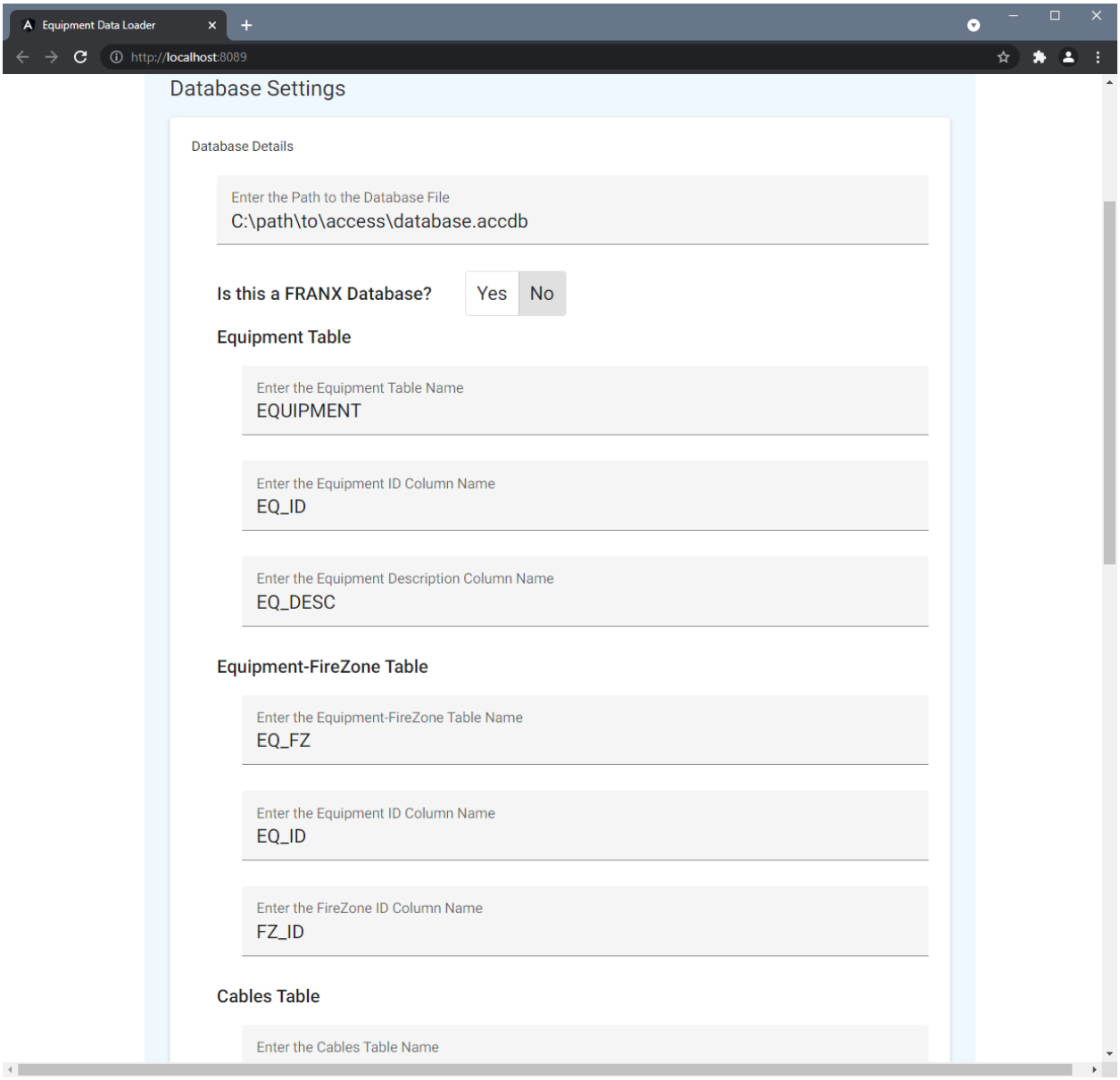

Figure 12. Screenshot of the "Database Details" section of the equipment data loader web app for a non-FRANX database.

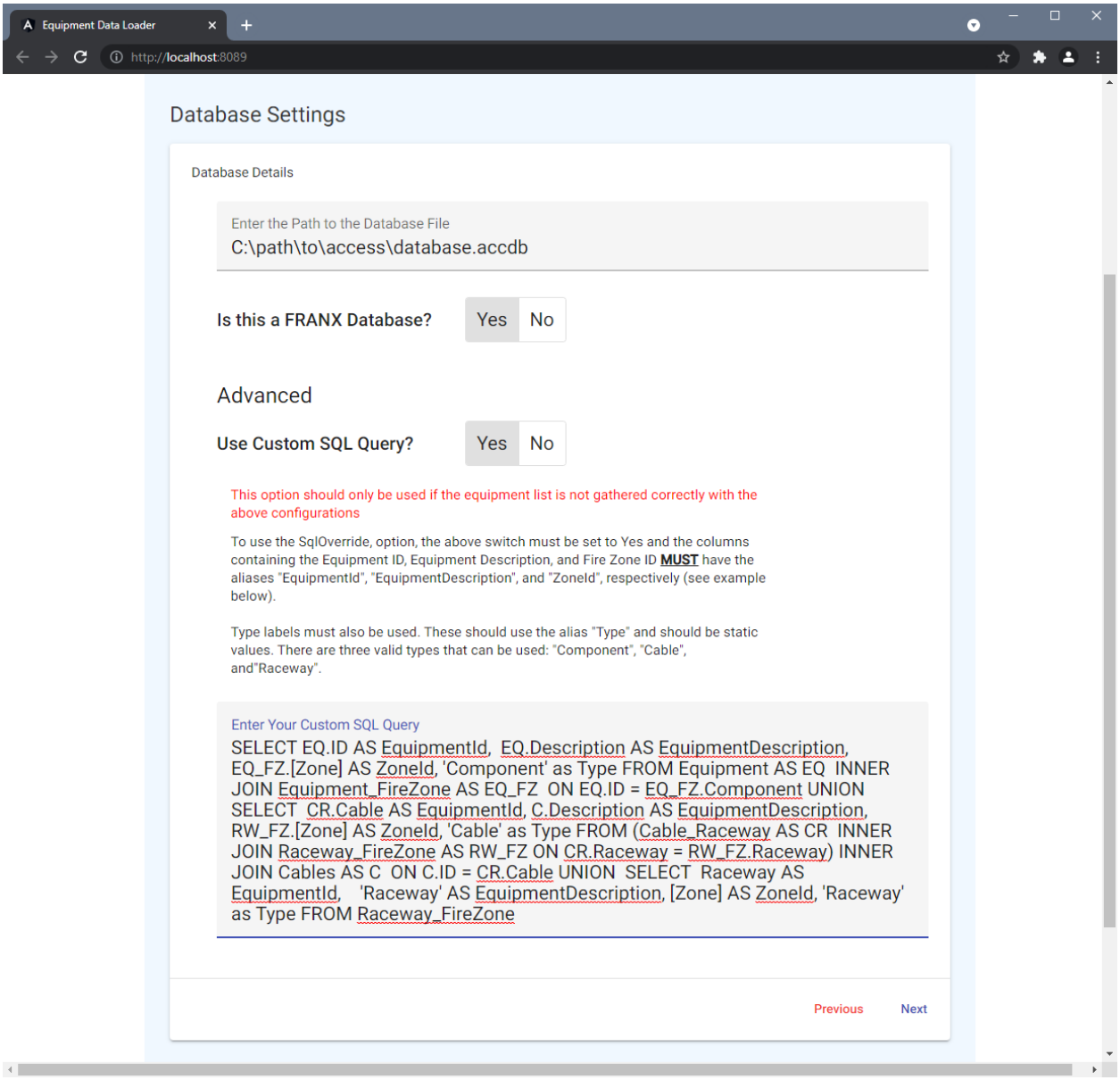

Figure 13. Screenshot of the "Database Details" section of the equipment data loader web app when using a custom SQL query.

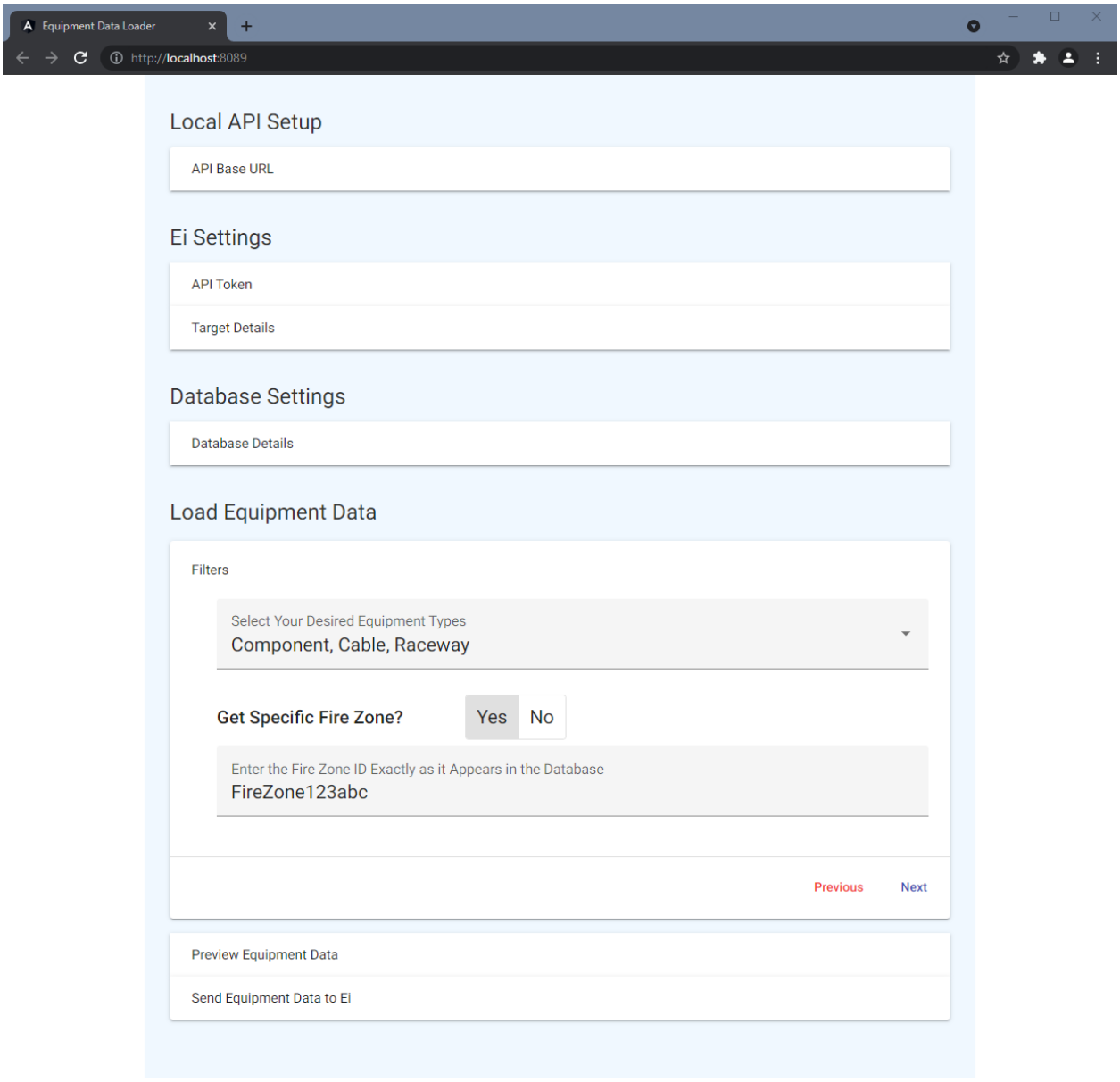

Figure 14. Screenshot of the "Filters" section of the equipment data loader web app.

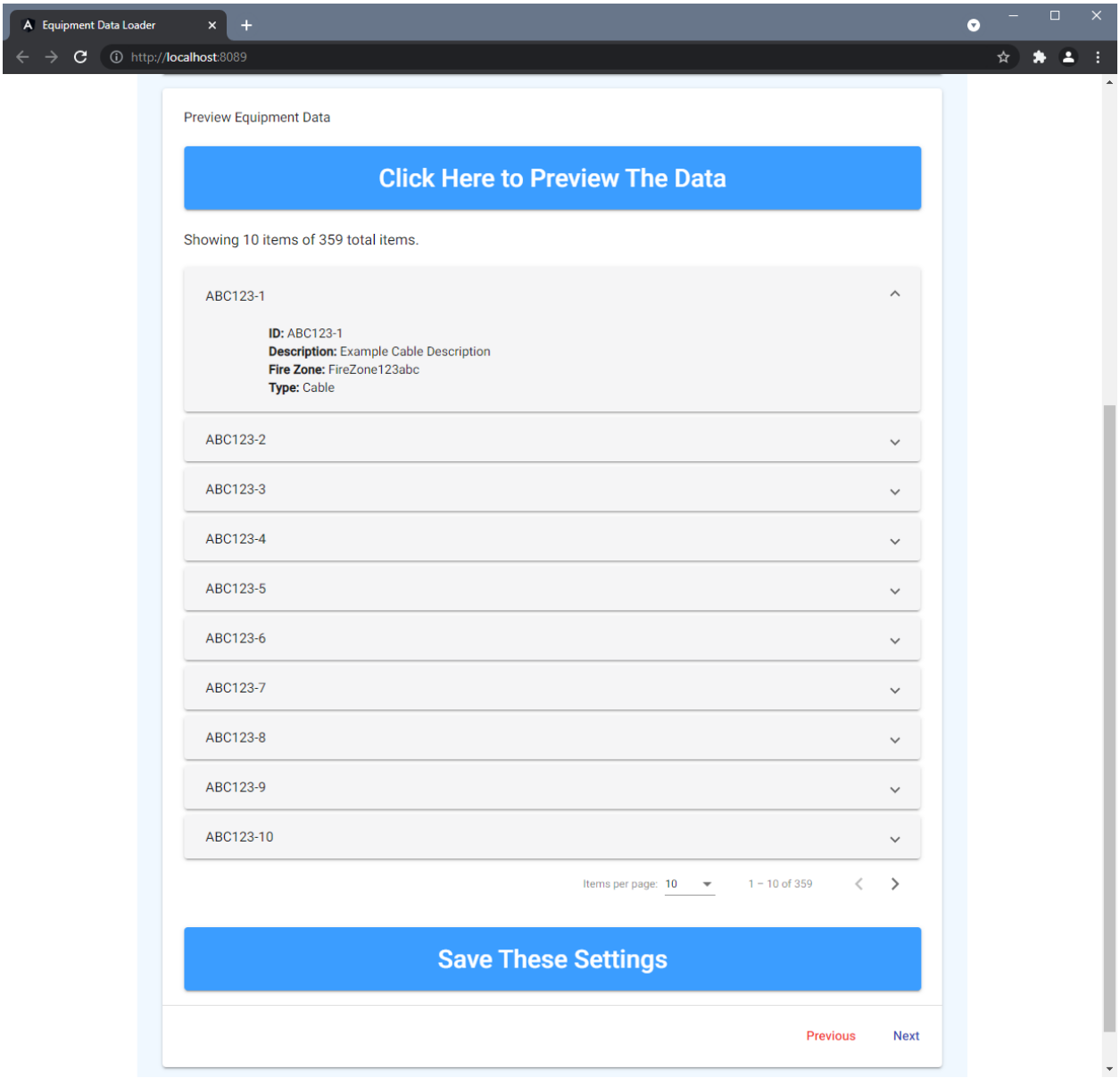

Figure 15. Screenshot of the "Preview Equipment Data" section of the equipment data loader web app.

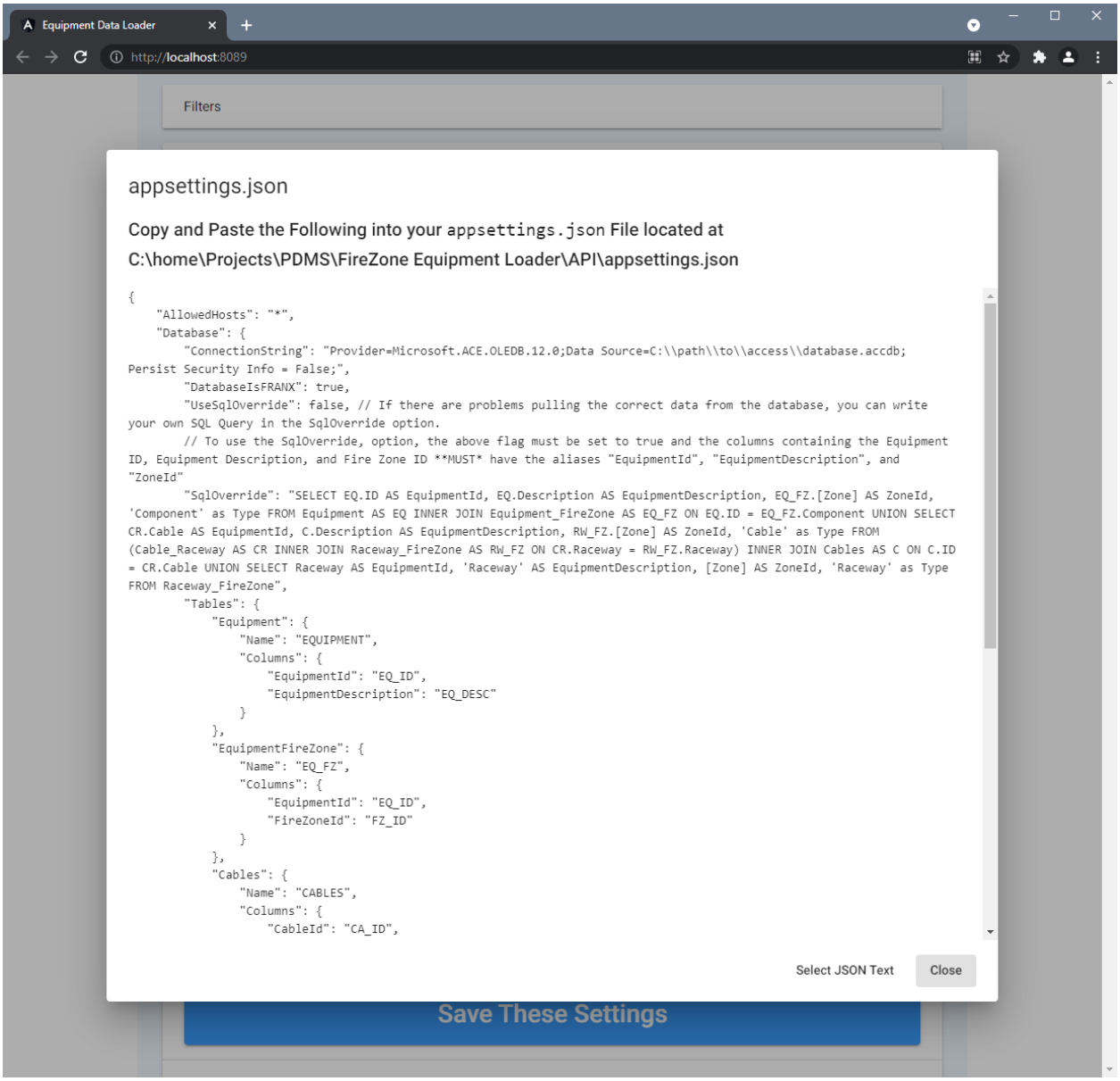

Figure 16. Screenshot of the generated JSON text modal of the equipment data loader web app.

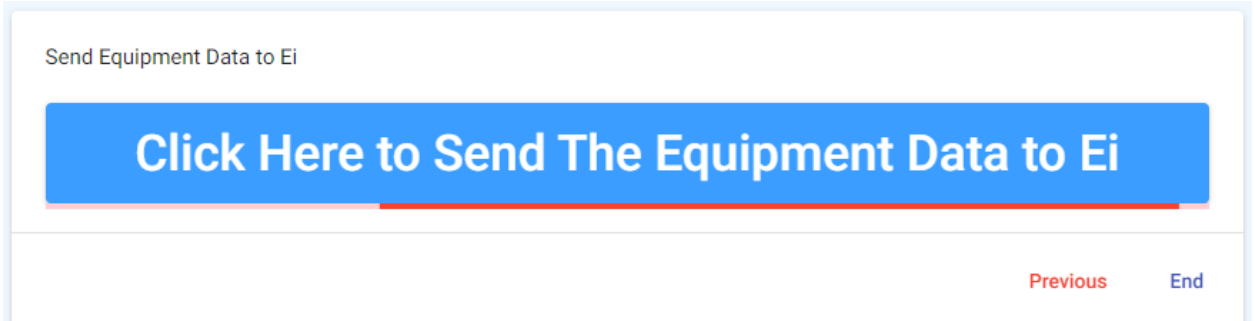

Figure 17. Screenshot of the final step of the equipment data loader web app to send data to Ei with loading bar.

3.3 Simplified Tagging Method

3.3.1 Filtering information

When the data is loaded into Ei, it will automatically have a "ZoneID" property added to it with the associated zone ID as its value. If an item is in multiple zones, the value will contain a comma-separated list of all the associated zone IDs. If an item does not have any zone associated with it, the "ZoneID" property value will be "ZoneNotSet."

Using this property, filters can be created to display only items in specific zones. This can be done by selecting the search box and clicking "Advanced Search." In the "Advanced Search" dialog, a condition can be added to look within the "Property" field. After selecting the "Property" field, the "Layer," specific property ("ZoneID"), comparison ("is" / "is not"), and desired property value can be set. Additional conditions can be added for a more specific set of results. The option is then available to "Save as Filter" to make the result set easily accessible by simply turning on that filter. These steps are shown in Figure 18.

When saving a filter, a filter name must be specified and an icon and color must be chosen. Additionally, the filter conditions can be edited from within the save dialog and a privacy option can be selected (this can be useful for sharing a filter with members of a team or keeping it for private personal use only). The save dialog is shown in Figure 19. Future work is planned to automatically create filters when loading data into Ei.

To display only the items in a filter, simply turn on that filter in the filter pane on the left side of the screen. When multiple filters are enabled, the results will be all items in any of the filters; this means if only items from a specific filter should be displayed, then all other filters must be disabled, as shown in Figure 20.

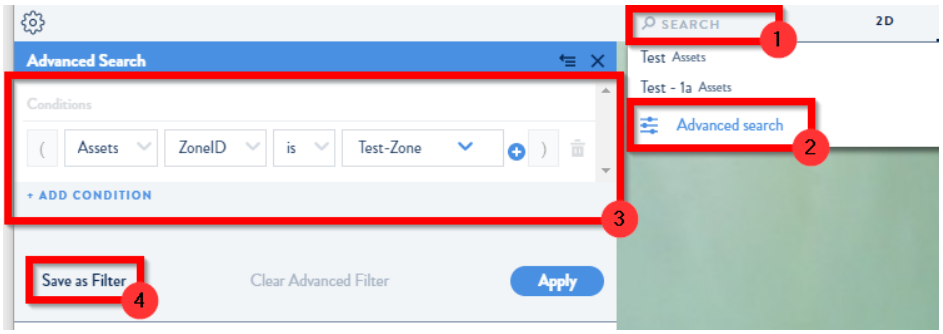

Figure 18. Steps for creating a new filter.

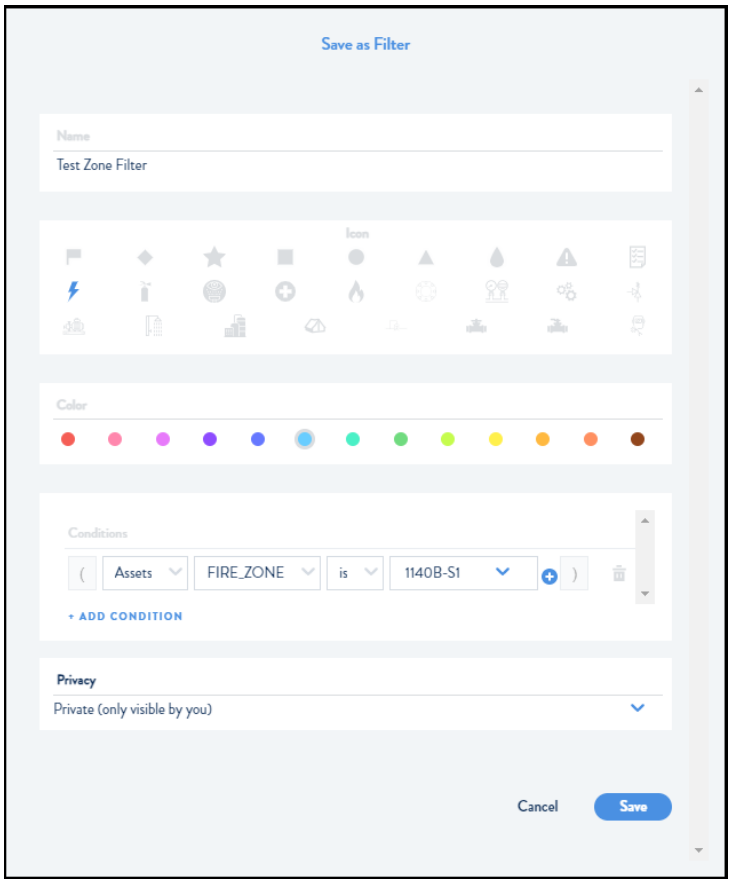

Figure 19. Save Filter Dialog.

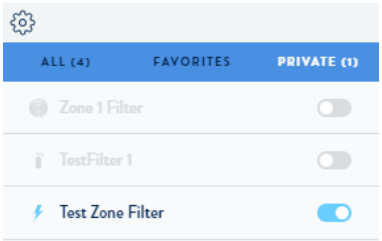

Figure 20. Only one filter enabled.

3.3.2 Tagging design

A new design has been developed to make the tagging process more intuitive and easier for nuclear power projects. This design was built around current capabilities in the Ei platform; other LiDAR software interface developers should design as desired using the data provided.

For the Ei implementation, when the user is in the 3D LiDAR view (this view is recommended for accuracy vs. picture view), objects can be placed simply by choosing an object and setting its location in the 3D view. To display a list of objects side-by-side with the 3D LiDAR view, create and/or enable any necessary filters as described in Section 3.3.1. Then, open the "Advanced Search" pane, as shown in steps 1 & 2 of Figure 18.

Once the appropriate filters have been enabled and the object list is displayed side-by-side with the 3D LiDAR view, the user can start the tagging process. The following steps outline how new object tags can be added to the 3D LiDAR view and are demonstrated in Figures 21–23.

1. The list of filtered items will appear below the filter options in the "Advanced Search" pane; select the desired object to tag, as shown in Figure 21, step 1 (in the example, the "Test3" asset is selected).

2. The item details pane should now appear on the right side of the screen, as shown in Figure 21. Select the three dots in the top-left corner of this pane to open the item's context menu, as shown in Figure 21, step 2.

3. Click the "SET LOCATION" button, as shown in Figure 21, step 3.

4. At the top of the screen will be displayed instructions for setting the item's location. Simply right-click on the picture of the object to tag it (in the example, the tagged location is shown in Figure 21, step 4).

5. After choosing a location, the top bar will change to ask the user to save the tag or cancel the tagging process. Click the "Save" button, as shown in Figure 22, step 5.

The tagging is now complete. The new tag should be displayed on the selected location in the 3D view and now, new options for the object are available, including the ability to change the

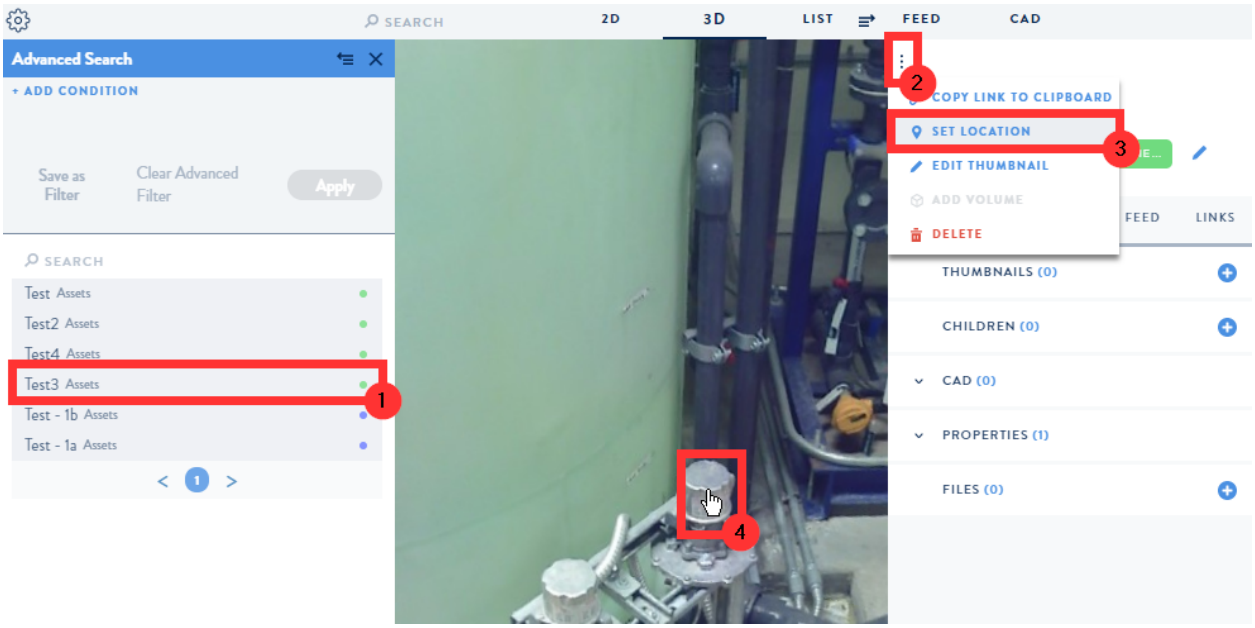

Figure 21. Steps 1–4 for the new tagging process.

Figure 22. Step 5 for new the tagging process.

item's location (the process is similar to Step 4) and locate the item, which will change the user's view to display the tagged object. The tag and new options are displayed in Figure 23.

Work is currently being planned to condense Steps 1 & 2 into a single step by having the context menu appear by right-clicking on the object in the list view and automatically saving. This would reduce the number of clicks needed by the user to 3 per item.

4. RETRIEVING SPACIAL INFORMATION

A key part of this research is to make the 3D data available for other tools to reduce modeling time and increase accuracy. Fire research and software at INL is incorporating spatial information into existing fire models. Currently, users must manually construct the 3D model using the items in the zone list. As with many modeling and simulation tools, the fire model does not require detailed shapes of items, just good location information and compartment boundaries. This data

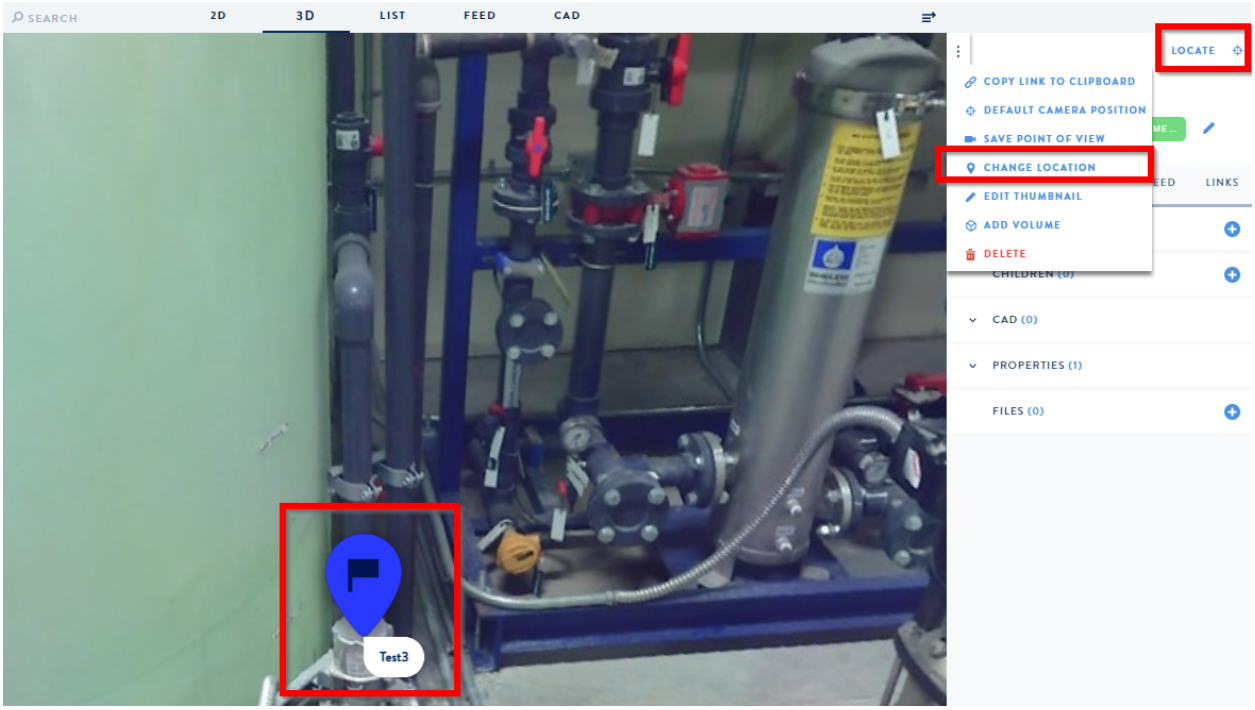

Figure 23. The placed object tag and new options for the tagged object, including "Locate" and "Change Location."

will be available through an API, which is in the initial design stages.

4.1 LiDAR Software API Requirements

It is anticipated different software platforms will have varying data and formats, so the design for retrieving data has a similar concept to that of uploading existing plant data. In this way, if an entity already has an API, only a small module will need to be customized to make it compatible. Even though it is customizable, the LiDAR platform will need to meet certain data requirements or options in order to implement an interface module successfully.

1. API calls for retrieving information on a single and set of tagged items, specified by IDs
2. Returned information must contain at least the ID/Name, position of the item/s
3. If available, information should include item bounding box information including shape type, size info, position, and quaternion
4. If data needs to be returned in batches, each data set will provide the batch range

```
[{
  "datasetName": "name",
  "pts3d": "[x,y,z]",
  "boundingBoxVolume": {
    "type": "cube",
    "depth": 1.375044407189236,
    "width": 0.7469334347255199,
    "center": [
      19.47986536718622,
      2.6500196254875075,
      0.490876226752806 84],
    "quaternion": [
      0,
      0,
      0.7776382353130729,
      0.6287119968468631]}}]
```


Figure 24. Example JSON data format from Ei's API.

5. Data points can be in local coordinates or Cartesian coordinates but must distinguish which and should be consistent for all data.

The Ei Platform provides a web API that returns a JSON array of data, as shown in Figure 24.

Other API calls are recommended as they will dramatically increase the options or capabilities available for the client tool. These are not required as they could involve substantial development from the platform. The standards for these calls will be developed.

1. Room Boundary – A call to get room or bounding box information

2. Point Cloud – Return a subset of the main point cloud model for the given criteria or area.

This research work initially focused on the required calls, but further development will be done to determine additional needs and logical formats for the optional functions.

4.2 Importing Data

The custom module interfacing with the LiDAR platform will be part of a middleware API to allow requesting software to obtain the 3D data. After obtaining the item data retrieved from the LiDAR platform, the coordinates will be adjusted using either a Cartesian offset coordinate

provided by the call or the center and bottom most point of the data. This coordinate adjustment allows the calling application to import items in relation to their model setup. It is anticipated several 3D format options will be available for importing.

4.3 Use Case Design

Along with the PDMS and FRANX models, a LiDAR scan of an existing facility's switchgear room and auxiliary feed water pump room are being used for testing of the API calls and tagging process. Using actual plant databases provides a large data set for testing scalability. The focus of the test case is to tag all cabinets, trays, and other equipment in the switchgear room and add item bounding boxes. Test code will retrieve the information for the specific room that would be needed for constructing a basic model.

Future work will involve adding a module to the FRI3D software to pull in all the items using the data retrieval API. The goal is for FRI3D to automatically place the equipment in its 3D model and make the links to the fire model and eliminate $> 90\%$ of manual modeling efforts.

5. FUTURE WORK

Several areas were investigated as additional features that could be beneficial for both tagging and external model use. The following were identified as possible future research tasks with a summary of potential benefits and challenges in the following subsections:

- Component type recognition
- Room boundary mapping
- Automatic object boundaries
- Label detection or tagging

5.1 Component Type Recognition

Modern Artificial Intelligence (AI) algorithms are effective at identifying objects in pictures after using an appropriate training set [3]. Nuclear power plants typically use a naming label or naming standard that can identify a component from the database as a specific type. By mapping

the convention to the AI type identification, items for tagging could be ranked according to the results.

The challenges for this would be first establishing a library of training set data. Many plant components such as pumps, valves, electrical cabinets, etc. are common in varying industry facilities. Exiting LiDAR models from other sectors and initial plant scans can be used to generate pictures from several different angles to develop training sets of data. Another challenge is plants having different conventions or schemes; a mapping to each component category would need to be developed for each.

While the benefits are high, the investment and risks are also high, so this task is low priority.

5.2 Room Boundary Mapping

Similar to the component location and bounding box, automatically generating the room boundary would be beneficial for external analysis. Manually constructing the room boundary for irregularly shaped rooms could require significant time; however, compared to the number of components in a room and other tasks, this should not be a large effort. Given this, the benefit is considered low.

The challenges for this task are similar to the automatic object boundaries, and how effective the algorithms are. The risk is considered low and overall priority is low.

5.3 Automatic Object Boundaries

To effectively use the model in various applications, the spatial location and dimensions needs to be exportable as described in Section 4.1. While constructing a bounding box of 3D items is not difficult, it would be the second most time consuming task in the previous process. Automatically identifying and creating a bounding box for the 3D objects is possible with current technology [4]. The goal would be to automatically create the bounding box for an item the user is tagging and then let them verify or adjust it as needed.

Possible challenges for this process would be the accuracy for oddly shaped objects such as conduits or racked items where a clear delineation cannot be obtained, but this should be a low percentage of components.

The benefits of this task are medium, and the risk is low, so this task is medium priority.

5.4 Label Detection or Tagging

Most items in a facility have labels on them with identification information. Machine learning and computer vision can identify labels and read the text [5]. Initial research has identified this as the most accurate way to assist the user in the tagging process by using that to filter or rank items and provide a spatial location.

The first step will be to do label detection, such that a user will be able to quickly click and review the label and tag the equipment in a more streamlined flow. The second step, to complement automatic label detection, is Optical Character Recognition (OCR). This will be extracting the label's text, which allows for the database items to be filtered accordingly. The user will be able to verify the correct match and tag with the simple click of a button.

Through tag identification, the goal will be to reduce the entire tagging process for a facility with an average to good labeling standard to less than 10 seconds per component.

Challenges include non legible tags caused by low resolution or obscuring, and computer vision accuracy levels. With current scan technology, risks are deemed as low with a high benefit. This task is rated as the highest priority.

Examples of labels that can be detected are shown in Figures 25 and 26.

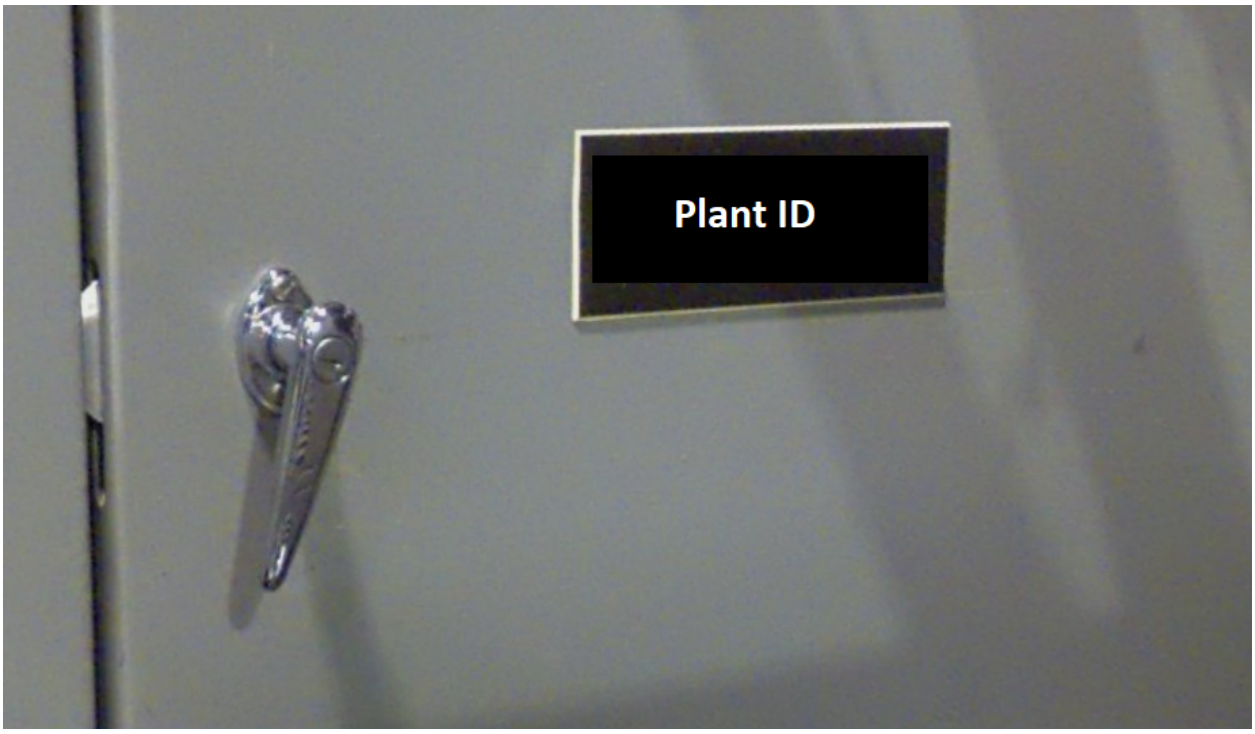

Figure 25. Example of a label for auto-detection.

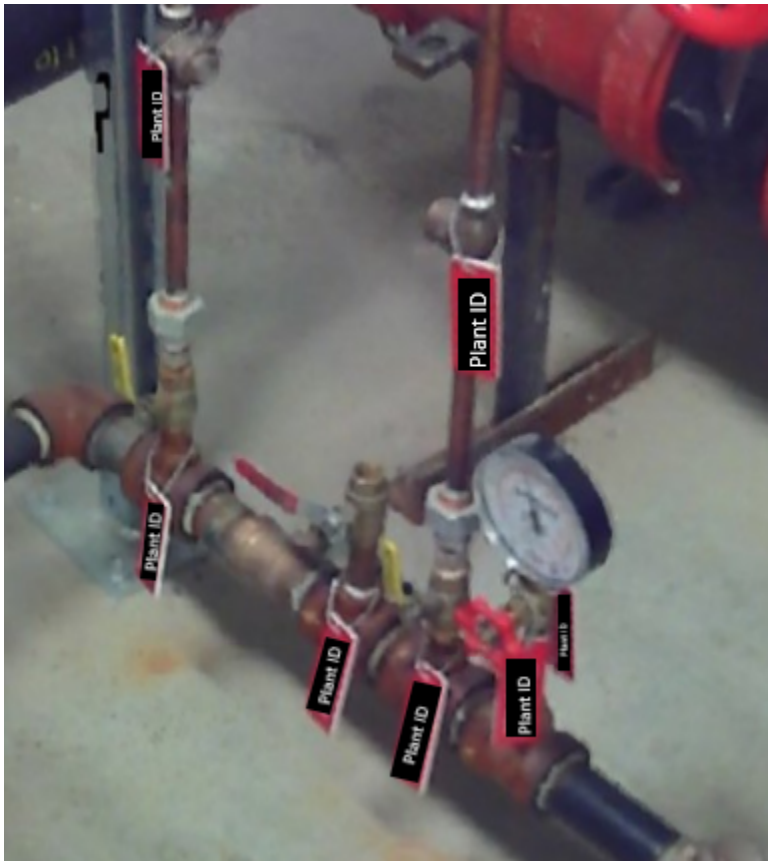

Figure 26. Example of labels for auto-detection.

REFERENCES

[1] H. Zhang, Y. Choi, C. Smith, S. Prescott, D. Mandelli, and A. Alfonsi, "Risk-informed systems analysis (RISA) pathway technical program plan," No. INL-EXT-19-55721, Idaho National Laboratory, 2013.

[2] Ei, "Ei everywhere application." https://www.env-int.com/app, n.d. Accessed on April 1, 2021.

[3] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[4] "A Steiner tree approach to efficient object detection." http://ai.stanford.edu/~olga/posters/cvpr10-poster.pdf, 2010.

[5] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng, "Text detection and character recognition in scene images with unsupervised feature learning," in *2011 International Conference on Document Analysis and Recognition*, pp. 440–445, 2011.