

---

# **TRACKIT Documentation**

*Release 17.02 build 03*

**TRACKIT**

**May 25, 2017**



---

## Contents:

---

<b>1</b>	<b>Intro</b>	<b>1</b>
1.1	About . . . . .	1
1.2	Asset Management in TRACKIT . . . . .	1
<b>2</b>	<b>Platform and Installation</b>	<b>5</b>
2.1	Installation Overview . . . . .	5
2.2	Cloud Installation . . . . .	5
2.3	Local Installation via Web . . . . .	5
2.4	Local Offline Installation . . . . .	10
<b>3</b>	<b>Layout</b>	<b>11</b>
3.1	Interface Overview . . . . .	11
3.2	Header Menu . . . . .	11
3.3	Sidebar . . . . .	12
3.4	Dashboard . . . . .	12
3.5	Location/Asset Breadcrumb . . . . .	12
3.6	Location/Asset Information Tabs . . . . .	13
<b>4</b>	<b>Wizards</b>	<b>15</b>
4.1	Wizard Overview . . . . .	15
4.2	Location Wizard . . . . .	16
4.3	Product Wizard . . . . .	17
4.4	Asset Wizard . . . . .	19
4.5	EAV Wizard . . . . .	22
<b>5</b>	<b>EAV (Entity Attribute Values)</b>	<b>23</b>
5.1	EAV Overview . . . . .	23
5.2	EAV Groups . . . . .	23
5.3	EAV Attributes . . . . .	24
<b>6</b>	<b>Locations</b>	<b>27</b>
6.1	Locations Overview . . . . .	27
6.2	Location Details . . . . .	28
6.3	Edit Floorplan . . . . .	28
6.4	Edit Borderpath . . . . .	29
6.5	Regions . . . . .	29

<b>7</b>	<b>Assets</b>	<b>31</b>
7.1	Asset Overview . . . . .	31
7.2	Placement Types . . . . .	31
7.3	Adding Assets . . . . .	32
7.4	Editing Assets . . . . .	34
7.5	Deleting Assets . . . . .	35
<b>8</b>	<b>Products</b>	<b>37</b>
8.1	Product Overview . . . . .	37
8.2	Maintaining Categories . . . . .	37
8.3	Maintaining Manufacturers . . . . .	37
8.4	Maintaining Models . . . . .	38
<b>9</b>	<b>Reports</b>	<b>39</b>
9.1	Reporting Overview . . . . .	39
9.2	Tabular Reports . . . . .	39
9.3	Saved/User Reports . . . . .	43
9.4	Trend Reports . . . . .	43
9.5	Dashboard Reports . . . . .	43
<b>10</b>	<b>Users and Groups</b>	<b>47</b>
10.1	Permissions Overview . . . . .	47
10.2	Managing Users . . . . .	47
10.3	Maintaining Groups . . . . .	48
<b>11</b>	<b>Mobile Apps</b>	<b>49</b>
11.1	TRACKIT Mobile Overview . . . . .	49
11.2	TRACKIT Mobile App . . . . .	49
11.3	TRACKIT Mobile . . . . .	49
<b>12</b>	<b>Administration</b>	<b>51</b>
12.1	Admin Overview . . . . .	51
12.2	Administration Options . . . . .	51
<b>13</b>	<b>Monitors and Metrics</b>	<b>55</b>
13.1	Monitors and Metrics Overview . . . . .	55
13.2	Configure a Monitor and Metric . . . . .	56
<b>14</b>	<b>Integration</b>	<b>59</b>
14.1	Integration Overview . . . . .	59
<b>15</b>	<b>Indices and tables</b>	<b>63</b>

New in version 16.02.

Changed in version 17.02.

## About

TRACKIT is an Asset Management tool, designed to be delivered via the Cloud as a Software as a Service (SaaS) application. It can be deployed locally and installed on a local virtual or physical machine within a Customer environment.

## PRO Version

TRACKIT PRO is an Enterprise level version of TRACKIT offering additional functionality such as a Scalable IT Equipment Library, Network and Power Connectivity and cable tracing. In addition, there are many integrations to Enterprise platforms built and configured by TRACKIT Service Partners available.

## Asset Management in TRACKIT

TRACKIT has been designed to provide an accurate, easily maintainable inventory of Locations and Assets. This could be any size, such as a Small Office environment or Store Room, through to a Large Enterprise scale Data Centre. Applications include Vehicle Fleet Management, IT Asset Management, Warehouse Stock Inventories or Mobile Equipment Monitoring.

## Locations

Locations are designed to contain either additional child Locations, such as a Building containing Floors and Rooms, and within these Assets can be placed. Locations could be physical such as a Building, or logical, such as an organisational unit or department.

## Assets

Assets are intended to be anything which requires managing, all Assets have a history, can be added, modified, removed. Assets can be placed within Locations or other Assets using various placement or mount types. To create an Asset, a Product must be created which determines all the physical attributes of the Asset.

## Products

Products are templates or symbols of any given Asset usually grouped by the Manufacturer of the Product and the Model of the Asset. They include, but are not limited to physical attributes, such as dimensions and weight and could include information such as graphical images, interfaces such as ports and connectivity or financial details such as cost or warranties. Anything could be defined against a Product.

Products should belong to an Asset Class, and can belong to more than one class if required.

## Asset Class

Products belong to an Asset Class, the following classes are system classes and define *Placement Type*, they should not be removed:

- Floor Mountable - Allows Assets to be mounted on a Floor
- Rack/Cabinet Mountable - Allow Assets to be mounted within a Rack/Cabinet Asset with U Space (for IT cabinets).
- Device/Instance Mountable - Allow Assets to be mounted within other Assets.
- Blade Servers - Allow Assets to be mounted within Asset Slots (for IT Assets)

The remaining are default classes, which can be altered or removed if required:

- IT equipment
- Software and Licenses
- Furniture and fixtures
- Machinery
- Office equipment
- Vehicles
- Intangible assets

Asset Classes determine some behavior of Assets within TRACKIT, for instance, how Assets can be mounted in Locations such as on the Floor, within other Assets or whether they can be mobile or not.

## Placement Type

To ensure Assets are placed correctly, the Product Model must belong to an appropriate system *Asset Class*.

It is possible for a Product to belong to more than one Asset Class, therefore ensure that the correct class is chosen to determine the placement type, and add any additional classes for reporting/grouping as required.

## Categories

Categories are the type of Asset which a Product belongs to, the Product must belong to a Sub Category, which belongs to a Parent Category.

For example, if a Car was being added to TRACKIT, the following configuration would be used:

- Manufacturer: Ford
- Model: GT Mustang
- Asset Class: Vehicles
- Category: Car
- Sub-Category: Performance

Another example, such as a Computer could use the following setup:

- Manufacturer: Hewlett Packard
- Model: Envy 15-AS104NA
- Asset Class: IT equipment
- Category: Computers
- Sub-Category: Laptops

## Additional Data

Any additional data can be added to TRACKIT either as additional data fields, called *EAV Overview* and imported by the User or through an Integration to another product. Examples of additional data are:

- Procurement Information, Warranties, Purchase Dates, Asset Values
- GPS Information, visually showing where an Asset is on a live map or showing its path over time.
- Power Information - Current and historic power consumption of an Asset from intelligent data sources.
- Environmental Information - Power, Heat, Humidity information from a Building Management System



New in version 16.02.

Changed in version 17.02.

### Installation Overview

Depending on the deployment, TRACKIT will have been either setup in the Cloud automatically, and can be accessed via a personalised url, i.e. <https://mycompany.trackit.cloud/>, or it will have been installed locally via a TRACKIT Engineer. For details on local installations please see [installation\\_steps](#).

Please contact your Administrator if you are unsure which installation type you have.

### Cloud Installation

There are no installation steps required, the application link will be delivered via email.

### Local Installation via Web

#### Hardware Pre-requisites

The TRACKIT architecture is container based. Containers can be placed on the same server or node, or split across multiple nodes. Typically, TRACKIT runs on Linux nodes, with a dedicated Reporting node running Windows. TRACKIT is database and web server agnostic, however we highly recommend PostgreSQL or Oracle as a database platform.

## Optimum Specification

The following is a recommended node specification for running the TRACKIT application utilising all features and functionality in a multiple location/user environment.

- Intel Xeon E5-2440 hexa-core 2.4 GHz or equivalent processor (dual processor if supported by chassis)
- 16GB DDR3 Memory
- 144GB HDD RAID1 minimum recommended
- Dual Gigabit NIC

Recommended Nodes:

- 2x Linux Based OS (Ubuntu 16.04) nodes
- 1x Windows Based OS (Windows 2016 Server Edition) node

External VPN Access for support/maintenance purposes (recommended)

Installation requires an internet connection, if this is not available please see the *Local Offline Installation* section for further details.

## Docker

TRACKIT uses Docker as the platform container.

Information on Docker can be found here: <https://www.docker.com/> and a list of support Operating Systems can be found here: <https://www.docker.com/community-edition>

## Installation Steps - Linux Node

### Install Docker

Install Docker for Ubuntu, instructions found here: <https://docs.docker.com/engine/installation/linux/ubuntu/>

Install Docker as per standard Ubuntu instructions. Ensure you add your system user to the docker group:

```
usermod -aG docker <username>
```

You will need to logout and back in again after doing so.

### Configure to AWS

Authorising to TRACKIT Image Repo. Firstly, install AWS CLI tools.

```
sudo apt-get install awscli
```

Configure with the relevant access keys, setting the default region to eu-west-1.

```
aws configure
```

Log Docker into the repo:

```
`aws ecr get-login --region eu-west-1`
```

## Install Database server (PostgreSQL)

PostgreSQL should be run on your native machine/VM:

```
sudo apt-get install postgresql
sudo -u postgres psql

create user trackit with password 'tr4ck17';
create database trackit with owner trackit;
```

Also install the PostGIS extension, currently this is 2.2, but adjust this for your Ubuntu version.

```
sudo apt-get install postgresql-9.5-postgis-2.2
sudo -u postgres psql
create extension postgis;
```

Enable local TCP connections to postgres:

```
sudo nano /etc/postgresql/9.5/main/postgresql.conf
```

Find `#listen_addresses="localhost"` replace with: `listen_addresses="*"`

```
sudo nano /etc/postgresql/9.5/main/pg_hba.conf
```

Add the following line to enable connections from docker IPs

```
host all all 172.0.0.0/8 md5
sudo service postgresql reload
```

## Create Docker Swarm

Once the native postgres instance is set-up, we can need to create a swarm and overlay network for our services to communicate across.

```
docker swarm init
docker network create --driver overlay trackit
```

Then we can proceed to create services.

- Pull Images (Repeated Step)

There is a convenience script in trackit-docker bitbucket repo, path `scripts/pull-from-ecr.py`. This will pull the relevant images and tag them automatically. It can be repeatedly used.:

```
vim ./pull-from-ecr.py
chmod +x ./pull-from-ecr.py
./pull-from-ecr.py
```

- App (Single Node Dev)

[Optional] Setup Dev environment. You can use your local trackit code and virtualenv. This is highly recommended. (Ensure your SSH key has been added to bitbucket)

Create SSH key for Bitbucket

Enter the following commands

```
ssh-keygen
```

Select default options

```
cat ./home/ubuntu/.ssh/id_rsa.pub
```

Copy the key and add to Bitbucket account.

Then

```
sudo mkdir /trackit
sudo chown <youruser> /trackit
cd /trackit
git clone git@bitbucket.org:trackit/trackit8.git
cd /trackit8
git checkout dev
git submodule update --init
git submodule foreach git checkout dev
sudo apt-get install python-virtualenv
virtualenv -p python3 /trackit/env-py3
source /trackit/env-py3/bin/activate

sudo apt-get -y install libopenblas-dev libfreetype6-dev build-essential
↳ postgresql-server-dev-9.5 libcurl4-openssl-dev libxml2-dev libxslt-dev
↳ libjpeg-turbo8-dev linux-kernel-headers python3-dev

pip install -r /trackit/trackit8/pip-requirements.txt
```

This will give you two folders on your machine (/trackit/trackit8 = dev code, /trackit/env-py3 = virtualenv)

```
docker service create --name app --publish 8000:8000 --network trackit -eTRACKIT_DB_
↳HOST=<your host IP> -eTRACKIT_DB_USER=trackit -eTRACKIT_DB_PASS=trackit -eTRACKIT_
↳DB_NAME=trackit -eDEVSERVER_OPTS="--settings=trackit.debug_settings" [--mount
↳type=bind,source=/trackit/trackit8,destination=/trackit/trackit8 --mount type=bind,
↳source=/trackit/env-py3,destination=/trackit/env-py3] trackit/appdev /bin/runserver.
↳sh
```

Square bracket part is optional, and will bind mount your local development folders to the docker environment.

/bin/runserver.sh executes the django manage.py runserver script. If you do not specify this command, it will launch using UWSGI.

Devserver is bound to port 8000 on the host machine.

A convenience manage command is supplied within the image, which will activate the virtualenv and run manage.py, passing through any command line arguments.

## Create the Services

- Create the Broker

```
docker service create --name broker --network trackit trackit/brokerdev
```

- Queue

Enter the following

```
docker service create --name queue --network trackit -eTRACKIT_DB_HOST=<your host IP>
↪-eTRACKIT_DB_USER=trackit -eTRACKIT_DB_PASS=trackit -eTRACKIT_DB_NAME=trackit --
↪mount type=bind,source=/trackit/trackit8,destination=/trackit/trackit8 trackit/
↪queuedev
```

- Cleanup Queue

Enter the following

```
docker service create --name cleanupqueue --network trackit -eTRACKIT_DB_HOST=<your_
↪host IP> -eTRACKIT_DB_USER=trackit -eTRACKIT_DB_PASS=trackit -eTRACKIT_DB_
↪NAME=trackit --mount type=bind,source=/trackit/trackit8,destination=/trackit/
↪trackit8 trackit/namedqueuedev
```

- Elasticsearch

Enter the following

```
docker service create --name elasticsearch --network trackit trackit/elasticsearchdev
```

Install Docker as per standard Ubuntu instructions. Ensure you add your system user to the docker group:

```
usermod -aG docker <username>
```

You will need to logout and back in again after doing so.

## Installation Steps - Windows Node

### Install Reporting App (Logi)

Install the Logi Reporting Service.

### Decompress the Archive

Unzip the file, and add the Reporting folder to the appropriate location on the disk. i.e. C:\trackit8reporting

### Add Application

Within Logi, add an application, point to the Reporting folder and follow the Wizard. If appropriate, complete the IIS Install.

## Install IIS

Install the IIS Role on the Windows machine, ensuring ASP.NET is checked.

## Check Settings

Check the Settings file under `C:\trackit8reporting\_Definitions\_Settings.lgx`, and ensure that the IP address of the Linux Application nodes have been added under `AuthenticationClientAddresses`.

Once complete, browse to `http://localhost/trackit8reporting` to check the service has installed correctly.

## Local Offline Installation

Should TRACKIT be installed into a secure environment where internet connectivity is not possible, TRACKIT provide an offline installer package. Please contact TRACKIT directly for assistance.

New in version 16.02.

Changed in version 17.02.

## Interface Overview

TRACKIT has a responsive layout, provided the browser is modern and supports WebKit, users can access the application. On entering a valid username and password into the login screen, users will be greeted with the standard application interface detailed below.

- Header Menu
  - Global Search
  - View Layout
  - Account Settings
- Sidebar
- Dashboard
- Location/Asset Breadcrumb
- Location/Asset Information

## Header Menu

The Header Bar is a persistent feature that appears at the top of the application throughout TRACKIT.



Fig. 3.1: How the Header Menu appears within TRACKIT.

## Global Search

This function allows users to search for Assets, Locations and Products. The results are returned in a list with the most relevant matches first. It's possible to refine the search using the Asset Type filter in the *Sidebar*.

## View Layout

This alters the view of the *Location/Asset Information Tabs*.

Sections can either be displayed as Tabs, as Horizontally or Vertically split sections on a single page, or a specific Image/Floorplan and Asset Grid view is available.

## Account Menu

Provides access to user settings, Administration (with appropriate permissions)

## Sidebar

The Sidebar contains controls and tools relevant to the main content. It can be hidden or pinned to maximise content space if required. Depending on permissions and content, the following features may be available:

- Add/Edit/Delete Asset
- Location Tree
- Product Search
- Asset Grouping
- Date Range Selectors

## Dashboard

The Dashboard is generally displayed as the landing page of TRACKIT once a user logs in. Typically this includes widgets and a Location Map.

## Location Map

Displays a global map (external Google access required) of all locations with address details.

## Location/Asset Breadcrumb

The breadcrumb beneath the Header Bar on most content pages within the application. It displays the current location within the location hierarchy and all the parent and grandparent objects. It's possible to drill further into child objects.

All objects displayed within the breadcrumb are hyperlinked and can be selected and opened, either in a new tab should the browser support it, or directly in the current window.

## Location/Asset Information Tabs

This area contains all the content and information stored against either the Location or Asset selected.

Using the *View Layout*, the user can swap between a tabbed layout, a horizontal layout or a vertical layout. There is also an option to view only the SVG element and the Asset Grid.

Depending on the information being stored, the following sections are available:

### Details

This section shows simple Asset or Location details, such as object name, code, asset/serial numbers and summary details relevant to the object, for instance an asset count for Locations.

### Assets

A table of all assets within either the Asset or Location selected. Any child assets of the listed assets can be viewed by expanding the row to show a nested table.

This view can be ordered by selecting a column header, filtered by using the text box immediately above the table and if the relevant permissions are available assets can be edited inline.

### Reports

Reports appropriate to the selected object are listed in this section. For example, if a Location this would be a Full Asset List and an Audit Trail report. Any reports built by and/or shared by Users are listed here also.

Reports can be run by simply selecting the report name, where date ranges are appropriate, some pre-selected ranges are listed adjacent to the report name.

### History

The entire audit log of either a Location or an Asset is viewable in this section. There are date selectors to choose the date range to search between.

The information available includes:

- Client Type - How the update was made, via the web interface, or via an API (i.e. an integration to another system)
- Action Type - Was the update modifying an object, or adding or removing an object.
- Timestamp - The server timestamp of when the change was made to the database.
- Model Type - What type of object was updated.
- User - The username of the User who made the update.
- Server Host Name - Name of the server which registered the change.
- IP Address - The IP of the client from which the change was made.

Where a modification to existing data has occurred, a hyperlink is available which can be selected to show the fields modified and the before and after values.

## Floor Plan

This section requires the object selected to be a Location. If a floorplan has been defined, it can be visualised here.

The floorplan may include layers from an AutoCAD dwg, an Adobe PDF, a Microsoft Visio or from an SVG. It may include a defined floor space, which may be gridded.

Floorplans can be dragged to pan around, zoom in and out of and layers switched on and off.

It's also possible to drag and drop Products which can be mounted on the floor into the correct physical location in this view.

## Image

If the object selected is an Asset and an SVG image is available, this would be visualised here.

## Photos

Photos of Assets can be uploaded through either a web browser, or are added automatically via the TRACKIT Mobile App.

## Notes

Notes added by Users can be viewed in this section. Each note includes the User, Timestamp and note content.

## Tasks

Any tasks, either current or historical are listed under this tab.

## Monitoring

If any Monitoring or Metric information have been linked to either an Asset or Location, it will be viewable here. Metrics can be selected from the list and the data values and their respective details can be viewed on the subsequent page.

## Alerts

If any Alerts have been configured against any Metrics linked to the object, they will be shown in this section. There is an option to toggle between viewing Archived or Current Alerts only.

## Tracking

Depending on connectivity, should any GPS data have been entered or populated against an Asset, it will be shown visually on a map in this section. On an individual Asset, this would include a time slider at the top of the map allowing the User to alter the range within which to display the GPS data.

If a Location has been selected containing multiple Assets, the last data point received for each Asset holding GPS data will be shown on the map, with labels identifying each point.

New in version 16.02.

Changed in version 17.02.

## Wizard Overview

There are a number of wizards in TRACKIT to assist users in completing common tasks and importing existing data.

Wizards can be accessed from the *Account Menu*.

On selecting the Wizards menu, a dashboard is displayed summarising each section:

### Users and Groups

Number of Users	10 Users
Number of Groups	12 Users
<a href="#">Configure Users/Groups (Advanced Admin)</a>	

### Locations

Number of Location Types	14 Location Types
Number of Locations	31 Locations
<a href="#">View/Edit Location Types</a> <a href="#">View/Edit Locations</a> <a href="#">View/Edit Location Structure</a>	

### Product Models

Number of Categories	332 Categories
Number of Manufacturers	928 Manufacturers
Number of Product Models	11237 Product Models
<a href="#">View/Edit Categories</a> <a href="#">View/Edit Manufacturers</a> <a href="#">View/Edit Product Models</a>	

### Custom Attributes

Number of Asset Attributes	8 Asset Attributes
Number of Location Attributes	4 Location Attributes
<a href="#">View/Edit Attributes</a>	

### Assets

Number of Assets	5159 Assets
<a href="#">View/Edit Asset Groups</a>	

Fig. 4.1: A view of the Wizard Overview.

## Location Wizard

TRACKIT has the ability to define locations and structure these however required. A Location could be a physical one, such as a Building or a Cupboard, or a Logical container, such as a Department or Virtual Container. The following sections describes how each Location attribute works.

### Locations

The Location Wizard allows the User to search for, add, view, edit and delete locations.

It displays the following information:

- Location Name
- Location Type
- Root Location - Whether the Location can be the top most parent in the Location Tree
- Allow Assets - Whether it's possible to add Assets to the Location
- Children - When the Location has child Locations, and a link to view any.
- Parent - The name and a link to the Parent Location.

### Location Types

Locations Types can be configured to help group, search or define different types of Locations.

Location Types contain the following fields:

- Name
- Attributes - the pre-existing fields available
- Allowed child types - which Location types can be added as children of this Location type, for instance a Building would allow a Room child type, but a Room would not allow a Building to be a child.
- Allow Device Placement - can Assets be placed within this Location, or is it effectively just a structural or group for other Locations.
- Require Postal Information - this Location Type requires Users to add an address when creating Locations using this type.
- Map Marker Colour - select the colour for this Location Type to be shown where mapping is viewable.
- Marker Icon - no longer relevant.

### Location Tree

This gives an overview of the current Location structure within TRACKIT.

It is possible to use this view to amend the Location structure.

## Product Wizard

### Categories

Categories are listed alphabetically, with all child Categories shown alongside each.

It is possible to add new, edit existing or remove Categories.

### Manufacturers

It is possible to search for, add, modify or remove Product Manufacturers from this wizard.

Information about Manufacturers are shown in this wizard, they include:

- Name
- Alias - Other known names for the Manufacturer
- Models - The number of Product Models using the Manufacturer

By selecting a Manufacturer, it's possible to view all Models and edit each directly if required.

### Models

This wizard provides the ability to search for (either Simple or Advanced), add, modify and update Product Models within TRACKIT.

The Advanced Search function allows the User to search across multiple Manufacturers, Product Families and/or Asset Classes.

### Import Models

This Wizard allows Users to import a list of Models via a text based CSV file. There is no fixed format to this file, it can be defined by the User and any columns contained in the file can either be matched to a field in TRACKIT or ignored.

There are 7 stages to this import.

1. Choose File - Select and upload a CSV file from the Users local machine.
2. Match Columns - TRACKIT will identify the columns of data contained in the file and prompt the User to match them to a Model field within TRACKIT.
3. Match Asset Classes - TRACKIT will attempt to match Asset Classes to existing Classes within TRACKIT or request the User make the match.
4. Match Manufacturers - TRACKIT will attempt to match the Manufacturer names in the CSV to existing Manufacturers in TRACKIT. It is not possible to create new Manufacturers automatically.
5. Match Categories - TRACKIT will attempt to match Categories to existing Categories in TRACKIT. It is not possible to create new Categories automatically.
6. Validate - TRACKIT will review the data to see if there is likely to be any duplication or conflict should the data be loaded. Any warnings are displayed.
7. Import - Any data which Validated successfully will attempt to be imported. Links to successfully loaded Models are created and displayed, any errors are listed at the bottom.

Completing this Wizard prior to importing an existing Asset Inventory is highly recommended.

## Match Columns

**i** Match the CSV columns to supported asset fields.

**CSV Columns:**

TID, Category, Asset Class, ID, Manufacturer, Model, Height

Tid

External Id

Category

Ignore

Fig. 4.2: An example of an analysed CSV.

## Match Asset Classes

Match the asset classes to the inputs in the CSV.

Goods

Goods

Fig. 4.3: TRACKIT attempts to match values based on names, this can be overridden.

## Validated data

**i** These product models are ready to import.

**x** Errors

- Line number 1 - Model with this Manufacturer and Name already exists.
- Line number 2 - Model with this Manufacturer and Name already exists.
- Line number 3 - Model with this Manufacturer and Name already exists.
- Line number 4 - Model with this Manufacturer and Name already exists.
- Line number 5 - Model with this Manufacturer and Name already exists.

Fig. 4.4: Any validation errors or warnings are shown.

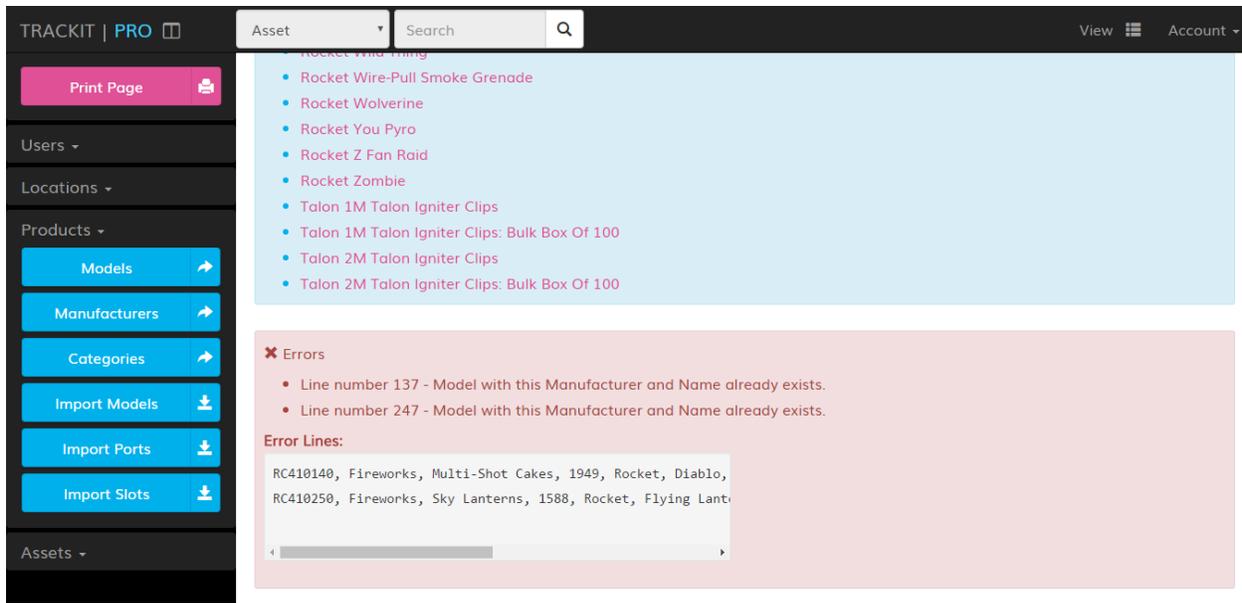


Fig. 4.5: An example of a Imported result set.

## Asset Wizard

### Asset Groups

This form allows the User to view, edit or remove existing Asset Groups or define new ones.

Each Asset Group has a colour defined, which is used in various parts of the application to help identify specific Asset Groups.

### Asset Group Types

The Asset Groups Types Wizard provide a form to create the Types available for Asset Groups.

### Asset Import

This Wizard allows Users to import a list of Assets via a text based CSV file. There is no fixed format to this file, it can be defined by the User and any columns contained in the file can either be matched to a field in TRACKIT, ignored or a new EAV field created.

There are 7 stages to this import.

1. Choose File - Select and upload a CSV file from the Users local machine.
2. Match Columns - TRACKIT will identify the columns of data contained in the file and prompt the User to match them to an Asset field within TRACKIT, ignore the field, or create a new field.
3. Match Locations - TRACKIT will attempt to match Location data to existing Locations within TRACKIT using the name. It's possible to load Assets into the system, but against a PLACEHOLDER Location. This is not recommended, Locations should be defined first.

## Match Columns

**i** Match the CSV columns to supported product model fields.

**CSV Columns:**

Location, TID, Category, Asset Class, Manufacturer, Model,

Location

Main Location

Tid

New field

Fig. 4.6: An example of an analysed Asset CSV.

## Set Locations

**i** In order to proceed we need some more information from you:  
For each location found in the CSV, match it to a location in the database, or set it to a "blank" import location to be sorted later.

Pike

pike

W: Pike - R: Ex Strong Virginian - None

Next →

Fig. 4.7: TRACKIT attempts to match values based on names, this can be override.

- Match Models - TRACKIT will attempt to match the Product Models in the CSV to existing Product Models in TRACKIT. It is not possible to create new Models automatically, so Models should be loaded into the application prior to running this wizard.

## Set Product Models: 59

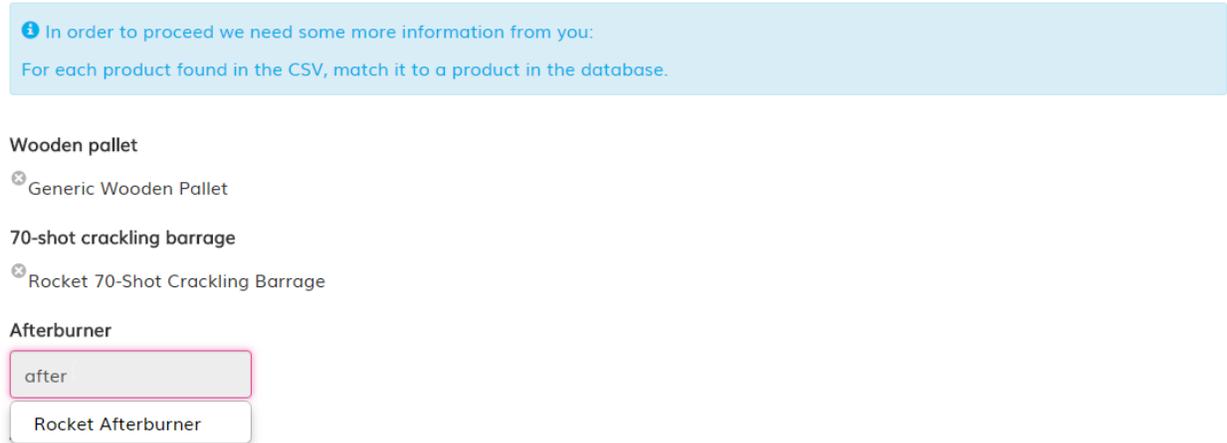


Fig. 4.8: TRACKIT attempts to match values based on names, this can be overridden.

- Match Parent Assets - TRACKIT will attempt to match the Parent Assets based on the earlier selected Parent field. Parents should be included in the CSV data.
- Validate - TRACKIT will review the data to see if there is likely to be any duplication or conflict should the data be loaded. Any warnings are displayed.

## Validate

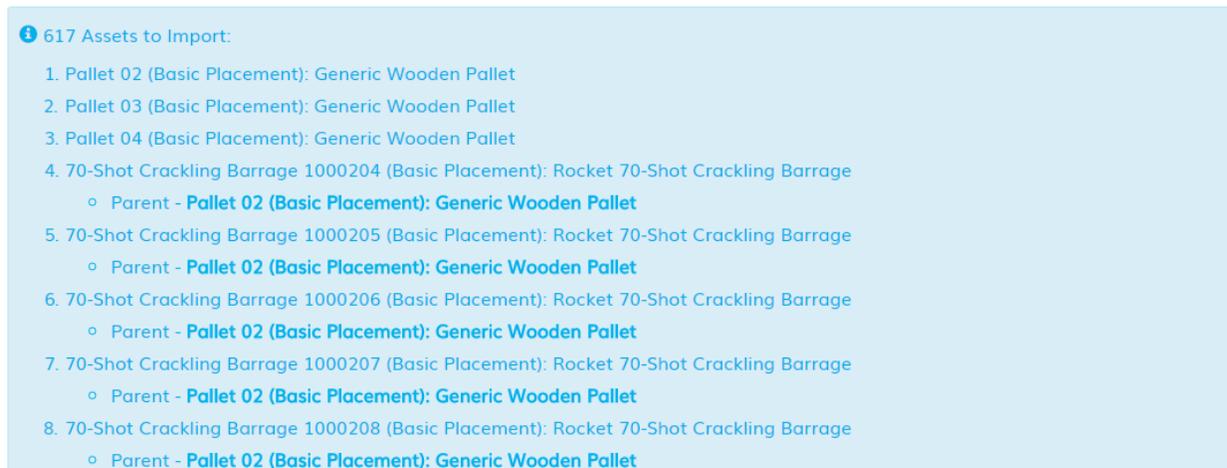


Fig. 4.9: Any validation errors or warnings are shown.

- Import - Any data which Validated successfully will attempt to be imported. Links to successfully loaded Models are created and displayed, any errors are listed at the bottom.

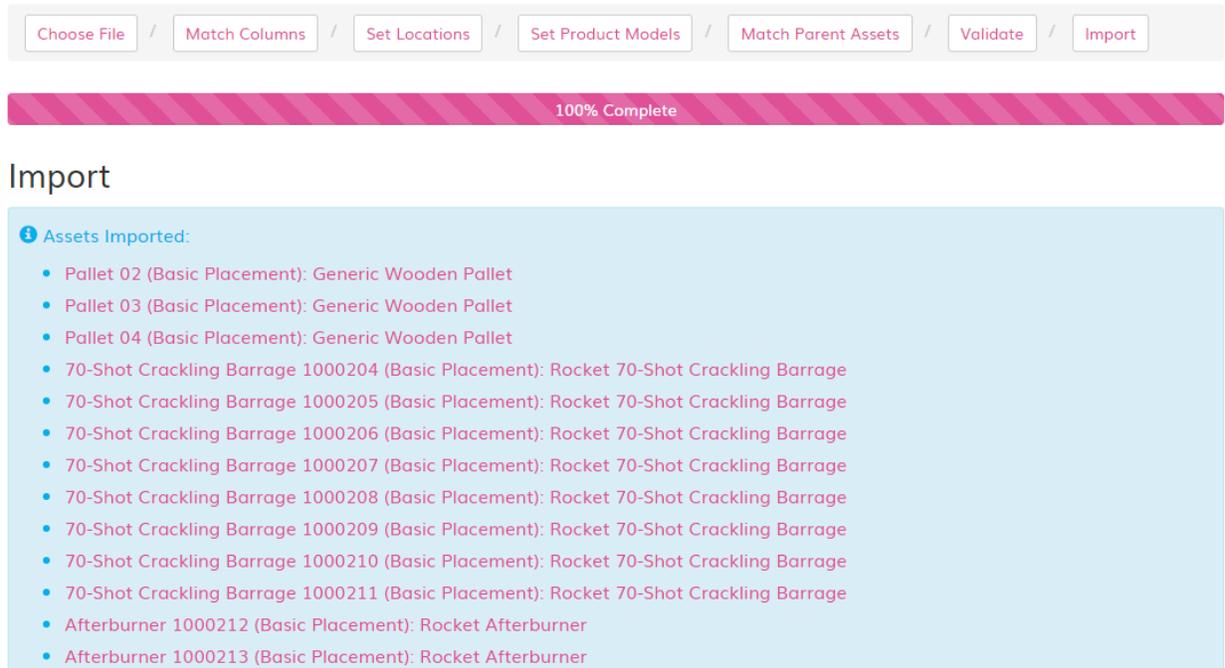


Fig. 4.10: An example of a Imported result set.

## EAV Wizard

### EAV Groups

This wizard allows Users to add and edit EAV Groups.

### EAV Fields

EAV fields can be added, edited or removed in this Wizard.

---

## EAV (Entity Attribute Values)

---

New in version 16.02.

Changed in version 17.02.

### **EAV Overview**

### **EAV Groups**

#### **Order**

This defines the priority order within which to show the Attribute Groups on the forms within which they're displayed.

#### **Name**

The name of the EAV Group.

#### **Bootstrap Class**

This determines the width of the field, if Small or Extra Small are chosen, the field will consume the full width of the screen (designed for small screens). Medium and Large are the opposite.

- Extra Small - Full width
- Small - Full width
- Medium - Half width
- Large - Half width

## EAV Attributes

### Name

The name of the field.

### Order

The priority order to display the field.

### Slug

A unique name, which does not allow spaces

### Required

Is the field required when the object is added. A validation error will appear if no *Default Value* is entered.

### Include in Main Grid

If this option is checked, the field will appear in the *Assets* grid table.

### Include in Sub Grid

If this option is checked, the field will appear in the *Assets* expanded grid table, i.e. where a Parent Asset has been expanded and the child Assets are listed.

### Description

Some text which will be displayed below the EAV field to help Users identify what the field is for.

## Edit Asset Step 3 of 3

### Finance

---

#### PO Number

986511

The PO Number from the Procurement System.

Fig. 5.1: An example of the description text underneath the EAV field.

## Attribute Group

The name of the *EAV Groups* which the field should belong to. The field can only be a member of one group.

## Data Type

The type of data that can be stored in this field. The following options are available:

- Text - A string of text
- Float - A number or decimal
- Integer - A whole number
- Date - A datetime value
- True/False
- Multiple Choice - If chosen, the *Choice Group* field must be populated.
- Distance Field

## Type

This determines what object type the field belongs to, it can be either:

**Asset** or **Location**.

## Choice Group

It is possible to create and edit Choice Groups. If the Multiple Choice option has been selected under *Data Type*, this field must have a value.

The screenshot shows a configuration form for a field. The 'Type' field is set to 'asset'. Below it, the 'Choice Field' section is active, showing 'Choice group' set to 'Orientation (Degrees)' with edit and add icons. The 'Default Value' section is empty. To the right, a separate window titled 'Enum group' is open, showing a list of values: 0, 90, 180, and 270, with a plus icon to add more.

Fig. 5.2: A selected Choice Group, showing the Edit Group option open in a new window.

## Default Value

A default value, if required. This could be 0 or None.



New in version 16.02.

Changed in version 17.02.

## Locations Overview

Locations can be created in the *Location Wizard*.. Ensure the correct *Location Types*. exists first, and consider how the Location structure should appear.

Typically, a structure would look something like the image below, however it could be generated to represent any parent > child relationship between objects:



Fig. 6.1: An example of a simple Location structure.

Ensure the correct properties are assigned to a Location Type, for example, the a Building type has the ability to accept Floor and/or Rooms as Child Location Types.

It is possible to amend a Location Structure at any time using the *Location Tree*, so it can be updated as required if the model structure isn't appropriate.

## Location Details

Location details can be updated using the *Location Wizard* or the Edit Location button in the *Sidebar* when viewing a Location.

Any Location fields, including EAV fields can be updated on this form.

## Edit Floorplan

It is possible to create a visual floorplan of a Location in the following situations:

- If the User wished to define a new area.
- If data has been imported from TRACKIT Mobile.
- If data has been imported from a previous version of TRACKIT.
- If data has been imported from another product, such as a CMDB or DCIM tool which has space information.

## Importing Floorplan Data

If existing data is available in a support format, this can be added to a TRACKIT Location.

Supported formats are as follows:

- AutoCAD DWG - All valid layers will be imported and the scale retained.
- Microsoft Visio - Everything is imported as a single layer which would require scaling to the correct size.
- Adobe PDF - Everything is imported as a single layer which would require scaling to the correct size.
- Scalable Vector Graphics (SVG) - Imported as a single layer which would require scaling to the correct size.

Floorplans can be imported by simply dragging and dropping from a Desktop or File Explorer on the Users PC into the web browser.

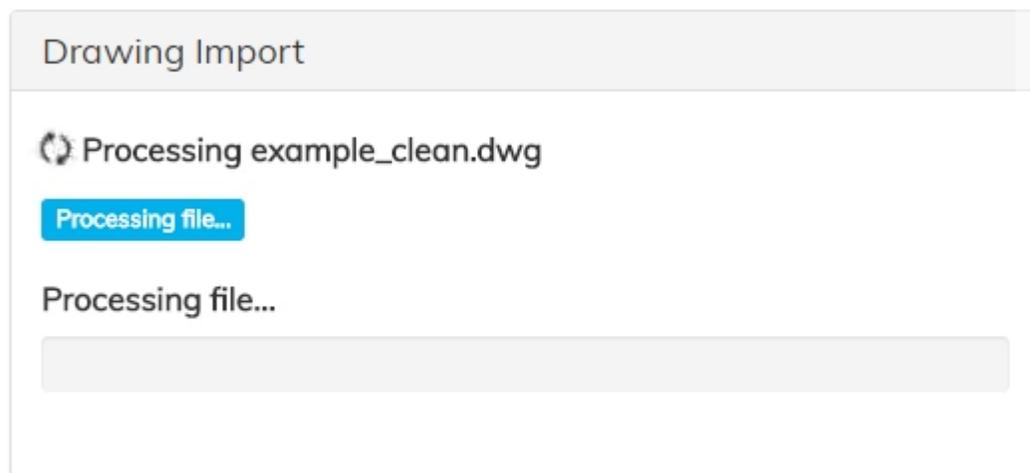


Fig. 6.2: The import screen of a Floorplan

Once the file has been imported, it can be scaled, rotated or moved as appropriate.

If the User requires, they can create an area on the floorplan to allow mounting of Asset using *Edit Borderpath*.

## Edit Borderpath

If a floorplan has been added, it's possible to define an area on the floorplan to designate as a set Locations mounting area.

To do so, select the Edit Border Path button on the Edit Floorplan form.

Once the floorplan has loaded, choose the option on the *Sidebar* to add points, creating enough to surround the area accurately.

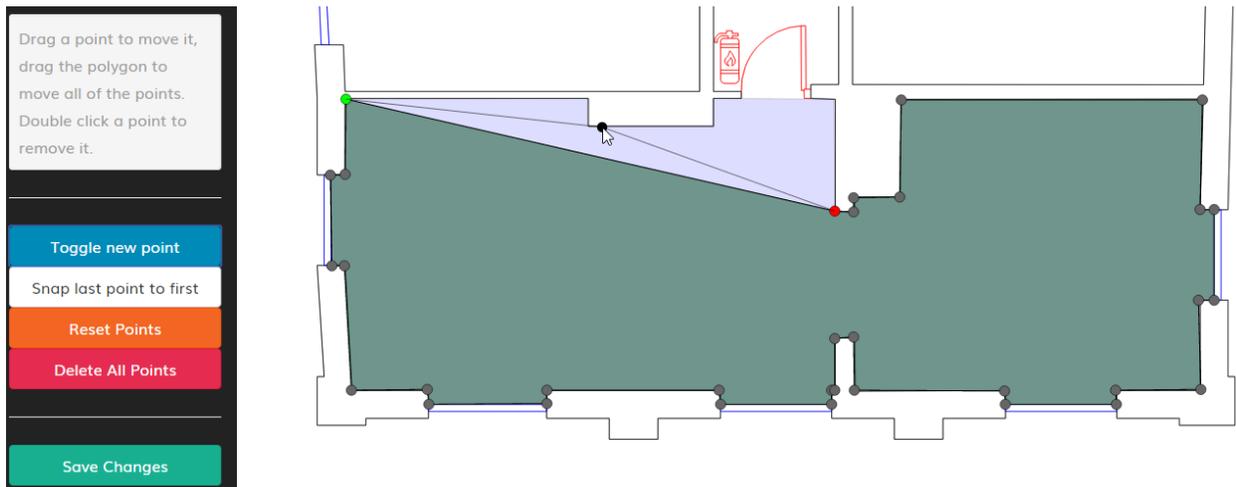


Fig. 6.3: The import screen of a Floorplan

## Regions

If the Map is used on the Dashboard screen, regions can be defined to place locations into. Regions are created in the Administration menu, under Location Management.



New in version 16.02.

Changed in version 17.02.

### Asset Overview

Assets can be added against either Locations or Parent Assets.

Data can added, edited and removed in a number of ways:

1. Using the TRACKIT Web Client.
2. Added via the TRACKIT Import *Asset Import*.
3. Via the TRACKIT RESTful API

### Placement Types

There are a number of methods of placing Assets in TRACKIT.

Placement Type	Main Location	Cell/Grid Placement	Parent Asset	Mount Location	Off-sets	Rotation/Orientation	U Position	Mounting Area
Floor Mounted	Required	Required			Required	Required		
Rack/Cabinet Mounted	Required		Required	Required	Required	Required	Required	
Asset Mounted	Required		Required			Required		Required
Basic Placement	Required		Optional					

## Adding Assets

The Add Asset button is available in a number of places through TRACKIT, usually in the *Sidebar*. Once selected, the Add Asset Form is presented, the User can complete this to Add an Asset to the database.

The Add Form comprised 3 parts:

1. Step One - Basic Details and Placement Type
2. Step Two - Product and Placement/Parent Details
3. Step Three - Additional Information

### Basic Details

This first step asks for some basic information on the Asset Name, any Serial or Asset Numbers.

It also requests the User add what Placement Type to use. This will determine what Products and Placement detail to collect on Step 2.

#### Add Asset Step 1 of 3

##### Basic Details

Name

Asset number

Please enter a asset number for this asset. Leave this blank for an asset Id to be automatically assigned.

Serial number

Please enter an optional manufacturer serial or part number for this asset.

Next



##### Placement Type

Placement Type

Floor Mounted

No Placement

Floor Mounted

Rack/Cabinet Mounted

Asset Mounted

Shelf Mounted

Basic Placement

Fig. 7.1: Step 1 of adding an Asset showing the different Placement Types

### Product and Placement Details

The next step requires the User to select the Product to use for the Asset.

Depending on which Placement Type was selected, any additional *Placement Types* data might be required at this step.

## Floor Placement Details

---

### Offset type

Cell/Grid Placement ▼

### Cell

Select a cell...

### Flush to cell

Flush to rear ▼

### Relative X Offset

0m

### Relative Y Offset

Fig. 7.2: An example of Floor Placement required fields in Step 2 of adding an Asset

## Additional Information

The final step allows the User to complete additional data fields. These typically include EAV fields for Assets.

The Add Asset Form will validate the data at each step and warn the User if there are any issues. Once completed, the Asset will be added to the system.

Sorry, the asset could not be saved. Please correct the errors below and try again.

### Add Asset Step 1 of 3

#### Basic Details

Name

- Please enter a valid asset name.

#### Placement Type

Placement Type

Please choose a placement type.

Fig. 7.3: An example of form validation on adding an Asset

## Editing Assets

### Edit Asset Button

When viewing an Asset, the Edit Asset button is visible in the *Sidebar*. This will take the User through the 3 steps of the *Basic Details* Add Asset form, however any existing data is pre-populated.

Field	Old Value	New Value
Floor Polygon	SRID=4326;POLYGON ((2.989 1.421, 4.309 1.421, 4.309 2.021, 2.989 2.021, 2.989 1.421))	SRID=4326;POLYGON ((2.99 1.421, 4.31 1.421, 4.31 2.021, 2.99 2.021, 2.99 1.421))
Name	Desk 1	Desk Alpha

Client Type	Action Type	Timestamp	Model Type	User	Server Host Name	IP Address
web	mod (View)	15 Mar 2017, 1:45 p.m.	location	support@trackit-solutions.com	bf290c37f5d1	10.255.0.2
web	mod (View)	15 Mar 2017, 2:47 p.m.	asset	support@trackit-solutions.com	bf290c37f5d1	10.255.0.2

Fig. 7.4: Viewing the details of an Asset Edit in the History tab.

## Assets Table

It's possible with appropriate permissions to edit Assets in the *Assets* Table.

Details Floor Plan **Assets** Photos Reports Tracking History

### Child Assets

Adding an asset using the button below will take you to another page, and you will lose any unsaved changes you have made. If you wish to add an asset directly to the floor plan, then please use the product model search in the left column and drag the relevant model onto the floor plan.

Cancel ✕
Save Changes ✓
Add Asset +

Asset Id	Device Name	Placement Type	Placement Details	Manufacturer	Product Model	Rotation
AST-0116502	Air conditioning unit	Logical	Logical	Emerson	Air Conditioning Uni	
<b>+</b> AST-0116504	cupboard	Logical	Logical	Ikea	- 0ucupboard 120x80x45	
<b>-</b> AST-0116423	Desk 1	Floor	X:2.99m, Y:1.421m	Ikea	White Desk	

Desk 1 Child Assets

Asset Id	Device Name	Placement Type	Placement Details	Manufacturer	Product Model	Serial
AST-0116442	Telephone 1	Logical	Logical	Grandstream	Grandstream IP Phone	20EY...

Fig. 7.5: An example of editing an Asset in the Asset Table.

## Deleting Assets

Assets can be deleted from TRACKIT, however they cannot be restored once deleted.

It is highly recommended that Assets are instead moved to a Decommissioned or Remove Location which can be reported on and retains the History of the Asset through it's lifecycle.

### Delete Asset Button

Should it be a requirement to delete an Asset, it can be done by selecting the Delete Asset button visible in the *Sidebar* when viewing an Asset.

Deleting an Asset will remove all Child Assets and should only be done where there is an error and/or the History of the Asset is not required.

Home / Organisation ▾ / Building / Floor / Room ▾ / Desk 5 ▾ / Monitor 3 ▾

## Delete Asset

Are you sure you wish to delete this asset (Monitor 3 (Basic Placement): Dell Dell 24" Monitor )?

All asset information and related details, such as tracking information and documents will also be removed. The history and audit trail will still be stored permanently, and a record of the deletion will be added.

### Are you sure?

This action *\*cannot\** be undone once confirmed, and the object cannot be restored.

If you wish to retain asset information, it is highly recommended you decommission the asset instead.

[No, return to the details page.](#) [Yes, permanently delete the asset information.](#)

Fig. 7.6: Confirmation and Warning message when Deleting an Asset.

New in version 16.02.

Changed in version 17.02.

### Product Overview

Ensure that the *Products*, *Asset Class* and *Categories* sections have been read and understood.

Products are fundamental to ensuring good quality data within TRACKIT. Correctly defining distinct Manufacturers, Models and Categories means data can be added, maintained and reported quickly and accurately.

To maintain Products, use the *Product Wizard*.

### Maintaining Categories

Categories can be maintained in the *Categories*.

The main purpose of Categories is to Group Products and Assets for Reporting.

### Maintaining Manufacturers

Manufacturers can be maintained in the *Manufacturers*.

It is possible to add Alias' for Manufacturers, i.e. Hewlett Packard, HP. This reduces the confusion and increases the consistency in the data.

By selecting a Manufacturer, it's possible to view all the Product Models created in the database using that Manufacturer.

## Maintaining Models

Models can be maintained in the *Models*.

New in version 16.02.

Changed in version 17.02.

## Reporting Overview

Reports

## Tabular Reports

### Columns

All data fields available against the report data, i.e. the Assets, Locations or Products are listed under Columns. By default, some Columns will be selected, however the User can show or hide any columns they choose.

Any EAV fields that have been configured within TRACKIT will also be listed and can be included in the report data.

### Formula

Formulas allow the User to create a new field of data in the report based on the data columns available.

It's possible to determine the Type of Data, i.e. date, number, text etc. and display the values in specific formats, as gauges or colours.

Any additional Formula columns created can be sorted, grouped, filtered etc as with any other fields.

Formulas are expressions made up of columns, constants, functions, and operators.

Columns are values that come from the data. Their names are enclosed in square brackets, like [ShippedDate]. Depending on their data type, they may be used as text strings, numbers, and date/time values.



Fig. 9.1: Columns can be selected or unselected.

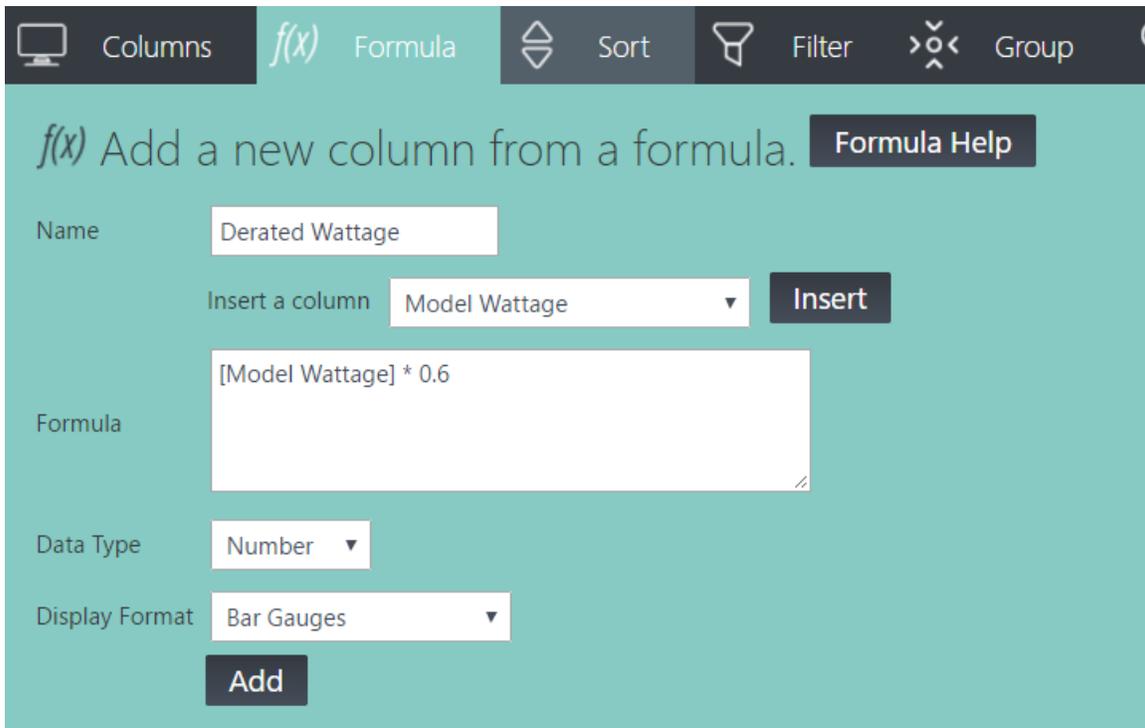


Fig. 9.2: An example of a Formula in the Reporting.

Here are some other examples:

```
[Price] * [Quantity]
```

Multiply two data columns, Price and Quantity, to make an ExtendedPrice column:

```
[Price] * .04
```

Multiply a data column by an constant value to calculate the tax applied to the price:

```
DateDiff("d", [OrderDate], [ExpiryDate] )
```

Get the number of days from the order to the shipment:

```
DateDiff("w", [ExpiryDate], Now )
```

Get the number of weekdays since the shipment date:

```
WeekdayName( Weekday( [ExpiryDate] ) )
```

Return the name of the day of the week of the shipment date:

```
[LastName] + ", " + [FirstName]
```

Concatenate columns and strings together. This might return: Smith, John:

```
UCase( [LastName] + ", " + [FirstName] )
```

Convert to upper case. This might return: SMITH, JOHN

## Sort

This option provides the functionality to Sort columns depending on the content. It's possible to sort on multiple columns in a specific order of priority.

## Filter

The filter function allows the User to display or hide specific data based on the data itself.

It's possible to filter on multiple columns and individual columns multiple times.

When filtering multiple times, selecting the And/Or text between the Filter row swaps the Filter rules from And (where the data being filtered must meet both requirements), or Or (where the data being filtered can meet either of the requirements).

The following types of Comparison operators are available:

- =, Equals, the data being filtered must match the given string exactly
- <, Less than, the data being filtered must be less than the given value
- <=, Less than or equal to, the data being filtered must be less than or equal to the given value
- >=, Greater than or equal to, the data being filtered must be greater than or equal to the given value
- > Greater than, the data being filtered must be greater than the given value
- Not =, The data returned will not exactly match the given string or value

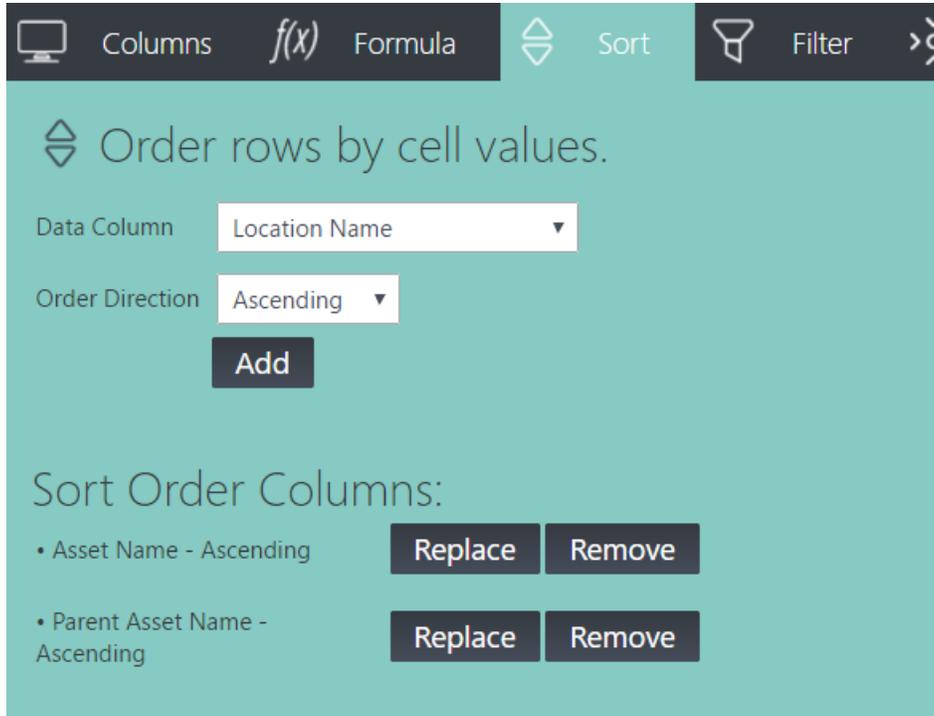


Fig. 9.3: An example of sorting on multiple columns.

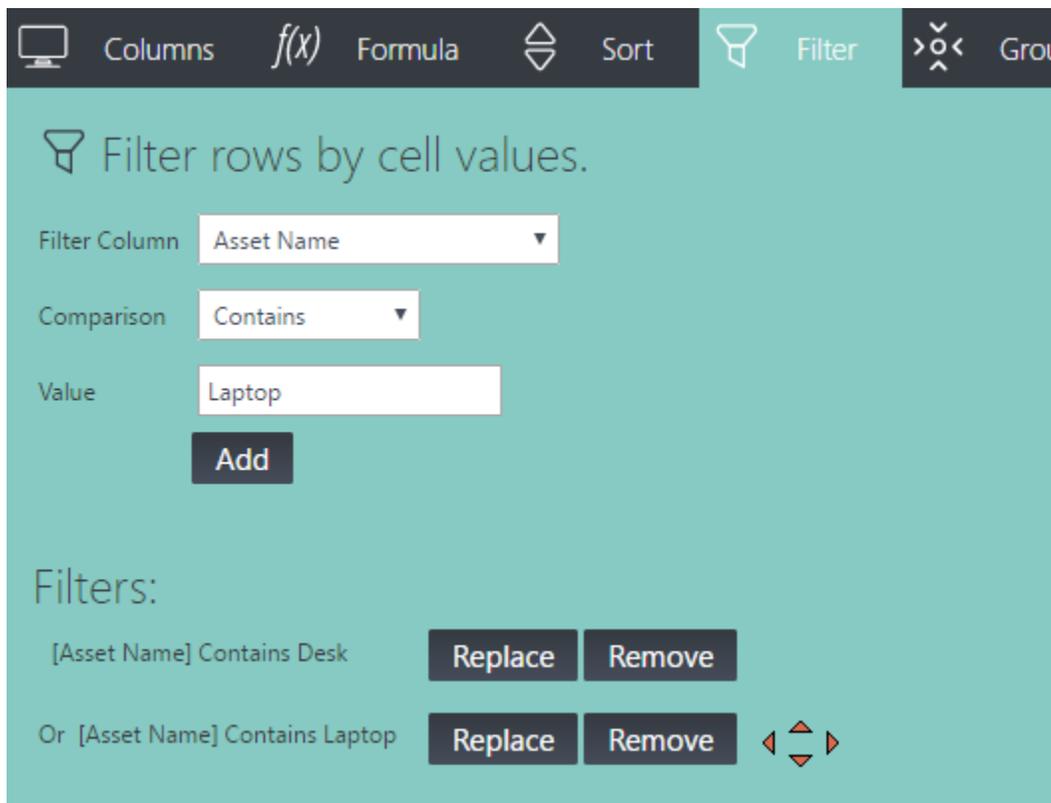


Fig. 9.4: An example of filtering on multiple values.

- Starts With, The data returned will start with the given string
- Contains, The data returned will contain the given string
- Not Starts With, The data returned will not start with the given string
- Not Contains, The data returned will not contain the given string

## Group

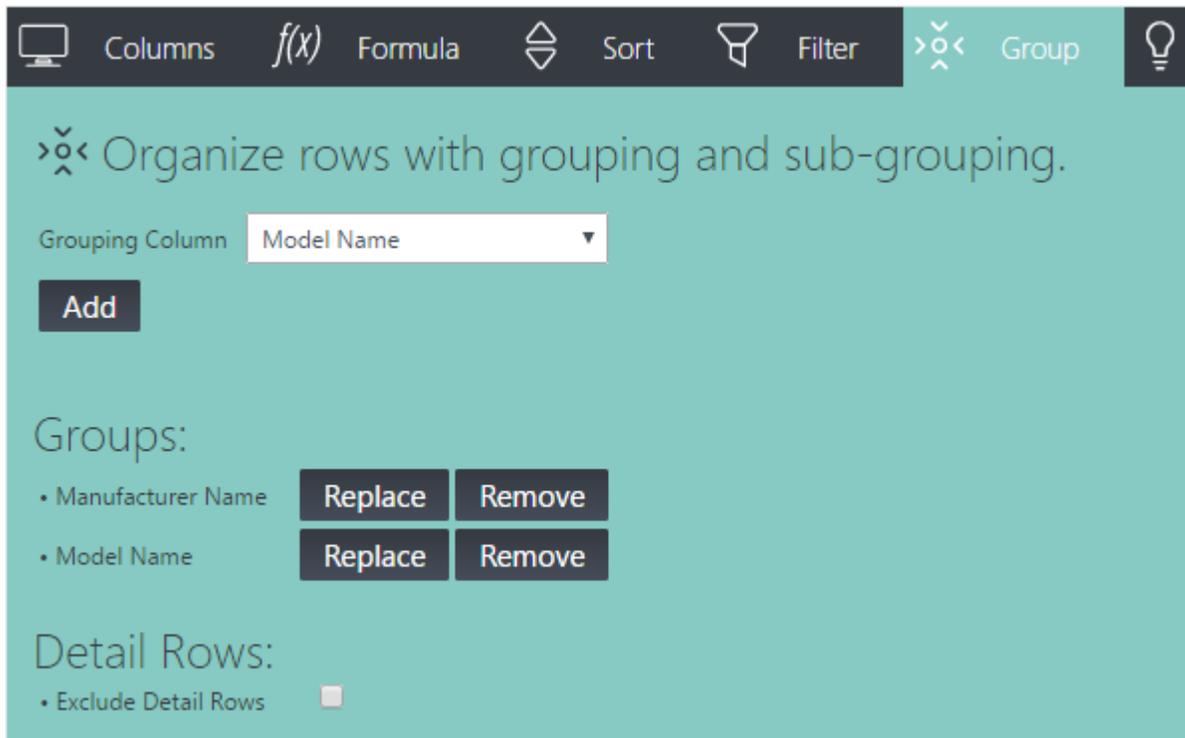


Fig. 9.5: An example of grouping on multiple columns.

## Aggregate

## Chart

## Crosstab

## Paging

## Saved/User Reports

## Trend Reports

## Dashboard Reports

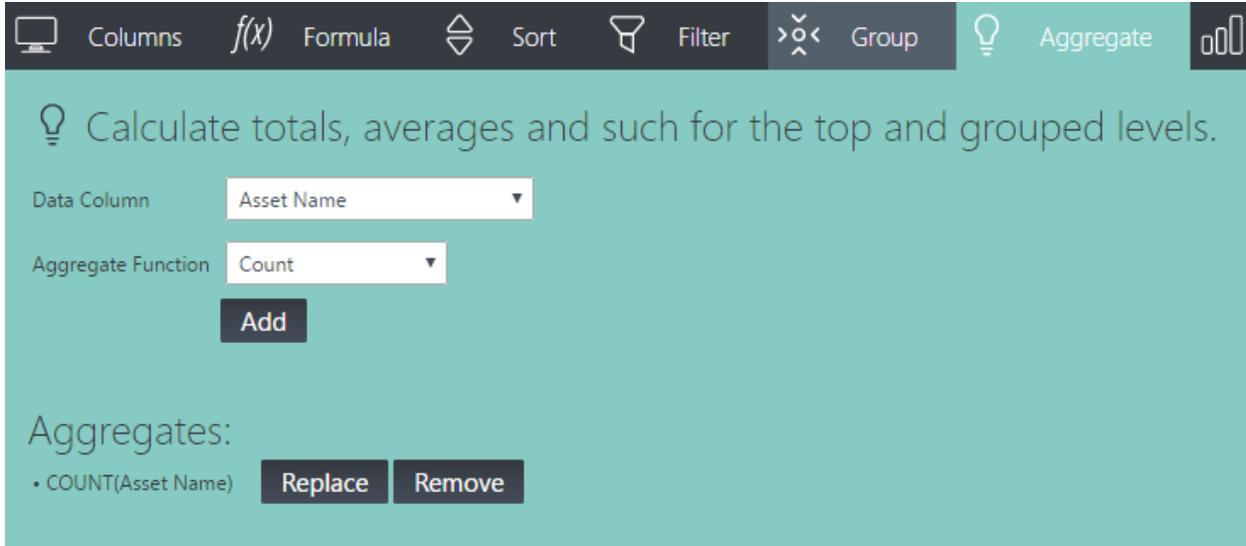
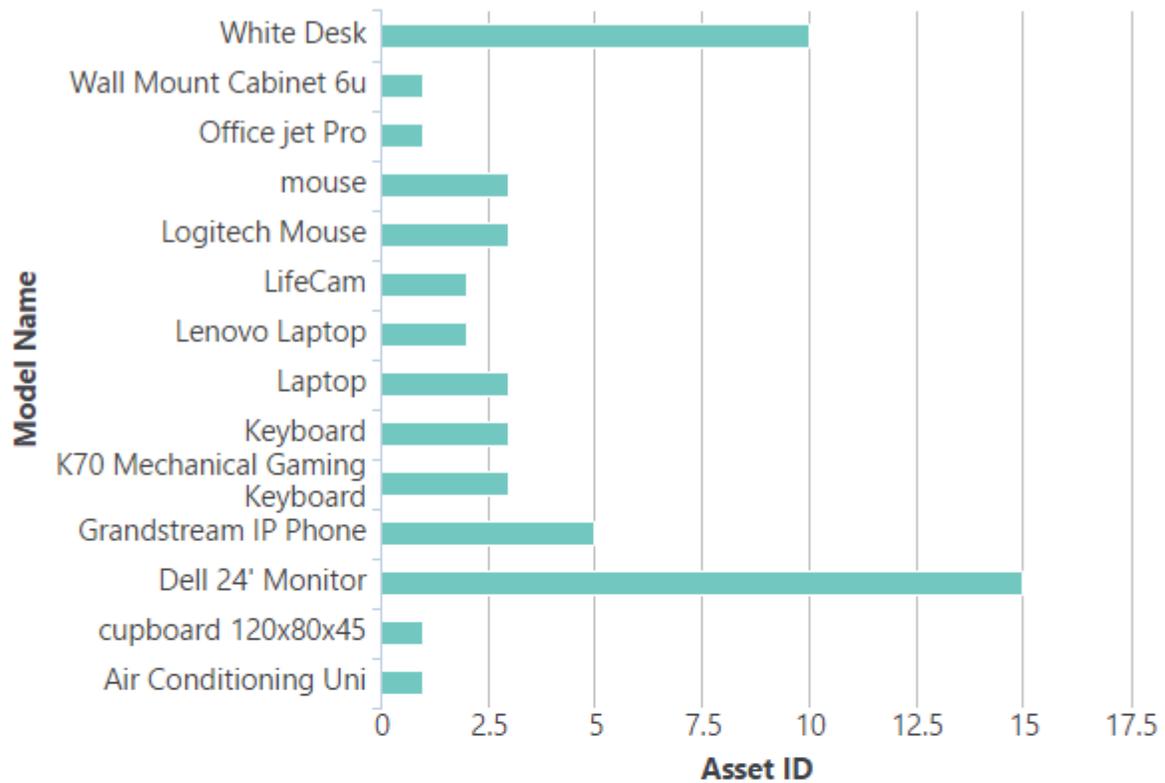


Fig. 9.6: An example of an aggregate value on grouped data.

<b>Dell</b>		<b>Count: 15</b>
	<b>Dell 24' Monitor</b>	<b>Count: 15</b>
	monitor	logical Desk 2
	Monitor 1	logical Desk 3
	Monitor 2	logical Desk 3

Fig. 9.7: The results of an aggregate value on grouped data.

## ⊖ Bar Chart - Count of Asset ID by Model Name



## ⊖ Pie Chart - Count of Asset ID by Model Name

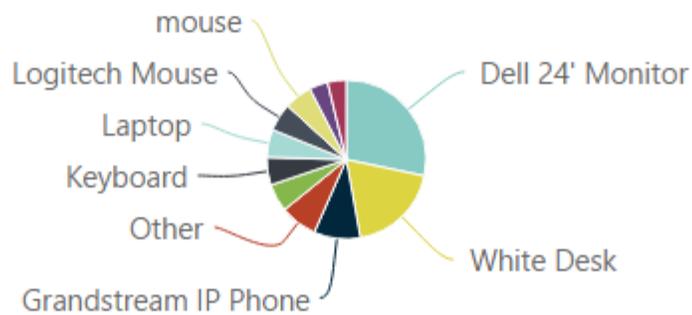


Fig. 9.8: A bar and pie chart, showing the same data, number of Asset per Product Manufacturer.

The screenshot shows the 'Crosstab' configuration panel with the following settings:

- Header Values Column: Parent Asset Name
- Label Values Column: Asset Placement Type
- Aggregate Values Column: Asset ID
- Aggregate Function: Count

Below the configuration is a list of crosstabs with the following entry:

- Parent Asset Name by Asset Placement Type on COUNT of Asset ID

The resulting crosstab table is as follows:

Asset Placement Type	Desk 2	Desk 3	Desk 4	Desk 5	Desk 6	Desk 7	Desk 8	Desk 9	Desk 10	Desk 1	cupboard
logical	1	5	2	2	4	6	6	5	5	1	1

Fig. 9.9: An example of a crosstab showing the number of Assets per Parent Asset in a Location.

The screenshot shows the 'Paging options' configuration dialog with the following settings:

- Show all rows:
- Show paging:
- Rows per Page: 20

An 'OK' button is visible at the bottom left of the dialog.

Fig. 9.10: A view of the Paging options.

New in version 16.02.

Changed in version 17.02.

### Permissions Overview

### Managing Users

Users can be created, edited or removed in the *Admin Overview* Section. The following fields are required against a User:

- Email Address (Username)
- First Name
- Last Name
- Active - Users can be disabled to temporarily or permanently revoke access but retain the User.
- Staff Status - Will the User have access to the Admin site
- Superuser status - Gives the User access to all permissions without explicitly assigning them.
- Groups - Lists the Groups the User is a member of.
- User Permissions - Individual Permissions granted to the User outside of any Group permissions they may inherit.

### Resetting Passwords

It is possible to reset a User password through the Administration Menu of TRACKIT.

## Deleting Users

Deleting a User from TRACKIT will remove the history and ability to restore the User. A more suitable option has been designed, where a User can be Suspended, but retained in the system for reporting and auditing.

## Maintaining Groups

Groups can be added, edited or removed in the *Admin Overview* Section.

Name:

Permissions:

<input type="checkbox"/>	admin   log entry   Can add log entry
<input type="checkbox"/>	admin   log entry   Can change log entry
<input type="checkbox"/>	admin   log entry   Can delete log entry
<input type="checkbox"/>	assets   asset   Can add asset

Hold down "Control", or "Command" on a Mac, to select more than one.

Fig. 10.1: A view of the Group Permissions selector in TRACKIT Admin.

It's possible to select multiple permissions by holding down the Ctrl key and selecting a number of permissions.

New in version 16.02.

### **TRACKIT Mobile Overview**

TRACKIT is an Asset Management tool, designed to be delivered via the Cloud as a Software as a Service (SaaS) application. It can be deployed locally and installed on a local virtual or physical machine within a Customer environment.

### **TRACKIT Mobile App**

TRACKIT has been designed to provide an accurate, easily maintainable inventory of Locations and Assets. This could be any size, such as a Small Office environment or Store Room, through to a Large Enterprise scale Data Centre. Applications include Vehicle Fleet Management, IT Asset Management, Warehouse Stock Inventories or Mobile Equipment Monitoring.

### **TRACKIT Mobile**

TRACKIT Mobile is a Windows tablet based application designed for Offline Data Collection of IT Rooms and Datacentre Assets.



Fig. 11.1: A view of the TRACKIT MOBILE App.

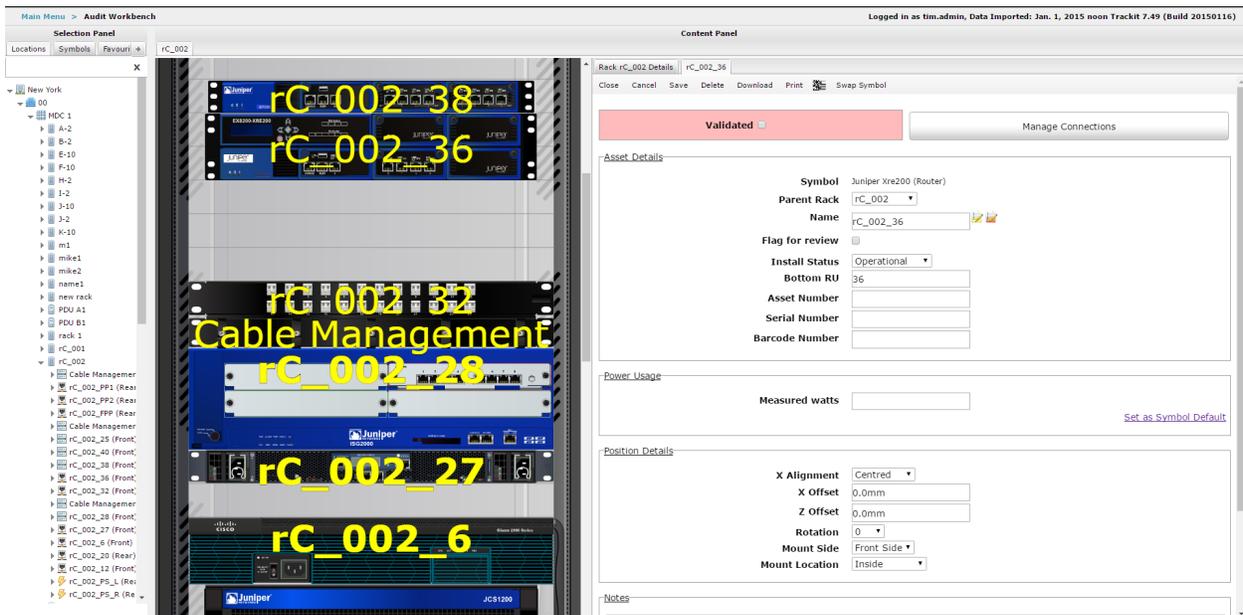


Fig. 11.2: A view of TRACKIT Mobile.

New in version 16.02.

Changed in version 17.02.

### **Admin Overview**

TRACKIT provides access to all features and functionality within the tool through the Administration section. Only Admin Users should be setup to have permissions to access these functions as they can determine the way TRACKIT works.

Do not update any settings in this section without confidence in what is being achieved or without guidance from TRACKIT Support or Services.

### **Administration Options**

Below is a list of each Admin section and a summary of the contents.

#### **Access and Authentication**

Allows updating of the Users and Groups.

#### **Asset Management**

This section covers everything to do with managing the assets, groups, environments, contracts and mount locations.

#### **Audit Trail**

An administrative view of the audit trail for everything within TRACKIT.

## Constance

This Config provides access to global system settings, usually this is pre-configured, however it's possible settings could be changed here if directed by TRACKIT Support or Services.

## Custom Forms

This should not be used.

## Djcelery

This section manages tasks and workflows within TRACKIT, it is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

## Dynamic Model Validation

This section manages rules and validation within TRACKIT, it is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

## EAV

This provides access to the EAV fields. It's possible to add, modify or remove EAV fields and groups in this section.

## Filer

This section manages folders within TRACKIT, it is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

## Import and Export

This section manages the import and export of data within TRACKIT, it is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

## Location Management

Many Location functions are available in this section.

- Location Cell Names - Provides the ability to modify cells mappings for Floorplans with grids and cells.
- Location Drawings - Allows the viewing/modification or removal of Floorplan drawings that have been added to Locations.
- Location Images - Allows the viewing/modification or removal of images that have been added to Locations.
- Location Mappings - Allows the viewing/modification or removal of Floorplan mappings, such as Grid definitions, that have been added to Locations spaces.
- Location Types - Full access to Location Types
- Locations - Full access to Locations

- Region - Provides functionality to define Regions
- Region Types - Allows Region Groups to be added/modified or removed.

## **Monitoring and Metrics**

### **Power**

This section is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

### **SVG Drawing/Annotations**

This section is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

### **Token\_Auth**

This section is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

### **Trackit**

This section is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.

### **Trackit External API**

### **Trackit Product Library**

### **Trackit Reporting**

### **Valuation**

### **Wizards**

### **Workflow and States**

This section is preconfigured and should not be tampered with except under guidance from TRACKIT Support or Services.



New in version 16.05.

Changed in version 17.02.

## Monitors and Metrics Overview

### Metric Classes

Metric classes determine what type a Metric is, for example, Power, Temperature, Security etc.

Any classes that have been created are listed here. The slug value is automatically generated and can't be edited, but the Name can be.

### Metric Values

This section lists the values that have been collected for any Metricsexist. It is possible to search for and find any problem values in here and delete them. It's also useful for checking to see what actual values are.

### Metrics

The view allows Metrics to be created and updated. It shows the latest value collected which can be useful for troubleshooting. Metrics can be edited or deleted from here too.

### Monitor Input Groups

These groups are generated by the monitoring naming standards to group items based on the same source device. If required, they can be altered, but it recommended to leave these.

## Monitor Input Values

This section lists all collected values from the monitoring system (often millions of them). It can be searched and filtered and it's possible to remove erroneous values from here.<sup>1</sup>

## Monitoring Source Types

The list of available monitoring API's loaded into the system.

## Monitoring Sources

Here it's possible to add additional, edit existing or remove any monitor sources in the system.

## Monitors

This sections provides the ability to create, update and remove monitors based on an existing monitoring source.

## Threshold Breaches

Allows creation, editing or removing of threshold settings for a specific metric.

## Configure a Monitor and Metric

This section describes how to set up Monitoring using an existing (in this case t-mac) API.

Ensure that the required Monitoring Source Type is available

Ensure that the correct Monitoring Source is available. If not, add a new one, the details required will be similar to as follows:

### Add a Monitor

Add the Monitor for the area or space required, the following details (or similar) will be required. Upon saving, the Monitor inputs will be calculated as Trackit connects to the Monitor. This may take a few moments. Once complete, the Number of inputs will no longer display 0.

### Add a Metric

Now Metrics can be created for the monitor.

A Name should be something unique that follows a standard

1. An Asset should be something within the correct space, either a monitoring rack (to link the metric to a space) or an individual asset such as a rack or device if appropriate.
2. The Monitor should be the correct Monitor create earlier
3. The Group is automatically generated, however can be overwritten. If in doubt, leave blank.
4. The Code is a short unique name that should follow a standard

5. The Metric Function can be one of the following:

- Mean Average - Will provide a single average value based on all input values added.
- Count of Values - Will generate a count of all input values added.
- Divide Values - Will generate a value based on dividing one input value by another i.e. Input Value 1 / Input Value 2.
- Maximum Value - Will show the maximum value based on all the input values provided.
- Minimum Value - Will show the minimum value based on all the input values provided.
- Pass through Input Value - Will simply pass through an input value as a metric value. Useful if the monitoring system already provided a metric such as PUE or Rack Temperature.
- Basic Formula - The most useful function, allows the user to generate a metric based on a formula of input values and then lists those inputs below.
- Sum of Values - Simply adds all provided input values together.

6. Classes refer to Metric Classes and can be added or new ones generated. A metric can belong to multiple classes

7. Minimum required value and Maximum required value are values which can be provided to prevent erroneous or anomaly values from being collected and producing incorrect charts. These should not be confused with Threshold values.

8. Description is the description of the Metric created.

9. Unit of Measure is automatically generated, however it's possible to override this if required.

10. Metric Inputs can be added, autocomplete attempts to search for the correct input. If using the Basic Formula option, the inputs are added automatically.

11. Metric Thresholds can be setup to trigger an alert when the Metric value is within a certain range. See the Threshold section for more details.

When complete, save the Metric and the value will be calculated.



New in version 16.02.

Changed in version 16.09.

### Integration Overview

In order to integrate both internal products (such as the Trackit 8 Web front-end and Trackit 8 Mobile Apps), and external products (such as a customer's own specific product), Trackit provides two main strands of API.

1. Data API - RESTful JSON API for retrieval/modification of data structures, based on Django Rest Framework. This is typically used for manipulation of data within the main application, and the mobile application, via standard XMLHttpRequests. Connectors are implemented for our backbone.js models within the basis module static files.
2. Application API - WSDL/JSONRPC API for programmatic and function calls, based on Spyne.
  - 2.1 This is used to implement Remote Procedure Calls.
  - 2.2 This could be anything from logging in, to retrieving or executing specific commands.
  - 2.3 It defines a set of commands which can be called using WSDL and or JSONRPC.

### Application API Autodiscovery

Spyne is a protocol agnostic library which allows quick and easy implementation of RPC based APIs in a Pythonic manner. It has good integration with Django, and is an excellent fit for the application. It also generates WSDL schemas for use with external SOAP clients.

The library provides a variety of transports and protocols which can be utilised without needing to implement extra code. Protocols include SOAP 1.1 and JSONDocument, and can be "mixed and matched" in terms of input and output documents. In order to simplify development, the trackit/trackit\_api folder within the main trackit module provides API autodiscovery within the AppConfig ready method. This builds on the Spyne Django implementation, and automatically discovers any Spyne methods within a module called api.py in any INSTALLED\_APPS. The process is as follows:

1. When Django calls the ready method, check that the `API_ENABLED` Django setting is True. If so, proceed to autodiscovery.
2. Iterate all of the installed AppConfigs. For each iteration, check if an `api.py` module exists within the application root folder.
3. If so, then iterate through each member of the module, and check if it's a Spyne service by checking if the member is an instance of `ServiceBase` or `DjangoServiceBase`.
  - Where possible, `spyne.util.django.DjangoServiceBase` should be used as a service base class, as this implements some additional exception handling specifically for Django errors, such as validation, and re-throws them as the relevant Spyne error. Please see the API Service Example that follows.
4. Look at the `API_ENABLED_PROTOCOLS` Django setting to determine what RPC protocols Trackit should create. This will very easily enable us to support additional protocols in the future, such as `Yaml` or `MessagePack`.
  - A protocol consists of three components:
    - An input protocol. This determines the format/protocol that data supplied by an API call should use. For example, `Soap11` or `JSON`. This defines how Spyne will parse the input HTTP request to determine a) what API call needs to be made, and b) what input data should be parsed/passed to the Service.
    - An output protocol. This determines the format/protocol that the API call should return to the callee. This determines how Spyne will serialize objects that the Service returns.
    - A validator. This determines what module will be used to verify valid input data - for example, `lxml` for `WSDL/Soap` calls.

For more information on Spyne high-level concepts, please see this documentation page.

- The default defined protocols are as follows:
  - `JSON`: Input protocol = `JSON`, Output Protocol = `JSON`, Validator = `Soft validator`. Name = `json`
  - `WSDL`: Input protocol = `Soap1.1`, Output Protocol = `Soap1.1`, Validation = `lxml`. Name = `wsdl`
- For each defined protocol in `API_ENABLED_PROTOCOLS`, create the relevant protocol instances, and store in an `AppConfig` property.
- Within the `urls.py` for the `trackit_api` module, iterate over the protocol instances.
  - For each protocol instance, create a standard Django URL pattern.

This is a combination of the application name according to the `AppConfig` (e.g. `prolib`) and the protocol name as above (e.g. `wsdl`), and is mapped as, for example, `^/prolib/wsdl/`. This is also a named URL pattern of the format `"prolib_wsdl"`. \* For the relevant URL, create a `Spyne DjangoView` to map directly to the defined Spyne services.

## Implementing API Services

Whilst knowledge of the API autodiscovery is useful, it is not a requirement to implement API services. An API service can be thought of as a group of one or more RPC methods, to perform specific functionality. Generally it is a Python class, which extends the `"DjangoServiceBase"` class provided by Spyne, and implement one or more functions. Each function can be mapped directly to an RPC function.

Please note, each Service is not currently namespaced, so if you implement more than one Service within an `api.py` module, you should ensure the function names are unique across all services. To implement an API service, first create an `api.py` file within the relevant module, if it does not already exist. Within the `api.py` file, implement one or more Spyne services, according to the Spyne documentation. An example Hello World service can be found here.

Please note, it is imperative that the `rpc` decorator be used over the `srpc` decorator if you wish to gain access to the Django request object.

The Django request object is stored within the Spyne context object, which is passed as the first argument to a service (if the `rpc` decorator is used to define the service). It can be accessed via the `req` property of the `transport` property of the context argument. For example:

```
class HelloService(DjangoServiceBase):
    @rpc(Unicode, _returns=Unicode)
    def whoami(context, input_argument):
        django_request = context.transport.req
        return "You are %s (%s)" % (django_request.user, input_argument)
```

This will be accessible via the URL endpoint determined by the URL mapping as described in the previous section. Depending on the input protocol, the method you use to call it may differ, but as an example, to call via the JSON protocol we could use curl to simulate this. Assuming the module is within the “trackit\_api” Django application, and the server is running on port 8000 on localhost, we would use the following CURL method to do so:

```
curl -s http://localhost:8000/en-us/api/trackit_api/json/ -d \
    '{"whoami": {"input_argument": "Hello!"}}'
```

The initial object key (“whoami”) determines the function to call within the service, and the associated object determines the parameters to be mapped to the function (e.g. `input_argument = “Hello!”`). The context argument is automatically provided by Spyne. This provides the following output:

```
"You are AnonymousUser (Hello!)"
```

This neatly demonstrates a) the Django request object access, and b) how Spyne automatically serializes the returned Python object into JSON format. You can test WSDL functionality using the Python library suds, via the same method, as follows:

```
from suds.client import Client
cli = Client("http://localhost:8000/en-us/api/trackit_api/wsd1/")
print(cli.service.whoami)
<suds.client.Method instance at xxx>
print(cli.service.whoami("Tester!"))
You are AnonymousUser (Tester!)
```

This demonstrates a) the automatic WSDL generation (which can be seen in a browser by navigation to [http://localhost:8000/en-us/api/trackit\\_api/wsd1/](http://localhost:8000/en-us/api/trackit_api/wsd1/)), and b) the automatic RPC discovery via suds, and how the results are serialized.



# CHAPTER 15

---

## Indices and tables

---

- `genindex`
- `modindex`
- `search`