Technical Design Document – AIR ALERT

Thingworx

Alpha Release

Table of Contents

[1. Application Overview 3](#_Toc51943076)

[1.1 Architecture 3](#_Toc51943077)

[1.2 ERD 4](#_Toc51943078)

[1.3 Thing Shape Implementation 6](#_Toc51943079)

[1.4 Entity Naming Convention 7](#_Toc51943080)

[2. Device Onboarding 8](#_Toc51943081)

[3. Data Storage 9](#_Toc51943082)

[4. Third Party Integration 12](#_Toc51943083)

[5. Rules Management 14](#_Toc51943084)

[5.1 Pre-Calculations 14](#_Toc51943085)

[5.2 Rules Implementation 15](#_Toc51943086)

[5.3 Alerts 16](#_Toc51943087)

[6. Roles & User Management 17](#_Toc51943088)

[6.1 Organization Tree 17](#_Toc51943089)

[7. Management Dashboards/ Admin Pages 21](#_Toc51943090)

[7.1 Landing Pages 21](#_Toc51943091)

[7.2 Machine List Pages 22](#_Toc51943092)

[7.3 Machine Details Pages 22](#_Toc51943093)

[7.4 Rules Pages 23](#_Toc51943094)

[7.5 Add/Update Machine Config Files 24](#_Toc51943095)

[7.6 Dashboard Calculations 25](#_Toc51943096)

[8. Services Used 27](#_Toc51943097)

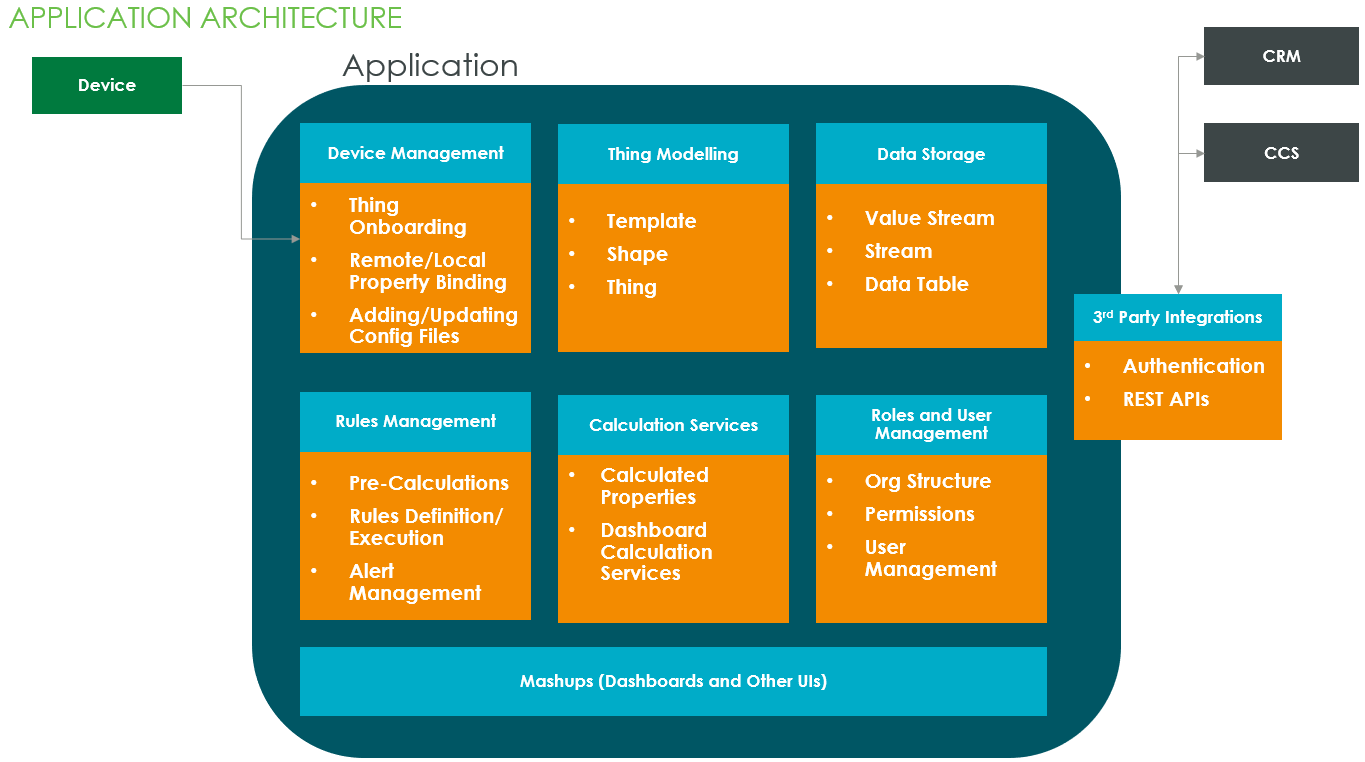
# Application Overview

## Architecture

A screenshot of a cell phone

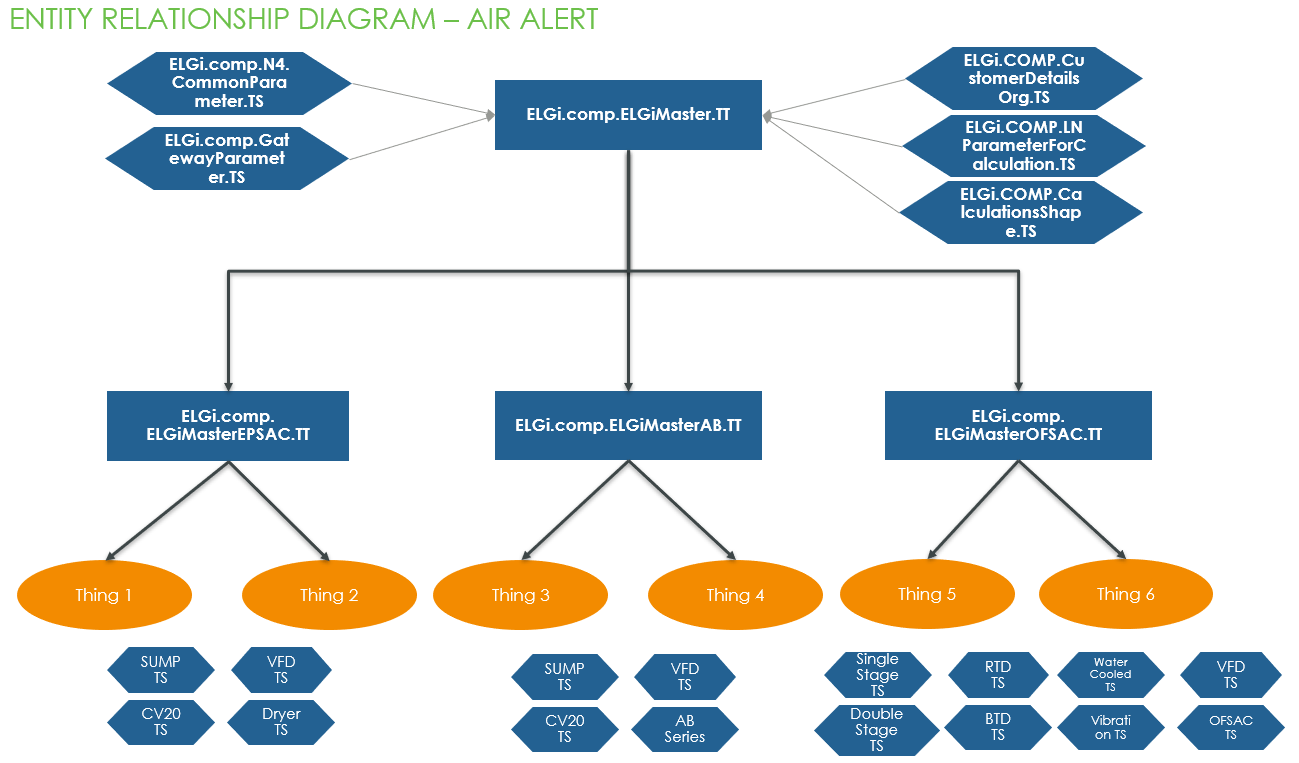
Description automatically generated

Below is the application architecture for the Air Alert Application:



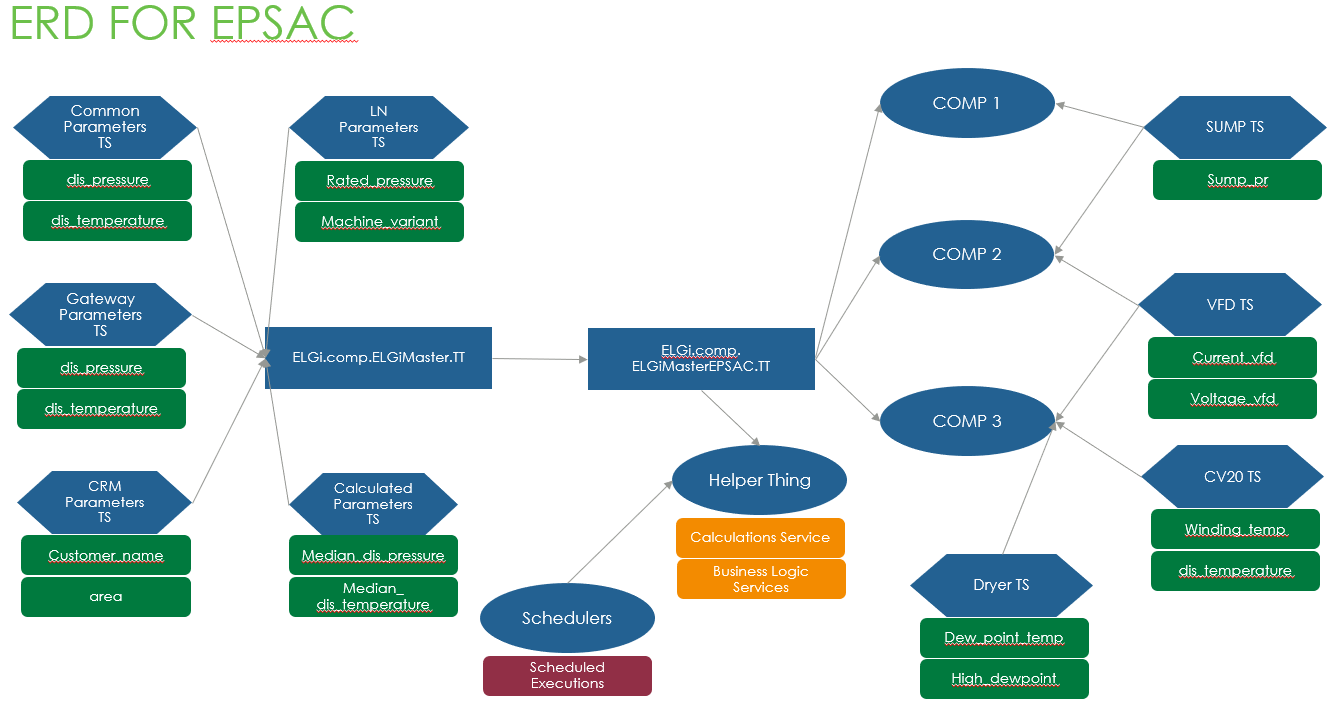
## ERD

This section provides information about the modelling of ELGi compressors in the thingworx platform. Here is a view of the Thing Model used for the Air Alert application:

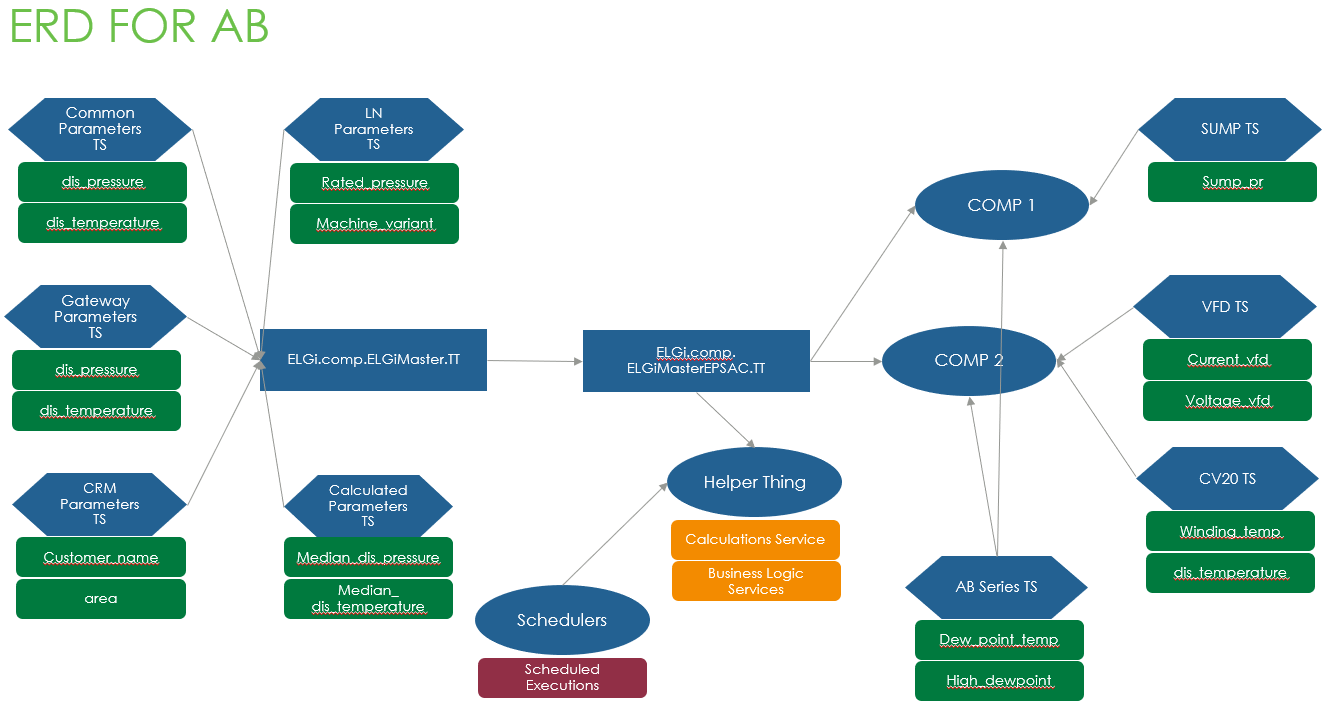


This is overall view of the thing model developed for Air Alert. Below is the machine group wise relationship diagram

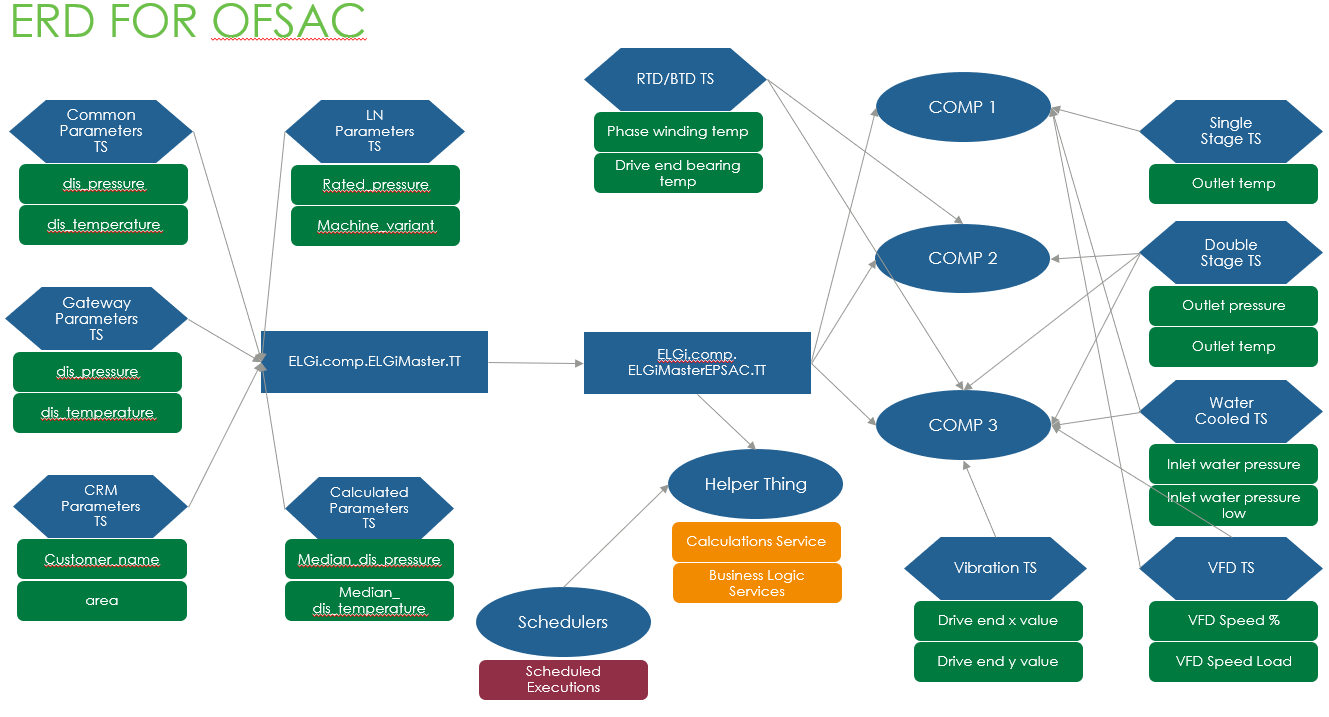
**EPSAC**:



**AB**:



**OFSAC**:



## Thing Shape Implementation

* Added at Master Template Level

Here the list of shapes added in ELGi.comp.ELGiMaster.TTtemplate

* 1. ELGi.comp.N4.CommonParameter.TS
  2. ELGi.comp.GatewayParameter.TS
  3. ELGi.COMP.CustomerDetailsOrg.TS
  4. ELGi.COMP.LNParameterForCalculation.TS
  5. ELGi.COMP.CalculationsShape.TS
* Added at Thing Level during onboarding

The below shapes are added at thing level during thing creation based on machine parameters:

* + 1. EPSAC Related
       1. ELGi.comp.SumpPr.TS
       2. ELGi.comp.VFD.TS
       3. ELGi.comp.VFD\_CV20.TS
       4. ELGi.comp.Dryer.TS
    2. AB Related
       1. ELGi.comp.ABSeriesParameters.TS
       2. ELGi.comp.SumpPr.TS
       3. ELGi.comp.VFD.TS
       4. ELGi.comp.VFD\_CV20.TS
    3. OFSAC Related
       1. ELGi.comp.OFSAC.TS
       2. ELGi.comp.SingleStage.TS
       3. ELGi.comp.DoubleStage.TS
       4. ELGi.comp.WaterCooled.TS
       5. ELGi.comp.RTD.TS
       6. ELGi.comp.BTD.TS
       7. ELGi.comp.Vibration.TS
       8. ELGi.comp.VFD.TS

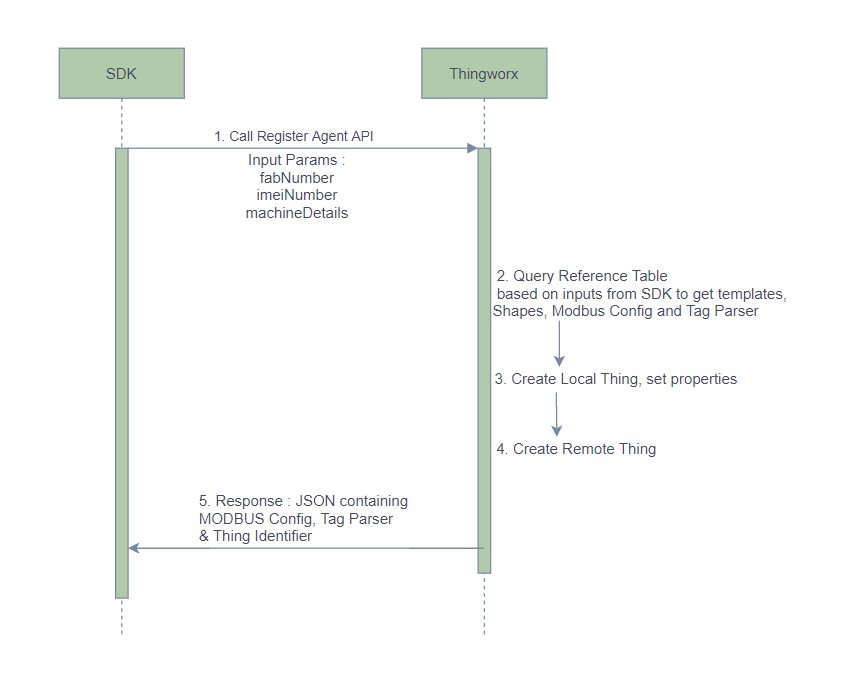
## Entity Naming Convention

Entity modelling will make main use of the objects illustrated on the following diagram. The below naming convention has been followed based on fully qualified names.

|  |  |
| --- | --- |
| Entity | Naming Convention |
| Mashup | ELGI.COMP.\*.MU |
| Master | ELGI.COMP.\*.Master |
| Thing | ELGI.COMP.<CustomerName>.<Location>.<ProjectNo>.Thing |
| Thing Template | ELGI.COMP.<CustomerName>.<Location>.<ProjectNo>.TT |
| Component Thing Template | ELGI.COMP.<ComponentName>.TT |
| Thing Shape | ELGI.COMP.<ComponentName>.TS |
| Stream | ELGI.COMP.\*.Stream |
| Value Stream | ELGI.COMP.\*.VS |
| Data Table | ELGI.COMP.\*.DT |
| Info Table | ELGI.COMP.\*.IT |
| Data Shape | ELGI.COMP.\*.DS |
| Style Definition | ELGI.COMP.\*.Style |
| State Definition | ELGI.COMP.\*.State |
| Menu | ELGI.COMP.\*.Menu |
| Blog | ELGI.COMP.\*.Blog |
| Wiki | ELGI.COMP.\*.Wiki |
| User | FirstNameLastName |
| User Group | ELGI.COMP.<OrganizationUnitName> .\*.UG |
| Organization | ELGI.COMP.\*.ORG |
| Organization Unit | ELGI.COMP.\*.OU |
| Networks | ELGI.COMP.\*.NW |
| Application Key | ELGI.COMP.\*.AppKey |
| Project | ELGI.COMP.\*.Project |
| Model Tag | ELGI.COMP.\*.MTag |
| Data Tag | ELGI.COMP.\*.DTag |

# Device Onboarding

Workflow for the device onboarding:



Use Case:

Create Remote Thing, Local Thing based on the machine details and return MODBUS config and tag parser to the SDK

**API:** <https://elgi-dev.cloud.thingworx.com/Thingworx/Things/ELGi.comp.RegisterAgentHelper.Thing/Services/RegisterAgent_v2>

**Request Body**:

{

"fabNumber": "BPHS030794",

"imeiNumber": "490154203237512",

“machineDetails\_1": 4160,

"machineDetails\_2": 8

}

**Response**:

{

"thing\_name": "",

"modbus\_config": "",

"imei\_no": "",

"success": true,

"thing\_identifier": "",

"fab\_no": "",

"tag\_parser": ""

}

Steps:

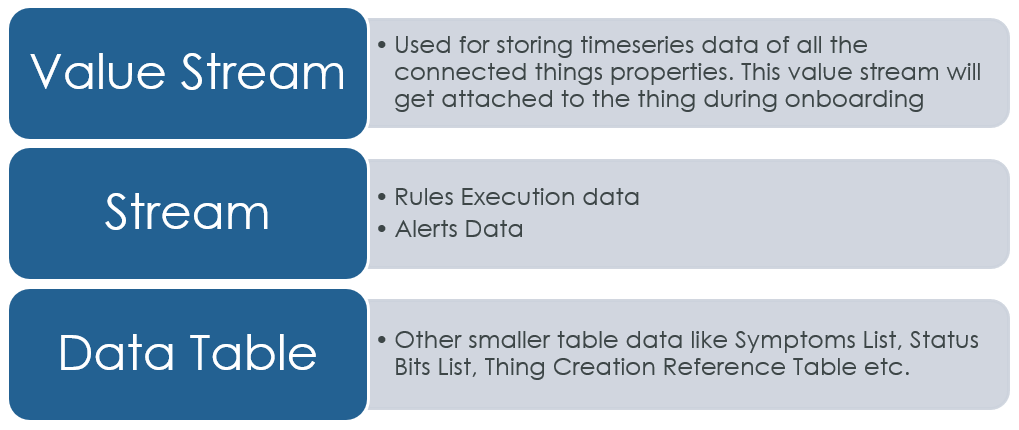
* SDK calls RegisterAgent Service on platform with fab Number, imei number and machine details
* Thing Template, Shapes, Modbus Config and Tag parser is queried from the Thing Creation reference (ELGI.COMP.CreateOrUpdateThingMappings.DT)
* Local Thing is created from the queried template and shapes with fabNumber as Thing name. Add ELGi.COMP.ThingData.VS value stream.
* Remote Thing is Created with RemoteThingWithFileTransfer
* Response sent to SDK with remote thing name, thing identifier, modbus config, tag parser and imei number
* After the device is connected, BindRemoteProperties service is called to bind the remote properties. These remote properties are then bound with Local Thing using BindLocalProperties
* After this flow is completed data is started to be pushed in the Local Thing

# Data Storage

Three different types of data are being considered:

* **Live Data**: Property values coming from the device. Data is ingested in various frequencies like 5s, 1Hr, On Change. Historical data is maintained for most of these properties.
* **Calculated Data**: Calculations are done on the historical data coming from device. These calculated values are then stored as a property on the thing. E.g. Median Calculation
* **Static Data/ Manual Entry**: These are data sets are entered manually in the platform. E.g. Reference Table for Thing Creation

**Data Storage Types Used**:



List of Storage entities used in Air Alert

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Entity Type | Name | Description | Features Used In | DataShape | Columns |
| Value Stream | ELGi.COMP.ThingData.VS | Stores property values of each connected thing | Device Property Data |  |  |
| Stream | ELGi.COMP.ExecutedRules.Stream | Maintains the record of all the executed rules and alerts | Rules Framework, Alerts | ELGi.COMP.ExecutedRules.DS | RuleId  RuleName  Frequency  Condition  ExecutionTimeStamp  PropertyValues  RuleExecutionId  FabNumber  RuleResult  Symptom  AlertStatus  TicketNumber |
| Data Table | ELGI.COMP.CreateOrUpdateThingMappings.DT | Reference Table for Thing Creation | Thing Onboarding | ELGI.COMP.CreateOrUpdateThingMappings.DS | machine\_group  machine\_type  neuron\_type  template\_name  shapes\_list  sump\_available  vfd\_available  winding\_temp\_available  dryer\_available  single\_stage\_available  double\_stage\_available  water\_cooled\_available  rtd\_available  btd\_available  vibration\_available  energy\_meter\_available  modbus\_config  tag\_parser\_config  machine\_details\_1  machine\_details\_2 |
| Data Table | ELGi.COMP.CalculationsServiceDataTable.DT | Maintains the list of Calculation Rules Added | Calculation Rules | ELGi.COMP.CalculationsServiceDataShape.DS | Name  Type  Thing\_OR\_ThingTemplates  INPUT\_Properties  OUTPUT\_Properties  Frequency  Interval  IntervalOption  Id  MachineStatus  Things  ThingTemplates |
| Data Table | ELGi.COMP.MasterRuleStructure.DT | Stores all the conditions to be used for Rules Creation | Rule Definition | ELGi.COMP.MasterRuleStructure.DS | ID  Condition  Variables |
| Data Table | ELGi.COMP.MasterRulesDefinition.DT | Maintains the record of all the rules created | Rules Framework | ELGi.COMP.MasterRulesDefinition.DS | ID  Entity  ConditionType  Condition  Frequency  Rule\_Created  Execution\_Date  Status  Variables  Symptom  Things  ThingTemplates |
| Data Table | ELGi.COMP.TrendsPropertiesList.DT | Maintains a list of all the properties for which charts to created | UI | ELGi.COMP.TrendsPropertiesList.DS | ID  DisplayName  PropertyName  EPSAC\_AVAILABLE  ABSeries\_AVAILABLE  OFSAC\_AVAILABLE |
| Data Table | ELGi.COMP.StatusBits.DT | Maintains the list of Machine Status | UI | ELGi.COMP.StatusBits.DS | Bit\_Position  General\_Status  Warning\_Status  Fault\_Status |
| Data Table | ELGi.COMP.SymptomList.DT | Stores all the machine symptoms | Rule Definition | ELGi.COMP.SymptomList.DS | Id  SymptomName  SymptomDescription |

# Third Party Integration

Air Alert consumes and exposes APIs for below systems:

* Dynamics 365 CRM
* CCS

All the integration related properties and services are created in ELGI.COMP.ThirdPartyIntegrationHelper.Thing

Properties List on the Thing:

|  |  |  |
| --- | --- | --- |
| Property Name | Base Type | Remarks |
| CRM\_Client\_Id | STRING | Client Id |
| CRM\_User\_Name | STRING | Username for accessing Bearer Token |
| CRM\_User\_Password | PASSWORD | Password |
| CRM\_Resource | STRING | Base URL |
| CRM\_GrantType01 | STRING | Grant Type |
| CRM\_GrantType02 | STRING | Grant Type |
| CRM\_Bearer\_Token | STRING | Bearer Token |
| CRM\_Refresh\_Token | STRING | Refresh Token |
| CRM\_GetBearerToken\_API | STRING | API to get Bearer Token |
| CRM\_PostFabToCRM\_API | STRING | API to post FAB Number to CRM |
| CCS\_User\_Name | STRING | Username for authentication |
| CCS\_Password | PASSWORD | CCS Password |
| CCS\_GetMachineDetails\_API | STRING | API to get machine details from CCS |
| CCS\_PostFabNumber\_API | STRING | API to post FAB Number to CRM |

Authentication Mechanism:

|  |  |  |
| --- | --- | --- |
| System | Mechanism | Authentication Method |
| Thingworx | App Key Based | Pass appkey parameter in the header |
| CCS | Username, password based basic authentication | Pass Username and password as basic authentication |
| CRM | Token based Authentication | - Fetch the Bearer token using login API  - Use the generated bearer token from previous step to call APIs |

Token Based Authentication Flow in Thingworx

* Call Api : https://login.microsoftonline.com/common/oauth2/token to fetch Bearer Token using GetCRMBearerToken. This API takes client id, username and password as input.
* Update CRM\_Bearer\_Token property and CRM\_Refresh\_Token properties
* Use the bearer token in header for authenticating APIs from CRM
* Refresh Bearer token every 1 Hr

Please refer AA\_API\_List.xlsx for getting the List of APIs used in Air Alert application.

# Rules Management

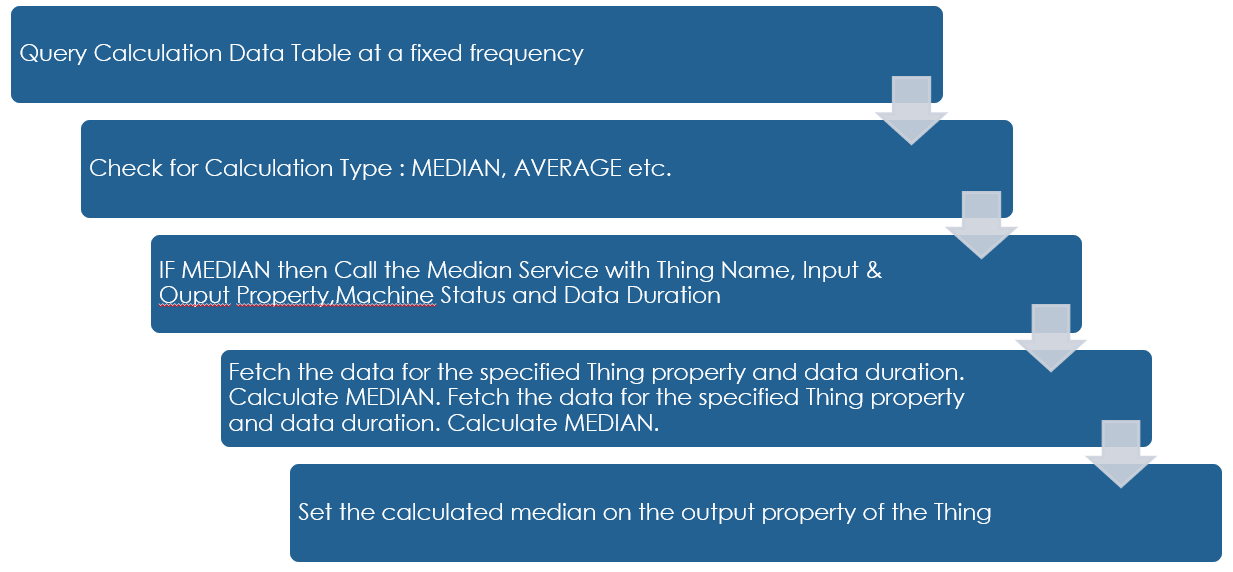
## 5.1 Pre-Calculations

UI Components Available:

* **Type**: MEDIAN, AVERAGE, LOAD/UNLOAD COUNT
* **Machine Status**: Load, Unload, Running
* **Thing Template**: AB, EPSAC, OFSAC
* **Thing**: List of commissioned compressors. Thing selection will be disabled when Thing Template is selected
* **Input Property**: List of properties available on commissioned things
* **Output Property**: List of Calculations property added in ELGi.COMP.CalculationsShape.TS
* **Frequency**: 24Hours (calculation service will execute in this frequency)
* **Data Duration**: 24Hours, 48Hours, 7Days, 30Days

User will select all these parameters and click on ADD. It will add an entry in ELGi.COMP.CalculationsServiceDataTable.DT. All added precalculation rules will be added in this Data Table

Backend Flow for Pre-Calculations:



* Each scheduler will query the calculations data table at a fixed frequency
* All queried rows available as an infotable will be iterated.
* If the Thing Templates are selected, then all the commissioned things implementing the Template will be fetched and Calculations will be applied for each thing.
* If Multiple Things are selected, then the calculation will be done for each thing.

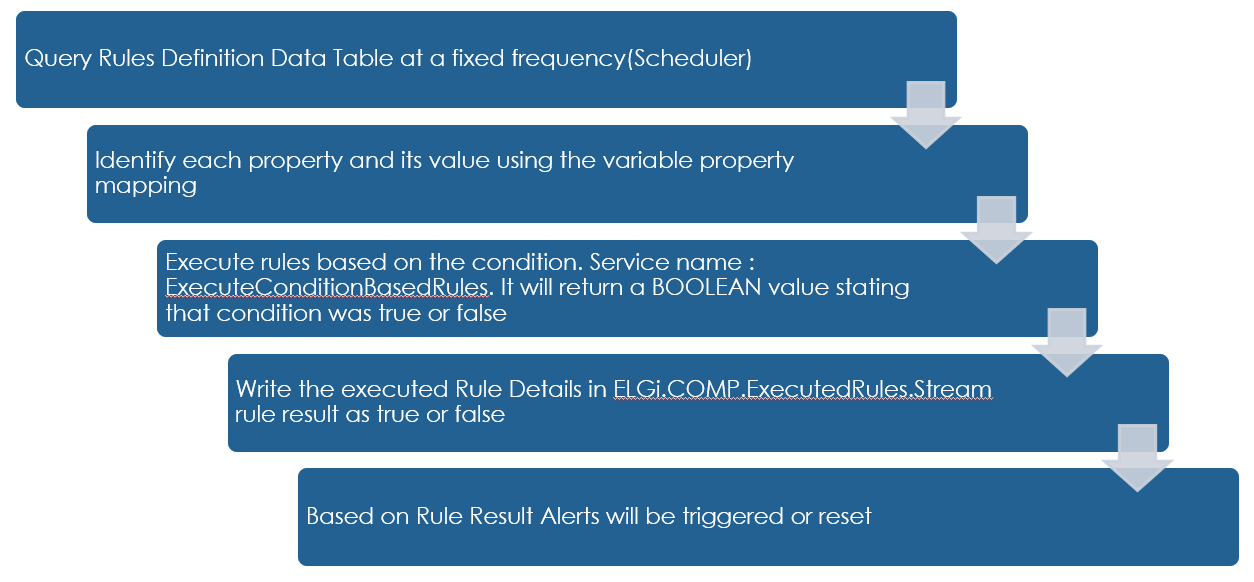
## Rules Implementation

UI Components Available:

* **Frequency**: 24Hours (calculation service will execute in this frequency)
* **Symptom:**
* **Thing Template** : AB, EPSAC, OFSAC
* **Thing**: List of commissioned compressors
* Thing selection will be disabled when Thing Template is selected
* **Condition:** Rule conditions like A>B, A<B, A>B-C, A<B+C where A,B,C are the variables
* **Map Variables:** Map the variables with thing property. Each variable from the selected condition will have to be mapped with a property present in the property list.

User will select all these parameters and click on Add Rule. It will add an entry in **ELGi.COMP.MasterRulesDefinition.DT**. All added rules will be added in this Data Table

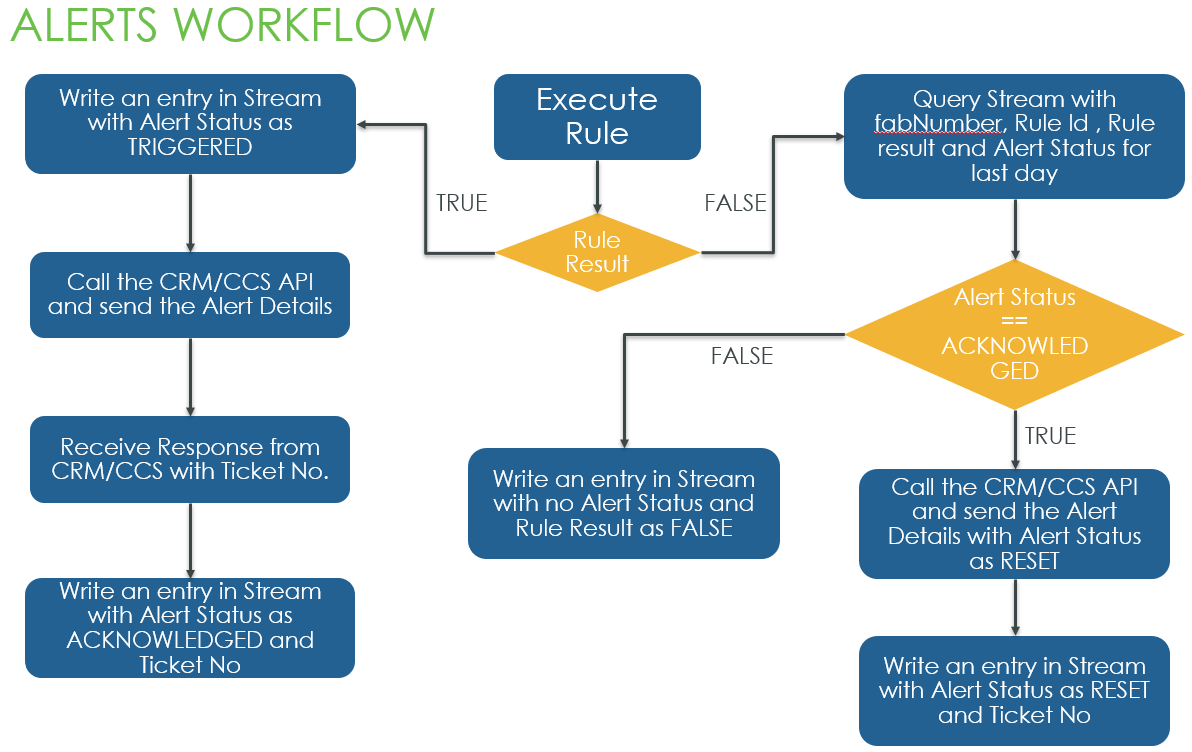
Rules Execution Workflow:



## Alerts

When a rule gets executed and if the rule result comes out as TRUE, then an Alert entry is made in the ELGi.COMP.ExecutedRules.Stream. Here are the details of the Stream:

|  |  |  |
| --- | --- | --- |
| Field Name | Data Type | Remarks |
| RuleId | STRING | Rule’s Unique Id, can be same for multiple entries |
| RuleName | STRING | Name of Rule |
| Frequency | STRING | Frequency of execution |
| Condition | STRING | Condition |
| ExecutionTimeStamp | DATETIME | Time of rule execution |
| PropertyValues | STRING | Property values at time of execution |
| RuleExecutionId | STRING | Unique id for executed rule, will be unique in the stream |
| FabNumber | STRING | Machine Name |
| RuleResult | BOOLEAN | Result of rule execution |
| Symptom | STRING | Symptom of machine |
| AlertStatus | STRING | If Rule Result is TRUE then Alert Status will be TRIGGERED. When ticket details will be received from the CRM/CCS Alert Status will be ACKNOWLDEGED. When Issue is resolved, Alert status will be RESET |
| TicketNumber | STRING | Service Ticket Number from CRM/CCS |



Both the alert data and executed rules data will be maintained in a Stream.

# Roles & User Management

## 6.1 Organization Tree

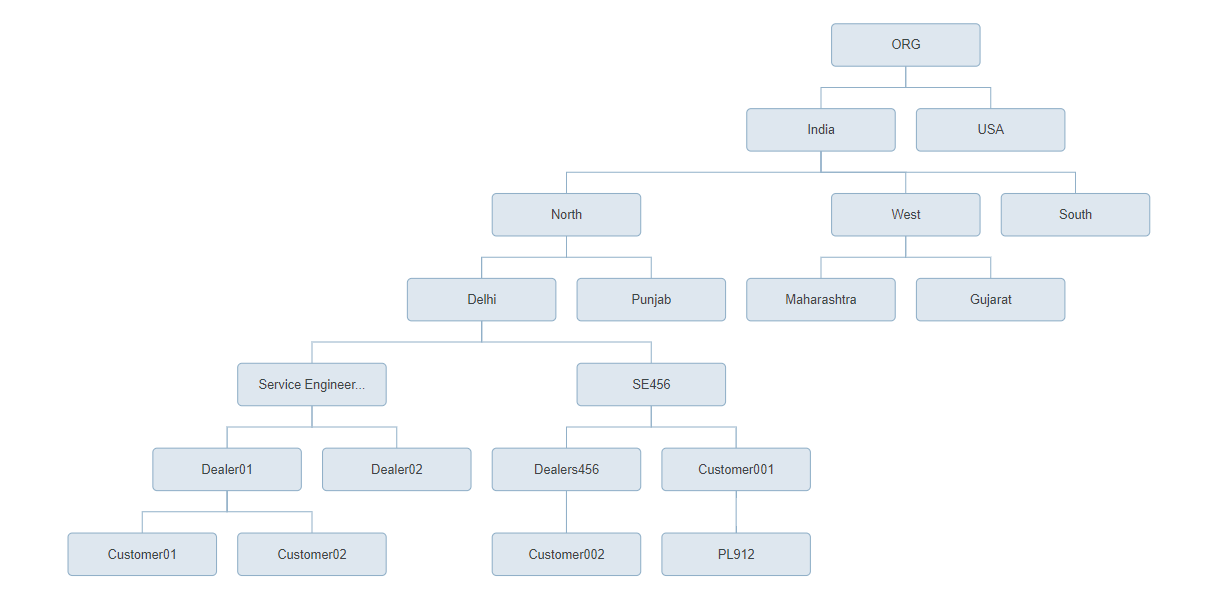
Components:

* Org Unit: It represents Country, Region, Area, ELGi Service Engineer, Dealer and Customer. Each unit can have multiple child but only one parent.
* User Group: A user group will be used to store one or more users. It is created for each Org Unit. Each user group created for Organization Structure will named as OrgUnit Name with \_UG as suffix.
* User: User represents actual user login. All the non-ELGI users will be created in Thingworx.

Below are some assumptions for creating the Org Structure in Air Alert:

* Each Country will be predefined in the Thingworx
* Each Country will have one or more Unique Region
* Each Region will have one or more Unique Areas
* Each Area will have one or more ELGi Service Engineers
* Each ELGi Service Engineer will have one or more Unique Distributors
* Each Dealer will have one or more Unique Customer
* Each Customer may have a Parent Company (unique Name/id)
* Each Customer will represent as unique Plant

Sample Org Structure created in ELGI\_ORG entity in Thingworx:



API Details for Device Commissioning:

* **API:** <https://elgi-dev.cloud.thingworx.com/Thingworx/Things/ELGI.COMP.ThirdPartyIntegrationHelper.Thing/Services/SetOrgStructureAndCustomerDetails>
* **Request Body:**

{

"input": {

"fabNumber": "",

"isCommisioning": true,

"commissioningDate": "",

"serviceDate": "",

"CRMorCCS": "CRM",

"isRetrofitted": false,

"warrantyStatus": "",

"isKeyCustomer": true,

"orgUnitDetails": [

{

"unitLevel": "REGION",

"unitName": "North",

"unit\_id": "REG001",

"parent\_id": "IND"

},

{

"unitLevel": "AREA",

"unitName": "Pune",

"unit\_id": "AR456",

"parent\_id": "REG001"

},

{

"unitLevel": "DEALER",

"unitName": "Sharad Enterprises",

"unit\_id": "DL276",

"parent\_id": "AR456"

},

{

"unitLevel": "CUSTOMER",

"unitName": "L & T",

"unit\_id": "CS858",

"parent\_id": "DL276"

}

],

"serviceEngineerDetails": {

"name": "Rohit Kumar",

"email": "rohit@elgi.com",

"mobileNo": "9999999999",

"area\_id": "AR456",

"region\_id": "REG001",

"dealer\_id": "DL276",

"customer\_id": "CS858"

},

"orgUserDetails": [

{

"userAction": "ADD",

"name": "Rajat Dhawan",

"email": "rajat@elgi.com",

"unit\_id": "REG001"

},

{

"userAction": "ADD",

"name": "Ravi Singh",

"email": "ravi@elgi.com",

"unit\_id": "AR456"

},

{

"userAction": "ADD",

"name": "Venkat Narsimha",

"email": "venkat@gmail.com",

"unit\_id": "DL276"

},

{

"userAction": "ADD",

"name": "Risabh Rai",

"email": "risabh@lnt.com",

"unit\_id": "CS858"

}

]

}

}

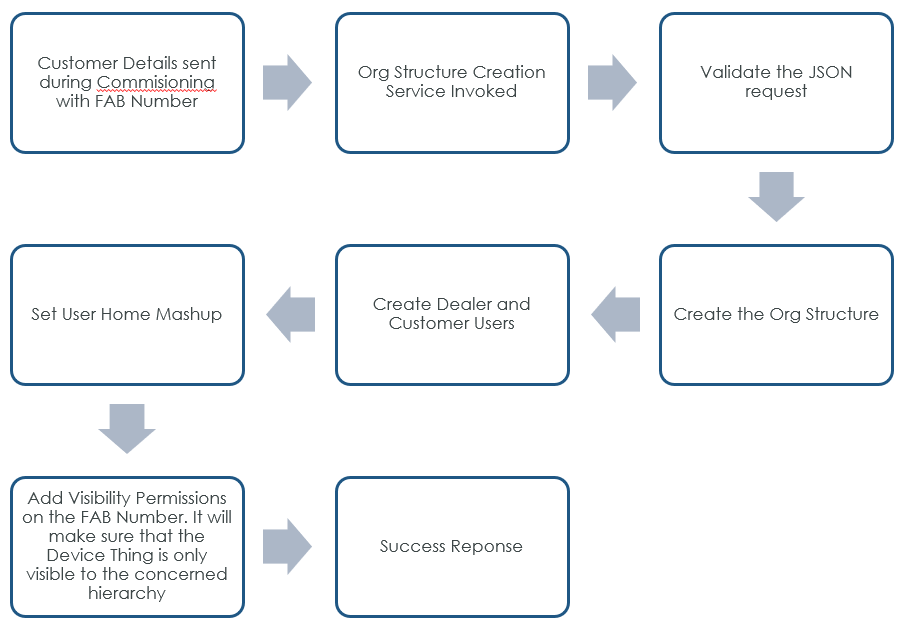
* **Response:** Success

**Applications Users:**

|  |  |  |
| --- | --- | --- |
| User | Level | User Type |
| Country Head | L3 | ELGi User |
| RSM | L2 | ELGi User |
| ASM | L2 | ELGi User |
| Dealer | L2 | Non ELGi User |
| Service Engg | L2 | ELGi User |
| Customer Top | L1A | Non ELGi User |
| Customer | L1B | Non ELGi User |

Note: All the dealer and customer users will be created in Thingworx.

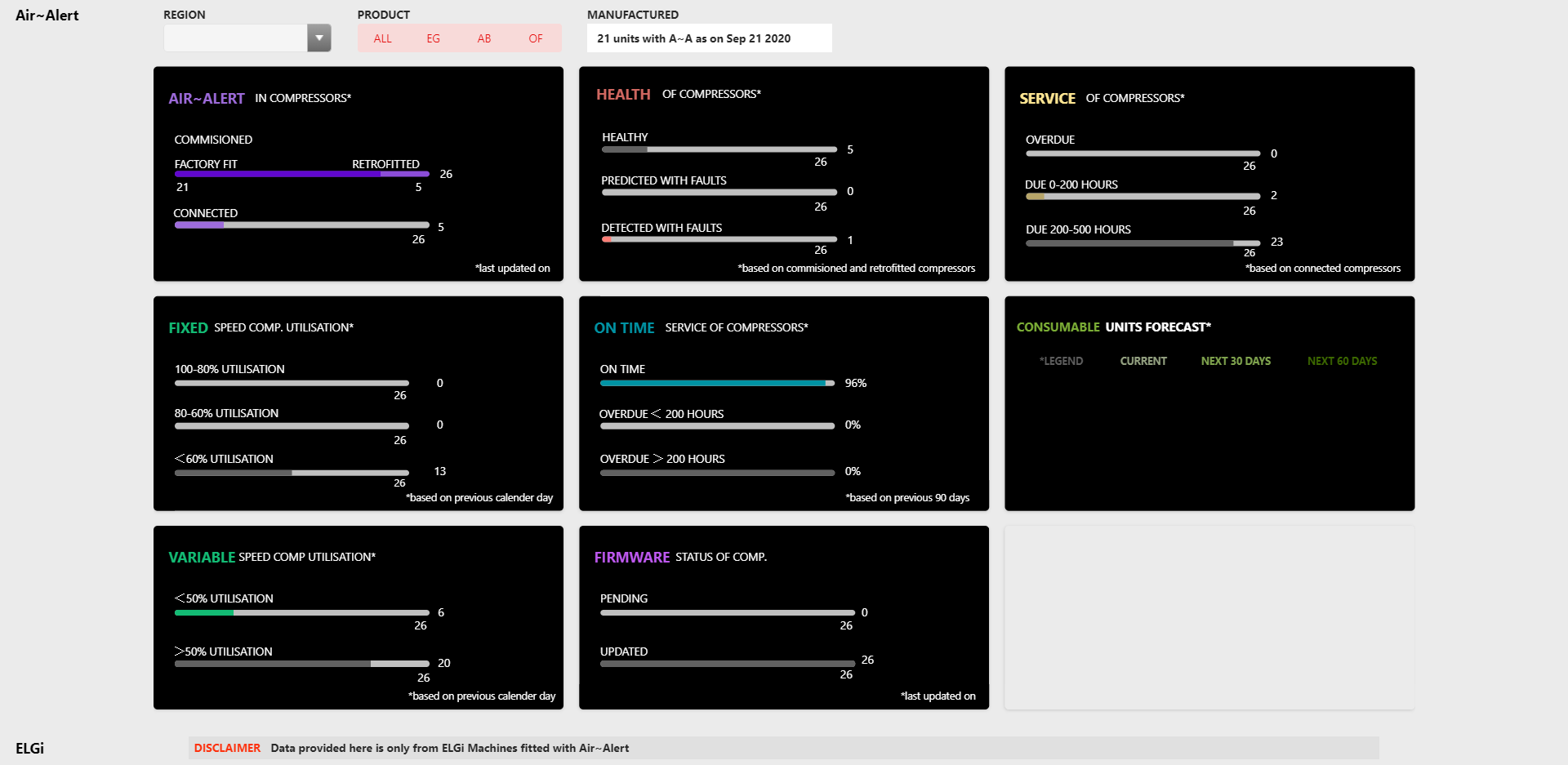
Below is the workflow of the service executions when Device Commissioning API is executed:



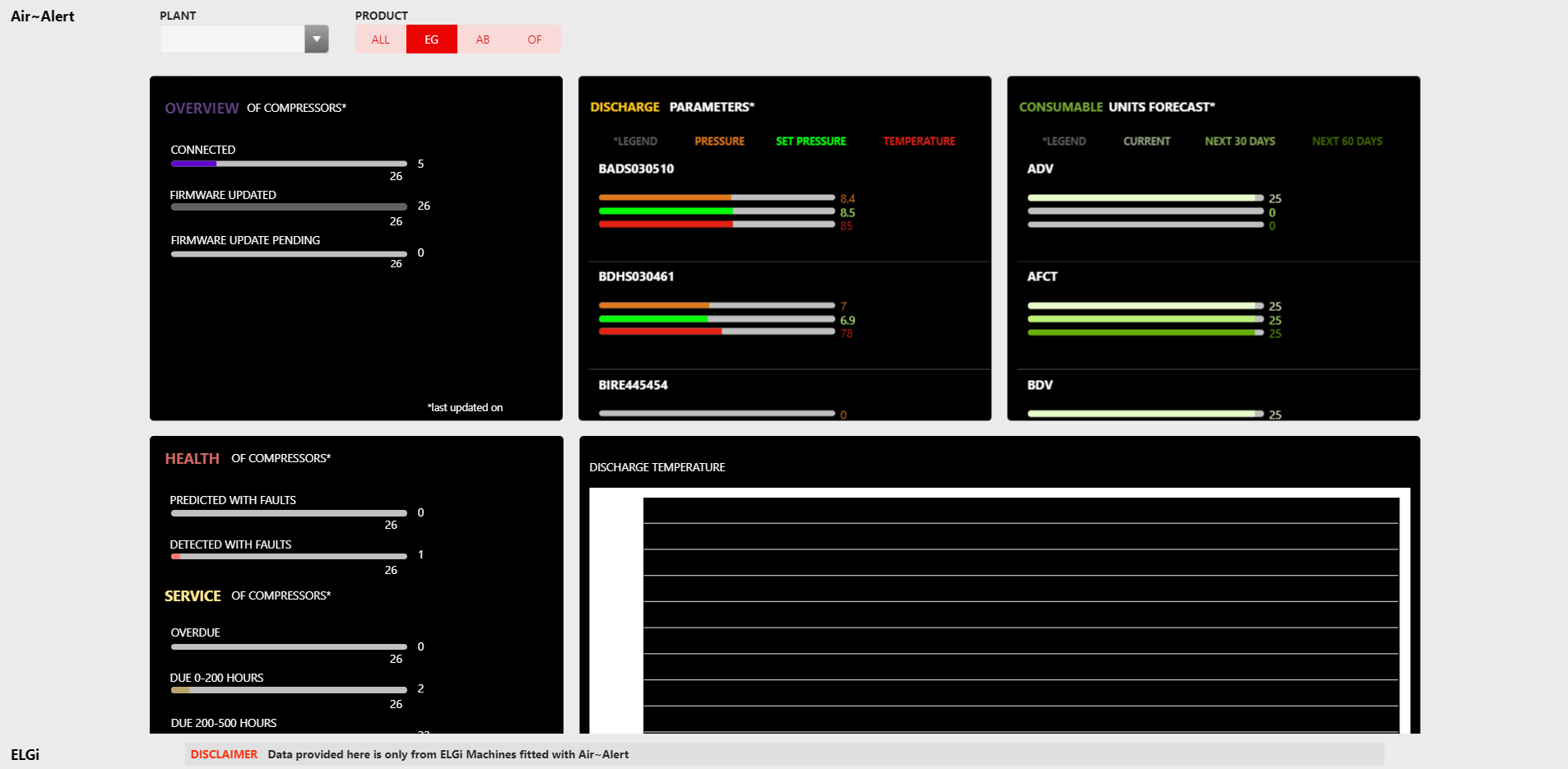
# Management Dashboards/ Admin Pages

## 7.1 Landing Pages

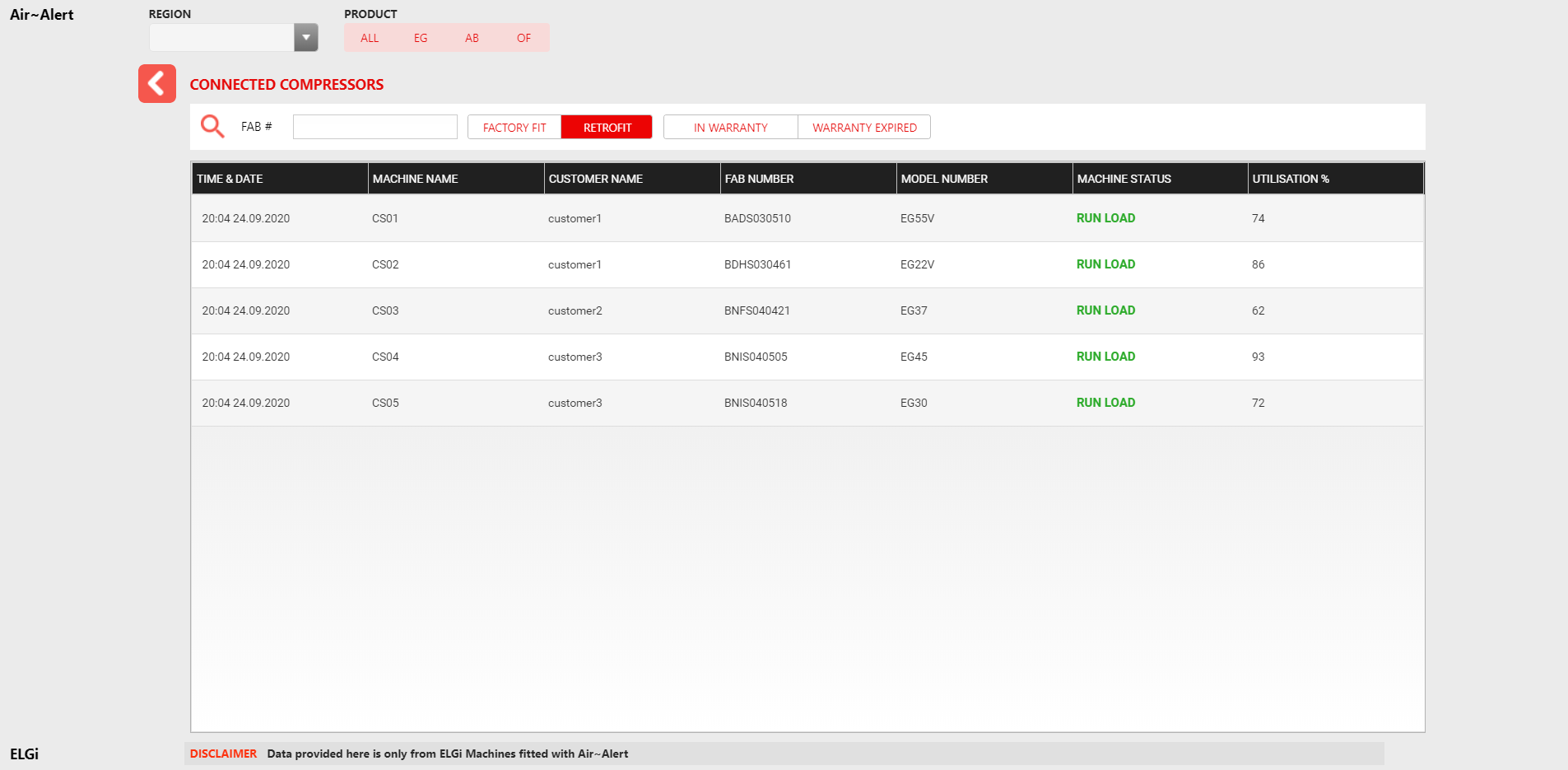
* L3/L2



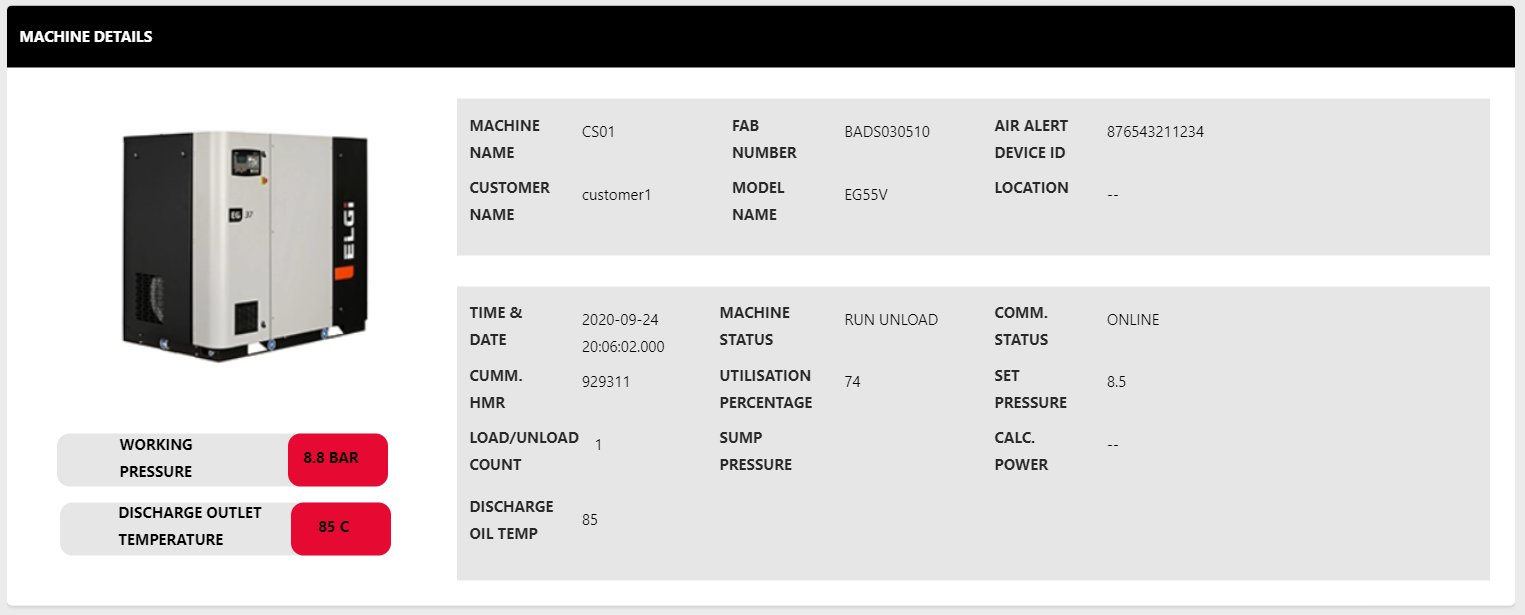
* L1B

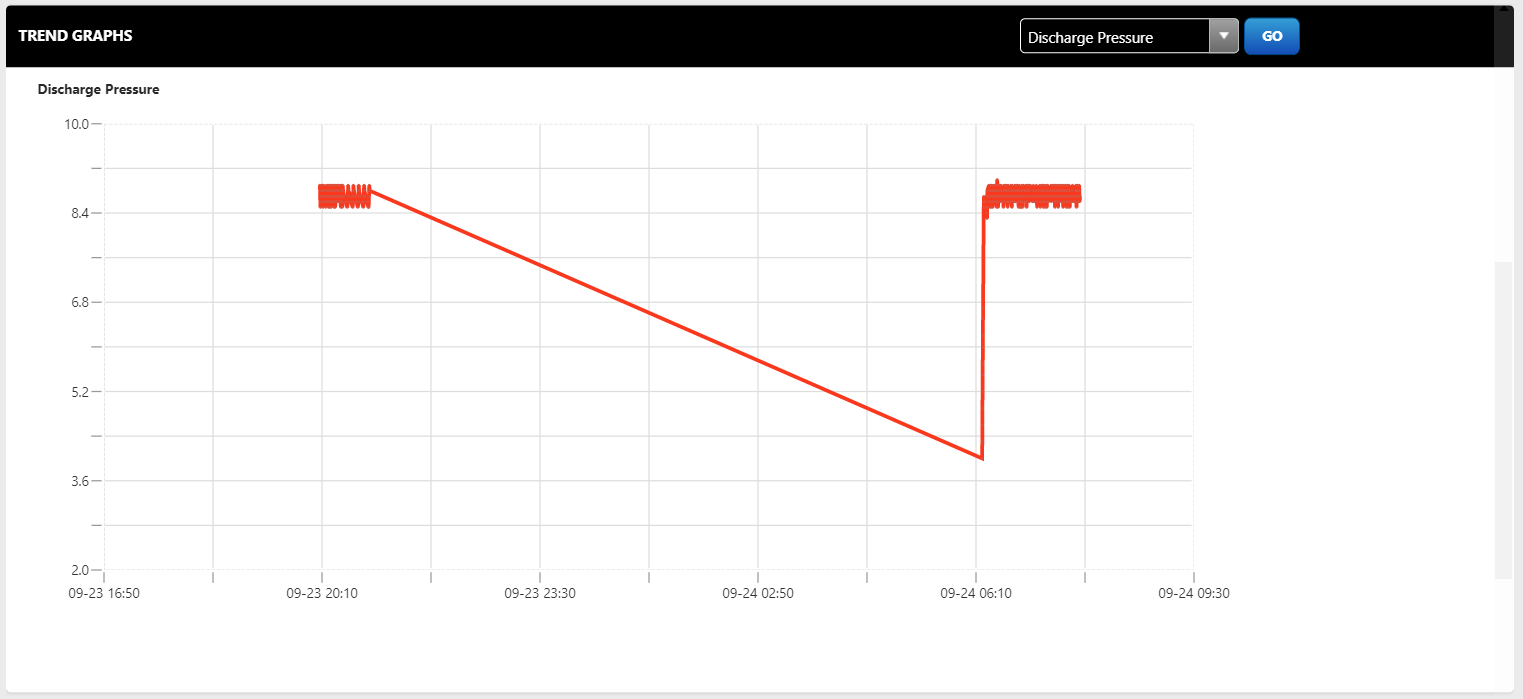


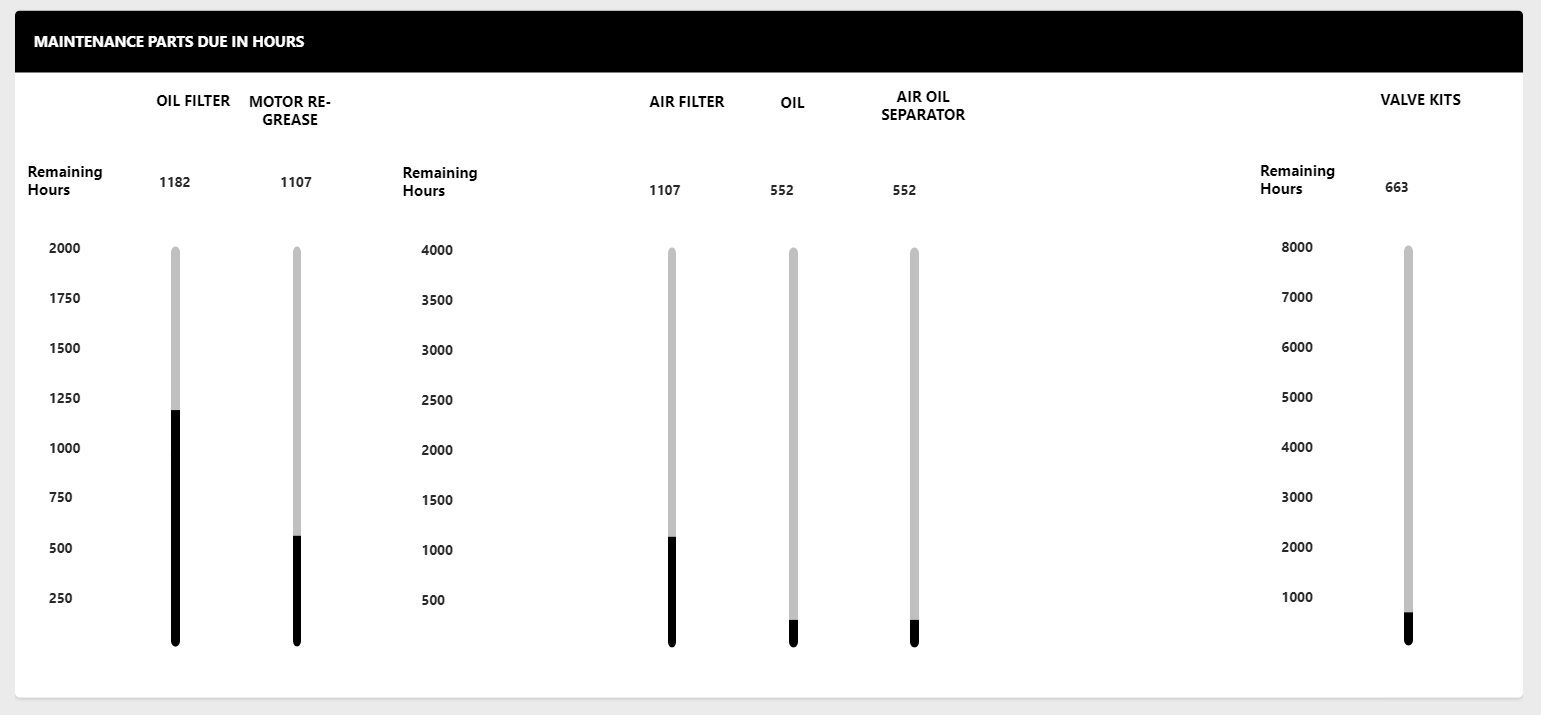
## Machine List Pages



## Machine Details Pages

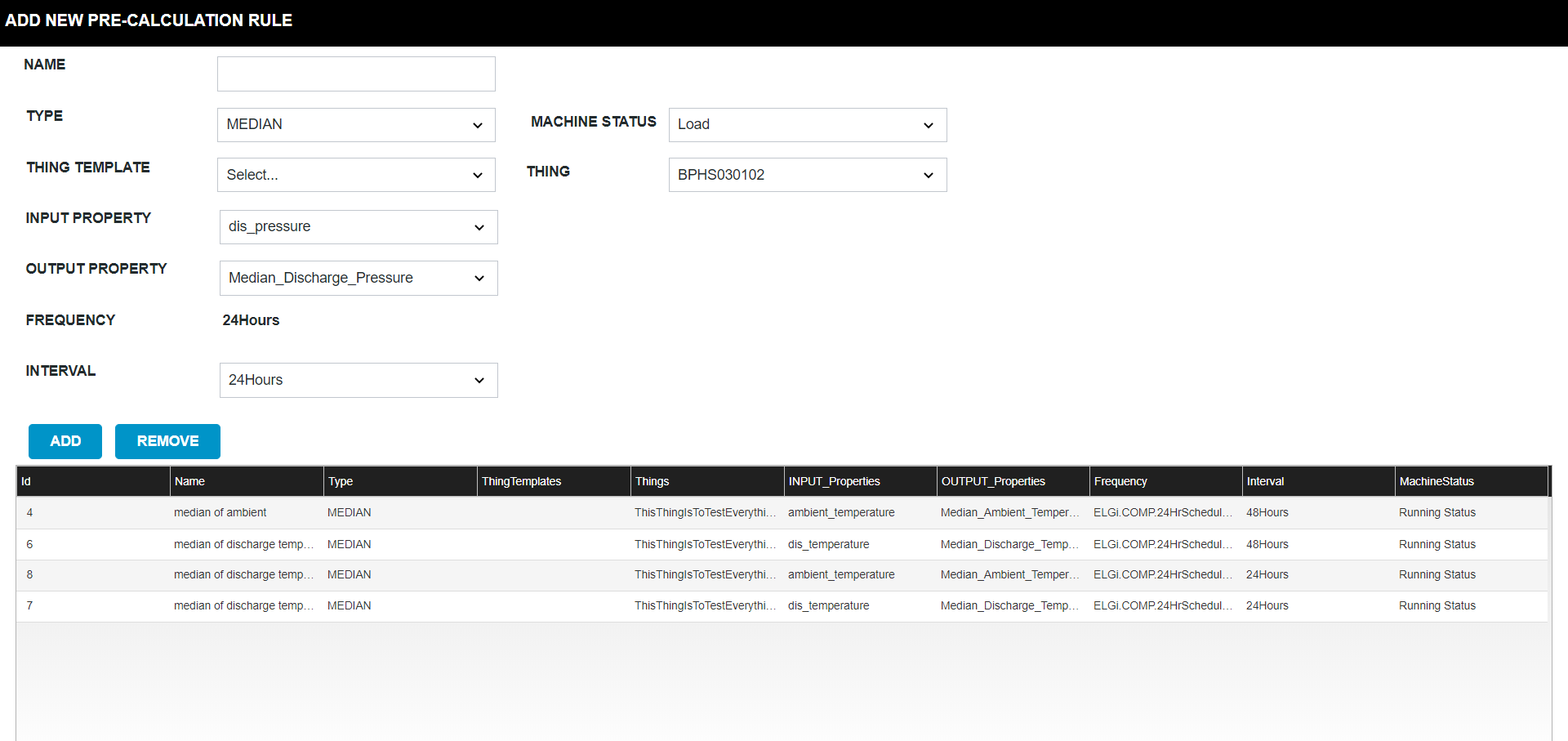




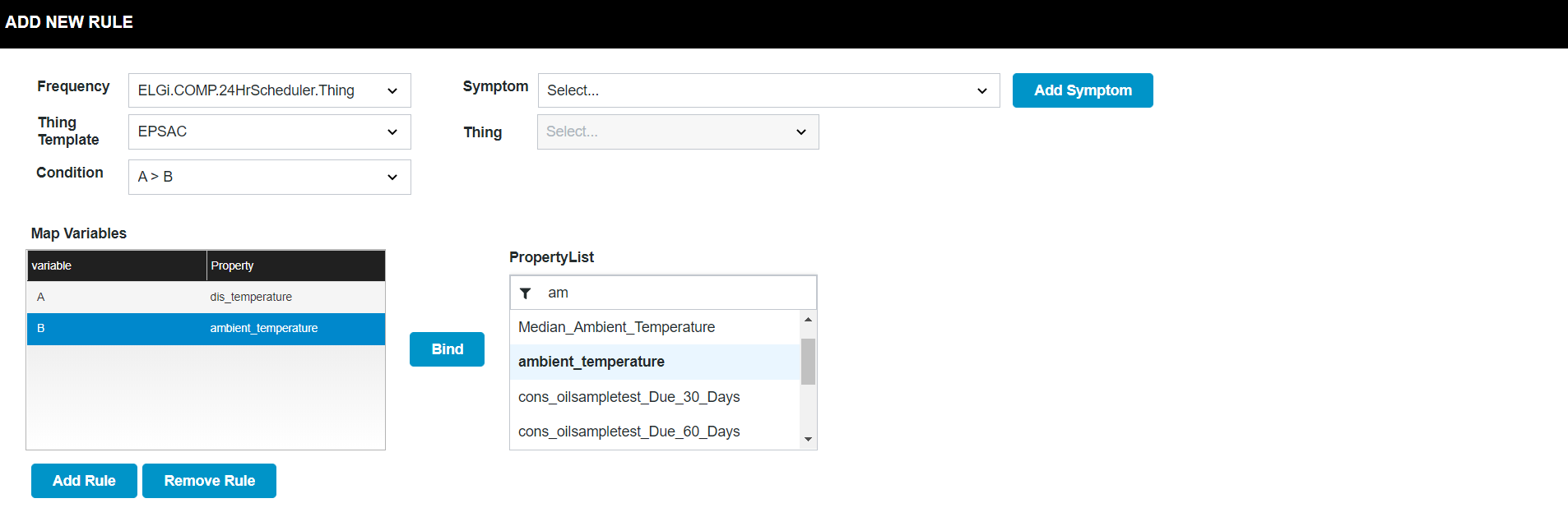


## Rules Pages

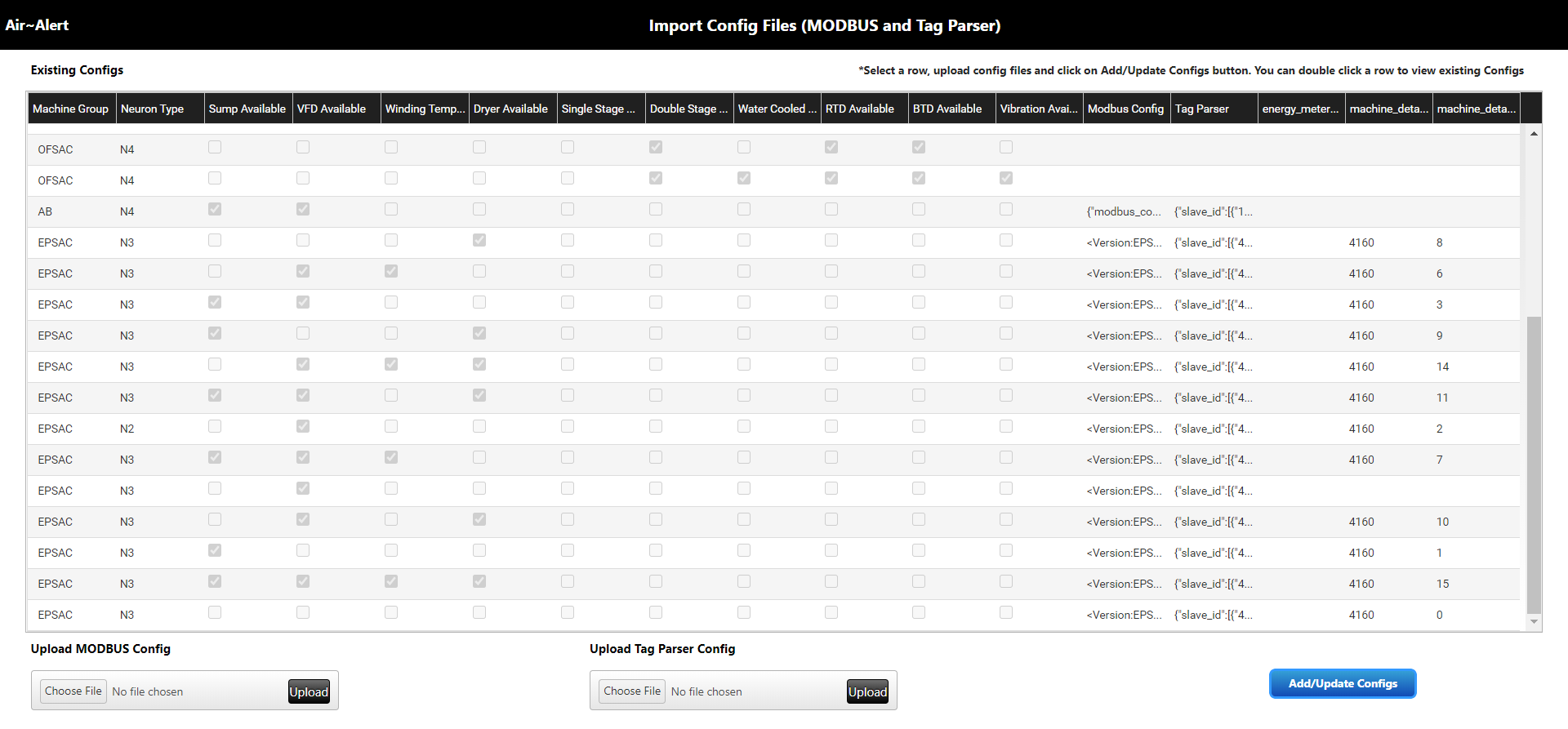
* PreCalculation UI



* Rule Definition



## Add/Update Machine Config Files



## Dashboard Calculations

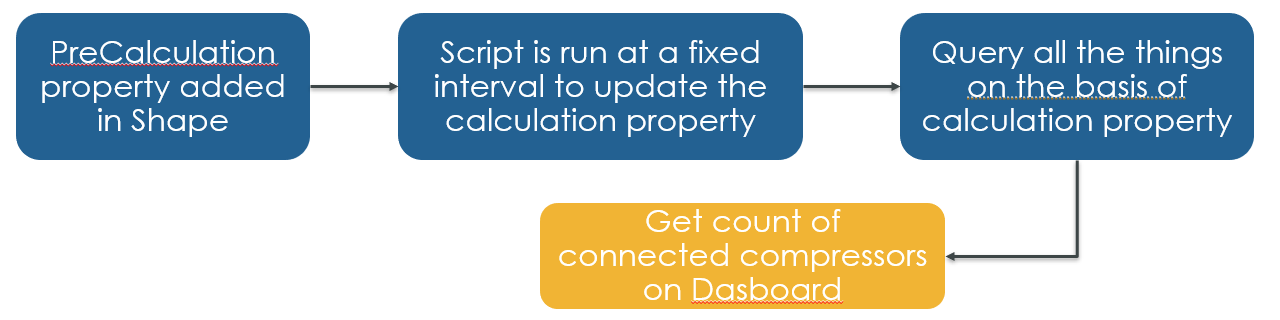
* All the calculations done are been stored as a property on ELGi.COMP.CalculationsShape.TS
* Each property is calculated at a fixed time interval through a scheduler service.
* These precalculated property are used to query the thing details and get count to display on the Dashboard.

Use Case 1

Get count of connected compressors

Steps:

* A property is added in ELGi.COMP.CalculationsShape.TS : is\_Machine\_Connected(BOOLEAN)
* Script in ELGi.COMP.CalculationsHelper.Thing : SetisConnectedTrue is executed on a frequency of 2 Hours. This service updates is\_Machine\_Connected property on each thing.
* is\_Machine\_Connected is used in a query at Runtime get the list of things where the property value is true.



Use Case 2

Get Compressors List with machine status and other details in a tabular view with filters.

Steps:

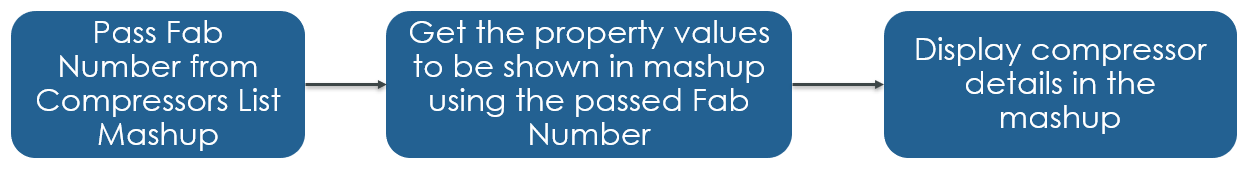
* User will land on this screen by clicking any of the Overview tiles to view the list of compressors.
* If any filter is there e.g Overdue for Service Overview mashup, then the clicked filter will be passed to the Compressors List mashup using Mashup Parameter.
* All the services to fetch machine details list are added in ELGi.COMP.MashupHelper.Thing
* Each service in this Helper Thing accepts the filter values as input and queries the list of compressors with details.



Use Case 3

Steps:

* User will be navigated to Machine details by clicking any row from the compressors list table.
* The FabNumber will be passed on to the Machine Details page through Session Parameters.
* A service will fetch all the details of Compressor using the FabNumber and the property values in mashup will be populated.



# Services Used

All the services used for Air Alert application have been detailed out in the LLD package where each Thing services have listed out as separate files. Below are the files list :

* LLD\_ELGi.COMP.CalculationsHelper.Thing
* LLD\_ELGi.COMP.GenericHelper.Thing
* LLD\_ELGi.COMP.MashupHelper.Thing
* LLD\_ELGi.COMP.OrgStructureHelper.Thing
* LLD\_ELGi.comp.RegisterAgentHelper.Thing
* LLD\_ELGi.COMP.RulesHelper.Thing
* LLD\_ELGI.COMP.ThingModelHelper.Thing
* LLD\_ELGI.COMP.ThirdPartyIntegrationHelper.Thing