



GE:AR Architecture &  
Scope Document  
*Version 0.4*



Prepared By  
Nishanth Saka  
Project Manager  
[x]cubeLabs

# 1. Introduction

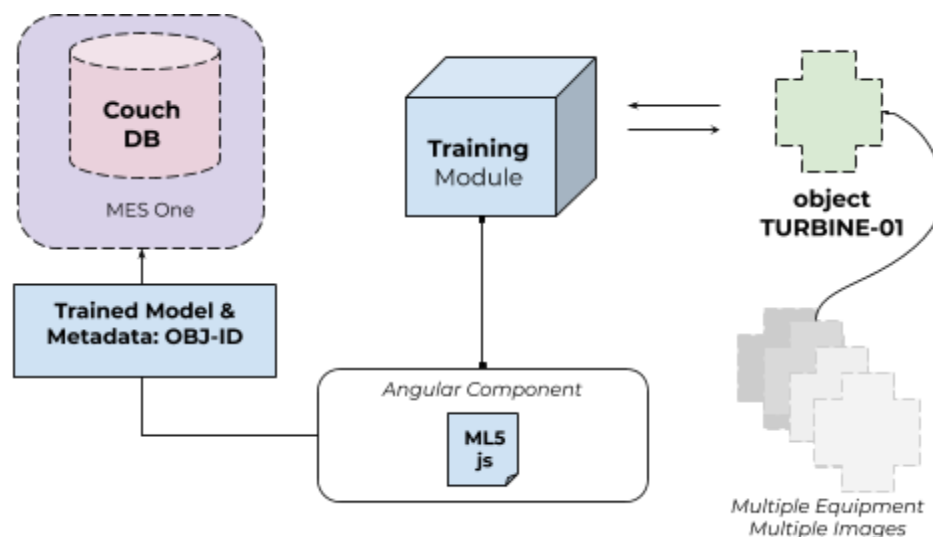
The GE:AR project has the following components...

- Training Module
- Learning Transfer - Trained Model on Cloud (CouchDB)
- Detection Module
- Integration Module

Details regarding each of these modules are given in further sections.

## 2. Training Model & Learning Transfer:

Below is the architecture that is followed during implementation of Training Model & Learning Transfer processes...



Training Module Architecture

### 2.1 Object

The object can be any of the equipment that needs to be used for training. For each object, a set of 4 images, in different angles, will be provided to model during training process. The idea is to train for a maximum of 4 such equipment.

## 2.2 Label (Meta Data)

While training, each image (for a specific equipment) shall be labelled with metadata, preferably 'ObjID'. This trained model shall associated the focal-points of the image with the corresponding metadata.

## 2.3 Training

Once all the images (for corresponding equipment) are added for training, the ML Model Training process can be started.

## 2.4 Test

Once the Training process is completed, the ML Model can be tested for accuracy - by checking with reference image and review if appropriate metadata is being displayed.

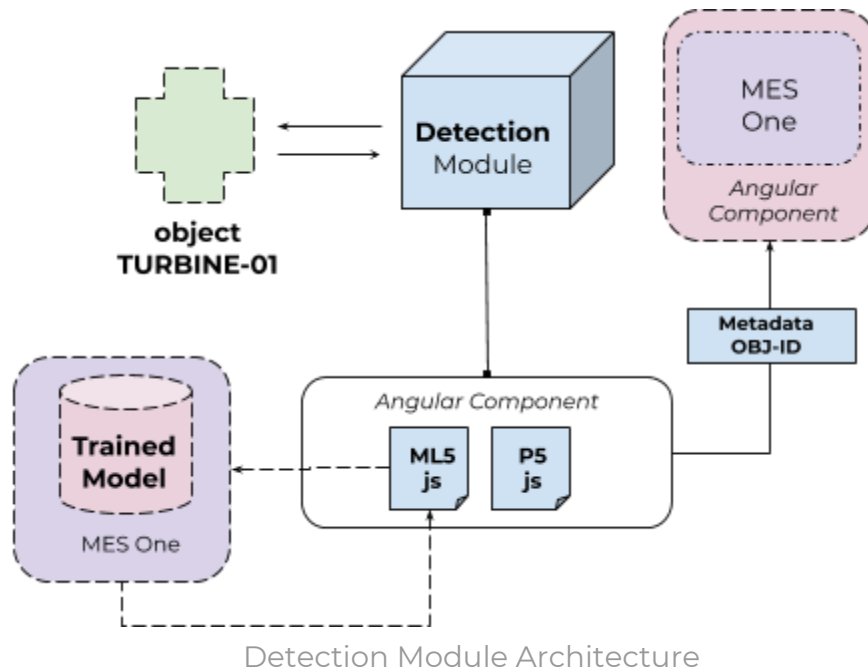
In case the model is providing inaccurate detection, during testing, the ML Model can be retrained, with different values for parameters like epoch, confidence etc., for better results.

## 2.5 Save/Upload Model

Once the ML Model has been tested, the model can then be saved/upload to CouchDB, along with its metadata.

# 3. Detection Model & Integration

Below is the architecture that is followed during implementation of Detection Model & Integration...



### 3.1 Object

The same objects used for training - images of various equipment - shall be used for detection - *refer to section 2.1.*

### 3.2 Detection

The model for detection shall be retrieved from the Trained ML Model stored in the CouchDB Setup - *refer to section 2.5*

### 3.3 Metadata

On successfully completing the Detection Process, the corresponding Metadata (ObjID) is then sent as an output parameter from the angular component.

### 3.4 MES One Integration

The component in MES One Integration shall be programmed (by GE) in order to retrieve the above parameter, from the child component, and perform further actions.

## 4. Need Clarifications

#	Item	Remarks
1	3 Equipment Images for Training - each Equipment should have 4 Images in different angles, and corresponding Metadata (ObjID).	Siva (GE) Clarified
2	CouchDB Setup/API Details for ML Model Upload & Access.	Siva (GE) Clarified
3	With respect to Section 3.4, the child component (which will be detecting the images for ObjID) will send the Metadata to its parent component (MES One).	Eswar (GE) Clarified

## 5. Delivery

#	Item	Date
1	Interim Build & Demo - Training Model & Learning Transfer	13 Jan, 2020
2	Final Build & Delivery - Training Model & Learning Transfer + Detection Model & Integration	17 Jan 2020

## 6. User Manual

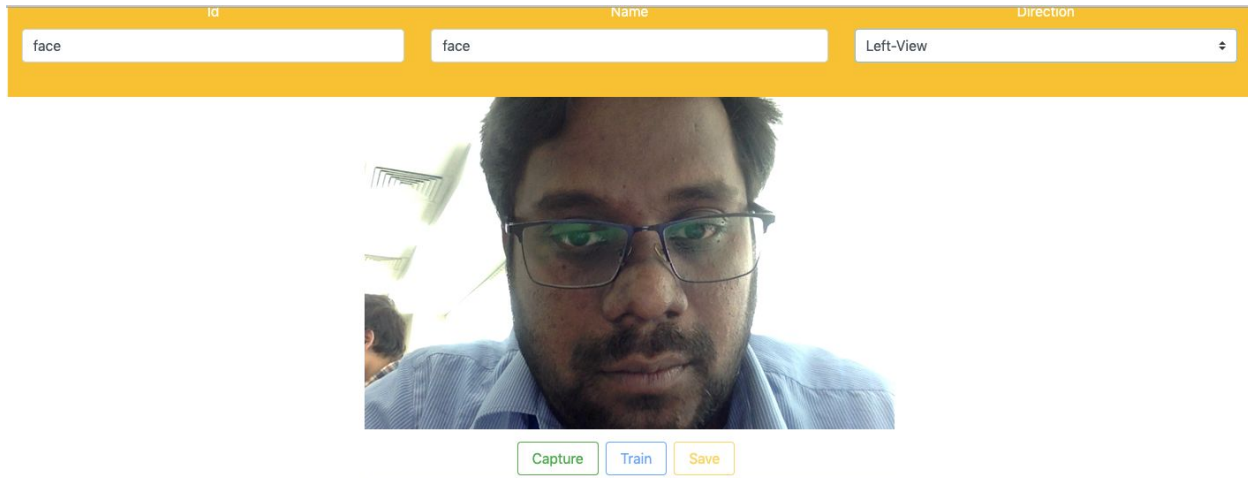
### 6.1 Select Operation

As soon as the application loads, there are a couple of options:

1. Click To Train - for training and creating a new model
2. Click To Scan - for scanning and detecting with metadata

### 6.2 Training Model

On selecting 'Click To Train', the Training Module is opened with the following options..



The screenshot shows the Training Module interface. At the top, there is a yellow header bar with three input fields: 'Id' (containing 'face'), 'Name' (containing 'face'), and 'Direction' (a dropdown menu showing 'Left-View'). Below the header is a large video feed showing a man with glasses and a beard. At the bottom of the video feed, there are three buttons: 'Capture' (green), 'Train' (blue), and 'Save' (yellow).

#### 6.2.1 Metadata ID

The ID parameter of Object Metadata.

#### 6.2.2 Metadata Name

The Name parameter of Object Metadata.

#### 6.2.3 Metadata Direction

The Direction parameter of Object Metadata.

## 6.2.6 Capture Image (Add to Model)

After entering/selecting the ID, Name and Direction, the Add Image button is enabled. On clicking the button, the image (view from camera) is added to the list of images that will be used for training.

Important Notes:

**The application will produce accurate results ONLY if the TRAINING is done properly.** Here are some points that are a must:

1. When training, the background of the object should be plain - ex. White wall, dark desk table etc.
2. Place the object properly, without shaking the camera/object, and take images. Each image should be clear and in-focus.
3. Blurry images, images with a lot of background elements will NOT be trained properly and hence application will not produce appropriate results.
4. Example of training 2 Different ID Card Images is shown below..

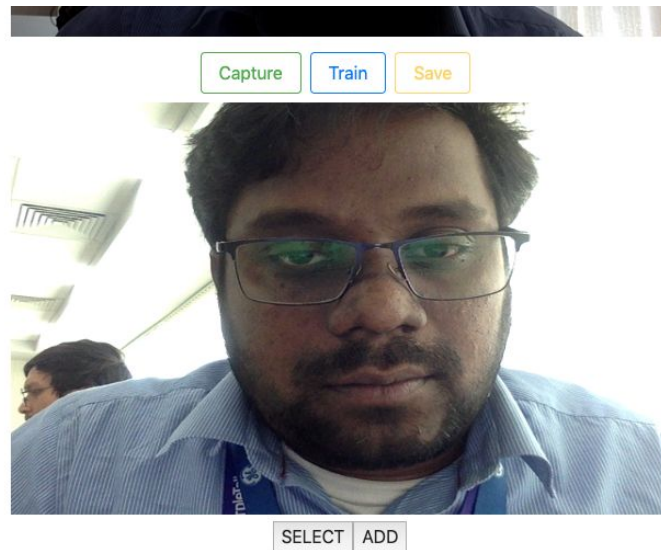


5. The more number of clear images, the better the results. As seen above, for a single item

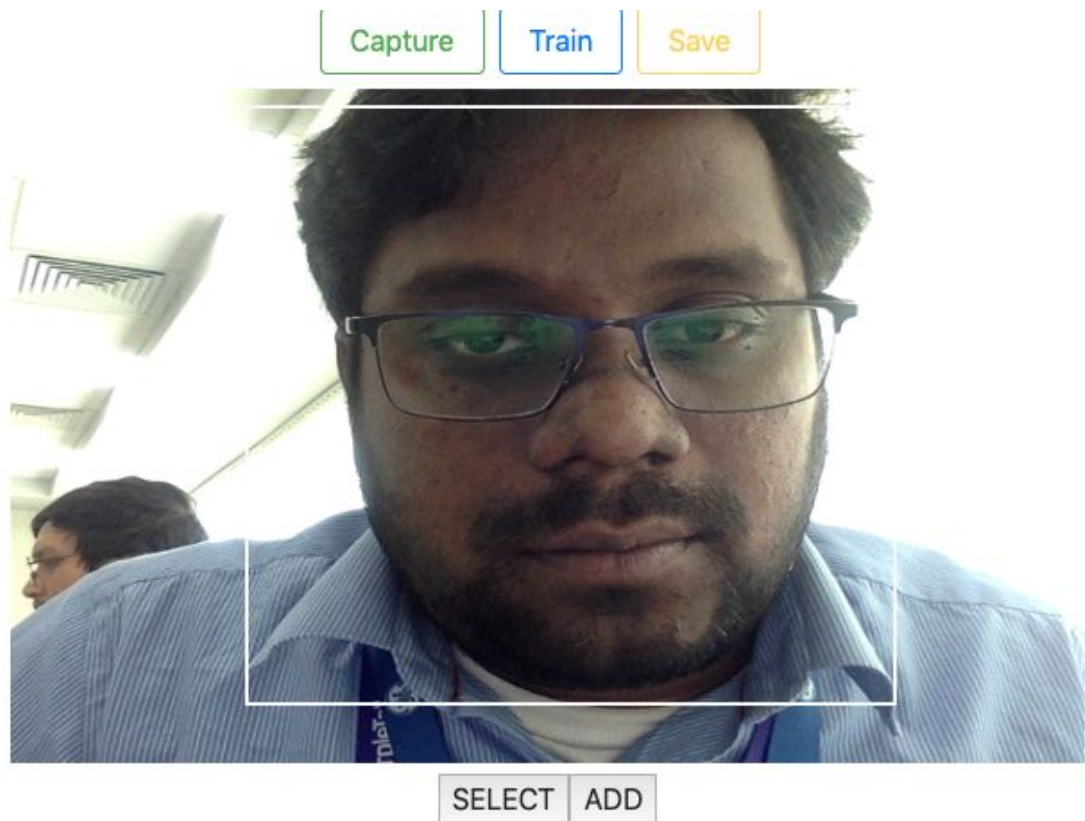
### 6.2.6.1 Select Area

User can select the area within an image - to be displayed on detection. Here are steps...

1. Add all training images for equipment - clear, without blur.
2. After adding the last image, scroll to the Area Select section...



3. Click on select, and on the image, click 3 points - left-top, right-top and right-bottom of a rectangle - to select the area of the image in a rectangle.



4. Click Add, to add this image as the image to be shown on detection.
5. Repeat the same for all equipment.

Note:

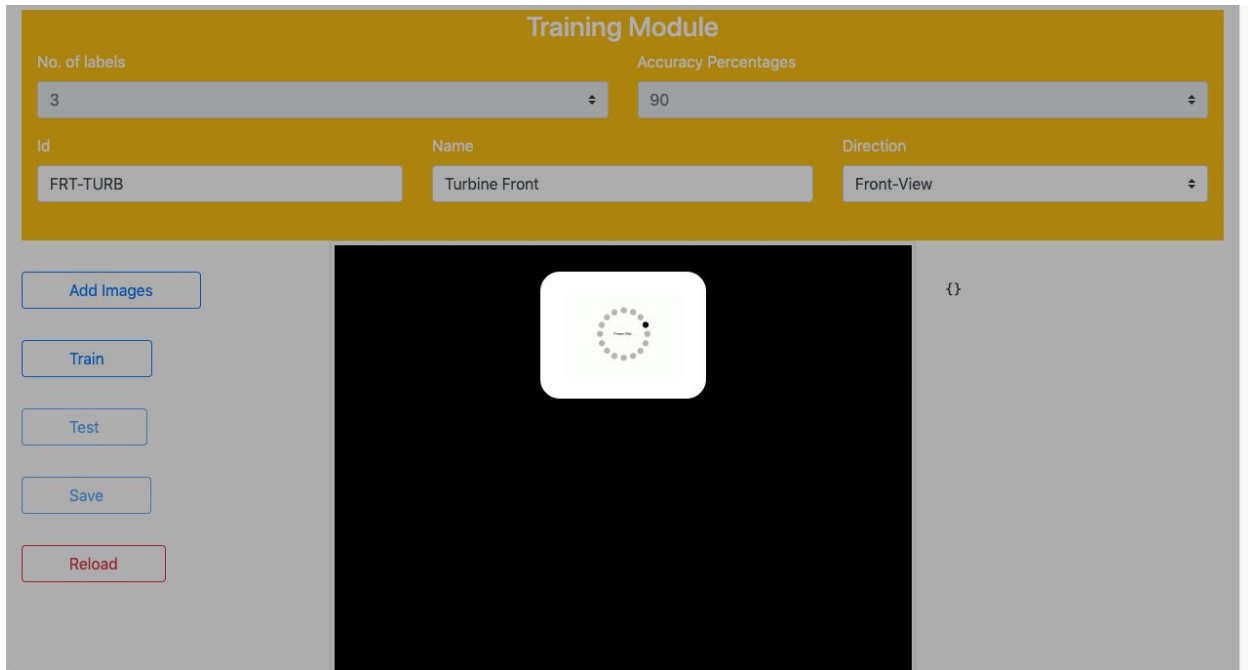


Getting the exact image detected which was used for scanning was not possible. Hence we've come up with this approach. There are advantages to this approach..

1. Selection of detection image would mean clarity in expectation, post detection.
2. The next feature, FOCAL POINTS, can be added to this image and hence seen on detection.

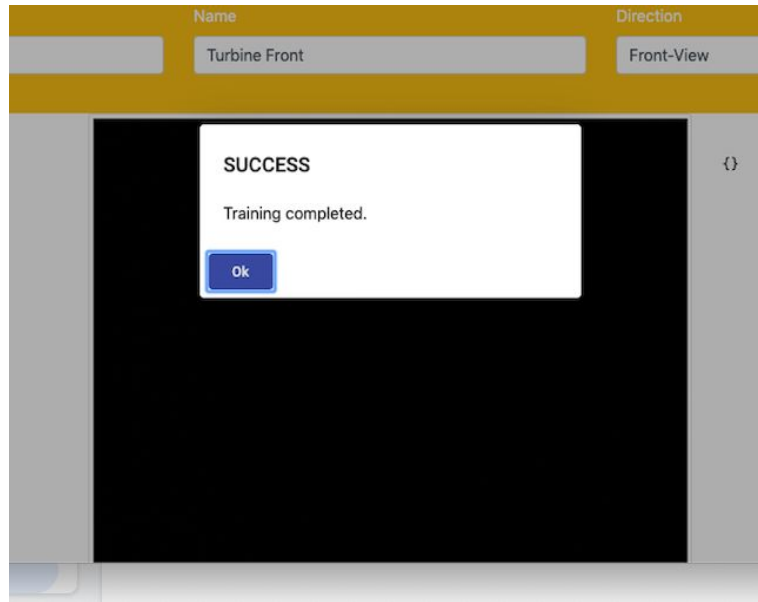
### 6.2.7 Train Model

Once the appropriate number of images are provided for each label, the Train button is enabled. On clicking, the Training Operation begins..



The screenshot shows the 'Training Module' interface. At the top, there are two dropdown menus: 'No. of labels' set to 3 and 'Accuracy Percentages' set to 90. Below these are three input fields: 'Id' with 'FRT-TURB', 'Name' with 'Turbine Front', and 'Direction' with 'Front-View'. On the left side, there is a vertical stack of buttons: 'Add Images', 'Train', 'Test', 'Save', and 'Reload'. The 'Train' button is highlighted. In the center, there is a large black rectangular area with a white circular progress indicator in the middle. To the right of this area, there is a small code editor with curly braces '{ }'.

Once the training is complete, the following popup is shown (and TEST, SAVE button are enabled)...



### 6.2.8 Test Model

In order to test the accuracy of the model (as mentioned in the section 6.2.2), one of the training images can be placed in from the camera and the Test button needs to be clicked. The model will then detect the object and display corresponding metadata - for verification.

### 6.2.9 Save Model

On clicking Save, the model (JSON & Binary Files) are uploaded to the configured CouchDB.

### 6.2.10 Reload

In order to reset/refresh the page and start over with training a new model, Reload option is provided.

## 6.3 Scanning Model

The scanning model, which is a component that can be easily embedded into an angular application, is used for object detection and displaying corresponding data.

.