|  |  |
| --- | --- |
| Api Version | /v1 |
| Base Path | /heimdall/ |
| SOLUTIONURL | http://52.177.169.81:5006 |
| Socket URL | http://52.177.169.81:5006 |
| Back end MQTT | http://52.177.169.81:1885 |

This document will provide API reference for below modules:

Backend: This module will provide api to North bound interfaces like web application etc. and South bound interfaces which would be for Aggregator and Compute Engine

Aggregator: Aggregator would be part of South bound section. This module will communicate with DVR and provide stream services for various cameras and push images for analytics.

Compute Engine: Compute Engine would be part of South bound section. This module will make analytics and deep learning detection of camera images.

Web app: Web app would be part of North bound section. It will mainly interact with user and backend module for numerous services from UI.

# Summary of API

# Back End Module

## Onboarding (NB)

### Add Compute Engine

### Add Aggregator

### Add Camera

### Enable algorithm

## Configuration (NB)

### Set camera configuration(FPS,resolution)

### set Bounding box

## Runtime

### Get list of compute engine(NB)

### Get list of aggregators(NB)

### Get list of camera(NB)

### Result update ( Web socket) (NB)

### Get status CE(NB)

### Get status aggregator(NB)

### Get raw image( MQTT) (NB)

### Get live image(POST) (NB)

## User management

### Add user(NB)

### Delete user(NB)

## Offboarding

### Delete Compute Engine(NB)

### Delete Aggregator(NB)

### Delete Camera(NB)

### Disable algorithm(NB)

## Onboot Registration (SB)

### Register compute engine

### Register aggregator

### Register algorithm

# Aggregator

## Discovery

### Heart bit update status (implemented by MQTT)

## Configuration

### Add camera (resolution, fps, location) (Backend)

### Update camera

### Delete camera

### Broadcast messages

## Status(Runtime)

### Get raw image

### Get live image

## Update

### Configure aggregator

# Compute Engine

## Bootup

### Max number of camera supported

## Discovery

### Heart bit update status

## Configuration

### Add algorithm

### Add shape selection for bb

### Add shape box

### Update shape box

### Delete shape box

### Stop algorithm

## Status(Runtime)

### Upload result

# Back End: This module will provide two types of API’s

# Northbound Api: To provide endpoint to webapp/mobileapp/custom app to interact.

### South bound Api: To provide endpoint to low level layer like Aggregator, compute engine etc.

## To Add Compute Engine

### POST /computeengine

### Description

### This api is part of North Bound Api set. It would be used by web application to onboard compute engine on solution.

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Type | Required | Type of compute engine:  Jetsontx2,  Movidius,  Cloud algorithm | String | Jetsontx2 |
| Name | Required | Name/Label of compute engine | String | Jetson 2 |
| Detection\_algo | Optional | Detection algorithm running on compute engine | String | Human detection |
| Location | Optional | Location of compute engine | String | Location details: pillar number 25 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created compute engine successfully |
| 409 | Compute engine already exist |

### Return format

String

### Errors

None

### Example

request

https://<url>/ComputeEngine

body

{

type: “jetson tx2”,

name:”Jetson 1”,

Location:”Pillar 23”

}

response

{

“return”:"success",

“id”:”1234”

}

## To delete Compute Engine

### POST /computeengine

### Description

### This api is part of North Bound Api set. It would be used by web application to offboard compute engine on solution.

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| cdId | Required | Compute engine ID to delete | String | 123 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Deleted compute engine successfully |
| 409 | Compute engine is in use, please remove dependant camera/algorithm |

### Return format

String

### Errors

None

### Example

request

https://<url>/ComputeEngine

body

{

“ceId”:”123”

}

response

{

“return”:"success"

}

## To Add Aggregator

### POST /aggregator

### Description

### This api is part of North Bound API set. It would be consumed by web application to onboard Aggregator to solution.

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Name | Required | Name/Label of aggregator | String | AggregatorDvr2 |
| URL | Required | Base URL of DVR/NVR to stream camera | String | <valid URL> |
| Channel ID | Required | Number of total camera supported | String | 16 |
| Location | Optional | Location of DVR/NVR | String | Location details: pillar number 25 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created aggregator successfully |
| 409 | aggregator already exist |

### Return format

String

### Errors

None

### Example

request

https://<url>/aggregator

body

{

“Name”:”aggregatorDvr2”,

“url”: “rtsp://admin:admin@192.168.1.20/user=admin&password=admin&channel=1&stream=0.sdp?”,

“channelid”:”32”,

Location:”Pillar 23”

}

response

{

“return”:"success",

“id”:”1234”

}

## To offboard Aggregator

### POST /aggregator

### Description

### This api is part of North Bound API set. It would be consumed by web application to offboard Aggregator to solution.

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| aggregatorId | Required | ID of aggregator | String | 1234 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | deleted aggregator successfully |
| 409 | aggregator in use |

### Return format

String

### Errors

None

### Example

request

https://<url>/aggregator

body

{

“aggregatorId”:”123”

}

response

{

“return”: “deleted successfully",

}

## To Add Camera

### POST cameras

### Description

### This would-be part of North Bound API set. To add camera to solution from web application

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Name | Required | Name/Label of camera | String | receptionCamera |
| Dvrid | Optional | DVR where this camera belongs to | String | 123 |
| Type | Required | Type of camera : IP/IR | String | ip |
| Ceid | Optional | Compute engine where we would like to process camra image | String | 25 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created camera successfully |
| 409 | camera already exist |

### Return format

String

### Errors

None

### Example

request

https://<url>/camera

body

{

“name”:”receptioncamera”,

“dvrid”:”123”,

“ceid”:”23”,

“type”:”ip”

}

response

{

“return”:"success",

“id”:”1234”

}

## To delete Camera

### POST camera

### Description

### This would-be part of North Bound API set. To remove camera from solution from web application

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| cameraId | Optional | Camera id to be removed | String | 25 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | removed camera successfully |
| 409 | camera already in use |

### Return format

String

### Errors

None

### Example

request

https://<url>/camera

body

{

“cameraId”:”123”

}

response

{

“return”: “removed successfully"

}

## To register Algorithm

### POST /devices/computeengine/algorithm

### Description

### This is part of South Bound API. Compute engine will register its capability in terms of algorithm supported to backend by using this api. Below table would be considered by Backend for Algorithm supported. Compute engine need to pass parameter id based on below table.

|  |  |
| --- | --- |
| **Algorithm** | **ID** |
| Human Detection | 1 |
| Face Detection | 2 |
| Object Detection | 3 |

### Below table shows different shapes would be supported by algorithm

|  |  |
| --- | --- |
| **Shape** | **ID** |
| Rectangle | 1 |
| Line | 2 |
| Tringle | 3 |
| Circle | 4 |

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| algoId | Required | Algorithm id for respective name | integer | 1 |
| fps | Optional | FPS supported by algorithm | integer | 2 |
| ceId | Required | Compute engine where algorithm is running | String | 23 |
| supportedShapes | Required | Shape supported by algorithm | String | [{1:rectangle},{2:line}] |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | register algorithm successfully |
| 409 | Algorithm is already existing on selected compute engine |

### Return format

String

### Errors

None

### Example

request

https://<url>/devices/computeengine/algorithm

body

{

“algoid”:”1”,

“fps”:”2”,

“ceid”:”12”,

“supportedShapes”:” [{1:rectangle},{2:line}]”

}

response

{

“return”:"success",

}

## To unregister Algorithm

### POST /devices/computeengine/algorithm

### Description

### This is part of **South Bound API**. Compute engine will unregister its capability in terms of algorithm supported to backend by using this api. Below table would be considered by Backend for Algorithm supported. Compute engine need to pass parameter id based on below table.

|  |  |
| --- | --- |
| **Algorithm** | **ID** |
| Human Detection | 1 |
| Face Detection | 2 |
| Object Detection | 3 |

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| algoId | Required | Algorithm id for respective name | integer | 1 |
| ceId | Required | Compute engine where algorithm is running | String | 23 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | unregister algorithm successfully |
| 409 | Invalid algorithm request |

### Return format

String

### Errors

None

### Example

request

https://<url>/devices/computeengine/algorithm

body

{

“algoid”:”1”,

“ceid”:”12”

}

response

{

“return”:"success",

}

## To get list of supported Algorithm

### GET supportedalgorithms

### Description

### This is part of North Bound API. From web application we can get list of algorithms like human detection, face detection, object detection etc supported by particular compute engine. Compute engine might be Jetson device, movidius or cloud based algorithm.

### Below table would be considered by Backend for Algorithm supported

|  |  |
| --- | --- |
| **Algorithm** | **ID** |
| Human Detection | 1 |
| Face Detection | 2 |
| Object Detection | 3 |

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Ceid | Required | Compute engine where algorithm is running | String | 23 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | List of algorithms supported |
| 409 | Invalid compute engine |

### Return format

String

### Errors

None

### Example

request

https://<url>/supportedalgorithms

body

{

“ceid”:”12”

}

response

{

“algo”:"1"

}

## To Enable/Disable Algorithm

### POST algorithm

### Description

### This is part of North Bound API. From web application we can enable/disable compute engine algorithm like human detection, face detection, object detection etc.

### Below table would be considered by Backend for Algorithm supported

|  |  |
| --- | --- |
| **Algorithm** | **ID** |
| Human Detection | 1 |
| Face Detection | 2 |
| Object Detection | 3 |

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| algoid | Required | Algorithm id for respective name | integer | 1 |
| FPS | Optional | FPS selected by use | String | 2 |
| Ceid | Required | Compute engine where algorithm need to be enable/start | String | 23 |
| Action | Required | Enable/disable (1,0 respectively) | integer | 1 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Enabled/disable algorithm successfully |
| 409 | Algorithm is already existing on selected compute engine |

### Return format

String

### Errors

None

### Example

request

https://<url>/algorithm

body

{

“algoid”:”1”,

“fps”:”2”,

“ceid”:”12”,

“action”:”1”

}

response

{

“return”:"success",

}

# Web Socket raw image

## Description

### This is part of North Bound API set.This feature helps to communicate the raw image from the backend to the webapp.

## Assumption:

Connected to SOCKETURL

## Topics

### Subscriber topic - rawImage

## Requires authentication

## Resource Information

|  |  |
| --- | --- |
| Type | Web socket |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Image base64 payload | required | The rawt image to be displayed on webapp for bounding box selection | BSON | Base64 payload |
| Raw image name | required | Raw image name | string | <image name> |

## Responses

### NA

## Return format

### NA

## Errors

### None

## Example

**Connection request**

https://<SOCKETURL>

**Topic to subscribe**

rawImage

**Response Body**

{

“imgName”:<image name>,

“imgBase64”: <Base64 payload>

}

## Web Socket Result image/metadata Update

## Description

#### This is part of North Bound API set. This feature helps to communicate the **result** image (along with result metadata) from backend to the webapp.

## Assumption:

Connected to SOCKETURL

## Topics

### Subscriber topic - liveImage

## Requires authentication

### Secure web socket

## Resource Information

|  |  |
| --- | --- |
| Type | Web socket |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Image base64 payload | required | The result image to be displayed on webapp | BSON | Base64 payload |
| Area of detection | required | Bounding boxes of detected objects | string | {“x”:”10”,“y”:”100”,“x2”:”150”,“y2”:”157”} |
| Total detections | required | Number of detections with respect to area of interest | integer | {“result”: [“1”]} |

## Responses

### NA

## Return format

### NA

## Errors

### None

## Example

**Connection request**

https://<SOCKETURL>

**Topic to subscribe**

liveImage

**Response Body**

{

“result”: [“2”],

“bbox”:[{“x”:”24”,“y”:”50”,“x2”:”67”,“y2”:”100”},{“x”:”150”,“y”:”150”,“x2”:”198”,“y2”:”450”}],

“imgBase64”: <Base64 payload>

}

# Web Socket add camera response

## Description

# Whenever the notification from camera is received from aggregator, this socket response will notify the webapp if the camera is valid or not. This is part of North Bound API set.

## Assumption:

Connected to SOCKETURL

## Topics

### Subscriber topic - addCameraResponse

## Requires authentication

## Resource Information

|  |  |
| --- | --- |
| Type | Web socket |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| status | required | If 0 - Invalid entry <camera name>  1 - Camera added <camera name> | string | {“stauts”: “1”} |
| Camera id | required | Camera id | string | {“camname”:<cameraid>} |

## Responses

### NA

## Return format

### NA

## Errors

### None

## Example

**Connection request**

https://<SOCKETURL>

**Topic to subscribe**

addCameraResponse

**Response Body**

{

“status”:”1”,

“camId”: <camera id>

}

# Web Socket Background camera results

## Description

# This feature helps to communicate the results of the background streaming cameras, from backend to the webapp. This is part of North Bound API set.

## Assumption:

Connected to SOCKETURL

## Topics

### Subscriber topic - backgroundCameraResults

## Requires authentication

## Resource Information

|  |  |
| --- | --- |
| Type | Web socket |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Camera name | required | Name of the camera which is streaming in the background | string | {“camname”:”Reception”} |
| Total detections | required | Number of detections with respect to area of interest | integer | {“result”: “1”} |

## Responses

### NA

## Return format

### NA

## Errors

### None

## Example

**Connection request**

https://<SOCKETURL>

**Topic to subscribe**

backgroundCameraResults

**Response Body**

{

“result”:”3”,

“camname”: <Camera name>

}

# Web Socket notifications

## Description

# This feature notifies the messages from backend to the webapp. This is part of North Bound API set.

## Assumption:

Connected to SOCKETURL

## Topics

### Subscriber topic - notification

## Requires authentication

#### secure socket based

## Resource Information

|  |  |
| --- | --- |
| Type | Web socket |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Notification | required | Notification to webapp | NA | {“message”: “Camera is down”} |

## Responses

### NA

## Return format

### NA

## Errors

### None

## Example

**Connection request**

https://<SOCKETURL>

**Topic to subscribe**

notification

**Response Body**

{

“message”: ”Camera is down”

}

# POST getRawImage

## Description

### This is North Bound api. Web app would request backend to provide RawImage. This API notifies the backend to fetch the raw image from the aggregator and the image will be received on socket.

## Requires authentication

Valid bearer token should be provided

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | 12 |
| feature | required | To specify the detection algorithm | String | “personDetection” |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Image send successfully |
| 401 | Invalid request |

## Return format

##### String

## Errors

**None**

## Example

request

https://<url>/api/getRawImage

body

{

“deviceName”: ”Reception”,

“camId”: <Unique ID>,

“feature”: “personDetection”,

“streamingUrl”: <Camera URL>

}

Response

NA

# MQTT get raw image

## Description

# This is part of Southbound API set. This feature publishes the required data to aggregator via topic getRawImage(Backend🡪Aggregator) and receives the base64 raw image from the aggregator via topic rawMQTT(Aggregator🡪Backend).

## Topics

# Publisher topic - getRawImage

Subscriber topic - rawMQTT

## Requires authentication

NA

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters(for getRawImage)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | 12 |
| feature | required | To specify the detection algorithm | String | “personDetection” |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |

## Parameters(for rawMQTT)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| imgName | required | The image name | String | <image name> |
| imgBase64 | required | Base64 payload of an image | BSON | <image payload> |

## Responses

NA

## Return format

NA

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - getRawImage

Request Body

{

“camId”: <Unique ID>,

“feature”: “personDetection”,

“streamingUrl”: <Camera URL>

}

**Topic** - rawMQTT

Request Body

{

“imgName”: <image Name>,

“imgBase64”: <Base64 payload>

}

# POST getResult

## Description

# This is part of Southbound API set, this would be consumed by Compute Engine to post result of Image to Backend(ComputeEngine🡪 Backend).

## Requires authentication

Valid bearer token should be provided

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| imgName | required | The image name | String | <image name> |
| bbox | required | The bounding box for the detection | String | {“bbox”:[{“x”:”10”,“y”:”100”,“x2”:”115”,“y2”:”300”}]} |
| result | required | Number of detections with respect to area of interest | integer | {“result” : [“1”]} |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Result received successfully |
| 401 | Invalid request |

## Return format

### String

## Errors

### None

## Example

request

https://<url>/api/getResult

body

{

“imageName”: <image Name>',

“result”: [“2”],

“bbox”:[{“x”:”24”,“y”:”50”,“x2”:”67”,“y2”:”100”},{“x”:”150”,“y”:”150”,“x2”:”198”,“y2”:”450”}]

}

Response

{

“201”:”Image result received”

}

### Refer to “Web Socket Result Update” section to cross-verify this functionality

## To register Aggregator

### POST /devices/aggregator

### Description

### This api is part of South Bound API set. It is used by Aggregator module. On boot aggregator will register itself with solution by using this api.

### Requires authentication

Valid bearer token should be provided

### Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

### Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| name | required | The aggregator name | String | <aggregator name> |
| url | required | Base URL of Aggregator | String | <url> |
| ip | required | IP address of Aggregator | String | 192.168.123.12 |
| macAddr | Optional | Mac id of Aggregator | String | 12:23:23:23:43:34 |
| Location | Optional | Physical location where aggregator is installed | String | Floor3 |
| Channel ID | Optional | Channel id for DVR/NVR | string | 32 |

### Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | deleted aggregator successfully |
| 409 | aggregator is whitelisted |

### Return format

String

### Errors

None

### Example

request

https://<url>/devices/aggregator

body

{

“Name”:”aggregatorDvr2”,

“url”: “rtsp://admin:admin@192.168.1.20/user=admin&password=admin&channel=1&stream=0.sdp?”,

“ip”:”192.158.123.23”,

“macid”:”23:23:23:23:23”,

“channelid”:”32”,

Location:”Pillar 23”

}

response

{

“return”: “registered successfully",

}

# POST Unregister aggregator

POST aggregators

## Description

### This api is part of South Bound API set. It is used by Aggregator module. On exit sequence aggregator will de-register itself with solution by using this api.

## Requires authentication

Valid bearer token should be provided in the headers

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| aggregatorId | required | The aggregator id | String | Agg23 |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Aggregator un-registered successfully |
| 401 | Invalid request |

## Return format

**String**

## Errors

**None**

## Example

“return”:"success",request

https://<url>/aggregators

Request Body

{

“aggId”:”agg23”

}

Response

{

201 : unregister successfully

}

# POST Register Compute Engine

POST /devices/computeengines

## Description

# This API helps to register Compute Engine. This is part of South Bound API set. It would be used by compute engine to register itself in bootup sequence.

## Requires authentication

Valid bearer token should be provided in the headers

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| name | required | The name of device | String | <device name> |
| deviceType | required | Type of device | String | <device type> |
| timestamp | required | Current timestamp |  | <epoch timestamp> |
| macId | required | Mac id of device | String | <mac id> |
| ipAddress | required | Ip Address of device | String | <ip> |
| Location | Optional | Physical location where aggregator is installed | String | Floor3 |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Compute engine registered successfully |
| 401 | Invalid request |

## Return format

**String**

## Errors

**None**

## Example

Request: https://<url> /devices/computeengines

Request Body

{

“name” : “jetsontx2”,

“deviceType” : “jetson”,

“timestamp”:”123123123”,

“macId” : “1231231123”,

“IpAddress”:”192.168.21.21”,

“location”:”floor3”

}

Response

201 : Created

# POST Unregister Compute Engine

POST /devices/computeengines

## Description

# This API helps to deregister Compute Engine. This is part of South Bound API set. It would be used by compute engine to deregister itself in shutdown sequence.

## Requires authentication

Valid bearer token should be provided in the headers

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| ceId | required | Id of compute engine | String | 12 |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Compute engine deregistered successfully |
| 401 | Invalid request or in use |

## Return format

**String**

## Errors

**None**

## Example

Request: https://<url> /computeengines

Request Body

{

“cdId”:”12”

}

Response

201 : unregistered

# GET Maximum number of Camera Supported

POST /computeengines/camerasupported

## Description

# This API is part of South Bound API set. It would be mainly used by compute engines to update capability of how many camera’s can be supported by each compute engine. This is runtime capability show up by compute engine.

## Requires authentication

Valid bearer token should be provided in the headers

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Id | required | Id of compute engine device | String | 123 |
| timestamp | required | Current timestamp |  | <epoch timestamp> |
| macId | Optional | Mac id of device | String | 00:0a:95:9d:68:16 |
| numberOfCameraSupported | required | Number of camera supported by compute engine | Integer | 3 |
| Location | Optional | Physical location where aggregator is installed | String | Floor3 |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Compute engine updated successfully |
| 401 | Invalid request |

## Return format

**String**

## Errors

**None**

## Example

“return”:"success",

Request: https://<url> /computeengines/camerasupported

Request Body

{

“id” : “123”,

“timestamp”:”123123123”,

“macId” : “1231231123”,

“numberOfCameraSupported”:”2”,

“location”:”floor3”

}

Response

201 : updated

# MQTT heartbeat update status

## Description

## This is part of South Bound Api. This feature enables devices such as compute engine, aggregator, etc to publish if they are alive or in running state.

## Topics

# Publisher topic - ping

Subscriber topic - ping\_ack

## Requires authentication

### Certificate based

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters(for ping)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| aggregatorId | optional | Aggregator Id | String |  |
| computeEngineId | optional | Compute Engine Id | String |  |

## Parameters(for ping\_ack)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| aggregatorId | optional | Aggregator Id | String |  |
| computeEngineId | optional | Compute Engine Id | String |  |

## Responses

NA

## Return format

**NA**

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - ping

Request Body

{

“aggregatorId”:”123”,

“computEngineId”: ”123”>

}

**Topic** - ping\_ack

Request Body

{

“aggregatorId”: ”123”

}

# MQTT check camera

## Description

# This is part of South bound api set. This would be used by backend to ensure the newly added camera is validated by aggregator.

## MQTT Topics

# Publisher topic - checkCamera

Subscriber topic - checkCameraResponce

## Requires authentication

### Certificate based

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters (for checkCamera)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | Unique ID |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |
| userId | Optional | User id who has initiated the request | String | 123 |

## Parameters(for checkCameraResponce)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | Unique ID |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |
| userId | Optional | User id who has initiated the request | String | 123 |
| camStatus | Required | Status of camera : Online-1 /Offline - 0 | Int | 1 |

## Responses

NA

## Return format

NA

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - checkCamera

Request Body

{

“camId”: <Unique ID>,

“streamingUrl”: <streamingUrl>,

“userId”: “123”

}

**Topic** - checkCameraResponse

Request Body

{

“camId”: <Unique ID>,

“streamingUrl”: <streamingUrl>,

“userId”: “123”,

“camStatus”: “0/1”

}

# MQTT check multiple camera status: online/offline

## Description

# This is part of South bound api set. This would be used by backend to ensure the selected camera is online/offline with aggregator. This is runtime check for all added cameras.

## Topics

# Publisher topic - cameraUrls

Subscriber topic - cameraStatus

## Requires authentication

NA

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters (for cameraUrls)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | Unique ID |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |
| userId | Optional | User id who has initiated the request | String | 123 |

## Parameters (for cameraStatus)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| camId | required | Unique ID generated for the device | String | Unique ID |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |
| userId | Optional | User id who has initiated the request | String | 123 |
| camStatus | Required | Status of camera : Online-1 /Offline - 0 | Int | 1 |

## Responses

NA

## Return format

NA

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - cameraUrls

Request Body

[

{

“camId”: “12”,

“streamingUrl”: <streamingUrl>,

“userId”: “123”

},

{

“camId”: “13”

“streamingUrl”: <streamingUrl>,

“userId”: “123”

}

]

**Topic** - cameraStatus

Request Body

[

{

“camId”: “12”,

“streamingUrl”: <streamingUrl>,

“userId”: “123”,

“camStatus”: “0/1”

},

{

“camId”: “13”,

“streamingUrl”: <streamingUrl>,

“userId”: “123”,

“camStatus”: “0/1”

}

]

# MQTT To select area of interest and start streaming engine

## Description

# This is part of South Bound Api set. This would be used by backend to pass on information about area of interest to Compute engine and ask Aggregator to start streaming. This will get confirmation from compute engine and Aggregator by MQTT subscriber topic.

## Topics

# Publisher topic - startStreaming

Subscriber topic - ackStartStreaming

## Requires authentication

### Certificate based

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters (for startStreaming)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Aggregator ID | Required | Aggregator id to start streaming | String | 123 |
| ComputeEngine ID | Required | Compute engine selection | String | 231 |
| camId | required | Select camera | String | Unique ID |
| Bouding box | Required | Area of interest from Northbound | Array of objects |  |
| streamingUrl | required | Camera URL to stream the camera | String | <camera Url> |
| userId | Optional | User id who has initiated the request | String | 123 |

## Parameters (for ackStartStreaming)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Aggregator ID | Required | Aggregator id to start streaming | String | 123 |
| ComputeEngine ID | Required | Compute engine selection | String | 231 |
| camId | required | Select camera | String | Unique ID |
| userId | Optional | User id who has initiated the request | String | 123 |

## Responses

NA

## Return format

NA

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - startStreaming

Request Body

{

“camId”: “13”,

“streamingUrl”: <streamingUrl>,

“userId”: “123”,

“boundingbox”:[{“x”:”24”,“y”:”50”,“x2”:”67”,“y2”:”100”}],

“computeEngineId”:”123”,

“aggregatorId”:”231”

}

**Topic** - ackStartStreaming

Request Body

{

“camId”: “13”,

“userId”: “123”,

“computeEngineId”:”123”,

“aggregatorId”:”231”

}

# MQTT to stop cameras

## Description

# This is part of South Bound Api set. This would be used by backend to stop camera streaming and processing.

## Topics

# Publisher topic - stopCameras

## Requires authentication

NA

## Resource Information

|  |  |
| --- | --- |
| Type | MQTT |
| Response formats | JSON |
|  |  |

## Parameters (for stopCameras)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Aggregator ID | Required | Aggregator id to stop streaming | String | 123 |
| camId | Required | Select camera/s | Array | Unique ID |
| userId | Optional | User id who has initiated the request | string | 123 |
| ComputeEngine ID | Required | Compute engine selection | String | 231 |

## Responses

NA

## Return format

NA

## Errors

**None**

## Example

**Connection request**

https://<MQTT url>

**Topic** - stopCameras

Request Body

{

“camId”: [“13”,”45”],

“userId”: “123”,

“computeEngineId”:”123”,

“aggregatorId”:”231”

}

# User API (North Bound)

### This section will cover below operations with respect to user

### Signup

### Login

### Logout

### Forgot password

# POST Signup

POST /user/signup

## Description

### This is North Bound API set. This feature helps to perform signup activity for new user to solution. This api allows new user to sign up on solution portal

## Requires authentication

### Valid password should be provided

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Username | Required | Username of user | String | mobiliyatest |
| Emailed | Required | Email id to of user,  Could you use for activation | String | user@mobiliya.com |
| Mobile number | Optional | This could be used for 2FA | integer | 9898989898 |
| Password | Required | Password for user account | String |  |
| Organization | Optional | Organization of user | String |  |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created user successfully |
| 409 | User already exist |

## Return format

### String

## Errors

### None

## Example

request

https://<url>/user/signup

body

{

“userName”:”mobiliyauser1”,

“mobileNumber”:”9898988998”,

“emailId”:”user@mobiliya.com”,

“password”:”1d\*23234(\*”,

“organization”:”mobiliya”

}

response

"success"

# POST login

POST /user/login

## Description

# This feature helps to perform login activity for user to solution. This is part of North Bound API set.

## Requires authentication

### Valid bearer token based

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Username | Required | Username of user | String | mobiliyatest |
| Password | Required | Password for user account | String |  |
| Organization | Optional | Organization of user | String |  |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created user successfully |
| 409 | User already exist |

## Return format

**String**

## Errors

**None**

## Example

request

https://<url>/user/login

body

{

“userName”:”mobiliyauser1”,

“password”:”1d\*23234(\*”,

“organization”:”mobiliya”

}

Response

{

“return”:"success",

}

# POST reset password

POST /user/resetpassword

## Description

# This feature helps to perform signup activity for new user to solution. This api allows new user to sign up on solution portal

## Requires authentication

Valid password should be provided

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Username | Required | Username of user | String | mobiliyatest |
| Emailed | Required | Email id to of user,  Could you use for activation | String | user@mobiliya.com |
| Mobile number | Optional | This could be used for 2FA | integer | 9898989898 |
| Password | Required | Password for user account | String |  |
| Organization | Optional | Organization of user | String |  |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created user successfully |
| 409 | User already exist |

## Return format

**String**

## Errors

**None**

## Example

request

https://<url>/user/resetpassword

body

{

“userName”:”mobiliyauser1”,

“mobileNumber”:”9898988998”,

“emailId”:”user@mobiliya.com”,

“password”:”1d\*23234(\*”,

“organization”:”mobiliya”

}

response

"success"

# POST logout

POST /user/logout

## Description

# This feature helps to perform logout activity for user to solution. This is part of North Bound API set.

## Requires authentication

Valid bearer token based

## Resource Information

|  |  |
| --- | --- |
| Type | HTTP/HTTPS |
| Response formats | JSON |
|  |  |

## Parameters

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | **Required** | **Description** | **Data Type** | **Example** |
| Username | Required | Username of user | String | mobiliyatest |

## Responses

|  |  |
| --- | --- |
| **Return Value** | **Description** |
| 201 | Created user successfully |
| 409 | User already exist |

## Return format

**String**

## Errors

**None**

## Example

request

https://<url>/user/logout

body

{

“userName”:”mobiliyauser1”

}

response

"success"