

Innovative Technology Ltd



ICU DEVICE API

version 1.0.27

Contents

Version History	3
System	4
Http Ports.....	4
REST response codes.....	4
API requests	5
Token	6
Set API Credentials.....	7
Get Device Detail	9
Set System Settings.....	10
Get System Settings	11
Get Device Settings	12
Set Device Settings.....	13
Get Device Video Steam Settings	17
Set Device Video Steam Settings.....	18
Set Face Processing enable/disable	19
Indicator Control	20
Get Device Status	21
Obtain Face ID Data	24
QR Code barcode detection	26
ID Document Session	27
Get ID Age Session	28
Set ID Age Session	29
Get Session Age Result	30
Get Session Scan Result	31
Single Image Session (1:1).....	32
Face Data Update.....	34
Face Data Delete.....	35
Action Data Update.....	36
FACE_ID.....	37
FACE_NON_ID	38
AGE.....	39
QR_DETECT	40
Action Parameters	41
Activate Action.....	43
Reset Device.....	44
Update Device.....	45

Version History

Version	comment	date
1.0.10	Get/Set Settings interface update	17 th Jan 2023
1.0.11	Status request – add FaceCount	25 th Jan 2023
1.0.12	Settings and Device update	3 rd Feb 2023
1.0.14	ID card verification session . Person object count added to status response	8th May 2023
1.0.15	Add Video Stream settings get/set . Add TrackingNumber to status response.	12 th June 2023
1.0.16	Add SpoofTerminated flag to status response	21 st July 2023
1.0.17	Add spoof terminate on/off option to settings request . Add reject reason to status . Add Indicator LED control	31 st August 2023
1.0.18	Add patch version to device details response	27 th November 2023
1.0.19	Add request to change API credentials	2 nd January 2024
1.0.20	Add AFD flag to settings	5 th January 2023
1.0.21	Add MyCheckr screen mode to settings . API action updates	20 th January 2024
1.0.22	Add FRT to settings	2 nd February 2024
1.0.23	Add Face Processing Enable/Disable	15 th February 2024
1.0.24	Add API credential table. Fix typos	23 rd February 2024
1.0.25	Add settings request parameter ComsAutoResetInterval	1 st March 2024
1.0.26	Add new QR Code reading feature	11 th March 2024
1.0.27	Update to ID Scan process. Typo fixes	15 th April 2024



System

The ICU Local API provides a REST interface to allow authorised users to create bespoke image recognition networks using ICU Devices

The API uses a token request system to authenticate the user. The user sends a token request and then uses that token string in subsequent request headers.

Http Ports

If SSL option is enabled:

ICU device API PORT is running at 44345

BASE_URL: https://192.168.137.8:44345

Else:

ICU device API PORT is running at 8060

BASE_URL: http://192.168.137.8:8060

BASE_PATH: api/v1_0/

The API credentials are the same as the admin user and password set by the configuration webserver app.

<api username> <api password>

REST response codes

This REST API use standard HTTP response codes in indicated the overall state of the request response. These include:

200	OK
201	Created
404	Not Found
500	Internal Server Error
400	Bad request
401	Unauthorised

Further response data is available as JSON object string in the response body giving detailed status codes and request information.



API requests

Token

Request an authorisation token from the API.

The user requires API username and password that has been set on the ICU device.

url: BASE_URL/Token

method: POST

headers: Content-Type: application/x-www-form-urlencoded

postdata:

grant_type=password&username=<api username>&password=<api password>

response: code 200 indicates success with JSON string data response.

```
{
  "access_token": "<string token> ", // use this in Bearer Authorization header for your API requests
  "token_type": "bearer",
  "expires_in": 1615907453940, // UTC time in millisecond since epoc (1.1.1970 00:00:00)
  "userName": "<api username>"
}
```

Set API Credentials

Allow the host to change the API token credentials. Note that the new credentials will apply in the server after a 200 success response is received from this request and any previous tokens will be invalidated so a new Token request will have to be made with the new credentials before any further request can be made.

The request also includes the SSLEnabled parameter to set the connection to either HTTP (false) or HTTPS(true).

The response will also include the new API and password that was used so that the host can verify the exact credentials used and use these for a new token request in case of error.

After a successful request, the device will automatically perform a full reset and after that, the host will need to use these new credentials to obtain a new Token for further requests.

url: BASE_URL/BASE_PATH/credentials (SWBuildVersion >= 1100 p20)

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: JSON POST body

```
{
  "Username": "new_username",
  "Password": "new_password",
  "SSLEnabled": false
}
```

Param name	Type	Comment
Username	string	The new API username. Must be a string of length <= 50 and >= 7
Password	string	The new API password. Must include 8 letters, at least 1 capital, 1 special character and 1 number. Must be a string of length <= 50 and >= 10
SSLEnabled	bool	Set to enable or disable API for https (SSL enabled)

Response: 200 success:

```
{  
  "Username": "new_username",  
  "Password": "new_password",  
  "SSLEnabled": false  
}
```


Get Device Detail

Returns the device data of the ICU for use in other API requests.

url: BASE_URL/BASE_PATH/device

method: GET

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: None

response: 200

```
{
  "DeviceId": "baa27042fe",
  "DeviceName": "ICU-baa27",
  "DeviceType": "ICUPro",
  "HWLevel": "10002",
  "SWBuildVersion": 1100,
  "PatchVersion": "pa1100.019",
  "Cameras": [
    {
      "Name": "camera-1",
      "Type": "usb",
      "Data": "/dev/video0",
      "Enabled": true
    },
    {
      "Name": "camera-2",
      "Type": "none",
      "Data": "none",
      "Enabled": false
    }
  ]
}
```

Set System Settings

Set the device system options settings.

url: BASE_URL/BASE_PATH/system (SWBuildVersion >= 1068)

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: JSON POST Body

POST data format

Item Name	Type	
<code>timeZoneID</code>	String	A string representing the desired time zone setting of the device. For example to set, the time zone for Germany set string to <i>Europe/Berlin</i> or <i>America/New_York</i> for Eastern Standard Time in the US. Send zero length string "" to make no change.
<code>utcDateTime</code>	String	A string representing the time to set the device in UTC time. String should be formatted as <i>YYYY-mm-dd HH:mm:ss</i> and in UTC. For example, the time zone of the device is set to Europe/Berlin and the host would like to manually set the device time to 6:15pm 25 seconds on 23 rd March 2021 (Berlin time) the UTC string to send would be <i>2021-03-23 17:15:25</i> (Berlin time to UTC) Send zero length string "" for no change.

```
{
  "timeZoneID": "Europe/Berlin",
  "utcDateTime": "2021/03/23 17:15:25"
}
```

Get System Settings

Get the device system options current settings.

url: BASE_URL/BASE_PATH/system (SWBuildVersion >= 1068)

method: GET

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: None

Returns the current time data settings for the system

```
{
  "localDateTime": "2022/03/02 01:49:15",
  "timeZoneID": "America/New_York",
  "utcDateTime": "2022/03/02 06:49:15"
}
```

Get Device Settings

Returns the device data of the ICU for use in other API requests.

url: BASE_URL/BASE_PATH/settings (SWBuildVersion >=1082)

method: GET

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: None

response: 200

```
{
  "DeviceName": "ICU-Touch",
  "ReactiveLED": "true",
  "ComsAutoResetInterval": 0,
  "Cameras": [
    {
      "Enabled": true,
      "Name": "ICU-Touch-camera",
      "Id_Index": 0,
      "Spoof_level": 2,
      "Spoof_terminate": true,
      "AFD_feature": false,
      "FRT_feature": true,
      "Camera_distance": 4,
      "Pose_filter": true,
      "Rotation": 0,
      "Resolution": "640_480",
      "Active": true,
      "Face_rec_en": true,
      "View_mode": "Biggest",
      "Video_mode": "Result",
      "ID_verify_capable": true
    }
  ]
}
```

Set Device Settings

Allow the API to modify system settings.

url: BASE_URL/BASE_PATH/settings

method: POST

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: JSON POST body

```
{
  "DeviceName": "ICU-Touch",
  "ReactiveLED": "false",
  "ComsAutoResetInterval": 0,
  "Cameras": [
    {
      "Enabled": true,
      "Name": "ICU-Touch-camera",
      "Id_Index": 0,
      "Spoof_level": 2,
      "Spoof_terminate": true,
      "AFD_feature": true,
      "FRT_feature": true,
      "Camera_distance": 4,
      "Pose_filter": true,
      "Rotation": 0,
      "Face_rec_en": true,
      "View_mode": "Kiosk",
      "Video_mode": "Video"
    }
  ]
}
```

For convenience, the request can be minimised into required parameters.

For instance, to just change the device name send `{"DeviceName": "NewName"}`

Camera parameters require that the **Id_Index** is always sent and some parameters are grouped.

E.g. to set the camera disable only: `{"Cameras": [{ "Id_Index": 0, "Enabled": false }]}`

Parameter table

Param name	Type	Comment
DeviceName	string	the name of the ICU device.
ReactiveLED	Bool	Enable/Disable the Indicator LED
ComsAutoResetInterval	integer	If greater than 0, the device will perform a full reset if any valid API command is not received during this interval. Min 10, Max 120 seconds. This setting does not persist between resets and defaults to zero at power up.
Enabled	Bool	Set to enable or disable the camera for ICU Operations
Name	string	The name of the ICU camera
Id_Index	integer	The id number of the camera obtained when using the Get Setting request
Spoof_level	integer	Valid values: (build <1100) 0 – Disabled 1 – Low 2 – Med 3 - High (build >=1100) 0- Disabled 1 to 4- Enabled
Spoof_terminate	Bool	Switches the feature that blocks detection after spoof triggers (build >= 1100)
AFD_feature	Bool	Enables or disables the Spoof Anti-Face Distortion feature. (build >= 1100)

		<ul style="list-style-type: none"> Ensures that a person's face is not manually distorted to alter face recognition characteristics. Frames with detected distortion are rejected. Operates independently of Spoof Detection.
FRT_feature	Bool	<p>Enables or disables the Face Recognition Tracking (FRT) feature (build >= 1100)</p> <ul style="list-style-type: none"> Tracks faces and assigns a unique ID when a new face is detected. Disabling this feature disables tracking, with the tracking ID always set to zero. Actions may be triggered multiple times by the same face. With AAE off, no age or gender is calculated. With AAE on, age and gender are calculated, but face recognition is disabled.
Camera_distance	integer	<p>Valid values: 0(0.5m) - 4(2m), zero is the closest to camera, 4 is the furthest.</p>
Pose_filter	bool	<p>Enable/disable the pose filtering (Only valid for firmware <1100)</p> <p>Valid values: Enable – true Disable - false</p>
Rotation	integer	<p>Value to set the camera rotation degree. Valid values 0,90,180,270</p>

Face_rec_en	bool	<p>Anonymous Age Estimation (AAE) by Disabling Face ID Generation</p> <p>Valid values: True - Enable Face ID Generation False - Disable Face ID Generation</p>
View_mode	string	<p>Set the current view mode of the device.</p> <p>Biggest: Will accept face closest to the device</p> <p>Multi: Will process all faces in camera view</p> <p>Kiosk: Will add an oval overlay for restricting processing area for acceptable faces.</p> <p>OCR: Will allow the device to process ID cards if certain conditions are met</p>
Video_mode	String	<p>MyCheckr only – set the screen mode:</p> <p>“Result” – No face video, only screen results</p> <p>“VideoResult” – face video is shown when detected</p> <p>“Video” – Constant camera view stream.</p> <p>“Ads” – screen cycles pre-loaded advert images.</p>
ID_verify_capable	Bool	<p>True if device has OCR scan feature</p>

Get Device Video Stream Settings

Returns the device video stream settings.

url: BASE_URL/BASE_PATH/streamsettings (SWBuildVersion >=1093)

method: GET

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: None

response: 200

```
{  
  "Enabled": true,  
  "HTTPEnabled": true,  
  "AuthenticationEnabled": true,  
  "DisplayBox": 1  
}
```

Parameter	type	description
Enabled	Boolean	True – stream is enabled, False - disabled
HTTPEnabled	Boolean	True – ssl enable (https://), False ssl disabled (http://)
Authentication	Boolean	True – token authentication credentials in url required
DisplayBox	Integer	0 = No face box, 1 = Face box

Set Device Video Stream Settings

Set the video stream settings for the device.

url: BASE_URL/BASE_PATH/streamsettings (SWBuildVersion >=1093)

method: POST

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: JSON post body

response: 200

```
{
  "Enabled":true,
  "HTTPSEnabled":true,
  "AuthenticationEnabled":true,
  "AuthUser": "<stream username>",
  "AuthPassword": "<stream password>",
  "DisplayBox":1
}
```

Parameter	type	description
Enabled	Boolean	True – stream is enabled, False - disabled
HTTPSEnabled	Boolean	True – ssl enable (https://), False ssl disabled (http://)
Authentication	Boolean	True – token authentication credentials in url required
DisplayBox	Integer	0 = No face box, 1 = Face box
AuthUser	String	The stream authentication username (max 30 char)
AuthPassword	String	The stream authentication password (max 30 char)

Please note:

After settings for HTTPSEnabled and/or Authentication and credentials are changed, the device must be [reset](#) in order for the new settings to be used.

These settings and any changes made are persistent between power cycles of the device.

Set Face Processing enable/disable

Enable or disable the face processing function. The current status of the Face Processing function is given in the [Get Device Status](#) response.

This will default to 'Enabled = true' on power up and a change in state is not saved so will need to set the desired state during the start-up process.

This value can be set at any time but if it is disabled during a current face process, that process will finish first before disabling future processing.

url: BASE_URL/BASE_PATH/faceprocess

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: JSON post body

response: 200

```
{
  "CameraIndex":0,
  "Enabled":false
}
```

Indicator Control

Control the device indicator LED by setting the colour for a variable duration.

The [settings](#) 'ReactiveLED' parameter must be set to false for this to work, otherwise a 400 response will be given. This disables the indicator for normal operation and allows control using this API request.

url: BASE_URL/BASE_PATH/indicator

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: JSON post body

response: 200

example post body:

```
{
  "Indicator_led":{
    "Color":"red",
    "Duration":3
  }
}
```

Parameter	type	description
Indicator_led	Object	JSON object of control data
Color	String	The required color to illuminate the indicator either "red", "green", "blue", "purple", "teal", "yellow", or "white"
Duration	float	The duration of the illumination in seconds (min 0.5 max 10.0)

Get Device Status

Returns the status of the ICU device.

Use this request to periodically poll the device for status and new detections.

The last update times of face data and action data for this device are also returned. These are used in determining if this device requires data updates.

url: BASE_URL/BASE_PATH/status

method: GET

headers: Content-Type: application/json
Authorization: Bearer *<token>*

Parameters: None

response: 200

```
{
  "DeviceState": "ready",
  "LastFaceUpdate": 1600869048,
  "LastActionUpdate": 1601224129,
  "SingleState": "none",
  "FaceInFrame": [
    {
      "CameraIndex ": 0,
      "Face": true,
      "FaceProcessingEnabled": true,
      "SpoonTerminated": false,
      "Spoon": false,
      "FaceCount": 1,
      "PersonCount": 1,
      "FaceReject": ""
    }
  ],
  "Detections": [
    {
      "Age": 37,
      "Camera": "camera-1",
      "Mask": false,
      "Gender": "male",
      "TrackingNumber": 4
      "Uid": "77803bf5-69a0-4222-a939-4dfb9c0c7790",
      "Image": "/9j/4AAQSkZJRgABAQAAQABAAD/2wBDAAIQAQEB...",
      "Feature":
"LTauMDQyMTE4MTYsLTauMDQxMTU2NjI3LDAuMDkyMzI5ODUsMC4wMjUyNTcyNjc5MC4wMTkxMDE5NDQsLTauMDAzMTcyMzAwOCwwL
..."
    }
  ]
}
```

```

    "FeatureUid": "1f0cd541-45a0-4a44-b13d-62ddb1aaa4ec"
  }
]
}

```

Table showing Status Detail

Name	Type	Description	Note
DeviceStatus	String	The current operational status of the ICU device 'initialising' – loading 'ready' – the device is ready for detection	
LastFaceUpdate	Long Integer	A timestamp showing the last time index that this device has for it's internal data.	
LastActionUpdate	Long integer	A timestamp showing the last time index that this device has for stored action data	
SingleState	String	Used in 1:1 compare mode. Shows the status of a 1:1 transaction: 'none' – mode not set, 'processing' – 1:1 in process, 'result' – status 1:1 result is ready, 'timeout' – 1:1 timed-out without seeing a face.	
FaceInFrame	JSON array	Array of items, one for each enabled camera shows if a face was detected in the camera frame at the time the request was made.	Build >= 1071
Detections	JSON Array	Array of items giving details of an ICU camera detection and result. See table for detail.	

FaceInFrame item

CameraIndex	Integer	The index of the camera	
Spoof	Boolean	True – System detected suspicious activity in the FOV for this camera, Detections void. False – detections valid.	Build >= 1075
Face	Boolean	True – face detected, false – no face seen.	
FaceProcessingEnabled	Boolean	True – enabled for face processing , false – disabled.	
SpoofTerminated	Boolean	True – a session was terminated due to multiple spoof attempts.	Build >= 1098
FaceCount	Integer	The number of pre-processed faces that were counted in the camera view at the time the request was made	Build >=1080

PersonCount	Integer	The number of 'person' objects seen in the camera view at the time the status request was made	Build >=1091
FaceReject	String	Describes a possible reason for live face not being detected	Build>=1100

Detection item

Age	Integer	The estimated age of the face (0 when FRT is disabled and AAE is off)	
Camera	String	The name of the camera detected	
Mask	Boolean	True: ICU calculates this face had a covering over the mouth	Functional on Build < 1090
Gender	String	The gender of the detected face, 'male' or 'female' ('N_A' when FTR is disabled and AAE is off)	
TrackingNumber	Integer	The sequence tracking id of this face	Build >=1093
Uid	String	'none': This face does not match any on the ICU system, otherwise give the unique id of the system face	
Image	String	Base64 encoded string converted from jpg of the detected face.	
Feature*	String	Base64 encoded string of the FaceID for this detection.	Enabled with ?e=1 added to url
FeatureUid*	String	A new UID that can be used along with the above feature data.	Enabled with ?e=1 added to url. Build>=1080

*This would be useful if the system wants to 'enroll' a face directly from the status response without having to set-up and submit a new enrolment image. The base64 decoded string of this feature data can be used as the input to the [Update Face Data](#) request, thus by-passing the [Obtain Face ID data](#) request in the enrolment flow. A UID string is also included in the response to be used with this feature data in the Update Face Request.

This would only be a valid operation if this detection was for a face not identified (Uid='none').

Obtain Face ID Data

Use this request to obtain face data strings for a submitted image. This will perform part of the ‘enroll’ process and will return data to be stored on your external database unique to the presented image. Note that this does not verify an image – just gives the unique ID data for that image.

url: BASE_URL/BASE_PATH/image

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: post data JSON base64 of a face image to get the ID data for. This must be a single face image no larger than 640 x 480 resolution.

```
{
  "ImageType": "face",
  "Image": "/9j/4AAQSkZJRgABAQEAYABgAAD/2wBDAAwMEAwMEBQgFBQQgEBQoHbWYIDAoMDAsKCwsNDhIQDQ4RDgsLEBYQER..."
}
```

Name	Type	Description
ImageType	String	The type of image: ‘face’ – a human face ‘qrcode’ – a QR Code image If this parameter is not given, then the ImageType defaults to ‘face’
Image	string	A base64 encoded string of a jpg or png image.

Response

response: 200

This process requires an ICU device on your network in ‘ready’ status in response to [status](#) request.

The device will perform the neural net calculation for this image and create the unique face id. It will test this against its stored id array. If there is a match – this face already exists in the system and the response status will show ‘already_exists’ together with image data and the uid of the image that it has matched so the user can check his database for user detail.

An ID that does not match any exiting data will return status ‘ok’ and face ID data to be stored in your database against the uid. Note that this data will not be added to the device internal data array

yet. The data needs to be stored in your database and distributed to your chosen devices using the update image request

```
[
  {
    "status": "ok",
    "image": "/9j/4AAQSkZJRgABAQAAQABAAD/2wB...",
    "faceid": "-0.061284766,0.09144592,-0.0012345896,0.006843344,0.038312104,0.0060733845,-0.030254085,-0.015731063,...",
    "uid": "31618a31-fe95-4b82-b7b8-85a16ff6b064",
    "version": "2.0"
  }
]
```

This response is given if the face already exists in the database:

```
[
  {
    "status": "already_exists",
    "image": "/9j/4AAQSkZJRgABAQAAQABAAD//AMmcgfg8L...Z//Z",
    "uid": "77803bf5-69a0-4222-a939-4dfb9c0c7790"
  }
]
```

response: 404 not success

```
{
  "error": "device not ready" // ICU needs to be Ready status
}
```

```
{
  "error": "no faces detected" // unable to detect a face on the given image
}
```

QR Code barcode detection

A feature to allow the ICU device to scan, read and decode a QR Code in front of the camera.

The QR Code data can be 'enrolled' onto the ICU device in the same way as a human face and can then be used in the same way as that enrolled face, i.e. actions can be performed when the QR Code is detected in the same way as when a face is detected.

This function is available in build versions \geq 1100 p032.

The QR code is read by default – no separate enable is required.

Enroll

For the ICU to recognise the QRCode, it must be enrolled onto the system. There are two ways to do this.

- 1) Enroll an image of the barcode in the same way as an image of a face using the [`Obtain Face ID Data`](#) request. Set the 'ImageType' parameter to 'qrcode' and receive the decoded text string and a new uid number back. This text string can then be used in the [`Face Data Update`](#) request to store the qrcode in the device recognition system. Make sure to set the 'Version' parameter to 'QR2.0'.
- 2) Directly store the QR Code data. Use the [`Face Data Update`](#) request to directly add the QR Code string. The 'Data' parameter must be the same as the text that the QR code is encoded to. This string must also be unique, if this data string already exists in the system, then this new string will not be stored. The 'Version' Parameter must be set to 'QR2.0' in order for the system to process the QR code string correctly.

Detection

Any QR Code scanned, decoded, and matching stored QR data will appear in the Get Device Status response. The detected uid will be given, Gender field will be empty, and the Age field will be set to zero.

Actions

Actions can be set on valid QR Code detections in using the QR_DETECT designator. See [Action Data Update](#) request.

ID Document Session

A new feature available in build versions ≥ 1100 will allow the ICU device to start a 'ID session'.

Using this ID session mode, a host system can initiate a ID detection and match session if the host decides that some ID Age verification is required.

The session will start by gathering a series of live face images. This can be started after the host obtains 'consent' using their own system UI dialogues.

Once the images are gathered the camera waits and for an ID card to be detected and then will scan the ID for its photo and text details. The photo image is then compared to the live user images and the ID age is calculated and the results of this are then given to the host for processing.

For this ID session mode, the ICU device will need to be setup with required settings before this mode is valid:

View_mode: OCR

Face_rec_en: true

These are set using the [Settings](#) request during the start-up procedure for the device.

The host [sets the session](#) to start and then then polls the device with a [session](#) request. This will give details of the various states of the session after which the host can take action based on these status result parameters.

Request sequence:

- Connect to ICU (token request)
- Get the [device settings](#).
- Poll [GetStatus](#) to retrieve ages from detected face.
- Once an underage is detected, show your consent dialogue for consent to proceed with ID scan and match.
- Set Face_rec_en:true and View_mode:"OCR" using the [set settings](#) request
- Set the session mode to "face_capture" using the [set session](#) request
- Keep sending the [get session](#) request at intervals. When the session_age_result flag is true in the get session response, send the [get session age result](#) request.
- Send the set session request to "id_scan". A graphic countdown will show while the ICU system changes configuration to enable ID Card scanning.
- The user now holds up their ID to the camera. A streamed video from the ICU displayed will give the user feedback of where to position their ID using a rectangle over the video.
- The ICU will scan the ID Card, attempting to match the photo ID face image with the user live image and scan for age information on the card.
- Keep sending the get session request and check for the session_scan_result flag set to true.
- Send the [get session scan result](#) request to see the ID card scan results.
- After the results are processed, send the set session request to "idle". This will clear the result flag and remove the user and ID card scans from memory.

Get ID Age Session

url: BASE_URL/BASE_PATH/session

method: GET

headers: Authorization: Bearer *<token>*

Parameters: None

Send this request at intervals to get current session statuses.

Response

response: 200 valid request

example response:

```
{
  "Session_mode": "idle",
  "FaceInFrame": true,
  "Session_age_result": false,
  "Session_scan_result": false,
  "Device_status": "ready"
}
```

Parameter	Type	Description
Session_mode	String	The current session mode set by the set session request (see table)
FaceInFrame	Boolean	A face has been detected in the FOV
Session_age_result	Boolean	True: A session age result is available (session age result)
Session_scan_result	Boolean	True: A session ID scan result is available (session scan result)
Device_status	String	Gives the current ICU device status ('initialising' or 'ready')

Session_mode	Description
idle	No action – the device is idling
face_capture	The device is capturing a live face and calculating an estimated age.
id_scan	The device is detecting and scanning an ID card for information

Set ID Age Session

url: BASE_URL/BASE_PATH/session

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: A JSON string containing the session request mode

Response

response: 200 valid request

example of a json request body:

```
{  
  "Session_mode": "face_capture"  
}
```

Session_mode	Description
idle	Puts the device in idle mode
face_capture	Puts the device into face capture and age estimation mode of the live face in the camera FOV. When an estimated age is ready, the session_age_result flag is set to true in the session request response
id_scan	Puts the device into ID Card scan mode. Only valid if a face_capture session has given a true session_age_result otherwise a 400 invalid_response code is given. When scan results are ready.

Get Session Age Result

url: BASE_URL/BASE_PATH/session_age_result

method: GET

headers: Authorization: Bearer *<token>*

Parameters: None

Send when the [session](#) request gives a true session_age_result flag response

Response

response: 200 valid request

```
{"CameraIndex": 0, "CameraName": "camera-2", "Age": 20, "Gender": "male", "Image":  
"/9j/4AAQSkZJRgABAQAAAQABAAD..."}
```

Parameter	Type	Description
CameraIndex	Integer	The device camera index on which the result was scanned
CaneraName	String	The set name for this camera
Age	Integer	The estimated age for this result
Gender	Gender	The measured gender of the face result
Image	String	A base64 encoded string thumbnail image of the face

Get Session Scan Result

url: BASE_URL/BASE_PATH/session_scan_result

method: GET

headers: Authorization: Bearer *<token>*

Parameters: None

Send when the [session](#) request gives a true session_scan_result flag response

Response

response: 200 valid request

- Successful face match and age response:
 - {"IDAge": 61, "IDFaceMatch": true, "IDFacelImage":
"/9j/4AAQSkZJRgABAQAAAQABAAD/
+8z8/wA63brZd2weRlwwx8prj0/ZtX1OjDc0Z3RhmykvQJbjUNjdxGMD9aKIJbWGWR
FkQkHB6UVSdVLQ9lQjbVnms9urPlu36mo9Smu7a1..."IDScanStatus": "complete"}
- No matching face found on scan:
 - {"IDAge": 0, "IDFaceMatch": false, "IDFacelImage":
"/9j/4AAQSkZJRgABAQAAAQABAAD/... ", "IDScanStatus": "no_face_match"}
- A card scan timeout has occurred:
 - {"IDAge": 0, "IDFaceMatch": false, "IDFacelImage": "", "IDScanStatus":
"scan_timeout"}
 -

Parameter	Type	Description
IDAge	Integer	The scanned age from the ID card info
IDFaceMatch	Boolean	Flag to show the status of the ID photo matched to the Live user image (true = matched)
IDFacelImage	String	The scanned photo ID image used to match with a live face
IDScanStatus	String	The status of the result (see table below)

IDScanStatus	Description
complete	An ID scan is complete with AGE result and face image match result
no_face_match	The ID scan session could not match the live face to the photo ID face
scan_timeout	An ID card could not be detected and scanned successfully before the timeout



Once the session is finished (either match, non-match counts reached or time-out) and the 'SingleStatus' shows 'result' The result will be given in the normal Status single detection array:

For a successful match:

```
{
  "Age": 47,
  "Camera": "camera-1",
  "Gender": "male",
  "Mask": false,
  "Uid": "1111-1111-1111-1111",           // the universal ID for 1:1 session match
  "Confidence": 98.3,
  "Image": "/9j/4AAQSkZJRgABAQAAAQABAAD..."
}
```

For non-match:

```
{
  "Age": 59,
  "Camera": "camera-1",
  "Gender": "male",
  "Mask": false,
  "Uid": "none",
  "Confidence": 0,
  "Image": "/9j/4AAQSkZJRgABAQAAAQABAAD/2wBDAAIBAQEBAQIBAQECAgICAgQD..."
}
```

Once the session has completed, the single face array is erased from the ICU device.

Face Data Update

Updates device face data to allow the device to recognise new faces.

url: BASE_URL/BASE_PATH/imageupdate

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: A JSON array of face data to be added to the ICU device system

```
{
  "last_update_time":1600869048,    // this is a UTC timestamp of the last update
  time of your database
  "updates":[
    {
      "Uid":"87ef5953-facd-48a2-b05c-c89f9b62d3a9",
      "Version":"2.0",
      "Data":"0.0861365,0.11962717,0.052714385,0.04335619,...",
      "Group":"Group1"
    },
    {
      "Uid":"b8f2c3ce-37c5-4952-ab7a-dc5edf9a974c",
      "Version":"2.0",
      "Data":"-0.044665553,0.027710449,0.022589475,-0.014485599...",
      "Group":"Group1"
    },
    {
      "Uid":"5ac367af-052a-423b-b040-77cbadd7a8fe",
      "Version":"QR2.0",
      "Data":"J Doe Company access. https:\\company.com aabb-cccc-11223344",
      "Group":"Group1"
    }
  ]
}
```

Name	Type	Description
Uid	String	The global unique identifier for this data. If not using the generated uid, please use a uuid4 id string.
Version	String	Designated the version of this data. Currently "2.0" for face data or "QR2.0" for QR CODE data.
Data	String	The string of data from Obtain Face ID Data response or decoded QR CODE data for QR2.0 type data.
Group	String	The ID Group for which this data is keyed to. This is used in any actions set for ID groups. Different groups can be made to perform different actions. MyCheckr uses fixed group names 'Group1' to 'Group5'.

Face Data Delete

Delete face data from the ICU device

url: BASE_URL/BASE_PATH/imagedelete

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: A JSON array of face uid to be deleted from this device

```
{
  "uid": ["229080f9-f4da-407b-ae86-c4b22e9c47f6", "627005fa-dd6d-4403-ad5e-6e9d21c614a2"]
}
```

An option to delete all face data from the device in one request:

Set the ID element to 'all' will purge all face definition data from this device. This will also reset the 'LastFaceUpdate' field in the [Status](#) response request to zero.

```
{
  "uid": ["all"]
}
```

Action Data Update

Sets the action data for the device.

The ICU device needs to know what to do when it sees a face on a certain camera.

This request allows the user to define what action will take place under what image result on a particular camera.

An ICU device can carry out these actions:

Action	
ICU-LED	Displays the selected colour on the ICU device centre LED
ICU-OUT	Activates the selected output pin on the ICU device
POST-URL-NONE	POST a json object of the result to a selected URL endpoint with no authorisation
POST-URL-BASIC	POST a json object of the result to a selected URL endpoint using basic authorisation headers
POST-URL-TOKEN	POST a json object of the result to a selected URL endpoint with token authorisation

An ICU device has these groups of parameters on which an image result can be acted upon.

Parameter	
FACE_ID	The face has been identified in the system
FACE_NON_ID	The face has not been identified in the system
AGE	The face conditional age
QR_DETECT	A valid QR code has been verified in the system

Each parameter group requires json object data on what to do when the image is processed by the ICU and a result id given

FACE_ID

This requires an array of json objects, one for each of the Group parameters you have defined for your enrolled members.

For example, you have defined 'Enrolled' for general members and 'Excluded' for members not allowed access.

You would need to compile FACE_ID parameter to actions for each Group and each camera in use:

This example shows an actions list for FACE_ID on a device with two active cameras and User Groups Enrolled and Excluded:

```
{
  "camera_index":0,
  "group":"Group1",
  "access":"allowed"
  "actions":[{
    {
      "action":"ICU-OUT",
      "pin":"ONE",
      "ontime":2,
    },
    {
      "action":"POST-URL-BASIC",
      "url":"https://testutl.com/bs",
      "username":"testuser",
      "password":"*****"
    }
  ]
},
{
  "camera_index":0,
  "group":"Group2",
  "access":"denied",
  "actions":[
    {
      "action":"ICU-LED",
      "color":"yellow"
    },
    {
      "action":"ICU-OUT",
      "pin":"TWO"
    }
  ]
}
}
```

FACE_NON_ID

This parameter requires a group of actions for each active camera:

```
{
  "camera_index":0,
  "actions":[
    {
      "action":"ICU-LED",
      "color":"purple"
    }
  ]
},
{
  "camera_index":1,
  "actions":[
    {
      "action":"ICU-LED",
      "color":"yellow"
    },
    {
      "action":"ICU-OUT",
      "pin":"TWO"
    }
  ]
}
```

AGE

This parameter contains conditional data for age actions:

```
{
  "camera_index":1,
  "condition":"less than",
  "age":25,
  "actions":[
    {
      "action":"ICU-LED",
      "color":"red"
    },
    {
      "action":"ICU-OUT",
      "pin":"TWO"
    }
  ]
},
{
  "camera_index":1,
  "condition":"greater than",
  "age":53,
  "actions":[
    {
      "action":"ICU-LED",
      "color":"purple"
    },
    {
      "action":"ICU-OUT",
      "pin":"ONE"
    }
  ]
}
```

QR_DETECT

This requires an array of json objects, one for each of the Group parameters you have defined for your `enrolled` QR Codes. This can be the same as for FACE_ID parameter or different if a different set of actions is required for a barcode over a face detection.

```
{
  "camera_index":0,
  "group":"Group1",
  "access":"allowed"
  "actions":[{
    {
      "action":"ICU-OUT",
      "pin":"ONE",
      "ontime":2,
    },
    {
      "action":"POST-URL-BASIC",
      "url":"https://testutl.com/bs",
      "username":"testuser",
      "password":"*****"
    }
  ]
},
{
  "camera_index":0,
  "group":"Group2",
  "access":"denied",
  "actions":[
    {
      "action":"ICU-LED",
      "color":"yellow"
    },
    {
      "action":"ICU-OUT",
      "pin":"TWO"
    }
  ]
}
}
```


Action Parameters

Parameter	
camera_index	The index of the camera to apply actions to (0 for all devices except PRO which is 0 or 1)
group	The name of the group that an identified face was enrolled with.
access	“allowed” or “denied” to set the access mode of the recognised face (used in MyCheckr to display different screens)
condition	The comparative condition against which to measure age
age	The threshold age for an AGE action

Actions:

ICU-LED

Parameter	
color	The required led COLOR

ICU-OUT

Parameter	
pin	The channel of operation ‘ONE’ or ‘TWO’
ontime	The pulse time in seconds of the output (0,1,2 or 3 only)

The Request:

url: BASE_URL/BASE_PATH/action/update

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: A JSON array of action data to be used on this device

```
{  
  "last_update_time":1601371294,  
  "FACE_ID": [ {...} ],  
  "FACE_NON_ID": [ {...} ],  
  "QR_DETECT":[ {...} ],  
  "AGE":[ {...} ]  
}
```

Activate Action

Give the ability to manually activate an ICU device action

url: BASE_URL/BASE_PATH/action

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

Parameters: A JSON object describing the action to activate

ICU device Actions

ActionType	ActionData	
ICU-LED	red,teal,yellow,green,blue or purple	The desired led colour
ICU-OUT	ONE or TWO	The output pin to activate

Example to activate the red led:

```
{  
  "ActionType": "ICU-LED",  
  "ActionData": "red"  
}
```

Reset Device

Performs a full reset cycle of the ICU device. The device will respond with 200 before the reset is started.

url: BASE_URL/BASE_PATH/reset

method: POST

headers: Content-Type: application/json

Authorization: Bearer *<token>*

post data:

```
{  
  "DeviceId": "38a28ac211"    // the DeviceId from Get Device Detail  
}
```

response: 200 - request accepted.



Update Device

Upload an ICU update package file to the device to perform an ICU device update (build \geq 1068)

url: BASE_URL/BASE_PATH/update/upload

method: POST

headers: Content-Type: multipart/form-data;
Authorization: Bearer <token>

Use multipart form-data to post the update file to the device.

The upload and then update process takes up to 4 minutes with the current package file sizes.`

After the update has run this – request will return 200 for success and the device will auto-restart.

response: 200 - request accepted