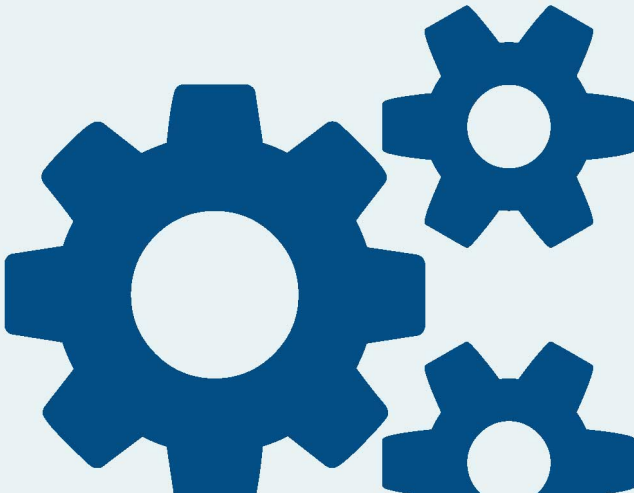
DATAWATCH SYSTEMS
Leading the Way in Access Control

D3000 REST API Guide

for 3rd Party Developer



Revision History

- 12/4/2020: This document was created initially with 20+ APIs which built for card/key management, visitor credentials management, history report and sending panel commands.

Contents

1. Introduction

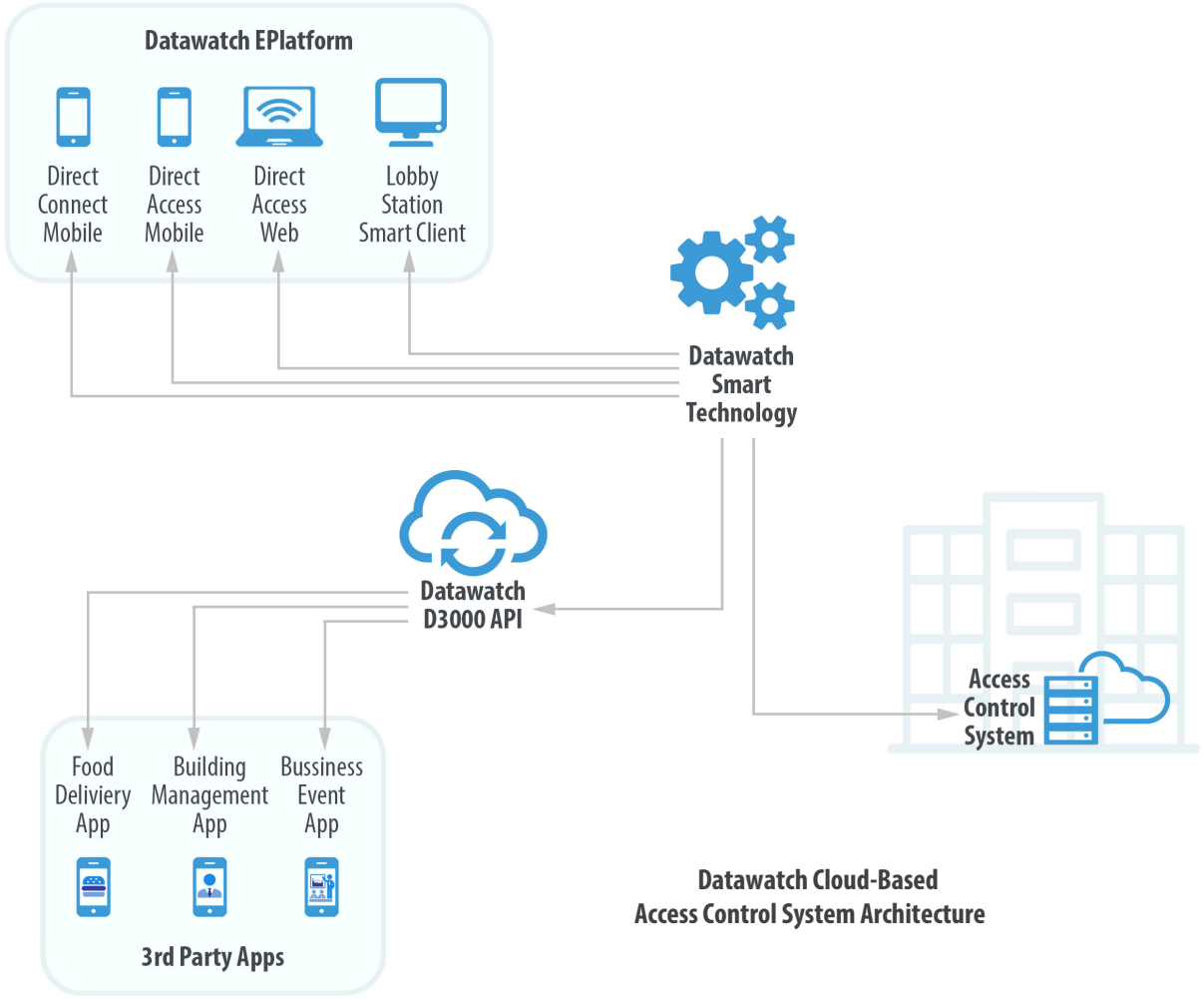
Datawatch API suite offers the features like credentials management, report and sending panel command, etc. Those REST APIs provide programmatic interface not only to create, retrieve, update and delete card holders and visitors but also be able to unlock the secured workforce via cloud command based on the user's privilege. The APIs can be integrated to 3rd party mobile apps for more features, 3rd party platform for eliminating double data entry, and more. This document focuses on getting started how to use the APIs to build 3rd party integrations.

Endpoints Base URL: https://d3000apilite.azurewebsites.net/API/D3000v10/

Allowed HTTPs Requests: POST

This document divided API suite into different sections such as pre-define information look up, employees management and visitor's management, reports and command actions. The responses of APIs are available in JSON. Each API should return any errors with attribute status in returned JSON data parameters when any unexpected conditions occurred. The status messages are well self-explained. It is highly recommended to check the returned status message before processing the payload.

2. How It Works



Datawatch Cloud-Based Access Control System Architecture

Datawatch cloud-based access control system consists of three major components which are on-premise access control panels, Datawatch EPlatform and Datawatch Smart Technology shown as above. Datawatch EPlatform was comprised by web-based client applications including DirectAccess website, DirectAccess app, DirectConnect app and smart client lobby solution. It is powered by Smart Technology engine running on the cloud. Smart Technology was developed to enable the cloud features for on-premise access control panels.

D3000 APIs serves as gateway to Datawatch Smart Technology for third party applications. By interfacing to D3000 APIs, third party applications would be able to provide their users the same experience like EPlatform offers. For example, a delivery service app may assign temp credential to the drivers so the packages can be dropped inside the building with minimum interruption to the clients. Another scheduling service app may send out meeting invites with virtual keys which have temp access to the facility. The imagination never stopped here.

3. About APIs

3.1 Description of Data Parameters

- **Lookup Information API**

Name	Type	Description
access_token	string	Access Token: assigned by Datawatch
user_id	string	User ID: DirectAccess login ID assigned by Datawatch
password	string	Password: the password for DirectAccess login
bldg_id	integer	Building ID: assigned by Datawatch
tent_name	string	Tenant name

- **Card Holders Management API**

Name	Type	Description
access_token	string	Access Token: assigned by Datawatch
user_id	string	User ID: DirectAccess login ID assigned by Datawatch
password	string	Password: the password for DirectAccess login
bldg_id	integer	Building ID: assigned by Datawatch
tent_name	string	Tenant name
card_holder	object	Represent card holder info
first_name	string	First Name of card holder
last_name	string	Last Name of card holder
sitecode	integer	Site code or facility code of credential
embossed	integer	Embossed Number of Credential

cardtype_id	integer	Type of credentials
expire	string	A string with valid date time format represents expiration
pin	integer	Pin code for dual-authorization
sendto_list	array	List of buildings the credential shall be sent to
sendto	object	Represents send to info
sendto_bldg_id	integer	The ID of building where the credential shall be sent to
al1st	integer	1st access level of credential
al2nd	integer	2nd access level of credential
al3rd	integer	3rd access level of credential
al4th	integer	4th access level of credential
al5th	integer	5th access level of credential
modify_al1st	bool	If 1st access level needs to be changed
modify_al2nd	bool	If 2nd access level needs to be changed
modify_al3rd	bool	If 3rd access level needs to be changed
modify_al4th	bool	If 4th access level needs to be changed
modify_al5th	bool	If 5th access level needs to be changed

- **Visitors Management API**

Name	Type	Description
access_token	string	Access Token: assigned by Datawatch
user_id	string	User ID: DirectAccess login ID assigned by Datawatch
password	string	Password: the password for DirectAccess login

bldg_id	integer	Building ID: assigned by Datawatch
tent_name	string	Tenant name
visitor	object	Represents visitor info
first_name	string	Visitor's first name
last_name	string	Visitor's last name
access_level	integer	The access level assigned to the visitor
modify_access_level	bool	If Access level needs to be changed
auth_by	string	The person authorized the visit
begin_date	string	A string in valid date format represents begin date
end_date	string	A string in valid date format represents end date
begin_time	string	Begin time in HHMM format
end_time	string	End time in HHMM format

- **History Report API**

Name	Type	Description
access_token	string	Access Token: assigned by Datawatch
user_id	string	User ID: DirectAccess login ID assigned by Datawatch
password	string	Password: the password for DirectAccess login
bldg_id	integer	Building ID: assigned by Datawatch
begin_time	string	A string in valid date format represents begin time
end_time	string	A string in valid date format represents end time
filter_by	object	Represents filter for the report

tent_name	string	Tenant name
first_name	string	First name
last_name	string	Last name
door	object	Represents a door
panel_id	integer	The id of panel the door wired to
zone_id	integer	The zone id of the panel

- **Action API**

Name	Type	Description
access_token	string	Access Token: assigned by Datawatch
user_id	string	User ID: DirectAccess login ID assigned by Datawatch
password	string	Password: the password for DirectAccess login
bldg_id	integer	Building ID: assigned by Datawatch
sitecode	integer	Site code or facility code of credential
embossed	integer	Embossed Number of Credential
bank_id	integer	Id of elevators bank or group
floor_id	integer	Id of the elevator floor
cab_id	integer	Id of the elevator cab
panel_id	integer	The id of panel the door wired to
zone_id	integer	The zone id of the panel

3.2 HTTP Status Code of Usual Server Responses

- **1xx: Informational – Communicates transfer protocol-level information.**
- **2xx: Success – Indicates that the client's request was accepted successfully.**
- **3xx: Redirection – Indicates that the client must take some additional action in order to complete their request.**
- **4xx: Client Error – This category of error status codes points the finger at clients.**
- **5xx: Server Error – The server takes responsibility for these error status codes.**

3.3 Examples of Status Code

Status	Description
200 OK	the request was successful
400 Bad Request	the request could not be understood or was missing required parameters
401 Unauthorized	authentication failed or user doesn't have permissions for requested operation
403 Forbidden	access denied
404 Not Found	resource was not found
405 Method Not Allowed	requested method is not supported for resource
500 Internal Server Error	generic REST API error response

4. Lookup information API Reference

The lookup information sector consists of lookup APIs could be used to get the pre-defined values like tenant name, access levels, types of credential from Datawatch database. To keep the data integrity, those values passing to those APIs data parameters must match the pre-defined.

4.1 Show Tenants List

Returns json data about the list of tenants. The example of data parameters shown how to get a list of tenants in the building.

** The value assigned to data parameter tent_name must match the values in this list.

- **URL**

  /GetTenantList

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321
}
```

- **Success Response**

  o **Code: 200**

```
Content: {
   "status": "OK",
   "tent_list": [
      {
         "tenant": "Admiral Security",
         "suite": "N/A"
      },
```

```
        {
          "tenant":  "Datawatch",
          "suite":  "500"
        },

        {
          "tenant":  "Red Coats",
          "suite":  "Ste 200"
        },
    ]
}
```


- **Example with error message (invalid building id was passed to API)**

  o **Code: 200**
  Content: {
  "status": "The building does not exist.",
  "tent_list": []
  }

- **Sample Call**

```
curl--location--request
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetTenantList' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id}
           }'
```

4.2 Show Card Type List

Returns json data about the list of card types. The example of data parameters shown how to get a list of available card types for tenant Datawatch.

** The value assigned to data parameter cardtype_id must match the values in this list.

- **URL**

  /GetCardTypeList

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
    "user_id": "demo",
    "password": "demo",
    "bldg_id": 123321,
    "tent_name": "Datawatch"
}
```

- **Success Response**

  - Code: 200
    Content: {

```
    "status": "OK",
    "cardtype_list": [
      {
        "cardtype_id": 2016,
        "card_type": "HID Bluetooth"
      },
      {
        "cardtype_id": 1004,
        "card_type": "HID PROX (34)"
      }
    ]
}
```

- **Example with error message (missed comma after parameter bldg_id)**

  - Code: 200
    Content: {

```
    "status": "After parsing a value an unexpected character was
              encountered: \". Path 'bldg_id', line 6, position 4.\r\n",
    "cardtype_list":  []
}
```

- **Sample Call**

```
curl--location--request
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetCardTypeList' \
--header 'Content-Type: application/json' \
--data-raw '{
              "access_token": "{your token}",
              "user_id": "{your login}",
              "password": "{your password}",
              "bldg_id": {your building id},
              "tent_name": "{your tenant name}"
            }'
```

4.3 Show Access Levels List

Returns json data about the list of access levels. The example of data parameters shown how to get available access levels for tenant Datawatch.

** The values assigned to data parameters al1st, al2nd, al3rd, al4th and al5th must match the values in this list.

- **URL**

  /GetAccessLevelList

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id":  "demo",
   "password":  "demo",
   "bldg_id":  123321,
   "tent_name": "Datawatch Systems"
}
```

- **Success Response**

  - **Code: 200**
    Content: {


```
"status": "OK",
"accesslevel_list": [
    {
        "accesslevel_id": 3,
        "description": "Building Access",
        "asAL1": true,
        "asAL2": false,
        "asAL3": false,
        "asAL4": false,
        "asAL5": false
    },
    {
        "accesslevel_id": 6,
        "description": "DW- Override",
        "asAL1": false,
        "asAL2": true,
        "asAL3": false,
        "asAL4": false,
        "asAL5": false
    },
    {
        "accesslevel_id": 9,
        "description": "DW- 2Fl Conference Room",
        "asAL1": false,
        "asAL2": true,
        "asAL3": false,
        "asAL4": false,
        "asAL5": false
    },
]
}
```


- **Sample Call**

```
curl--location--request
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetAccessLevelList' \
--header 'Content-Type: application/json' \
--data-raw '{

             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "tent_name": "{your tenant name}"

            }'
```

4.4 Show Visitor Access Levels List

Return json data about the list of visitor's access levels. The example of data parameters shown how to get available access levels for Datawatch's visitors.

** The values assigned to data parameter access_level must match the values in this list.

- **URL**

  /GetVisitorAccessLevelList

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "tent_name": "Datawatch Systems"
}
```

- **Success Response**

```
o Code: 200
  Content: {
     "status": "OK",
     "visitor_accesslevel_list": [
        {
           "accesslevel_id": 23,
           "description": "Suite 500 only"
        },
        {
           "accesslevel_id": 49,
           "description": "Visitor Access (2nd Floor Elevator)"
        },
     ]
  }
```

- **Sample Call**

```
curl--location--request
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetVisitorAccessLevelList' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "tent_name": "{your tenant name}"
            }'
```

5. Employee Management API Reference

This section contains the basic CRUD operations for card holder management. The modify operation would not change those attributes if they were passed by empty or blank value. After the API had been called successfully, the backend Smart Technology components would push the changes to on-premise panels automatically.

5.1 Enroll a new card holder

Create a new card holder. The new created card number can be sent to multiple buildings defined by SendTo list. The example of data parameters shown how the credential was sent to two buildings.

- **URL**

  /EnrollCardHolder

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "card_holder": {
      "tent_name": "Datawatch",
      "first_name": "John",
      "last_name": "Doe",
      "sitecode": 2223,
      "embossed": 12347,
      "cardtype_id": 1004,
      "expire": "12/10/2020 5:25 PM",
      "pin": 0,
      "sendto_list": [
         {
            "sendto_bldg_id": 123321,
            "al1st": 6,
            "al2nd": 8,
```

```
            "al3rd": 0,
            "al4th": 0,
            "al5th": 0
        },
        {
            "sendto_bldg_id": 213312,
            "al1st": 4
            "al2nd": 0,
            "al3rd": 0,
            "al4th": 0,
            "al5th": 0
        }
    ]
  }
}
```

- **Success Response**
  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/EnrollCardHolder' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "card_holder": {
              "tent_name": "{your tenant name}",
              "first_name": "{first name of the employee}",
              "last_name": "{last name of the employee}",
              "sitecode": {site code of employee's credential},
              "embossed": {embossed number of employee's credential},
              "cardtype_id": {card type_id of the credential},
              "expire": "{expiration of credential}",
              "pin": "{pin code for dual auth}"
              "sendto_list": [
                {
                  "sendto_bldg_id": {1st building the credential needs to be sent},
                  "al1st": {1st access level of 1st building assigned to credential},
                  "al2nd": {2nd access level of 1st building assigned to credential},
                  "al3rd": {3rd access level of 1st building assigned to credential},
                  "al4th": {4th access level of 1st building assigned to credential},
                  "al5th": {5th access level of 1st building assigned to credential},
                },
```

```
                {
                  "sendto_bldg_id": {2nd building the credential needs to be sent},
                  "al1st": {1st access level of 2nd building assigned to credential},
                  "al2nd": {2nd access level of 2nd building assigned to credential},
                  "al3rd": {3rd access level of 2nd building assigned to credential},
                  "al4th": {4th access level of 2nd building assigned to credential},
                  "al5th": {5th access level of 2nd building assigned to credential},
                },
                   ……
              ]
            }
          }'
```

5.2 Show Card Holder

Return json data about a single card holder. The data parameters require to return card holder details based on credential numbers. The developers may use this API for debugging purpose.

- **URL**

  /GetCardHolder

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "sitecode": 2223,
  "embossed": 12347
}
```

- **Success Response**

  o **Code: 200**
  Content: {
  "status": "OK",
  "card_holder": {
  "tent_name": "Datawatch",
  "first_name": "John",

```
        "last_name": "Doe",
        "sitecode": 2223,
        "embossed": 12347,
        "cardtype_id": 1004,
        "expire": "12/10/2020 5:25 PM",
        "pin": 0,
        "sendto_list": [
          {
            "sendto_bldg_id": 123321,
            "al1st": 6,
             "al2nd": 8,
            "al3rd": 0,
            "al4th": 0,
            "al5th": 0
          },
          {
            "sendto_bldg_id": 213312,
                  "al1st": 4,
                  "al2nd": 0,
                  "al3rd": 0,
                  "al4th": 0,
                  "al5th": 0
          }
        ]
      }
    }
```


- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetCardHolder' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "sitecode": {site code or facility code of the credential},
             "embossed": {embossed number of the credential}
            }'
```

5.3 Modify Card Holder

Update an existing card holder. The modification includes changing card holder information and the access levels assigned to the credential for each building. The example of data parameters shown how to change last name and 1st access level of building #123321 to 16 but leave 2nd access level with no change. All other info were not changed.

- **URL**

  /ModifyCardHolder

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "sitecode": 2223,
   "embossed": 12347,
   "card_holder": {
      "last_name": "Smith",
      "sendto_list": [
         {
            "sendto_bldg_id": 123321,
            "al1st": 16,
            "modify_al1st": true
         }
      ]
   }
}
```

- **Success Response**

  o **Code: 200**
  Content: {
  "status": "OK"
  }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/ModifyCardHolder' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "sitecode": {site code or facility code of the credential},
             "embossed": {embossed number of the credential}
             "card_holder": {
               "tent_name": "{your tenant name}",
               "first_name": "{first name of the employee}",
               "last_name": "{last name of the employee}",
               "sitecode": {site code of employee's credential},
               "embossed": {embossed number of employee's credential},
               "cardtype_id": {card type_id of the credent
               "expire": "{expiration of credential}",
               "pin": "{pin code for dual auth}"
                "sendto_list": [
                  {
                    "sendto_bldg_id": {1st building the credential needs to be sent},
                    "al1st": {1st access level of 1st building assigned to credential},
                    "modify_al1st": {if 1st access level needs to be changed},
                    "al2nd": {2nd access level of 1st building assigned to credential},
                    "modify_al2nd": {if 2nd access level needs to be changed},
                    "al3rd": {3rd access level of 1st building assigned to credential},
                    "modify_al3rd": {if 3rd access level needs to be changed},
                    "al4th": {4th access level of 1st building assigned to credential},
                    "modify_al4th": {if 4th access level needs to be changed},
                    "al5th": {5th access level of 1st building assigned to credential},
                    "modify_al5th": {if 5th access level needs to be changed}
                  },
                  {
                    "sendto_bldg_id": {2nd building the credential needs to be sent},
                    "al1st": {1st access level of 2nd building assigned to credential},
                    "modify_al1st": {if 1st access level needs to be changed},
                    "al2nd": {2nd access level of 2nd building assigned to credential},
                    "modify_al2nd": {if 2nd access level needs to be changed},
                    "al3rd": {3rd access level of 2nd building assigned to credential},
                    "modify_al3rd": {if 3rd access level needs to be changed},
                    "al4th": {4th access level of 2nd building assigned to credential},
                    "modify_al4th": {if 4th access level needs to be changed},
                    "al5th": {5th access level of 2nd building assigned to credential},
                    "modify_al5th": {if 5th access level needs to be changed}
                  },
                     ……
                ]
              }
            }'
```

5.4 Delete Card Holder

Delete an existing card holder. When the card holder is deleted, the access permission to each building would be removed as well.

- **URL**

  /DeleteCardHolder

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "sitecode": 2223,
   "embossed": 12347
}
```

- **Success Response**

  - **Code: 200**
    Content: {
      "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/DeleteCardHolder' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "sitecode": {site code or facility code of the credential},
             "embossed": {embossed number of the credential}
            }'
```

5.5 Add Send To

Send the credential to a building.

- **URL**

  /AddSendTo

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "sitecode": 2223,
  "embossed": 12347,
  "sendto": {
    "sendto_bldg_id": 123312,
    "al1st": 8,
    "al2nd": 16
  }
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/AddSendTo' \
--header 'Content-Type: application/json' \
--data-raw '{
              "access_token": "{your token}",
              "user_id": "{your login}",
```

```
    "password": "{your password}",
    "bldg_id": {your building id},
    "sitecode": {site code or facility code of the credential},
    "embossed": {embossed number of the credential}
    "sendto": {
      "sendto_bldg_id": {the building needs to be sent},
      "al1st": {1st access level of credential},
      "al2nd": {2nd access level of credential},
      "al3rd": {3rd access level of credential},
      "al4th": {4th access level of credential},
      "al5th": {5th access level of credential}
    }
  }'
```

5.6 Modify Send To

Modify the access levels of the credential sent to a building already.

- **URL**

  /ModifySendTo

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "sitecode": 2223,
  "embossed": 12347,
  "sendto": {
    "sendto_bldg_id": 123312,
    "al1st": 20,
    "modify_al1st": true,
    "al2nd": 0,
    "modify_al2nd": true,
    "al3rd": 48,
    "modify_al3rd": true
  }
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10//ModifySendTo' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "sitecode": {site code or facility code of the credential},
             "embossed": {embossed number of the credential}
             "sendto": {
               "sendto_bldg_id": {the building needs to be sent},
               "al1st": {1st access level of credential},
               "modify_al1st": {if 1st access level needs to be modified},
               "al2nd": {2nd access level of credential},
               "modify_al2nd": {if 2nd access level needs to be modified},
               "al3rd": {3rd access level of credential},
               "modify_al3rd": {if 3rd access level needs to be modified},
               "al4th": {4th access level of credential},
               "modify_al4th": {if 4th access level needs to be modified},
               "al5th": {5th access level of credential},
               "modify_al5th": {if 5th access level needs to be modified},
             }
           }'
```

5.7 Remove Send To

Remove the access privilege from a building.

- **URL**

  /RemoveSendTo

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "sitecode": 2223,
   "embossed": 12347
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/RemoveSendTo' \
--header 'Content-Type: application/json' \
--data-raw '{
              "access_token": "{your token}",
              "user_id": "{your login}",
              "password": "{your password}",
              "bldg_id": {your building id},
              "sitecode": {site code or facility code of the credential},
              "embossed": {embossed number of the credential}
              "sendto": {
                "sendto_bldg_id": {the access to the building needs to be removed},
              }
             }'
```

6. Visitor Management API Reference

This section contains the basic CRUD operations and check in action for visitor management. The modify operation would not change those attributes if they were passed by empty or blank value. After the API had been called successfully, the backend Smart Technology components would push the changes to on-premise panels automatically.

6.1 Enroll a new visitor

Create a new visitor. When a new visitor is enrolled via API, the visitor's encoded number shall be returned from API and it can be printed in bar code or QR code to issue to visitors as temporary access badges by 3rd party software. The example of data parameters shown how to enroll a new visitor.

- **URL**

  /EnrollVisitor

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "visitor": {
     "tent_name": "Datawatch",
     "first_name": "John",
     "last_name": "Doe",
     "access_level": 201,
     "auth_by": "John Smith",
     "begin_date": "12/2/2020",
     "end_date": "12/2/2020",
     "begin_time": "1000",
     "end_time": "1200"
  }
 }
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    "encoded": 11125940
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/EnrollVisitor' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "visitor": {
               "tent_name": "{your tenant name}",
               "first_name": "{visitor's first name}",
               "last_name": "{visitor's last name}",
               "access_level": {visitor's access level created by Datawatch},
               "auth_by": "{full name of office admin}",
               "begin_date": "{begin date of the visit}",
               "end_date": "{end date of the visit}",
               "begin_time": "{begin time of the schedule}",
               "end_time": "{end time of the schedule}"
             }
            }'
```

6.2 Show Visitor Info

Return json data about a single visitor. The data parameters require to return visitor details based on credential numbers. The developers may use this API for debugging.

- **URL**

  /GetVisitor

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id": "demo",
   "password": "demo",
   "bldg_id": 123321,
   "encoded": 11125940
}
```

- **Success Response**

  - **Code: 200**
    Content: {

```
   "status": "OK",
   "visitor": {
      "tent_name": "Datawatch",
      "first_name": "John",
      "last_name": "Doe",
      "access_level": 201,
      "auth_by": "John Smith",
      "begin_date": "12/02/2020",
      "end_date": "12/02/2020",
      "begin_time": "1600",
      "end_time": "1800"
   }
}
```

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetVisitor' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "encoded": {visitor credential number}
           }'
```

6.3 Modify Visitor

Update an existing visitor. The modification includes changing the access privilege to secured space. The example of data parameters shown how to change last name and access level. All other info was not changed.

- **URL**

  /ModifyVisitor

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id":  "demo",
   "password":  "demo",
   "bldg_id":  123321,
   "encoded": 11125940
   "visitor": {
         "last_name": "Smith",
         "access_level": 201,
         “modify_access_level”: true
      }
}
```

- **Success Response**

  o **Code: 200**
  Content: {
  "status": "OK"
  }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/ModifyVisitor' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "encoded": {encoded number of visitor credential},
             "visitor": {
               "tent_name": "{your tenant name}",
               "first_name": "{visitor's first name}",
               "last_name": "{visitor's last name}",
               "access_level": {visitor's access level},
               "modify_access_level": {if the access level needs to be modified},
               "auth_by": {the person who authorized the visit},
               "begin_date": "{begin date of the visit}",
               "end_date": "{end date of the visit}",
               "begin_time": "{begin time of the schedule}",
               "end_time": "{end time of the schedule}"
             }
            }'
```

6.4 Delete Visitor

Delete an existing visitor.

- **URL**

  /DeleteVisitor

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id":  "demo",
  "password":  "demo",
  "bldg_id":  123321,
  "encoded": 11125940
 }
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/DeleteVisitor' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "encoded": {encoded number of visitor credential}
           }'
```

6.5 Check-In Visitor

Check in a pre-scheduled visitor. This API shall be used if the building requires the visitors to check in first before their credentials become active.

- **URL**

  /CheckInVisitor

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "encoded": 11125940
 }
```

- **Success Response**
  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/CheckInVisitor' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "encoded": {encoded number of visitor credential}
           }'
```

7. History Report API Reference

This section comprises the report APIs. For history report, it is recommended to retrieve the history records page by page. If the page size is over the limit, the API would throw a exception. GetHistoryCount shall be used to determine the proper page size.

7.1 Get Total Number of History Records

Get the total number of history records. Before calling GetHistoryList, this API should be called so the proper page size can be set based on the total records.

- **URL**

  /GetHistoryCount

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
   "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
   "user_id":  "demo",
   "password":  "demo",
   "bldg_id":  123321,
   "begin_time":  "11/01/2020 08:00",
   "end_time":  "11/10/2020 17:00",
   "filter_by":  {
      "tent_name": "",
      "first_name": "",
      "last_name": "",
      "door":  {
         "panel_id": 0,
         "zone_id": 0
      }
   }
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK",
    "records_total": 1486
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetHistoryCount' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "begin_time": {begin time for the history report},
            "end_time": {end time for the history report}
            "filter_by": {
              "tent_name": "{optional, tenant name}",
              "first_name": "{optional, first name of employee or visitor}",
              "last_name":  "{optional, last name of employee or visitor}",
              "door":  {
                "panel_id": {optional, the panel id the door is wired to},
                "zone_id": {optional, the zone id of the panel}
              }
            }
           }'
```

7.2 Get History List

Return json data about a list of history.

- **URL**

  /GetHistoryList

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "begin_time": "11/01/2020 08:00",
  "end_time": "11/10/2020 17:00",
  "records_page_index": 0,
  "records_page_size": "3",
  "filter_by": {
    "tent_name": "",
    "first_name": "",
    "last_name": "",
    "door": {
      "panel_id": 0,
      "zone_id": 0
    }
  }
}
```

- **Success Response**

  - **Code: 200**

```
Content: {
  "status": "OK",
  "history_list": [
    {
      "datetime": "11/01/2020 08:00:00 AM",
      "unit_id": 12345
      "zone_id": 3,
      "event": "Access Granted At 5Fl Data Processing Main Dr",
      "event_address": "1234 Main Street",
      "home_address": "1234 Main Street",
      "first_name": "Vince",
      "last_name": "Jackson",
      "tent_name": "Datawatch Systems",
    },
    {
      "datetime": "11/01/2020 08:11:42 AM",
      "unit_id": 23456,
      "zone_id": 1,
      "event": "Access Granted At MAIN LBY DOOR",
      "event_address": "1234 Main Street",
      "home_address": "1234 Main Street",
      "first_name": "Mike",
      "last_name": "Hillman",
      "tent_name": "Datawatch Systems",
    },
```

```
        {
            "datetime":"11/01/2020 08:12:58 AM",
            "unit_id":12345,
            "zone_id":4,
            "event":"Access Granted At 5FL CENTRAL IN RDR",
            "event_address": "1234 Main Street",
            "home_address": "1234 Main Street",
            "first_name":"John",
            "last_name":"Smith",
            "tent_name":"Datawatch Systems",
        }
    ]
}
```


- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/GetHistoryList' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "begin_time": {begin time for the history report},
             "end_time": {end time for the history report},
             "records_page_index": {zero-based page index},
             "records_page_size": {total records per page},
             "filter_by": {
               "tent_name": "{optional, tenant name}",
               "first_name": "{optional, first name of employee or visitor"},
               "last_name": "{optional, last name of employee or visitor}",
               "door":  {
                 "panel_id": {optional, the panel id the door is wired to},
                 "zone_id": {optional, the zone id of the panel}
               }
             }
           }'
```


8. Panel Command API Reference

This section contains a set of APIs are able to send commands to access control panels. Those commands include calling elevators, select in-cab floor buttons, unlock the doors, etc.

8.1 Make UP Call for an Elevator Bank

Make UP call request to the elevator bank. The UP button on the wall shall be illumined after a successful call. The example of data params shown a person with credential 1234-34567 made a UP call at 2nd floor.

- **URL**

  /CallElevatorUp

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
  "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
{ "user_id":  "demo",
  "password":  "demo",
  "bldg_id":  123321,
  "sitecode": 1234,
  "embossed": 34567,
  "bank_id": 1,
  "floor_id": 2
}
```

- **Success Response**

```
o Code:  200
  Content:  {
     "status":  "OK"
  }
```

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/CallElevatorUp' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "sitecode": {site code of credential},
            "embossed": {embossed number of credentials},
            "bank_id": {elevators bank id},
            "floor_id": {floor id of source floor}
          }'
```

8.2 Make DOWN Call for an Elevator Bank

Make DOWN call request to the elevator bank. The DOWN button on the wall shall be illumined after a successful call. The example of data params shown a person with credential 1234-34567 made a DOWN call at 5th floor.

- **URL**

  /CallElevatorDown

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id":  "demo",
  "password":  "demo",
  "bldg_id":  123321,
  "sitecode": 1234,
  "embossed": 34567,
  "bank_id": 1,
  "floor_id": 5
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/CallElevatorDown' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "sitecode": {site code of credential},
            "embossed": {embossed number of credentials},
            "bank_id": {elevators bank id},
            "floor_id": {floor id of source floor}
          }'
```

8.3 Push an In-Cab Floor Button

Select an in-cab floor button for destination. The floor button shall be illumined after a successful call. The example of data params shown a person with credential 1234-34567 selected 5th floor for destination.

- **URL**

  /PushInCabButton

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token": "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id": "demo",
  "password": "demo",
  "bldg_id": 123321,
  "sitecode": 1234,
  "embossed": 34567,
  "cab_id": 1,
  "floor_id": 5
}
```

- **Success Response**

  - **Code: 200**
    Content: {
    "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/PushInCabButton' \
--header 'Content-Type: application/json' \
--data-raw '{
            "access_token": "{your token}",
            "user_id": "{your login}",
            "password": "{your password}",
            "bldg_id": {your building id},
            "sitecode": {site code of credential},
            "embossed": {embossed number of credentials},
            "cab_id": {elevators cab id},
            "floor_id": {floor id of destination floor}
          }'
```

8.4 Unlock A Door

Unlock a secured door. The example of data params shown a person with credential 1234-34567 unlocked the door wired to zone #1 of panel #100.

- **URL**

  /Unlock

- **Method**

  POST

- **URL Params**

  None

- **Data Params**

```
{
  "access_token":  "CCCCCCCCCCAAAAAAAAAAAAAABBBBBBBB",
  "user_id":  "demo",
  "password":  "demo",
  "bldg_id":  123321,
  "sitecode": 1234,
  "embossed": 34567,
  "panel_id": 100,
  "zone_id": 1
}
```

- **Success Response**

  - **Code: 200**
    Content: {
      "status": "OK"
    }

- **Sample Call**

```
curl--location--request POST
'https://d3000apilite.azurewebsites.net/API/D3000v10/Unlock' \
--header 'Content-Type: application/json' \
--data-raw '{
             "access_token": "{your token}",
             "user_id": "{your login}",
             "password": "{your password}",
             "bldg_id": {your building id},
             "sitecode": {site code of credential},
             "embossed": {embossed number of credentials},
             "panel_id": {panel id of the door wired to},
             "zone_id": {zone id of the panel}
           }'
```