The Beginner's Guide on BioStar API

April 2017
Hailey Park

suprema

Contents

Basic Concept

Possible integration options

BioStar 2 SDK

When you want..

- To have your own server program for Suprema devices
- To directly access to the device to control and retrieve information
- To have a single system, not multiple

BioStar 2 API

When you want..

- To use the BioStar 2, while having certain UI for specific features
- To use the BioStar 2 Mobile Application with your system
- To integrate BioStar 2 with your system

Basic Concept

BioStar 2 API version 1 vs Biostar 2 API version 2

- BioStar 2 API version 1 (V1 API)
- BioStar 2 API version 2 (V2 API)

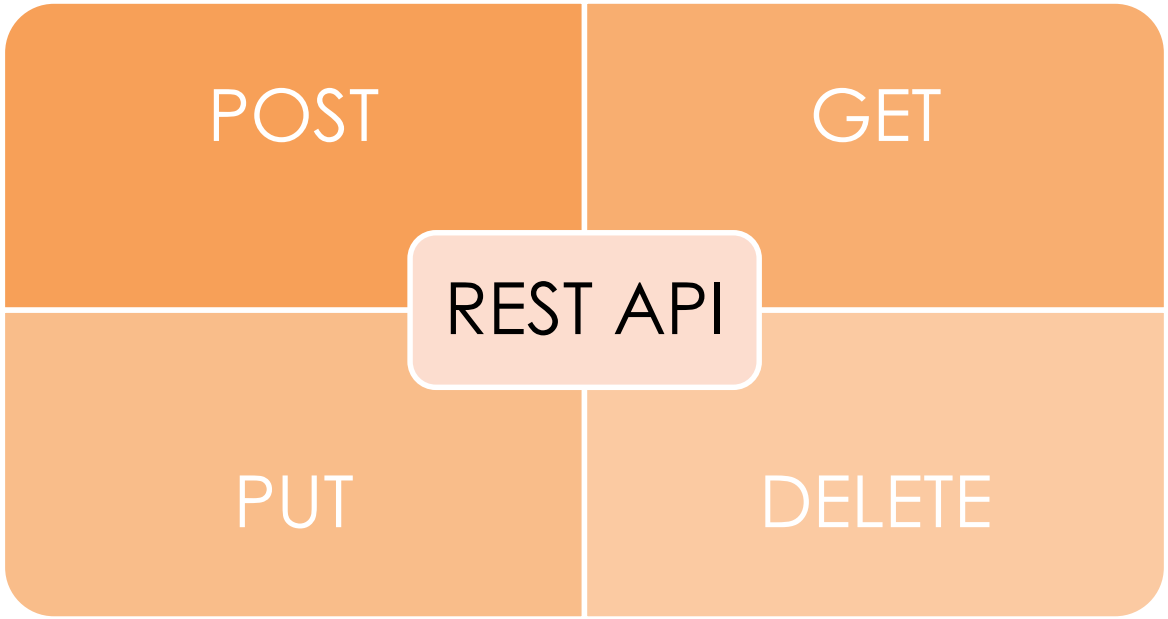
Item	V1 API	V2 API
Compatible Version	BioStar v2.3 or below	BioStar v2.4 or above
Custom Admin	Not Supported	Supported
Authorized	Multi Signing	Single Signing

<Major Differences>

Basic Concept

BioStar 2 API

- REST API (Representational State Transfer API)
- Call BioStar 2 API using HTTP METHOD

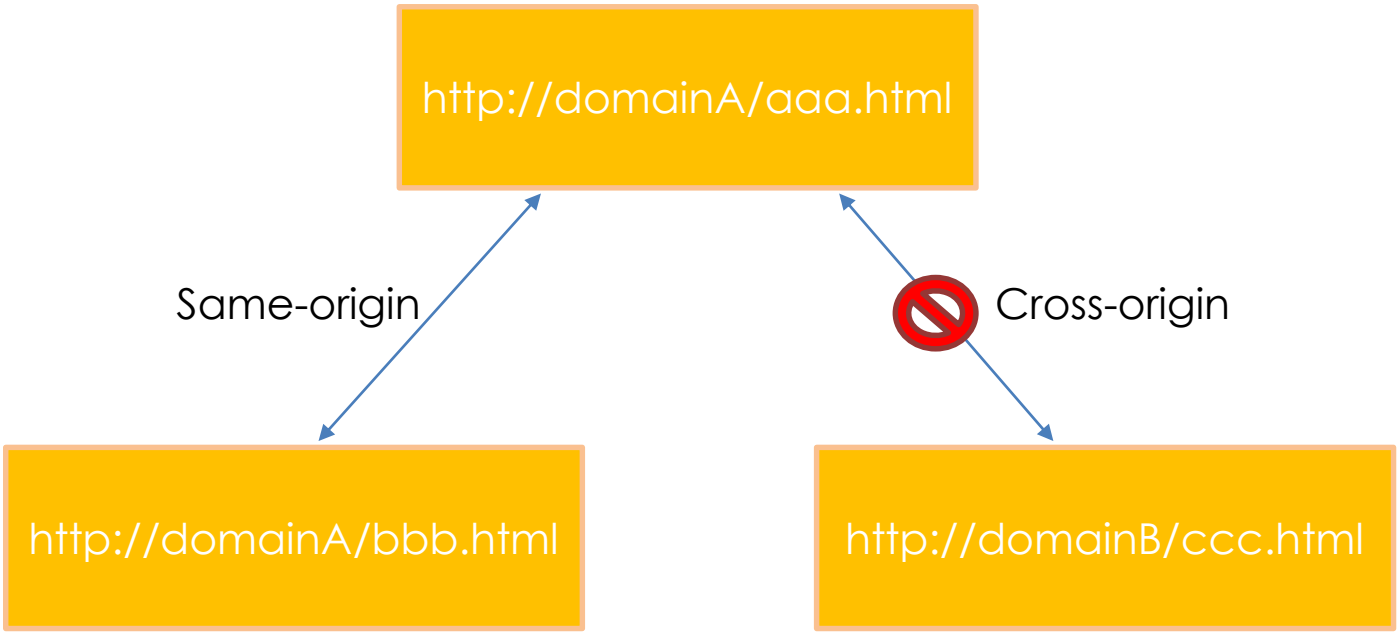


<HTTP Method>

Basic Concept

Cross Domain Issue

- You cannot call BioStar 2 API using JavaScript
- Due to the Same-Origin Policy



Basic Concept

Types of BioStar 2 API

BioStar 2 Cloud API

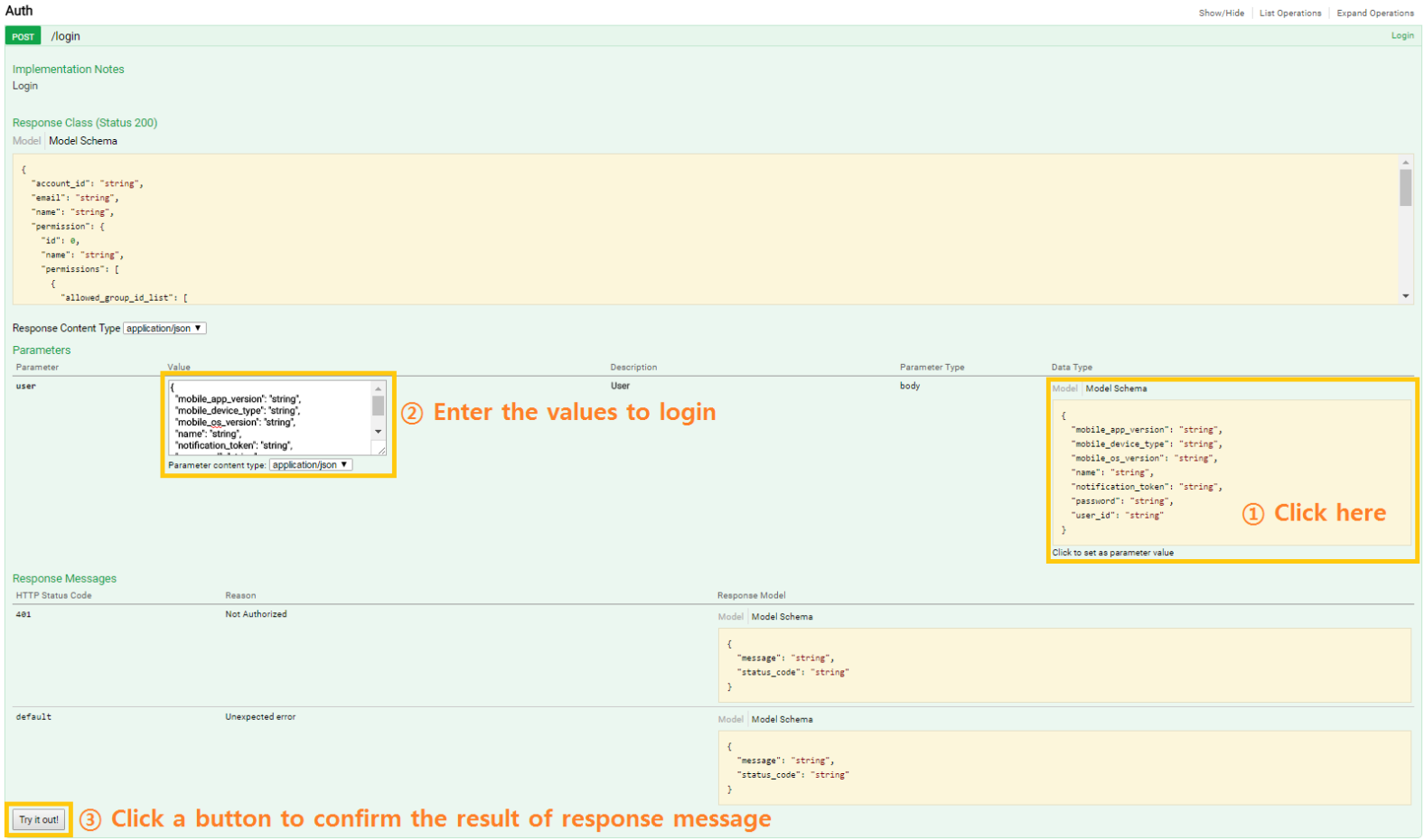
VS

BioStar 2 Local API

Basic Concept

How to utilise “Swagger UI”

- https://api.biostar2.com/v2/docs/
- POST/PUT message (e.g. POST /login API)



Basic Concept

Usage of Chrome App "Postman"

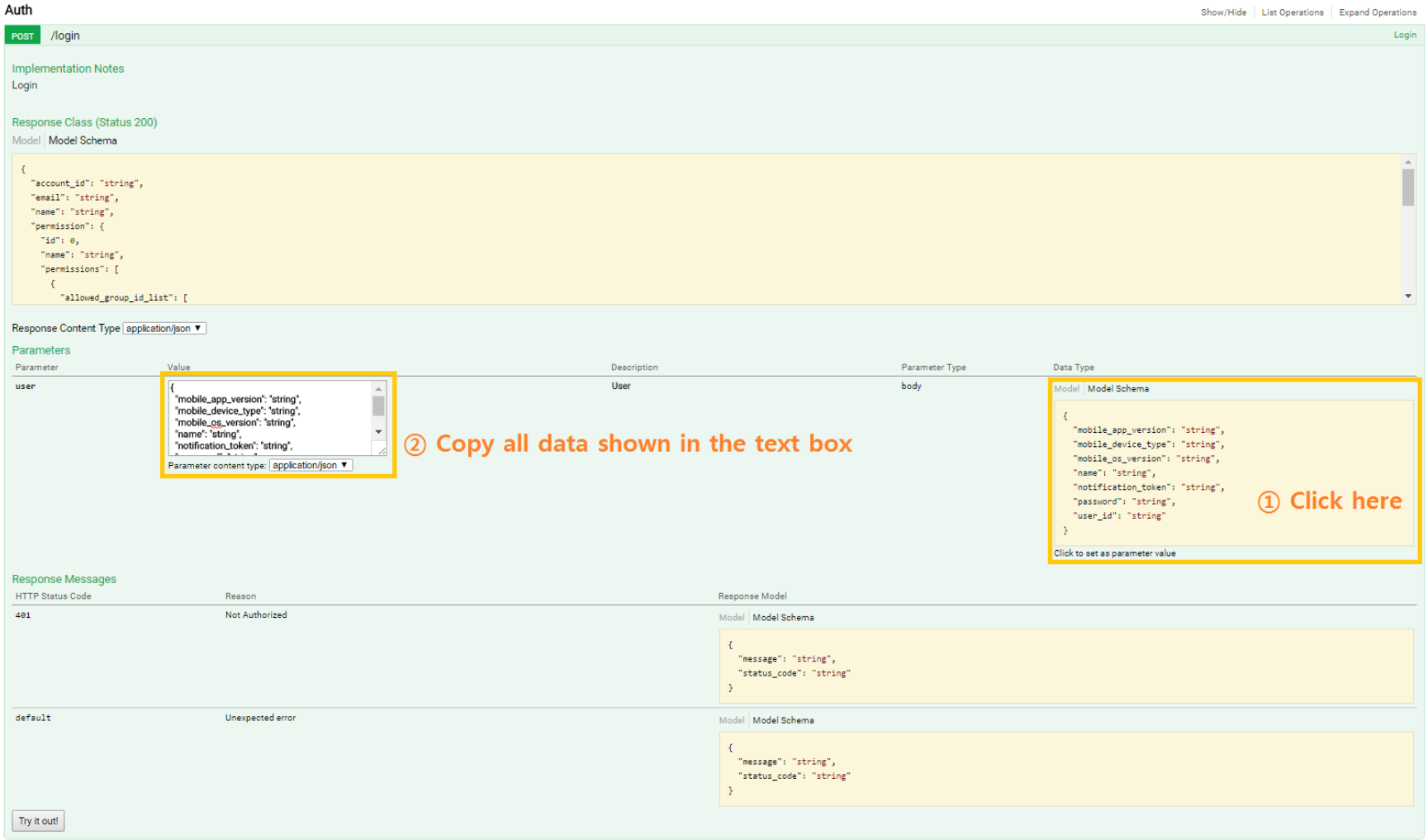
- https://chrome.google.com/webstore/search/Postman?hl=en
- You can confirm the result of calling API easily via Postman

<Source: www.getpostman.com>

Basic Concept

How to utilise “Postman” with “Swagger UI” [1/5]

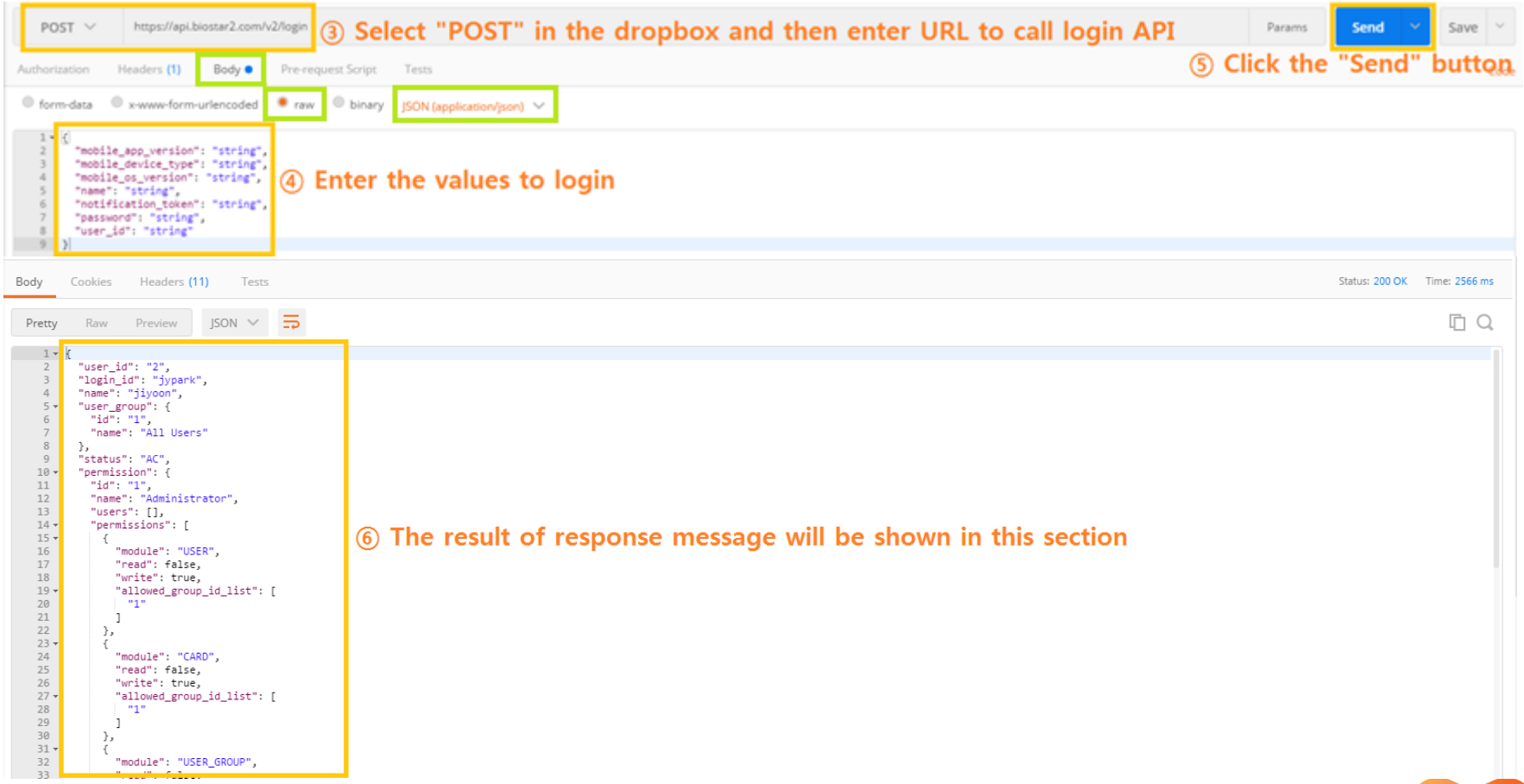
- POST/PUT message (e.g. POST /login API)



Basic Concept

How to utilise “Postman” with “Swagger UI” [2/5]

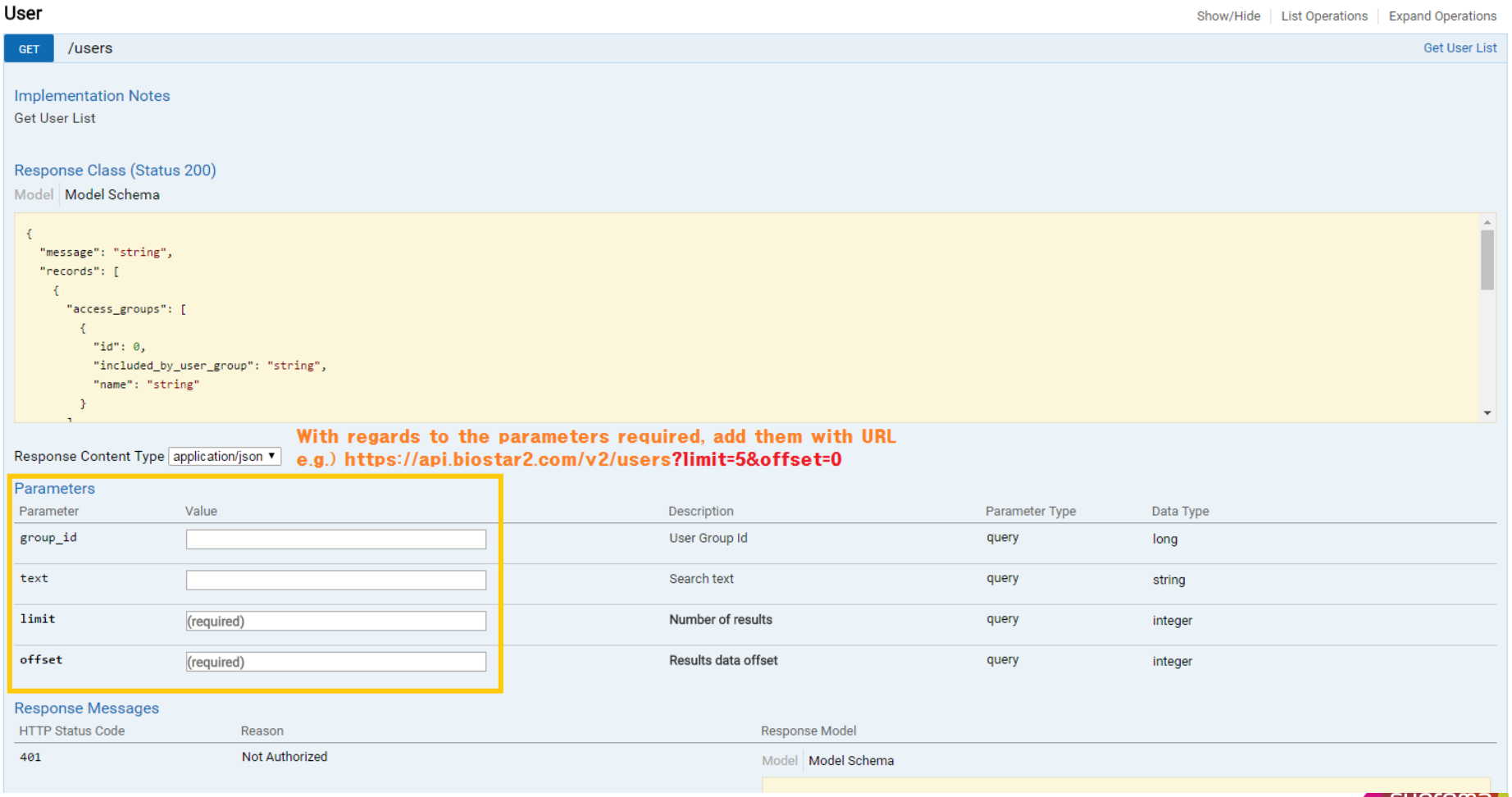
- POST/PUT message (e.g. POST /login API)



Basic Concept

How to utilise “Postman” with “Swagger UI” [3/5]

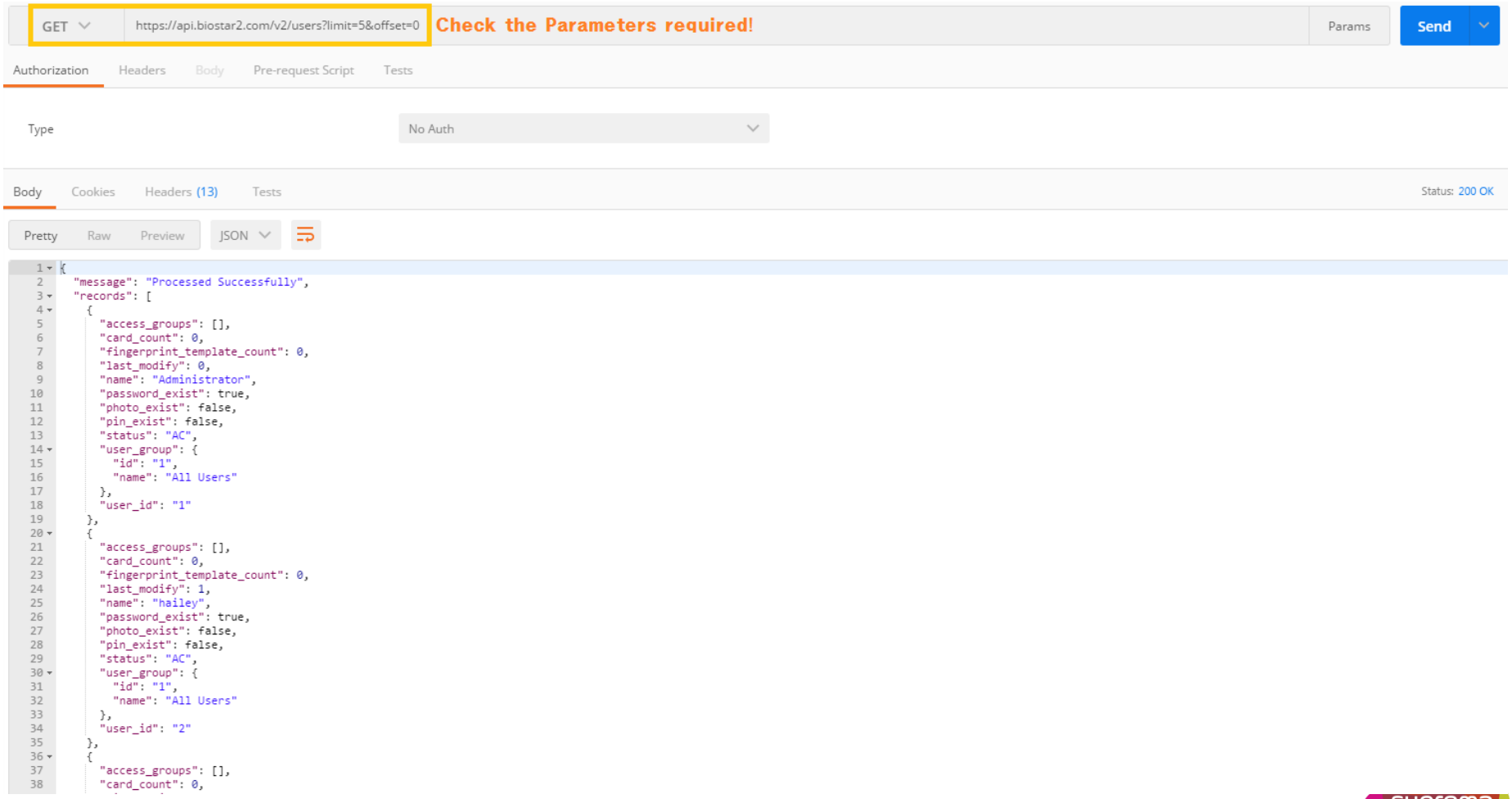
- GET message (e.g. GET /users API)



Basic Concept

How to utilise “Postman” with “Swagger UI” [4/5]

- GET message (e.g. GET /users API)



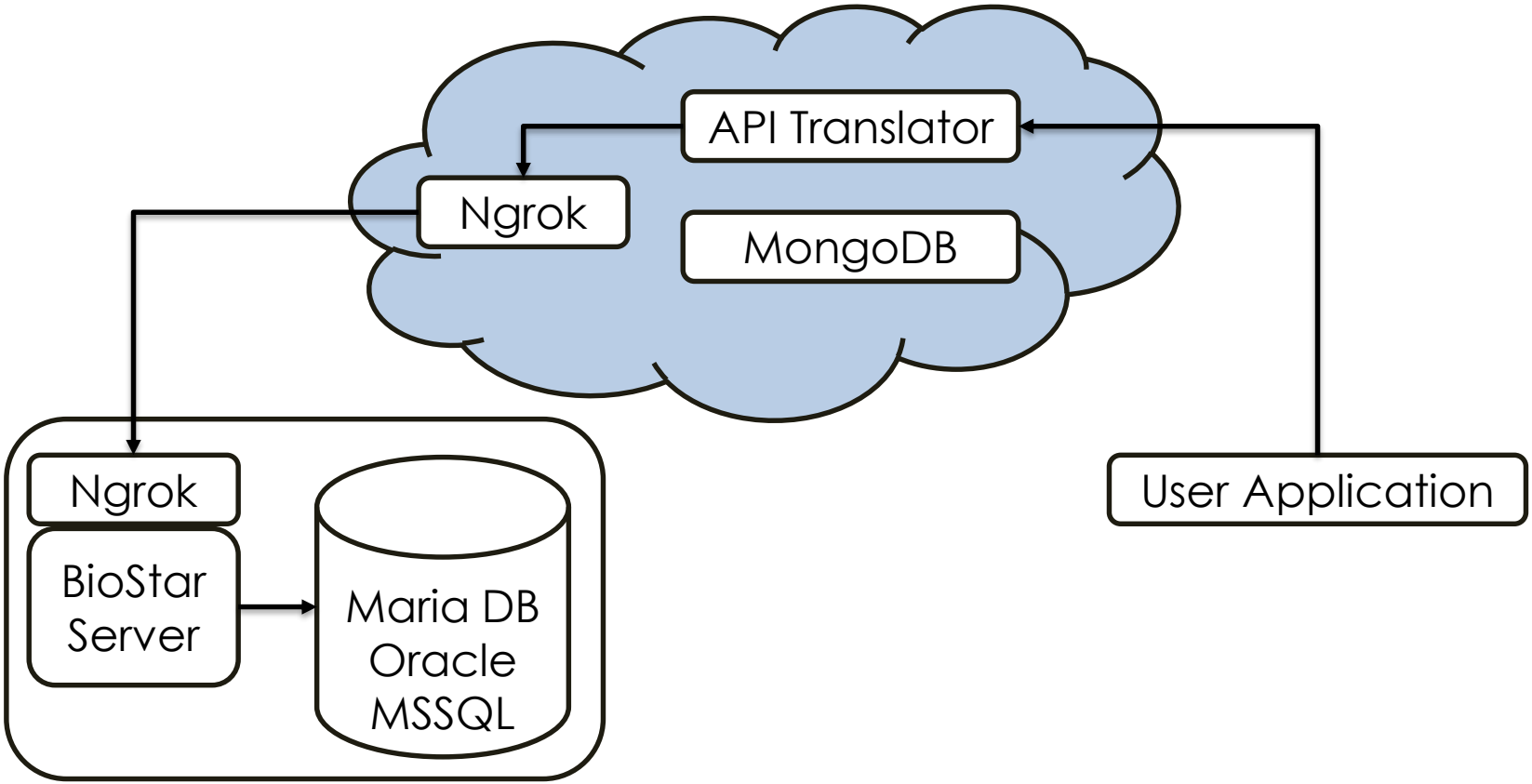
Basic Concept

How to utilise “Postman” with “Swagger UI” [5/5]

Video Demo

Icon made by SimpleIcon from www.flaticon.com

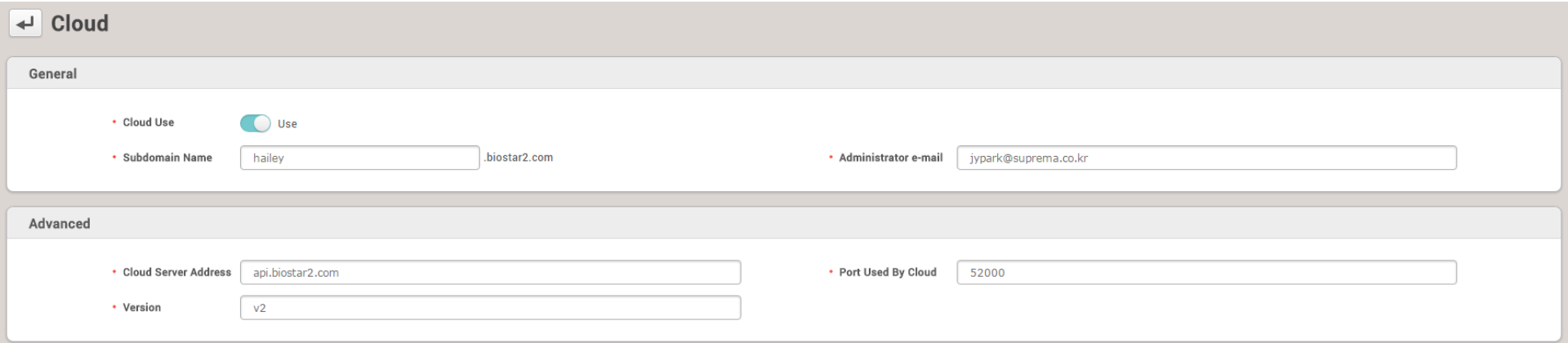
BioStar 2 Cloud API



BioStar 2 Cloud API

Cloud Setting [1/2]

- Go to **Setting** > **SERVER** and then set the **Password Level** to either **Medium** or **Strong**
- Go to **Setting** > **CLOUD** and change Cloud Setting
  - Cloud Use : Use
  - Subdomain Name : Enter this used for url
  - Administrator e-mail : Enter email address for cloud administrator
  - Cloud Server Address : api.biostar2.com (In case of using local API, this should be changed)
  - Port Used By Cloud : 52000 (default value, but you can change it only if the port is used by other application.
  - Version : v2



BioStar 2 Cloud API

Cloud Setting [2/2]

NOTICE

- ✓ In case of reinstalling BioStar 2, we recommend the 'setting.conf' file should be backed up since the cloud setting will be initialized like subdomain, account id and so forth

- ✓ After reinstalling BioStar 2, follow several steps below
  1) Go to Setting > SERVER and then set the password to either Medium or Strong
  2) Stop BioStar 2's service
  3) Open the 'setting.conf' file and modify into the existing setting

BioStar 2 Cloud API - Example

Auth [1/2]

- [POST] /login
- Whenever you call other API, you have to login first

<Major Parameters>

Name	Type	*M/O	Explanation	Value
mobile_app_version	string	O	Mobile App Version	
mobile_device_type	string	O	Mobile Device Type	'ANDROID' = Android Device, 'IOS' = Apple Device
mobile_os_version	string	O	Mobile OS Version	
name	string	M	**Subdomain Name**	
notification_token	string	O	Notification Token (Google GCM Registration ID or Apple Push Notification Device Token)	
password	string	M	User Password	
user_id	string	M	Login ID	

* M - Mandatory, O - Optional

BioStar 2 Cloud API - Example

Auth [2/2]

- [POST] /login

```
public static string sessionID;                                   // Global variable for maintaining a session

static async void LoginTask()
{
    string resourceAddress = "https://api.biostar2.com/v2/login"; // Request URL

    HttpClient httpClient = new HttpClient();

    JavaScriptSerializer serializer = new JavaScriptSerializer();

    Dictionary<string, string> dicLoginUser = new Dictionary<string, string>();   // Parameters
    dicLoginUser.Add("name", "hailey");
    dicLoginUser.Add("password", "administrator!1");
    dicLoginUser.Add("user_id", "hailey");

    string jsonLoginUser = serializer.Serialize(dicLoginUser);

    StringContent sc = new StringContent(jsonLoginUser, Encoding.UTF8, "application/json");
    HttpResponseMessage httpResponse = await httpClient.PostAsync(resourceAddress, sc);
                                                                  // HTTP Method - POST
    if (httpResponse.IsSuccessStatusCode == true)
    {
        Console.WriteLine(httpResponse.ToString());
        string httpResponseBody = await httpResponse.Content.ReadAsStringAsync();
        Console.WriteLine(httpResponseBody);

        MemoryStream responseMemoryStream = new MemoryStream();
        StreamWriter sw = new StreamWriter(responseMemoryStream);
        sw.Write(httpResponse.ToString());
        sw.Flush();
                                                                  // Process to get session ID
        bool isSessionIDContained = httpResponse.Headers.Contains("Set-Cookie");
        if (isSessionIDContained == true)
        {
            IEnumerable<string> sessionEnum = httpResponse.Headers.GetValues("Set-Cookie");
            foreach (string element in sessionEnum)
            {
                Console.WriteLine("Set-Cookie: " + element);
                string[] strCookieArr = element.Split(new string[] { "bs-cloud-session-id=" }, StringSplitOptions.None);
                string[] strCookieArr2 = strCookieArr[1].Split(new string[] { ";" }, StringSplitOptions.None);
                sessionID = strCookieArr2[0];
            }
        }
        else
        {
            Console.WriteLine("Session ID not found");
        }
    }
    else
    {
        Console.WriteLine("Failed to log in");
        Console.WriteLine(httpResponse.ToString());
    }
}
```

BioStar 2 Cloud API - Example

Access Control [1/2]

- [GET] /access_groups

<URL Parameters>

Name	Type	*M/O	Explanation
text	string	O	Search text
limit	integer	M	Number of results
offset	integer	M	Results data offset

* M - Mandatory, O - Optional

BioStar 2 Cloud API - Example

Access Control [2/2]

- [GET] /access_groups

```
static async void AccessGroupsTask()
{
    if (sessionID == null)
    {
        Console.WriteLine("You must log in first!");
        return;
    }

    CookieContainer cookieContainer = new CookieContainer();

    HttpClientHandler handler = new HttpClientHandler();
    handler.CookieContainer = cookieContainer;

    HttpClient client = new HttpClient(handler);

    Console.WriteLine("Input limit: ");
    string limit = Console.ReadLine();
    Console.WriteLine("Input offset: ");
    string offset = Console.ReadLine();

    cookieContainer.Add(new Uri("https://api.biostar2.com"), new Cookie("bs-cloud-session-id", sessionID));
    HttpResponseMessage httpResponse = await client.GetAsync("https://api.biostar2.com/v2/access_groups?limit=" + limit + "&offset=" + offset);

    if (httpResponse.IsSuccessStatusCode == true)
    {
        string httpResponseBody = await httpResponse.Content.ReadAsStringAsync();
        Console.WriteLine(httpResponseBody);
    }
    else
    {
        Console.WriteLine("Retrieving Access Groups Failed");
        Console.WriteLine(httpResponse.ToString());
    }
}
```

If you are successful in login, the session ID is not null

The parameters, both limit and offset, are essential

HTTP Method - GET

BioStar 2 Cloud API - Example

User [1/5]

- [POST] /users
- The information on the Created new user is automatically synchronized with BioStar 2
- JSON Structure

Type 1 Basic Level

```
{
   Key : Value,
   Key : Value,
   Key : Value
}
```

Type 2 Multiple Values

```
{
   Key : [
     Value,
     Value,
     Value
   ]
}
```

Type 3 Multi Level

```
{
   Key : {
     Key :
Value,
     Key :
Value
   }
}
```

Type 4 Multi Level

```
{
   Key : [
     {
       Key :
Value,
       Key :
Value
     }
   ]
}
```

BioStar 2 Cloud API - Example

User [2/5]

- [POST] /users
- JSON Structure (Multi Level)


```
{
  "access_groups": [
    {
      "id": 0,
      "included_by_user_group": "string",
      "name": "string"
    }
  ],
  "email": "string",
  "expiry_datetime": "string",
  "login_id": "string",
  "name": "string",
  "password": "string",
  "permission": {
    "id": 0,
    "name": "string",
    "permissions": [
      {
        "allowed_group_id_list": [
          "long"
        ],
        "module": "string",
        "read": true,
        "write": true
      }
    ]
  },
  "phone_number": "string",
  "pin": "string",
  "security_level": "string",
  "start_datetime": "string",
  "status": "string",
  "user_group": {
    "id": 0,
    "name": "string"
  }
}
```

BioStar 2 Cloud API - Example

User [3/5]

- [POST] /users

<Major Parameters>

Name	Type	*M/O	Explanation	Value
user_id	string	M	ID	
access_groups	array	M	id, one of elements, is neccesary	
expiry_datetime	string	M	Expiry Datetime in UTC in ISO-8601 format	e.g.) 2015-06-10T02:14:05.268Z
permission	class	O	**It has the similar role compared to the parameter 'role' in V1 API**	
security_level	String	M	If this value is empty, this value set the default 0	'DEFAULT' = Device default 'LOWER' = 1/1000 'LOW' = 1/10000 'NORMAL' = 1/100000 'HIGH' = 1/1000000 'HIGHER' = 1/10000000
start_datetime	String	M	Start Datetime in UTC in ISO-8601 format	e.g.) 2015-06-10T02:14:05.268Z
status	String	M	Default value is 'AC'	'AC' = Active 'IN' = Inactive
user_group	class	M	Yon can find the existing user group by sending the GET message to user_groups through API	

* M - Mandatory, O - Optional

BioStar 2 Cloud API - Example

User [4/5]

- [POST] /users

NOTICE

- ✓ About the key "**permission**"
  - **id** is mandatory
  - Even if **id** is '**0**', you will get a success message, new user actually is **not** added in BioStar 2
- ✓ API for getting permission list
  - [GET] /setting/permission_list

BioStar 2 Cloud API - Example

User [5/5]

- [POST] /users
- JSON Structure (Multi Level)

<wrong example>

```
Dictionary<string, string> dicNewUser = new Dictionary<string, string>();
dicNewUser.Add("user_id", "1");
dicNewUser.Add("access_groups", "test");
dicNewUser.Add("expiry_datetime", "2019-06-10T02:14:05.268Z");
dicNewUser.Add("security_level", "1/10000000");
dicNewUser.Add("start_datetime", "2015-06-10T02:14:05.268Z");
dicNewUser.Add("login_id", "hailey");
dicNewUser.Add("password", "hailey12345");
dicNewUser.Add("status", "AC");
dicNewUser.Add("user_group",("1"+ "All Groups") );
dicNewUser.Add("name", "hailey");
string payload = seerializer.Serialize(dicNewUser);
```

```
static async void CreateUserTask()
{
    if (sessionID == null)
    {
        Console.WriteLine("You must log in first!");
        return;
    }

    CookieContainer cookieContainer = new CookieContainer();

    HttpClientHandler handler = new HttpClientHandler();
    handler.CookieContainer = cookieContainer;

    HttpClient httpClient = new HttpClient(handler);

    HttpClient client = new HttpClient(handler);
    cookieContainer.Add(new Uri("https://api.biostar2.com"), new Cookie("bs-cloud-session-id", sessionID));

    string resourceAddress = "https://api.biostar2.com/v2/users";

    Console.WriteLine("Input User ID: ");
    string userInputID = Console.ReadLine();

    JavaScriptSerializer serializer = new JavaScriptSerializer();

    Dictionary<string, object> dicNewUser = new Dictionary<string, object>();
    dicNewUser.Add("user_id", userInputID);

    List<object> access_groups_list = new List<object>();
    dicNewUser.Add("access_groups", access_groups_list);
    Dictionary<string, dynamic> dicAccessGroups = new Dictionary<string, dynamic>();
    access_groups_list.Add(dicAccessGroups);
    dicAccessGroups["id"] = 1;
    dicAccessGroups["included_by_user_group"] = "YES";
    dicAccessGroups["name"] = "dm";

    dicNewUser.Add("expiry_datetime", "2017-12-10T02:14:05.268Z");
    dicNewUser.Add("security_level", "DEFAULT");
    dicNewUser.Add("start_datetime", "2017-01-10T02:14:05.268Z");
    dicNewUser.Add("status", "AC");

    Dictionary<string, object> dicUserGroup = new Dictionary<string, object>();
    dicNewUser.Add("user_group", dicUserGroup);
    dicUserGroup.Add("id", 1);
    dicUserGroup.Add("name", "All Users");

    string payload = serializer.Serialize(dicNewUser);
```

Correct!

BioStar 2 Cloud API - Example

Monitoring [1/2]

- [POST] /monitoring/event_log/search
- [POST] /monitoring/event_log/search_more
- [POST] /monitoring/event_log/search_by_device (**Recommend**)

<Major Parameters>

Name		Type	*M/O	Explanation	Value
device_query_list		array	M		
	device_id	string	M	Device id	
	End_datetime	string	M	Expiry Datetime in UTC in ISO-8601 format	e.g.) 2015-06-10T02:14:05.268Z
	Start_datetime	string	M	Start Datetime in UTC in ISO-8601 format	e.g.) 2015-06-10T02:14:05.268Z
event_type_code_list		array	M	Refer to GET /references/event_types	
limit		long	M	Number of results	
offset		long	M	Results data offset	

* M - Mandatory, O - Optional

BioStar 2 Cloud API - Example

Monitoring [2/2]

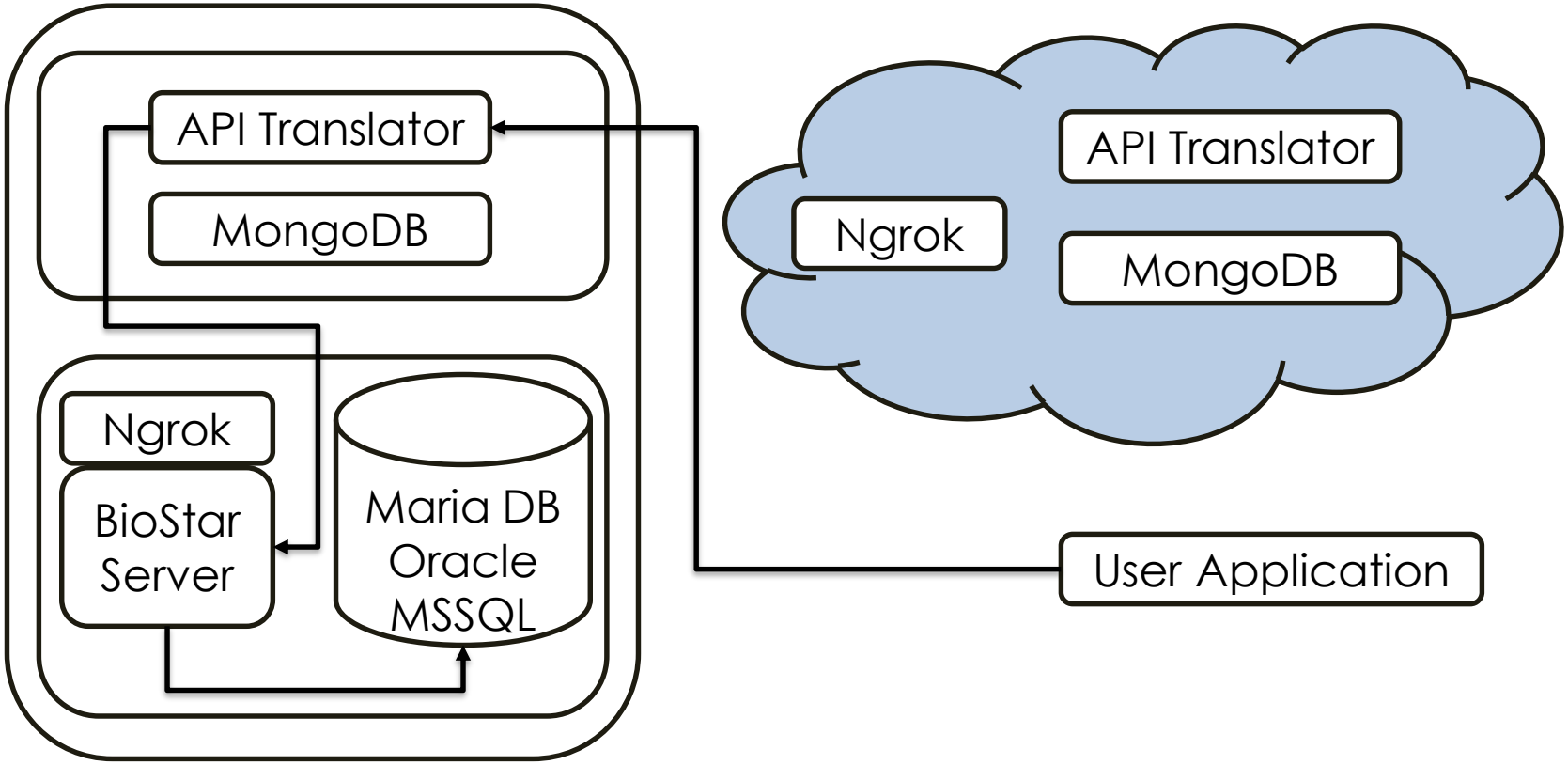
- [POST] /monitoring/event_log/search_by_device

```
static async void GetLogTask()
{
    if (sessionID == null)
    {
        Console.WriteLine("You must log in first!");
        return;
    }

    string resourceAddress = "https://api.biostar2.com/v2/monitoring/event_log/search_by_device";

    CookieContainer cookieContainer = new CookieContainer();

    HttpClientHandler handler = new HttpClientHandler();
    handler.CookieContainer = cookieContainer;

    HttpClient httpClient = new HttpClient(handler);

    JavaScriptSerializer serializer = new JavaScriptSerializer();

    List<string> event_type_code_list = new List<string>();
    Console.WriteLine("Input the number of event code you want to search logs");
    int eventCodeCNT = Convert.ToInt32(Console.ReadLine());

    for (int i = 0; i < eventCodeCNT; i++)
    {
        Console.WriteLine("Input Event Code("+ (i+1) + "): ");
        event_type_code_list.Add(Console.ReadLine());
    }
    Console.WriteLine(event_type_code_list);

    Console.WriteLine("Input limit: ");
    string limit = Console.ReadLine();
    Console.WriteLine("Input offset: ");
    string offset = Console.ReadLine();

    cookieContainer.Add(new Uri("https://api.biostar2.com"), new Cookie("bs-cloud-session-id", sessionID));

    Dictionary<string, object> dicGetLog = new Dictionary<string, object>();

    List<object> device_query_list = new List<object>();
    dicGetLog.Add("device_query_list", device_query_list);
    Dictionary<string, dynamic> dicDevice_query_list = new Dictionary<string, dynamic>();
    device_query_list.Add(dicDevice_query_list);
    dicDevice_query_list["device_id"] = "541531003";
```

BioStar 2 API Server

Local API [1/3]

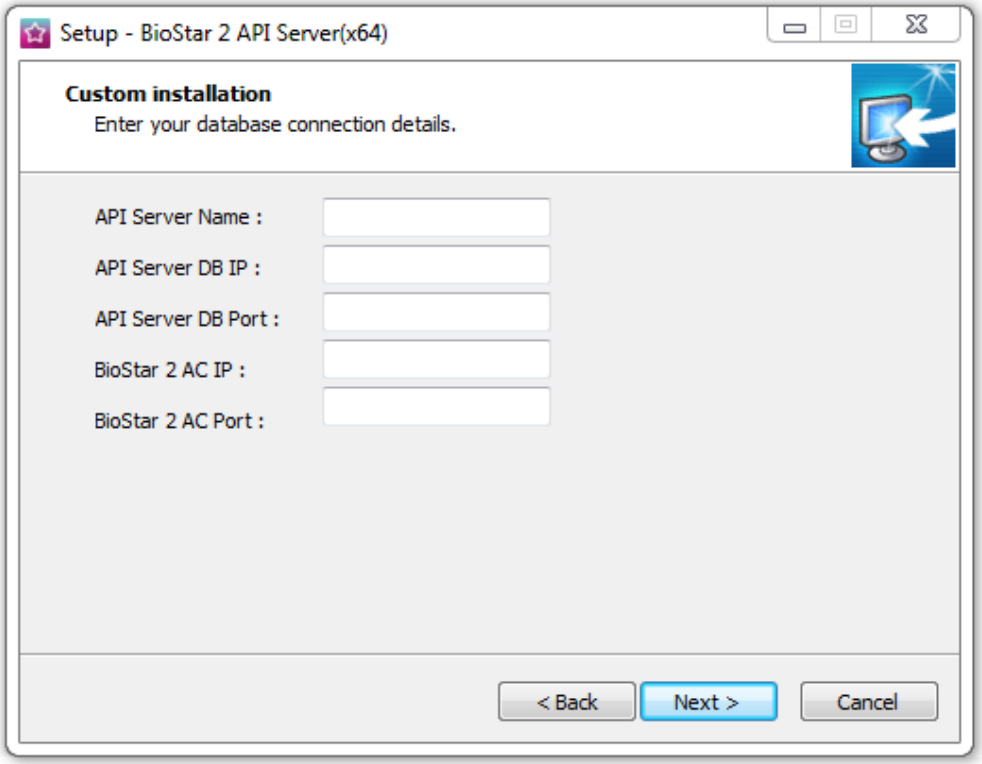
- BioStar 2 API Server is called as **Local API**



BioStar 2 API Server

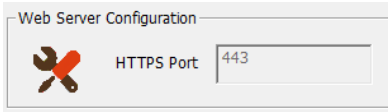
Local API [2/3]

- Install BioStar 2 API Server on a PC which BioStar 2 is already installed
- Download setup program (e.g. BioStar 2 API Server(x64).x.x.x.xxx.exe)
- Biostar 2 API is supported under a 64-bit Operating System



NOTE

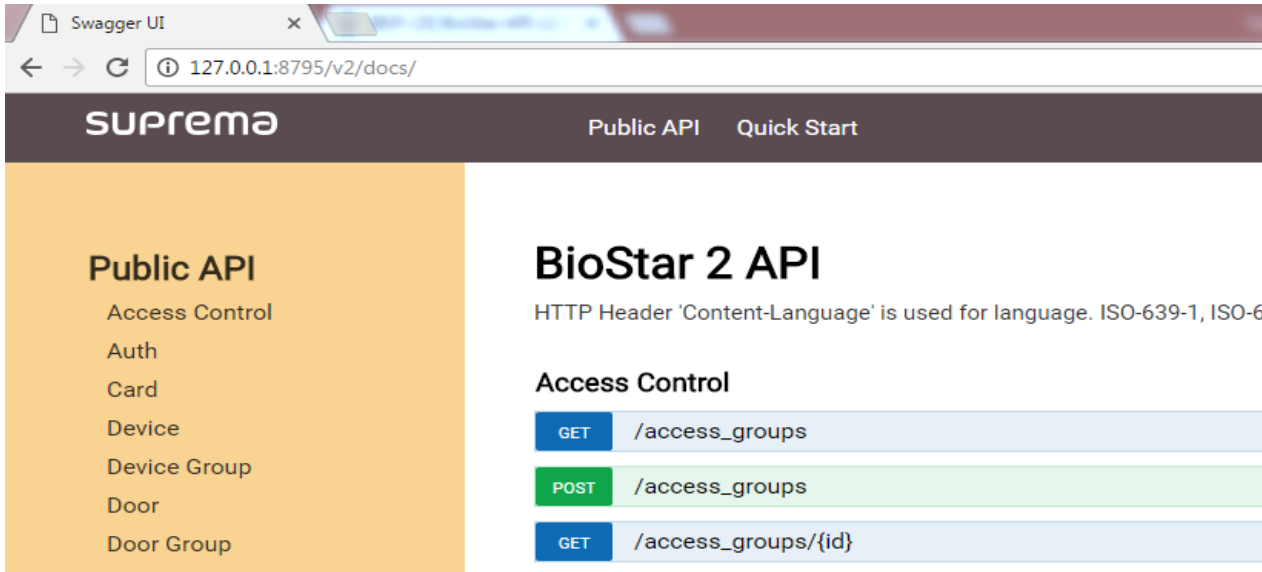
- ✓ API Server Name : **Subdomain Name**
- ✓ API Server DB IP : 127.0.0.1
- ✓ API Server DB Port : 27077 (Recommand)
- ✓ BioStar 2 AC IP : 127.0.0.1
- ✓ BioStar 2 AC Port : Refer to the Web Server Configuration of BioStar Setting



BioStar 2 API Server

Local API [3/3]

- We recommend you try to utilize POSTMAN
- Swagger UI
  - [BioStar v2.4 or above] http://*[BioStar 2 API Server IP]*:**8795**/**v2**/docs/
  - [BioStar v2.3 or below] http://*[BioStar 2 API Server IP]*:**8790**/**v1**/docs/
- Request URL
  - [BioStar v2.4 or above] http://*[BioStar 2 API Server IP]*:**8795**/**v2**/*[Enter API you want to call]*
  - [BioStar v2.3 or below] http://*[BioStar 2 API Server IP]*:**8790**/**v1**/*[Enter API you want to call]*



Introduction of BioStar 2 API v2.4.1

Add registration procedure for System Account

Available to login with admin account

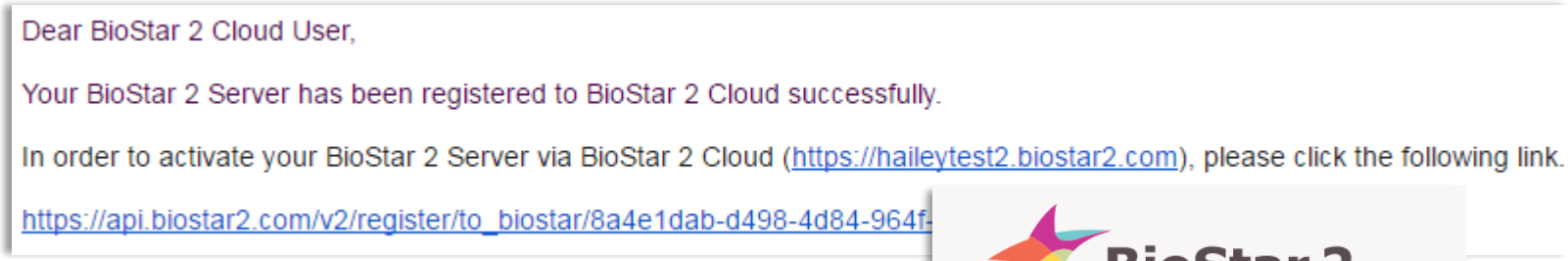
Add new API related to face

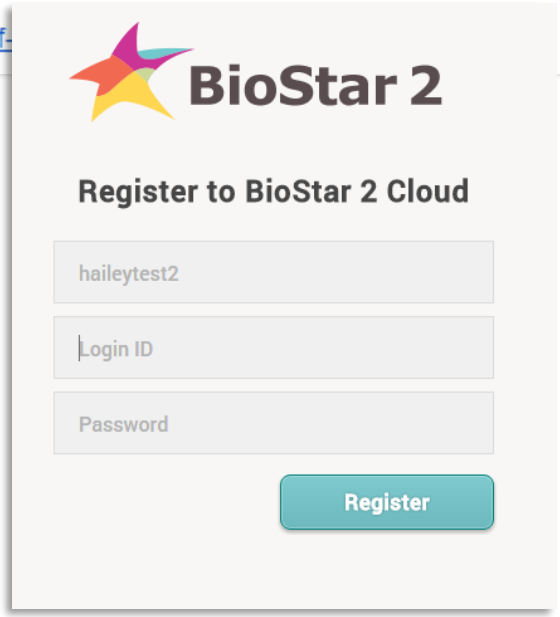
Major Features

Introduction of BioStar 2 API v2.4.1

Major Features [1/3]

- **Add registration procedure for System Account**
- Available to login with admin account
- Add new API related to face

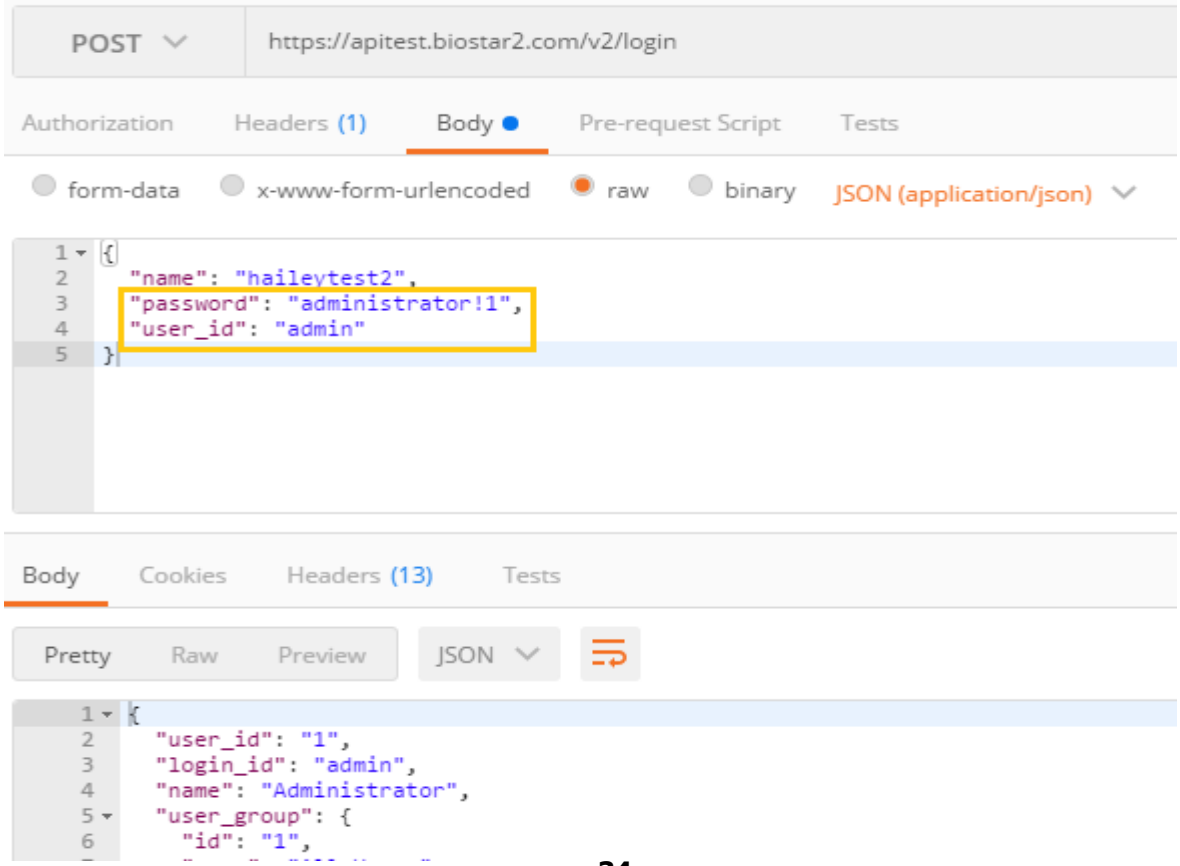




Introduction of BioStar 2 API v2.4.1

Major Features [2/3]

- Add registration procedure for System Account
- **Available to login with admin account**
- Add new API related to face



Introduction of BioStar 2 API v2.4.1

Major Features [3/3]

- Add registration procedure for System Account
- Available to login with admin account
- **Add new API related to face**

Category	Message Type	Sub URL
Device	POST	/devices/{id}/scan_face
User	POST	/users/update
	GET	/users/{user_id}/face_templates
	PUT	/users/{user_id}/face_templates

<Swagger UI>

Q & A

Any Questions?

Thank You

suprema