Interface Specification

<Server>

***Revision History***

|  |  |  |
| --- | --- | --- |
| **Date** | **Author** | **Description** |
| 4/30/2021-5/1/2021 | XinHU(Sean) | Made the first version of IS. |
| 5/6/2021 | XinHU(Sean) | Make the document complete. Revise UID in Cookie to SID in Access Interfaces. |
| 5/8/2021 | XinHu(Sean) | Communicate with DB group, define filters to search data about user profile/messages/doctors |
| 5/9/2021 | XinHu(Sean) | Checked some spelling errors |

1. **Introduction**

**This document describes all server interfaces in detail. These interfaces enable clients to implements functions which require server to process data. Interfaces are mainly divided into five parts including- User Profile APIs, Algorithm APIs, Treatment APIs, Message APIs and Backup APIs which will be introduced specifically later.**

**All the interfaces are designed in the form of HTTP requests.**

**This document incudes 4 parts – Introduction, Service Summary, Access Interfaces and Appendices.**

1. **Services Summary**

**As mentioned in Introduction, all interfaces are divided into five parts. This section briefly describes what these parts of interfaces provide.**

* 1. **User Profile APIs**

User Profile APIs provides basic functions for users to register/login/logout/manage their accounts.

* 1. **Algorithm APIs**

Algorithm APIs mainly receive images, which need to be detected, then send back automatic detection result through ML algorithm. Algorithm APIs also provides Algorithm management API to choose which algorithm to be used.

* 1. **Treatment APIs**

Treatment APIs mainly serve for user to find their upload history and provide records for clients to generate line chart. For admin, Treatment APIs allow administrators to search records of all users.

* 1. **Message APIs**

Message APIs mainly serve for patients to send messages to doctor and for doctors to reply to patients’ message.

* 1. **Backup APIs**

Backup APIs are designed for administrators to back up all data in the system including user profiles, upload records and messages.

1. **Access Interfaces**

**3.0 Before You Read Specific Definition of Interfaces**

**Notice that all of our APIs provided to clients are in the form of HTTP requests. In order to use our services, you must at least specify these fields- HTTP Method, Path, Cookie and Body- before you send requests to the server. More details are listed in following diagrams.**

* 1. **User Profile APIs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SID** | **SN** | **Method** | **Path** | **Cookie** | **Body** | **RetVal** |
| **1.1** | **Register an Account** | **POST** | **/users/register** | **None** | **RegisterForm** | **UserProfile** |
| **1.2** | **User Login** | **POST** | **/users/login** | **None** | **LoginForm** | **UserProfile** |
| **1.3** | **Change Password** | **POST** | **/users/cpass** | **SI** | **CpassForm** | **UserProfile** |
| **1.4.1** | **(AD)Search User** | **GET** | **/users/search** | **SI** | **UserSearchForm** | **UserProfilesList** |
| **1.4.2** | **(AD)** **Modify User Profile** | **POST** | **/users/modify** | **SI** | **UserModifyForm** | **UserProfile** |
| **1.4.3** | **(AD)delete specific users** | **POST** | **/users/delete** | **SI** | **DeleteUserList** | **UserProfilesList** |

**\* ID = Service ID, SN = Service Name, RetVal = Return Value, SI=Session Identifier**

**\* SN with (AD): This function is provided to administrator.**

**\* Contents in Cookie, Body and RetValue are defined in Appendices.**

* 1. **Algorithm APIs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SID** | **SN** | **Method** | **Path** | **Cookie** | **Body** | **RetVal** |
| **2.1** | **Upload Image** | **POST** | **/images/upload** | **SI** | **RawImage** | **DetectResult** |
| **2.2** | **(AD)** **Manage Algorithms** | **POST** | **/algorithms/manage** | **SI** | **AlgorithmSelector** | **NULL** |

**\* ID = Service ID, SN = Service Name, RetVal = Return Value, SI= Session Identifier**

**\* SN with (AD): This function is provided to administrator.**

**\* Contents in Cookie, Body and RetValue are defined in Appendices.**

* 1. **Treatment APIs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SID** | **SN** | **Method** | **Path** | **Cookie** | **Body** | **RetVal** |
| **3.1** | **Find Upload Records** | **GET** | **/images/find** | **SI** | **ImageSearchForm** | **ImagesList** |
| **3.2** | **Revise Treatment** | **POST** | **/images/modify** | **SI** | **RevisedResult** | **Image** |
| **3.3.1** | **(AD)Find Upload Records** | **POST** | **/images/manage** | **SI** | **ImageSearchForm** | **ImagesList** |
| **3.3.2** | **(AD)Modify Records** | **POST** | **/images/modify** | **SI** | **RevisedResult** | **Image** |
| **3.3.3** | **(AD)Delete records** | **POST** | **/images/delete** | **SI** | **ImagesList** | **ImagesList** |

**\* ID = Service ID, SN = Service Name, RetVal = Return Value, SI = Session Identifier**

**\* SN with (AD): This function is provided to administrator.**

**\* Contents in Cookie, Body and RetValue are defined in Appendices.**

* 1. **Message APIs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SID** | **SN** | **Method** | **Path** | **Cookie** | **Body** | **RetVal** |
| **4.1** | **Search Doctors** | **GET** | **/doctors/find** | **SI** | **DoctorSearchForm** | **DoctorList** |
| **4.2** | **Send Message** | **POST** | **/messages/send** | **SI** | **Message** | **NULL** |
| **4.3** | **Find Messages** | **GET** | **/messages/find** | **SI** | **MessageSearchForm** | **MessagesList** |
| **4.4.1** | **(AD) Find Messages** | **GET** | **/messages/find** | **SI** | **MessageSearchForm** | **MessagesList** |
| **4.4.2** | **(AD) Delete Messages** | **POST** | **/Messages/delete** | **SI** | **MessagesList** | **MessagesList** |

**\* ID = Service ID, SN = Service Name, RetVal = Return Value, SI = Session Identifier**

**\* SN with (AD): This function is provided to administrator.**

**\* Contents in Cookie, Body and RetValue are defined in Appendices.**

* 1. **Backup APIs**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **SID** | **SN** | **Method** | **Path** | **Cookie** | **Body** | **RetVal** |
| **5.1** | **(AD)Create Backup** | **POST** | **/backup/create** | **SI** | **NULL** | **Backup** |
| **5.2** | **(AD)Find Backups** | **GET** | **/backup/find** | **SI** | **NULL** | **BackupsList** |
| **5.3** | **(AD)Restore Backups** | **POST** | **/backup/restore** | **SI** | **Backup** | **Backup** |

**\* ID = Service ID, SN = Service Name, RetVal = Return Value, SI = Session Identifier**

**\* SN with (AD): This function is provided to administrator.**

**\* Contents in Cookie, Body and RetValue are defined in Appendices.**

**APPENDICES**

**A.1 Data Structures**

Data Structures in Access Interfaces can be divided to two formats: json (used in POST) and web form (used in GET). More details will be given later.

**A.1.2 Overall**

This section introduces how request HTTP package and response HTTP package like.

**Request HTTP Package-**

**+-----------Header(POST)-----------+**

**POST PATH HTTP； Version**

**Some key-values that won’t be used in our structure.**

**Cookie: cookie about SID**

**+------------Body------------+**

**<Json object of type specified in**

**“Access Interfaces”>**

**+-----------Header(GET)-----------+**

**GET PATH HTTP； Version**

**Some key-values that won’t be used in our structure.**

**Cookie: cookie about SID**

**+------------Body------------+**

**Data in web form**

**Response HTTP Package-**

**+-----------Header-----------+**

**Some key-values that won’t be used in our structure.**

**Cookie information**

**+------------Body------------+**

**{**

**“result”: true/false,**

**“info”: “String of extra information”,**

**“retval”: < Json object of type specified**

**in “Access Interfaces”>**

**}**

***Annotation:***

***“Result” is a Boolean value. IF the server successfully processed the request this value will be set to true with data in RetVal. IF there are some errors happened in server “Result” will be set to false with Error Description in “info” and null in RetValue.***

***Traditional HTTP return code in this structure represents connectivity between the server and the client. It does not represent logical errors in our system and clients may deal with this code by themselves.***

**A.1.3** **UserProfile**

**User profile is used in HTTP APIs to represent all the basic information of an account including user\_id, name, birthdate, phone number, email address, credential, register date, password, account\_type.**

**User profile in json –**

|  |  |  |
| --- | --- | --- |
| **key** | **type** | **description** |
| user\_id | **String** | **User’s identifier, generated by server** |
| name | **String** | **User’s real name** |
| birthdate | **String** | **Use String like "yyyy-mm-dd" to represent date** |
| phone\_number | **String** | **User’s telephone number** |
| email\_address | **String** | **User’s email address** |
| credential | **String** | **User’s credential, it can be Identity Card ID or passport ID.** |
| register\_date | **String** | **Date that this account being registered** |
| account\_type | **String** | **"Doctor" or "Patient" or "Admin"** |

**An example of User Profile-**

{

"user\_id": "20210501675157",

"name": "John",

"birthdate ": "2001-05-01",

"phone\_number ": "+86 18657115478",

"email\_address ": "example@hotmail.com",

"credential": "330825200505260267",

"register\_date": "2021-04-23"

"account\_type": "patient"

}

**A.1.4 RegisterForm**

**RegisterForm contains basic information which is required to register an account including name, password, birthdate, phone\_number or email address and credential.**

**RegisterForm in json –**

|  |  |  |
| --- | --- | --- |
| name | **String** | **User’s real name** |
| password | **String** | **Password Encrypted by SHA256** |
| birthdate | **String** | **Use String like "yyyy-mm-dd" to represent date** |
| phone\_number | **String** | **User’s telephone number** |
| email\_address | **String** | **User’s email address** |
| credential | **String** | **User’s credential, it can be Identity Card ID or passport ID.** |
| account\_type | **String** | **"Doctor" or "Patient" or "Admin"** |

**Example of this structure is similar to User Profile.**

**A.1.4 LoginForm**

**Login Form contains information required in login including user\_name(phone number or email address used to register account), password.**

|  |  |  |
| --- | --- | --- |
| user\_name | **String** | **Phone number or email address** |
| password | **String** | **Password Encrypted by SHA256** |

**A.1.5 CpassForm**

**CpassForm(Change Password Form) contains information required in changing password, including user\_id, old\_password, new\_password.**

**CpassForm in json-**

|  |  |  |
| --- | --- | --- |
| user\_id | **String** | **User’s identifier generated by server** |
| old\_password | **String** | **Old Password Encrypted by SHA256** |
| new\_password | **String** | **New Password Encrypted by SHA256** |

**An example of CpassForm -**

{

"user\_id": "20210501675157",

"old\_password": "167677e1479e32ed1ff758b224735b3c2730681754181f9a9cf848e1989d9c62",

"new\_password ": "87cbfa4e880cde2b3a39dc8563c9959fe0b1cbbdc8f976cbdcf86774bea64987 "

}

**A.1.6 UserSearchForm**

**UserSearchForm is a form in web style.**

**UserSearchForm contains parameters to search users. It is a key-values form in which keys are domains in User Profile and values are requirement description Strings.**

***Annotation:***

***Format of requirement String in the form depends on domain’s date type. See A.2 about it.***

**An example of UserSearchForm –**

name=John&phone\_number=+86\* (to search users whose name is John and phone\_number in China)

**A.1.7 UserProfilesList**

**UserProfilesList contains a list of User Profiles.**

**UserProfilesList in json-**

|  |  |  |
| --- | --- | --- |
| users | **List of User Profile** | **A list of User Profile** |

**An example of UserProfilesList–**

{

"users":

[

{

"user\_id": "20210501675157",

"name": "John",

"birthdate ": "2001-05-01",

"phone\_number ": "+86 18657115478",

"email\_address ": "example@hotmail.com",

"credential": "330825200505260267",

"register\_date": "2021-04-23"

"account\_type": "patient"

},

{

"user\_id": "20210501685377",

"name": "Jerry",

"birthdate": "2001-02-18",

"phone\_number": "+86 17357115478",

"email\_address": "example@hotmail.com",

"credential": "330825200504360254",

"register\_date": "2021-04-22"

"account\_type": "patient"

}

]

}

**A.1.8 UserModifyForm**

**UserModifyForm contains renewed data of a specific user.**

**UserModifyForm in json-**

|  |  |  |
| --- | --- | --- |
| user\_id | **String** | **User’s Identifier** |
| renewed\_profile | **User Profile** | **Renewed user profile** |
| new\_password | **String** | **null(not to update) or New Password Encrypted by SHA256** |

**An example of UserModifyForm–**

{

"renewed\_profile":

{

"user\_id": "20210501675157",

"name": "John",

"birthdate ": "2001-05-01",

"phone\_number ": "+86 18657115478",

"email\_address ": "example@hotmail.com",

"credential": "330825200505260267",

"register\_date": "2021-04-23"

"account\_type": "patient"

},

"new\_password": null

}

**A.1.9 DeleteUserList**

**DeleteUserList contains a list of user\_ids which represent users that will be deleted from database.**

|  |  |  |
| --- | --- | --- |
| user\_ids | **List of String** | **A list of user\_id** |

**An example of UserModifyForm–**

{

“user\_ids”:

[

"20210501675157",

"20210501675158",

"20210501675159",

"20210501675170"

]

}

**A.1.10 RawImage**

**RawImage contains user\_id and image data encrypted by BASE64.**

|  |  |  |
| --- | --- | --- |
| user\_id | **String** | **user\_id of who own this image** |
| data | **String** | **Encrypted binary image data by BASE64** |

**An example of UserModifyForm–**

{

"user\_id": "20210501675157",

"data": "YXNmc2R6dnp0ZWd2Zg==",

}

**A.1.11 DetectResult**

**DetectResult contains image\_id and detect result generated from algorithm.**

|  |  |  |
| --- | --- | --- |
| image\_id | **String** | **ID of target image** |
| result | **String** | **Result from algorithm group** |

**A.1.12 AlgorithmSelector**

**This structure contains id of algorithm that will be selected.**

|  |  |  |
| --- | --- | --- |
| algorithm\_id | **String** | **ID of target algorithm** |

**A.1.13 Image**

**Image contains all the information about an image(being detected or being revised by Doctor) including record\_id, user\_id, url, cobb\_result algorithm\_result, detect\_date, checked\_by\_doctor, doctor\_id, doctor\_result.**

**Image in json-**

|  |  |  |
| --- | --- | --- |
| record\_id | **String** | **Image’s identifier** |
| user\_id | **String** | **User’s Identifier. Who uploaded this image** |
| url | **String** | **Image’s url where user can get image data** |
| cobb\_result | **String** | **Detect result from algorithm** |
| detect\_date | **String** | **Date the image being detected or uploaded** |
| checked\_by\_doctor | **Boolean** | **Whether this image has been checked by doctor** |
| doctor\_id | **String** | **Doctor who has checked** |
| doctor\_result | **String** | **Result provided by doctor** |

**A.1.14 ImageSearchForm**

**ImageSearchForm is a form in web style.**

**ImageSearchForm contains parameters to search images. It is a key-values form in which keys are domains in Image and values are requirement description Strings.**

***Annotation:***

***Format of requirement String in the form depends on domain’s date type. See A.2 about it.***

**A.1.15 ImagesList**

**ImagesList is a list of Images in json form.**

|  |  |  |
| --- | --- | --- |
| images | **List of Images** | **A list of Images** |

**An example of ImagesList –**

{

"images":

[

<json object of Image>,

<json object of Image>,

…

]

}

**A.1.16 RevisedResult**

**RevisedResult contains image\_id of an image and revised result of the image together with doctor id.**

|  |  |  |
| --- | --- | --- |
| image\_id | **String** | **ID of target image** |
| doctor\_id | **String** | **Doctor\_id who revised the result** |
| cobb\_result | **String** | **null or renewed result** |
| doctor\_result | **String** | **Result/Advice from doctor** |

**A.1.17 Doctor**

**Doctor contains basic information about a doctor.**

|  |  |  |
| --- | --- | --- |
| doctor\_id | **String** | **same to user\_id** |
| name | **String** | **Doctor’s name** |
| doctor\_profile | **String** | **set it to null-not used** |

**A.1.18 DoctorList**

**DoctorList is a list of Doctors.**

|  |  |  |
| --- | --- | --- |
| doctors | **List of Doctor** | **A list of doctors** |

**A.1.19 Message**

**Message is a message record between user and doctor. It contains message\_id, sender\_id, receiver\_id, consult\_record\_id, time, unread.**

|  |  |  |
| --- | --- | --- |
| message\_id | **String** | **A message’s identifier** |
| sender\_id | **String** | **Sender’s user\_id** |
| receiver\_id | **String** | **Receiver’s user\_id** |
| content | **String** | **Content in this message** |
| time | **String** | **Use String like "yyyy-mm-dd 00:00:00 " to represent date** |
| unread | **Boolean** | **Whether this message been read** |

**A.1.20 MessagesList**

**MessagesList is a list of Messages**

|  |  |  |
| --- | --- | --- |
| messages | **List of Message** | **A list of messages, may not be in order** |

**A.1.21 MessageSearchForm**

**MessageSearchForm is a form in web style.**

**MessageSearchForm contains parameters to search messages. It is a key-values form in which keys are domains in Message and values are requirement description Strings.**

***Annotation:***

***Format of requirement String in the form depends on domain’s date type. See A.2 about it.***

**A.1.22 Backup**

**Backup contains basic information about a specific backup.**

|  |  |  |
| --- | --- | --- |
| backup\_id | **String** | **Identifier of a backup** |
| create\_date | **String** | **Use String like "yyyy-mm-dd" to represent date** |
| size | **String** | **Disk size of this backup** |
| description | **String** | **Description admin can add when creating the backup** |

**A.1.23 BackupsList**

**BackupsList is a list of Backups**

|  |  |  |
| --- | --- | --- |
| backups | **List of Backups** | **A list of backups** |

**A.2 Search Parameters**

**This section describes details about search parameters in three search form – UserSearchForm, ImageSearchForm, MessageSearchForm.**

**A.2.1 UserSearchForm**

**In this form server and DB only provide function to search user profiles by specific user\_id and account\_type.**

**For example:**

**If clients provide "user\_id=100243455&name=john&account\_type=doctor" in its GET request, the server will ignore name=john in request and only find the one whose user\_id equals to 100243455 and account type is doctor.**

**A.2.2 ImageSearchForm**

**In this form server and DB only provide function to search image records by specific user\_id and timestart, timend.**

**For example:**

**If clients provide "user\_id=100243455&record\_id=asvcx", the server will only find such records whose user\_id equals to 100243455.**

**A.2.3 MessageSearchForm**

**In this form server and DB provide function to search messages by sender\_id, receiver\_id, timestart, timend and unread to search required messages.**

**For example:**

**If clients provide "sender\_id=10024345&receiver\_id=46457231&timestart=2021-05-08&timend=2021-05-09&unread=true", the server will provide messages whose sender id is 10024345, receiver\_id is 46457231, time is later than timestart and earlier than timend and unread is true.**