IRAS API SERVICES INTERFACE SPECIFICATIONS

Auto-Inclusion Scheme (AIS) For Employment Income - “Submission of Employment Income Records” Plan

Last updated on : 10 Aug 2018
Version No: 1.5
## Table of Contents

Table of Contents ............................................................................................................................ 2  
1. Introduction ................................................................................................................................. 3  
2. Create an Account at API Portal .................................................................................................. 3  
3. API Services ............................................................................................................................... 4  
   3.1 “Submission of Employment Income Records” Plan............................................................. 4  
   3.1.1 JSON Request for AIS for Employment Income API Service ........................................ 7  
   3.1.2 JSON Responses for AIS for Employment Income API Service ................................ 11  
      3.1.2.1 Success Response ...................................................................................................... 13  
      3.1.2.2 Error Response: Content Validation ....................................................................... 15  
      3.1.2.3 Error Response: Header or Trailer ......................................................................... 17  
      3.1.2.4 Error Response: Exceed Max Records ................................................................... 18  
      3.1.2.5 Error Response: Server Error ................................................................................ 19  
      3.1.2.6 Error Response: Not Found .................................................................................. 19  
4. Status Codes for API Response .................................................................................................... 20  
5. Sample Code (C#) ....................................................................................................................... 21
1. Introduction

The Inland Revenue Authority of Singapore (IRAS) provides application programming interface (API) services to allow application developers to submit and retrieve tax related matters using HTTP requests. Most of the APIs will be in the form of a RESTFUL web service which reduces client/server coupling and thus enabling easier integration between IRAS’s service with external developers.

There will be a variety of services available in due time, while some services require a simple GET, others may be secured and require credentials that can be passed via HTTP header parameters, as follows:

<table>
<thead>
<tr>
<th>X-IBM-Client-Id</th>
<th>String containing the client ID of the application invoking IRAS’ API. This value will be provided to the application vendor by IRAS.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E.g. a1234b5c-1234-abcd-efgh-a1234b5cdef</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>X-IBM-Client-Secret</th>
<th>String containing the client secret of the application invoking IRAS’ API. This value will be provided to the application vendor by IRAS.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>E.g. a12345bC67e8fG9a12345bC67e8fG9a12345bC67e8fG9a12345bC67e8fG9a12345bC67e8fG9</td>
</tr>
</tbody>
</table>

This document serves to help developers consume the API services provided by IRAS.

2. Create an Account at API Portal

Application developers are required to create a developer’s account at https://apisandbox.iras.gov.sg/ to subscribe to IRAS API services for Sandbox Testing, and a developer’s account at https://apiservices.iras.gov.sg/ to subscribe to IRAS API services for Production.

Note: In order for IRAS to identify your API subscriptions, please follow the steps below to create an account:

**Step 1**
- Create an account with your personal particulars. A computer-generated email will be sent to you for account activation.

**Step 2**
- After account activation, click on your Username at the top-right corner of the page > My organization > Edit organization.
- In the “Organization name” field, key in your organisation’s details by adhering to the following name convention:
  “Name of Organisation_Tax Reference No. of Organisation”
  E.g. ABC PTE LTD_200312345A
3. API Services

The following sections describe the request and response for the API service.

<table>
<thead>
<tr>
<th>Name of API Service</th>
<th>Description</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Submission of Employment Income Records</td>
<td>This API allows payroll software developers to incorporate within their payroll software the feature for employers to validate and submit employment income information (IR8A, IR8S, Appendix 8A and Appendix 8B) to IRAS directly.</td>
<td>For Sandbox Testing: <a href="https://apisandbox.iras.gov.sg/iras/sb/AISubmission/submit">https://apisandbox.iras.gov.sg/iras/sb/AISubmission/submit</a> For Actual Submission (Production): <a href="https://apiservices.iras.gov.sg/iras/prod/AISubmission/submit">https://apiservices.iras.gov.sg/iras/prod/AISubmission/submit</a> This service is available only when the AIS submission portal is open. AIS Employers can refer to IRAS' website for the submission period: <a href="https://www.iras.gov.sg/irashome/Businesses/Employers/Auto-Inclusion-Scheme--AIS--for-Employment-Income/#title6">https://www.iras.gov.sg/irashome/Businesses/Employers/Auto-Inclusion-Scheme--AIS--for-Employment-Income/#title6</a></td>
</tr>
</tbody>
</table>

3.1 “Submission of Employment Income Records” Plan

This service allows AIS employers to validate and submit their employees’ employment income information (IR8A, IR8S, Appendix 8A and Appendix 8B). Each submission allows up to 800 records in total (regardless of form type) and cannot be more than 8 MB in total data size.

Examples of Acceptable Submissions (total size < 8 MB)
- 800 IR8A records
- 500 IR8A records + 100 IR8S records + 50 Appendix 8A records + 50 Appendix 8B records
- 100 IR8A records + 30 IR8S records + 10 Appendix 8A records

Examples of Rejected Submissions
- 700 IR8A records and total size > 8 MB
- 801 IR8A records
- 201 IR8A records + 200 IR8S records + 200 Appendix 8A records + 200 Appendix 8B records
For submissions of more than 800 records or 8 MB in total data size, your application should ‘chunk’ it into 2 submissions.

Example of IR8A submission for 1,000 employees
- 1st submission – IR8A for 800 employees (total size < 8 MB)
- 2nd submission – IR8A for 200 employees (total size < 8 MB)

In the event that a ‘chunked’ submission fails, error details will be returned in the response message, and the failed submission is not accepted by IRAS.

Example:
- 1st submission – IR8A for 800 employees (Passed)
- 2nd submission – IR8A for 200 employees (Failed)

Application user will have to resubmit the 2nd submission data chunk.

**Submission Guidelines**

- If an employee has excess CPF contribution, Benefits-in-Kind and/or Stock Options Gains, submit all the relevant form types (IR8A/ IR8S/ Appendix 8A/ Appendix 8B) of this employee within the same request.

- Ensure the amounts reflected in the IR8A and the supporting forms (IR8S/ Appendix 8A/ Appendix 8B) tally.

- Do NOT submit duplicate data.

- Original data (i.e. Batch Indicator = O) must be submitted before submitting Amendment data.

- When submitting Amendment data, only prepare the affected employees’ records and provide the difference in amounts. Leave other numeric fields not affected by the error blank. If the amendment for the supporting forms (IR8S/ Appendix 8A/ Appendix 8B) affects the amounts submitted for Form IR8A, an IR8A amendment data has to be submitted within the same request as well.

**Sandbox Testing**

The "Submission of Employment Income Records (Sandbox)" plan in the sandbox environment is designed to mimic the actual Production environment for developers to test the API integration before submitting the actual data to the Production environment. IRAS' approval is not required for subscription to this plan.

To perform the sandbox testing, developers will require the testing entity assigned by IRAS, their client ID and client secret, and the API sandbox URL (refer to section 3). Developers can refer to IRAS' website for details on the testing process: [https://www.iras.gov.sg/irashome/Businesses/Employers/Auto-Inclusion-Scheme--AIS-/Vendors-Supporting-AIS-Employers--Submission/#title5](https://www.iras.gov.sg/irashome/Businesses/Employers/Auto-Inclusion-Scheme--AIS-/Vendors-Supporting-AIS-Employers--Submission/#title5).

Developers who have passed the sandbox testing can proceed to subscribe to the “AISubmission” plan in the API Portal for Production.
**Production**
The “Submission of Employment Income Records” plan in Production requires approval by IRAS and is subjected to successful completion of sandbox testing. The subscription approval turnaround time is 10 days.

After approval is granted by IRAS, client application can POST a JSON request object to the production URL (refer to section 3).

HTTP Header parameters

<table>
<thead>
<tr>
<th>X-IBM-Client-Id</th>
<th>String containing the client ID of the application invoking IRAS’ API. This value will be provided to the application vendor by IRAS.</th>
</tr>
</thead>
<tbody>
<tr>
<td>X-IBM-Client-Secret</td>
<td>String containing the client secret of the application invoking IRAS’ API. This value will be provided to the application vendor by IRAS.</td>
</tr>
</tbody>
</table>
3.1.1 JSON Request for AIS for Employment Income API Service

Client invoking this service will be expected to submit the following JSON request object with the following fields in the HTTP request. Note that the field names are case sensitive.

The JSON request need to be modified with escape strings and serialised in order to be acceptable by API web service. Refer to Section 5 for sample code.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>validateOnly</td>
<td>Boolean Indicator to indicate whether to validate JSON message only. If validateOnly = true, the API will perform validation of the employment income information without submission. If validateOnly = false, the API will perform validation of the employment income information and submission to IRAS.</td>
</tr>
<tr>
<td>bypass</td>
<td>Boolean Indicator to indicate whether to bypass warning message thrown and proceed with submission. If bypass = true, hit warning and proceed with submission. If bypass = false, respond as error with no submission.</td>
</tr>
<tr>
<td>userID</td>
<td>String containing the ID Number of the person doing the submission. ID Number must be valid.</td>
</tr>
<tr>
<td>userIDType</td>
<td>String containing the type of User ID Number. Acceptable type are “1” is for NRIC, “2” is for FIN, “4” is for WP, “A” is for ASGD and “11” is for MIC</td>
</tr>
<tr>
<td>ir8aInput</td>
<td>String containing the content of the IR8A This will be the same as the one that is generated for IRAS’ Validation and Submission Application.</td>
</tr>
<tr>
<td>ir8sInput</td>
<td>String containing the content of the IR8S This will be the same as the one that is generated for IRAS’ Validation and Submission Application.</td>
</tr>
<tr>
<td>a8aInput</td>
<td>String containing the content of the Appendix 8A This will be the same as the one that is generated for IRAS’ Validation and Submission Application.</td>
</tr>
<tr>
<td>a8bInput</td>
<td>String containing the content of the Appendix 8B This will be the same as the one that is generated for IRAS’ Validation and Submission Application.</td>
</tr>
<tr>
<td>inputType</td>
<td>String to indicate if the content is a XML or TEXT format. Expected value includes “XML” and “TEXT”.</td>
</tr>
<tr>
<td>clientID</td>
<td>String containing the ID of the application invoking IRAS’ API. This value will be provided to the application vendor by IRAS. This value will be the same as the Client-Id in the HTTP header parameter.</td>
</tr>
</tbody>
</table>

^ Refer to the current file format published at https://www.iras.gov.sg/IRASHome/Businesses/Employers/Auto-Inclusion-Scheme--AIS-/Technical-File-Format/Specifications/
# At least one of the form inputs must be filled. All inputs must be of the same input type (TEXT / XML).

JSON request object schema

```json
{
    "properties": {
        "validateOnly": {
            "type": "boolean",
            "description": "Validate Only Indicator"
        },
        "bypass": {
            "type": "boolean",
            "description": "Bypass Indicator"
        },
        "userID": {
            "type": "string",
            "description": "User ID"
        },
        "userIDType": {
            "type": "string",
            "description": "User ID Type"
        },
        "ir8aInput": {
            "type": "string",
            "description": "ir8a XML or TEXT string"
        },
        "ir8sInput": {
            "type": "string",
            "description": "ir8s XML or TEXT string"
        },
        "a8aInput": {
            "type": "string",
            "description": "a8a XML or TEXT string"
        },
        "a8bInput": {
            "type": "string",
            "description": "a8b XML or TEXT string"
        },
        "inputType": {
            "type": "string",
            "description": "XML or TEXT indicator"
        },
        "clientID": {
            "type": "string",
            "description": "Client ID"
        }
    }
}
```
Sample JSON request object

```json
{
    "validateOnly": false,
    "bypass": false,
    "userID": "S11111111K",
    "userIDType": "1",
    "a8aInput": null,
    "ir8sInput": null,
    "userIDType": "1",
    "bypass": false,
    "validateOnly": false
}
```
```xml
<taxablevaluehotelaccommodation>2500.00</taxablevaluehotelaccommodation>
<costofleavepassageandincidentalbenefits/>
<noofleavepassageself/>
<noofleavepassagespouse/>
<noofleavepassagechildren/>
<ohqstatus/>
<interestpaidbyemployer/>
<lifeinsurancepremiumspaidbyemployer/>
<freedorsubsidisedholiday/>
<educationalexpenses/>
<nonmonetaryawardsforlongservice/>
<entranceortransferfeesto socialclubs/>
<gainsfromassets/>
<fullcostofmotorvehicle/>
<carbenefit/>
<othersbenefits/>
<totalbenefitsinkind>9347.25</totalbenefitsinkind>
<filler/>
<fieldreserved/>
</a8a2015st>
</esubmissionsdsc>
</a8arecord>
</details>
</a8a2015>
```
### Sample JSON request object

```
{
    "validateOnly": false,
    "bypass": false,
    "userID": "S1111111K",
    "userIDType": "1",
    "ir8aInput": null,
    "ir8sInput": "+",
    "a8aInput": "0620158200312345A TOMMY LEE                     FINANCE MANAGER
ABC PTE LTD
O20150101                              A8A
\n\n11S1234567A   SAMPLE 1
\n\nNOVENA
201503012015052909002000493150F000024657500000000000397250020000000005397250
00025000000000000000001200000000145000000025000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000
\n",
    "a8bInput": "+",
    "inputType": "TEXT",
    "clientID": "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxx"
}
```

### 3.1.2 JSON Responses for AIS for Employment Income API Service

For API services that process or persist data, a JSON response object will be returned to the client after service is invoked in the HTTP response. Clients invoking these API services will be required to consume this JSON response object to retrieve the status and output. The schema of the response object varies across services and will be covered in detail in each specific service in the later sections.

Depending on the request to this service, the following different responses can be expected from the service based on the input provided to it from the request.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusCode</td>
<td>A string representing the status of the service output returned to the client. The default value is 200 (OK). For a listing of valid status codes, see Status Codes section below.</td>
</tr>
<tr>
<td>msgError</td>
<td>A string containing general error message.</td>
</tr>
<tr>
<td>ir8a</td>
<td>An array containing the output, warning, error related to the ir8a form submitted.</td>
</tr>
<tr>
<td>ir8s</td>
<td>An array containing the output, warning, error related to the ir8s form submitted.</td>
</tr>
</tbody>
</table>
### a8a
An array containing the output, warning, error related to the a8a form submitted.

### a8b
An array containing the output, warning, error related to the a8b form submitted.

### Detailed description of the output, warning and error are listed the table below.

<table>
<thead>
<tr>
<th><strong>output</strong></th>
<th>An array containing pipe delimited string/s containing the Submission Reference Number and the processed information of the submission, as shown in sample below. Client is expected to store the Submission Reference Number for future reference.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>The output is delimited in the following order:</td>
</tr>
<tr>
<td></td>
<td>• Submission Reference Number</td>
</tr>
<tr>
<td></td>
<td>• Organisation Name</td>
</tr>
<tr>
<td></td>
<td>• Organisation Reference Number</td>
</tr>
<tr>
<td></td>
<td>• Basis Year + 1</td>
</tr>
<tr>
<td></td>
<td>• File Type - “O” for original submission, “A” for amendment</td>
</tr>
<tr>
<td></td>
<td>• Date and time of submission in ddMMyyHHmm format</td>
</tr>
<tr>
<td></td>
<td>• User ID</td>
</tr>
<tr>
<td></td>
<td>• Number of Records</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>error</strong></th>
<th>An array list of error messages. Each array item will contain a JSON error object with the following fields:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• RecordType – a string containing the type of record that caused the error, record type will be specific to each service.</td>
</tr>
<tr>
<td></td>
<td>• RecordField – a string containing the field that caused the error.</td>
</tr>
<tr>
<td></td>
<td>• RecordIdentifier – a string containing the ID of the record or row that caused the error</td>
</tr>
<tr>
<td></td>
<td>• error – a string containing the error message or number.</td>
</tr>
<tr>
<td></td>
<td>(0 = Header,1 = Detail, 2 = Trailer)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>warning</strong></th>
<th>An array list of warning messages. Each array item will contain a JSON error object with the following fields:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• RecordType – a string containing the type of record that caused the error, record type will be specific to each service.</td>
</tr>
<tr>
<td></td>
<td>• RecordField – a string containing the field that caused the error.</td>
</tr>
<tr>
<td></td>
<td>• RecordIdentifier – a string containing the ID of the record or row that caused the error</td>
</tr>
<tr>
<td></td>
<td>• error – a string containing the warning message or number.</td>
</tr>
<tr>
<td></td>
<td>(0 = Header,1 = Detail, 2 = Trailer)</td>
</tr>
</tbody>
</table>
3.1.2.1 Success Response

A successful call to the service means that the AIS submission is accepted. A JSON response object that contains the following fields will be returned in the HTTP response.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusCode</td>
<td>A string containing the value “200”, representing STATUS_OK.</td>
</tr>
<tr>
<td>msgError</td>
<td>Empty</td>
</tr>
<tr>
<td>ir8a</td>
<td>An array containing the output, warning, error related to the ir8a form submitted.</td>
</tr>
<tr>
<td>ir8s</td>
<td>An array containing the output, warning, error related to the ir8s form submitted.</td>
</tr>
<tr>
<td>a8a</td>
<td>An array containing the output, warning, error related to the a8a form submitted.</td>
</tr>
<tr>
<td>a8b</td>
<td>An array containing the output, warning, error related to the a8b form submitted.</td>
</tr>
</tbody>
</table>

Sample JSON response object

```json
{
    "statusCode": "200",
    "msgError": "",
    "ir8a": {
        "output": "ES15080216IR8A1234567890|SAMPLE PTE.
LTD.|20312345A|2016|O|0120161600|S1111111K|30",
        "warnings": [],
        "errors": []
    },
    "ir8s": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8b": {
        "output": "",
        "warnings": [],
        "errors": []
    }
}
```
OR

If bypass=true, warning message will be shown and submission is allowed.

```
{   "statusCode": "200",   "msgError": "",   "ir8a": {     "output": "15080216IR8A1234567890|SAMPLE PTE. LTD.|200312345A|2016|O|0102161600|S1111111K|30",     "warnings": [{       "recordType": 1,       "recordField": " IDNo",       "recordIdentifier": "S1234567A",       "error": "ir8s is required"     }],     "errors": [   ]   },   "ir8s": {     "output": "",     "warnings": [   ],     "errors": [   ]   },   "a8a": {     "output": "",     "warnings": [   ],     "errors": [   ]   },   "a8b": {     "output": "",     "warnings": [   ],     "errors": [   ]   } }
```

For successful production submission, an email will also be triggered to both the company contact representative as well as the individual that submit the salary file through payroll software.

The format of the email is shown below:
3.1.2.2 Error Response: Content Validation

When the records in the content has failed validations, the call to the service is unsuccessful. A JSON response object that contains the following fields will be returned in the HTTP response.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusCode</td>
<td>A string containing the value “400”, representing STATUS_BAD_REQUEST.</td>
</tr>
<tr>
<td>msgError</td>
<td>String which may contain a simplified error message or an error number.</td>
</tr>
<tr>
<td>ir8a</td>
<td>An array containing the output, warning, error related to the ir8a form submitted.</td>
</tr>
<tr>
<td>ir8s</td>
<td>An array containing the output, warning, error related to the ir8s form submitted.</td>
</tr>
<tr>
<td>a8a</td>
<td>An array containing the output, warning, error related to the a8a form submitted.</td>
</tr>
<tr>
<td>a8b</td>
<td>An array containing the output, warning, error related to the a8b form submitted.</td>
</tr>
</tbody>
</table>
Sample JSON response object

```json
{
    "statusCode": "400",
    "msgError": "",
    "ir8a": {
        "output": "",
        "warnings": [],
        "errors":{
            "recordType": "2",
            "recordField": "BasisYear",
            "recordIdentifier": "200312345A",
            "error": "Invalid"
        }
    },
    "ir8s": {
        "output": "",
        "warnings": [],
        "errors":[]
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors":[]
    },
    "a8b": {
        "output": "",
        "warnings": [],
        "errors":[]
    }
}
```

If bypass=false, warning message will be shown and submission will not be allowed.

```json
{
    "statusCode": "400",
    "msgError": "",
    "ir8a": {
        "output": "",
        "warnings": {
            "recordType": 1,
            "recordField": "IDNo",
            "recordIdentifier": "S1234567A",
            "error": "ir8s is required"
        },
        "errors":[]
    },
    "ir8s": {
        "output": "",
        "warnings": [],
        "errors":[]
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors":[]
    }
}
```
3.1.2.3 Error Response: Header or Trailer

When the Header or Trailer in the content has failed validations, the call to the service is unsuccessful. A JSON response object that contains the following fields will be returned in the HTTP response.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>statusCode</td>
<td>A string containing the value “400”, representing STATUS_BAD_REQUEST.</td>
</tr>
<tr>
<td>msgError</td>
<td>String which may contain a simplified error message or an error number.</td>
</tr>
<tr>
<td>ir8a</td>
<td>An array containing the output, warning, error related to the ir8a form submitted.</td>
</tr>
<tr>
<td>ir8s</td>
<td>An array containing the output, warning, error related to the ir8s form submitted.</td>
</tr>
<tr>
<td>a8a</td>
<td>An array containing the output, warning, error related to the a8a form submitted.</td>
</tr>
<tr>
<td>a8b</td>
<td>An array containing the output, warning, error related to the a8b form submitted.</td>
</tr>
</tbody>
</table>

Sample JSON response object

```json
{
    "statusCode": "400",
    "msgError": "",
    "ir8a": {
        "output": "",
        "warnings": [],
        "errors": [
            {
                "recordType": "2",
                "recordField": " BasisYear ",
                "recordIdentifier": "200312345A",
                "error": " Invalid"
            }
        ]
    },
    "ir8s": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8b": {
        "output": "",
        "warnings": [],
        "errors": []
    }
}```
3.1.2.4 Error Response: Exceed Max Records

When the number of records submitted is more than 800, the call to the service is unsuccessful. A JSON response object that contains the following fields will be returned in the HTTP response. There will be only one error record in an error array, as shown in sample below.

<table>
<thead>
<tr>
<th><strong>statusCode</strong></th>
<th>A string containing the value “413”, representing STATUS_REQUEST_TOO_LARGE.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>msgError</strong></td>
<td>String which may contain a simplified error message or an error number.</td>
</tr>
<tr>
<td><strong>ir8a</strong></td>
<td>An array containing the output, warning, error related to the ir8a form submitted.</td>
</tr>
<tr>
<td><strong>ir8s</strong></td>
<td>An array containing the output, warning, error related to the ir8s form submitted.</td>
</tr>
<tr>
<td><strong>a8a</strong></td>
<td>An array containing the output, warning, error related to the a8a form submitted.</td>
</tr>
<tr>
<td><strong>a8b</strong></td>
<td>An array containing the output, warning, error related to the a8b form submitted.</td>
</tr>
</tbody>
</table>

Sample JSON response object

```json
{
    "statusCode": "413",
    "msgError": "NoOfRecords Exceed 800",
    "ir8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "ir8s": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8b": {
        "output": "",
        "warnings": [],
        "errors": []
    }
}
```
3.1.2.5 Error Response: Server Error

When an exception occurs during the processing of the request, a JSON response object that contains the following fields will be returned in the HTTP response.

<table>
<thead>
<tr>
<th>statusCode</th>
<th>A string containing the value “500”, representing STATUS_SERVER_ERROR.</th>
</tr>
</thead>
<tbody>
<tr>
<td>msgError</td>
<td>String which may contain a simplified error message or an error number.</td>
</tr>
</tbody>
</table>

Sample JSON response object

```
{
    "statusCode": "500",
    "msgError": "Internal Server Error Occurred. Please try again later.",
    "ir8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "ir8b": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8a": {
        "output": "",
        "warnings": [],
        "errors": []
    },
    "a8b": {
        "output": "",
        "warnings": [],
        "errors": []
    }
}
```

3.1.2.6 Error Response: Not Found

When the API has been unpublished or turned off, a JSON response object that contains the following fields will be returned in the HTTP response.

<table>
<thead>
<tr>
<th>httpCode</th>
<th>A string containing the value “404”, representing STATUS_SERVER_ERROR.</th>
</tr>
</thead>
<tbody>
<tr>
<td>httpMessage</td>
<td>String which may contain a simplified error message or an error number.</td>
</tr>
<tr>
<td>moreInformation</td>
<td>String which may contain more details about the error</td>
</tr>
</tbody>
</table>

Sample JSON response object

```
{
    "httpCode": "404",
    "httpMessage": "Not Found",
    "moreInformation": "The requested URL was not found on this server"
}
```
4. Status Codes for API Response

The `statusCode` field will always contain an integer representing the processed state of the request. The list of possible status codes and what they represent are listed below.

<table>
<thead>
<tr>
<th>Status</th>
<th>Status Code</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>STATUS_OK</td>
<td>200</td>
<td>The request completed successfully.</td>
</tr>
<tr>
<td>STATUS_NO_CONTENT</td>
<td>204</td>
<td>The server has fulfilled the request, but there is no new information to send back.</td>
</tr>
<tr>
<td>STATUS_MOVED</td>
<td>301</td>
<td>The requested service has been assigned to a new permanent Uniform Resource Identifier (URI), and any future references to this service should be done using one of the returned URIs.</td>
</tr>
<tr>
<td>STATUS_REDIREC</td>
<td>302</td>
<td>The requested service resides temporarily under a different URI.</td>
</tr>
<tr>
<td>STATUS_NOT_MODIFIED</td>
<td>304</td>
<td>The requested did not make any modification.</td>
</tr>
<tr>
<td>STATUS_BAD_REQUEST</td>
<td>400</td>
<td>The request could not be processed by the server due to invalid inputs.</td>
</tr>
<tr>
<td>STATUS_DENIED</td>
<td>401</td>
<td>The requested service requires user authentication/authorisation.</td>
</tr>
<tr>
<td>STATUS_GONE</td>
<td>410</td>
<td>The requested service is no longer available at the server, and no forwarding address is known.</td>
</tr>
<tr>
<td>STATUS_REQUEST_TOO_LARGE</td>
<td>413</td>
<td>The server cannot process the request because the submitted entity is larger than the server is able to process.</td>
</tr>
<tr>
<td>STATUS_SERVER_ERROR</td>
<td>500</td>
<td>The server encountered an unexpected condition that prevented it from fulfilling the request.</td>
</tr>
</tbody>
</table>
5. Sample Code (C#)

Escape strings and serialize the data before calling API

```csharp
// Step 1 Read IR8A file
string input = File.ReadAllText(file);

// Step 2 Escape string
StringBuilder escapeString = new StringBuilder(input);
escapeString.Replace("\", "\\");
escapeString.Replace("\r", "\r");
escapeString.Replace("\n", "\n");
escapeString.Replace("\t", "\t");
input = escapeString.ToString();

// Step 3 Create submission object
ApiSubmission inputObj = new ApiSubmission()
{
    ClientID = txtClientID.Text,
    UserID = txtBoxUserId.Text,
    UserIDType = txtBoxUserIdType.Text,
    InputType = "XML",
    Bypass = true,
    ValidateOnly = true,
    IR8AInput = input
};

// Step 4 Serialize the object using JavaScriptSerializer
var serializer = new JavaScriptSerializer();
StringBuilder serializedData = new StringBuilder();
```
serializer.Serialize(inputObj, serializedData);

//Step 5 Send serialized data - function from specs
callWebAPI(serializedData);

Calling the Web API
using System;
using System.Net;
using System.IO;
using System.Text;

// jsonData – contains data from Section 3.1.1 of this document
public static void callWebAPI(string jsonData)
{
    //Step 0 : Call ServerCertificateValidationCallback
    ServicePointManager.ServerCertificateValidationCallback +=
        (sender, cert, charIn, sslPolicyErrors) =>
            true;

    // Step 1: Construct URL
    String url = "https://apisandbox.iras.gov.sg/iras/sb/AISubmission/submit";

    try
    {
        var httpWebRequest = (HttpWebRequest)WebRequest.Create(url);
        httpWebRequest.ContentType = "application/json;";
        httpWebRequest.Method = "POST";

        //Step 2: Enter the Client-Id given by IRAS
        httpWebRequest.Headers["X-IBM-Client-Id"] = "a4434a0b-1514-4dc6-b4f7-db5316bf647";

        //Step 3: Enter the Client-Secret given by IRAS
        httpWebRequest.Headers["X-IBM-Client-Secret"] =
            "qO3eB0gC2eF5cV7tR1oY6tG5wL8nX6cR5rN0tH1hF3pF6dB6wL";

        // Step 4: Call API using POST
        using (var streamWriter = new StreamWriter(httpWebRequest.GetRequestStream()))
        {
            streamWriter.Write(jsonData);
            streamWriter.Flush();
            streamWriter.Close();
        }

        // Step 4a: Output response
        var httpResponse = (HttpWebResponse)httpWebRequest.GetResponse();
        using (var streamReader = new StreamReader(httpResponse.GetResponseStream()))
        {
            var result = streamReader.ReadToEnd();
            //print the received response
            Console.WriteLine(result);
        }
    }
    catch
    {
        // handle exceptions
    }
}
catch (WebException e)
{
    if (!string.IsNullOrEmpty(e.Message))
    {
        // Step 4b: Print general errors
        Console.WriteLine("Exception - ");
        Console.WriteLine(e.Message);
    }

    if (e.Response != null)
    {
        // Step 4c: Print Output response exception
        Stream receiveStream = e.Response.GetResponseStream();
        StreamReader readStream = new StreamReader(receiveStream, Encoding.UTF8);
        // print the error received from Server
        Console.WriteLine("Response error received - ");
        Console.WriteLine(readStream.ReadToEnd());
    }
}

The information provided is intended for better general understanding and is not intended to comprehensively address all possible issues that may arise. The contents are correct as at 10 Aug 2018 and are provided on an “as is” basis without warranties of any kind. IRAS shall not be liable for any damages, expenses, costs or loss of any kind however caused as a result of, or in connection with your use of this document.

While every effort has been made to ensure that the above information is consistent with existing policies and practice, should there be any changes, IRAS reserves the right to vary our position accordingly.