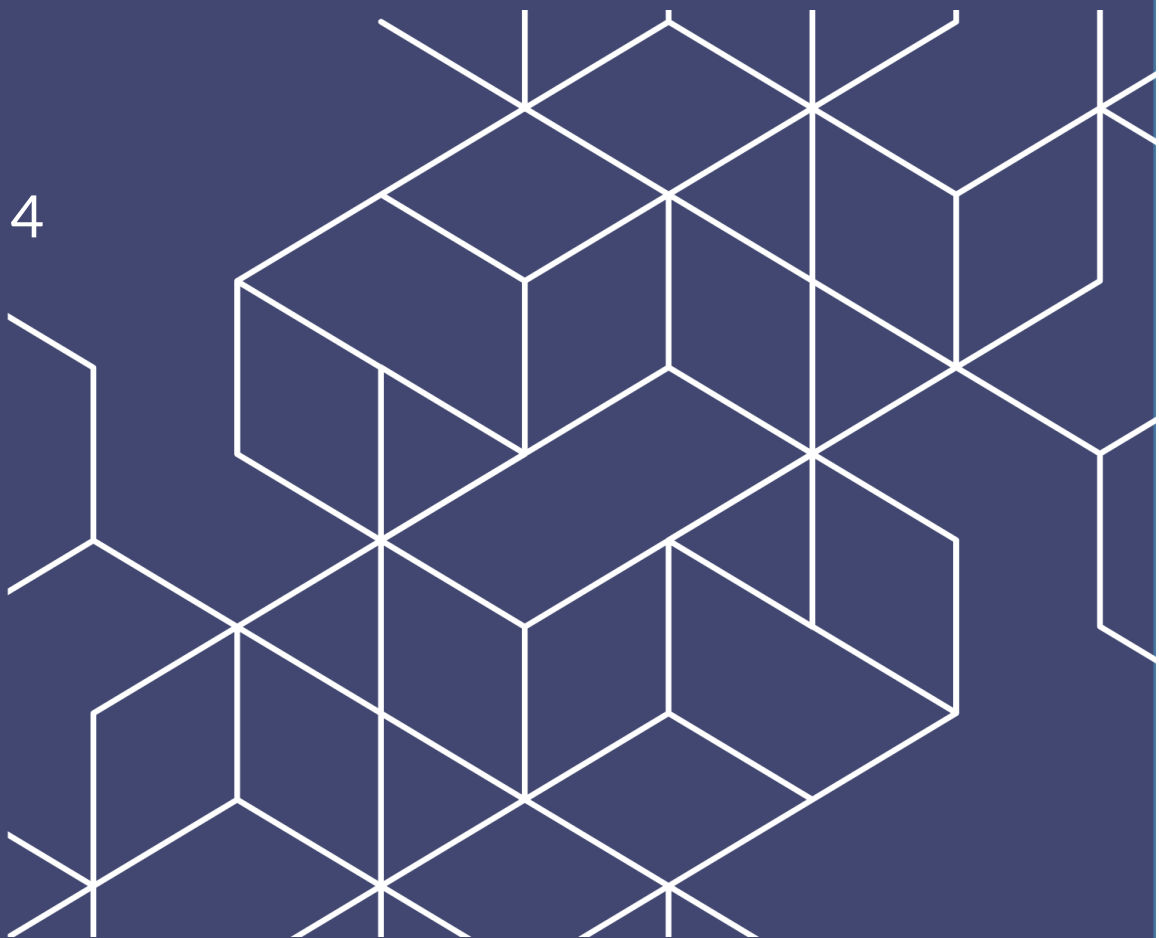
eCASE

OPEXUS

REST Services API

v11.3.0

May 2024



eCASE 11.3.0 REST Services API

Notice of Rights

Notice of Liability

The information in this publication is believed to be accurate and reliable. However, the information is distributed by the publisher (AINS, LLC.) on an “As Is” basis without warranty for its use, or for any infringements of patents or other rights of third parties resulting from its use.

While every precaution has been taken in the preparation of this publication, neither the author (or authors) nor the publisher will have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused, directly or indirectly, by the information contained in this publication or by the computer software and hardware products described in it.

Notice of Trademarks

The publisher’s company name, company logo, company patents, and company proprietary products are trademarks or registered trademarks of the publisher: AINS, LLC. All other trademarks or registered trademarks are the property of their respective owners.

Non-Disclosure Statement

This document’s contents are confidential and proprietary to AINS, LLC. This document cannot be released publicly or outside the purchasing agency without prior written permission from AINS, LLC.

Images in this manual are used as examples and may contain data and versioning that may not be consistent with your version of the application or information in your environment.

Additional Notice

Information in this documentation is subject to change without notice and does not represent a commitment on the part of AINS, LLC.

Notwithstanding any of the foregoing, if this document was produced as a Deliverable or other work for hire under a contract on behalf of a U.S. Government end user, the terms and conditions of that contract shall apply in the event of a conflict.

Contents

1 Introduction

eCASE REST services Application Programming Interface (API) allow developers to integrate enterprise applications or mobile applications with the eCASE platform using simple HTTP methods. eCASE REST services user JSON format payloads, making it an ideal integration service for many mobile or web application development frameworks. eCASE REST services use the case data model configured for the case type in eCASE when working with cases. REST Services are used for creating, searching, updating, and deleting case folders. These services also provide capability to query system data, such as the User Logins Report and Audit Trail actions, so that enterprise monitoring applications can monitor the system events.

Case Information-related REST Services are secured over HTTP/S with user authentication, similarly to eCASE Application authentication. The user is authenticated in the context of the eCASE application with login credentials and is allowed to perform only authorized actions on the case folders in the application. All actions are audited, just like the eCASE application.

System Information-related REST services are secured over HTTP/S, with authentication mode supported by the web server and enterprise monitoring system.

1.1 eCASE REST Resources and URIs

A REST resource is a case folder in eCASE. Each case folder in REST services is identified by a named Uniform Resource Identifier (URI) and is accessed using standard HTTP methods (HEAD, GET, POST, DELETE). REST Services are based on the configured case types and their URIs.

For example, eCASE REST services can:

- Retrieve case type data model information about the case type.
- Get Case Folder information.
- Get a Contact information.
- Perform a query or search on Case folder or Contact resources.
- Update or delete a Case Folder or Contact.
- Get System information like user login report or audit trail actions.

REST URIs for each eCASE instance depends on configured case types. The REST URIs are formatted as shown below:

Type	Format
Case URI	https://<eCaseServerDNSName>/eCaseOData
Case Folder Resource URI	https://<eCaseServerDNSName>/eCaseOData/CaseFolder
Contact Resource URI	https://<eCaseServerDNSName>/eCaseOData/Contact
System Data Resource URI	https://<eCaseServerDNSName>/eCaseOData/systemodata/<methodname>?parameters

1.2 Authentication

All REST Services calls that perform any operations on Case Folder or Contact resources are required to be authenticated. All REST operations are performed in a valid application user context. Access and authorization control for users is required to be configured in eCASE platform by the application administrators. As a first step, you need to get a security access token that can be used for all REST Service calls on Case Folder or Contact Resources. The lifespan of the security token is controlled as per the security settings configured in eCASE.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{BaseURI}}/Authenticate ",
  "method": "POST",
  "headers": {
    "content-type": "application/x-www-form-urlencoded",
    "accept": "application/json"
  },
  "data": {
    "applicationName": "invoice",
```

```
    "loginId": "jsmith",
    "password": "jsmithpassword"
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
"ab387ce8-a37c-48d3-9f46-473defe235cf"
```

The result will be security token which is used for all REST service calls until it expires. This security token is posted as “sectoken” field in all HTTP POST requests.

All REST service calls that fetch system data should be secured through web service authentication and SSL/TLS certificates.

2 Case Folder Services

2.1 GET Case Model

The security token obtained from the *Authenticate* operation is used in all the case folder and contact operations. The GET case model REST Services method is used to get the case folder model details.

The following is jQuery example for getting case folder by folder number, where CaseTypeName are Case Type in eCASE. In this example, we assign the following values:

CaseTypeName: EmployeeTraining

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url":
"{{baseURL}}/CaseFolder/Metadata/<CaseTypeInternalName>",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "content-type": "application/json",
    "sectoken": "<sectoken>"
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
 {
```

```
"CaseTypeInternalName": "leaverequest",
"CoreFields": [
  {
    "InternalName": "action_verb",
    "FieldDataType": 1
  },
  {
    "InternalName": "action_reference_id",
    "FieldDataType": 1
  },
  {
    "InternalName": "action_reference_type",
    "FieldDataType": 1
  },
  {
    "InternalName": "action_reference_name",
    "FieldDataType": 1
  },
  {
    "InternalName": "db_record_id",
    "FieldDataType": 3
  },
  {
    "InternalName": "db_record_version",
    "FieldDataType": 1
  },
  {
    "InternalName": "ACTIVITY_ID",
    "FieldDataType": 3
  },
  {
    "InternalName": "WORKITEM_ID",
    "FieldDataType": 3
```


```
},
{
  "InternalName": "WORKITEM_STATUS_ID",
  "FieldDataType": 12
},
{
  "InternalName": "ACTIVITY_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "ACTIVITY_DUE_DATE",
  "FieldDataType": 5
},
{
  "InternalName": "ASSIGNED_TYPE",
  "FieldDataType": 12
},
{
  "InternalName": "ASSIGNED_ID",
  "FieldDataType": 12
},
{
  "InternalName": "ASSIGNEE_OFFICE_ID",
  "FieldDataType": 3
},
{
  "InternalName": "ASSIGNED_DATE",
  "FieldDataType": 5
},
{
  "InternalName": "COMPLETED_DATE",
  "FieldDataType": 5
},
```

```
{
  "InternalName": "COMPLETED_BY",
  "FieldDataType": 12
},
{
  "InternalName": "ASSIGNED_BY",
  "FieldDataType": 12
},
{
  "InternalName": "REFERENCE_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "REFERENCE_TYPE",
  "FieldDataType": 1
},
{
  "InternalName": "case_monitored_by_user",
  "FieldDataType": 3
},
{
  "InternalName": "infocopy_for_user",
  "FieldDataType": 3
},
{
  "InternalName": "linked_folder_number",
  "FieldDataType": 1
},
{
  "InternalName": "sub_folder_number",
  "FieldDataType": 1
},
{
```

```
  "InternalName": "on_behalf_of",
  "FieldDataType": 14
},
{
  "InternalName": "db_record_id",
  "FieldDataType": 3
},
{
  "InternalName": "folder_id",
  "FieldDataType": 12
},
{
  "InternalName": "workflow",
  "FieldDataType": 12
},
{
  "InternalName": "folder_owner",
  "FieldDataType": 12
},
{
  "InternalName": "folder_created_by",
  "FieldDataType": 12
},
{
  "InternalName": "folder_modified_by",
  "FieldDataType": 12
},
{
  "InternalName": "folder_closed_by",
  "FieldDataType": 12
},
{
  "InternalName": "folder_requester",
```

```
  "FieldDataType": 12
},
{
  "InternalName": "contact",
  "FieldDataType": 12
},
{
  "InternalName": "action_office",
  "FieldDataType": 12
},
{
  "InternalName": "responsible_office",
  "FieldDataType": 12
},
{
  "InternalName": "folder_status",
  "FieldDataType": 1
},
{
  "InternalName": "subject",
  "FieldDataType": 1
},
{
  "InternalName": "workflow_start_date",
  "FieldDataType": 5
},
{
  "InternalName": "created_date",
  "FieldDataType": 4
},
{
  "InternalName": "response_due_date",
  "FieldDataType": 5
```


```
},
{
  "InternalName": "received_date",
  "FieldDataType": 4
},
{
  "InternalName": "priority_code",
  "FieldDataType": 12
},
{
  "InternalName": "secondary_due_date",
  "FieldDataType": 5
},
{
  "InternalName": "all_closed",
  "FieldDataType": 6
},
{
  "InternalName": "all_open",
  "FieldDataType": 6
},
{
  "InternalName": "parent_folder_id",
  "FieldDataType": 12
},
{
  "InternalName": "closed_date",
  "FieldDataType": 4
},
{
  "InternalName": "case_type",
  "FieldDataType": 12
},
```


```
    {
      "InternalName": "application_type",
      "FieldDataType": 12
    }
  ],
  "CustomFields": [
    {
      "InternalName": "employee_name",
      "FieldDataType": 1
    },
    {
      "InternalName": "employee_id",
      "FieldDataType": 1
    },
    {
      "InternalName": "date_of_birth",
      "FieldDataType": 4
    },
    {
      "InternalName": "gender",
      "FieldDataType": 12
    },
    {
      "InternalName": "street_address",
      "FieldDataType": 1
    },
    {
      "InternalName": "city",
      "FieldDataType": 1
    },
    {
      "InternalName": "state",
      "FieldDataType": 12
```

```
    },
    {
      "InternalName": "certification_date",
      "FieldDataType": 4
    },
    {
      "InternalName": "certified_by",
      "FieldDataType": 1
    },
    {
      "InternalName": "certificate_type",
      "FieldDataType": 12
    }
  ],
  "CustomRecords": [
    {
      "InternalName": "Certifications",
      "RecordFields": [
        {
          "InternalName": "certification_date",
          "FieldDataType": 4
        },
        {
          "InternalName": "certified_by",
          "FieldDataType": 1
        },
        {
          "InternalName": "certificate_type",
          "FieldDataType": 12
        }
      ]
    }
  ]
```

```
}
```

2.2 GET Case Folder

The GET case folder REST services method is used to get the case folder details for a specified case folder. The Folder ID is used to retrieve the case folder resource.

The following is a jQuery example for getting case folder by folder number. In this example, we assign the following values:

FolderNumber: 2017-1234-00091

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
     "url": "{{BaseURI}}/CaseFolder/<FolderNumber>",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "content-type": "application/json",
    "sectoken": "<sectoken>"
  }
}
```

Result:

```
[
  {
    "Core": {
      "Id": 91,
      "FolderId": "2017-1234-00091",
      "Created": "2016-10-27T14:56:13.783",
      "Modified": "2016-10-27T14:56:13.84"
```

```
},
"Data": {
  "Fields": [
    {
      "InternalName": "employee_name",
      "Value": "John"
    },
    {
      "InternalName": "employee_id",
      "Value": "12354"
    },
    {
      "InternalName": "date_of_birth",
      "Value": "2001-06-06T00:00:00.0000000"
    },
    {
      "InternalName": "gender",
      "Value": "Male"
    },
    {
      "InternalName": "street_address",
      "Value": "123 Main Street"
    },
    {
      "InternalName": "city",
      "Value": "Fairfax"
    },
    {
      "InternalName": "state",
      "Value": "Virginia"
    }
  ],
  "RecordInfo": {
```


```
"Certifications": [
  {
    "RowNumber": 1,
    "RowId": 93,
    "Fields": [
      {
        "InternalName": "certification_date",
        "Value": "2015-11-02T00:00:00.0000000"
      },
      {
        "InternalName": "certified_by",
        "Value": "Microsoft"
      },
      {
        "InternalName": "certificate_type",
        "Value": "Expert"
      }
    ]
  },
  {
    "RowNumber": 2,
    "RowId": 94,
    "Fields": [
      {
        "InternalName": "certification_date",
        "Value": "2016-10-11T00:00:00.0000000"
      },
      {
        "InternalName": "certified_by",
        "Value": "JQuery"
      },
      {
        "InternalName": "certificate_type",
```


```
                "Value": "Professional"
              }
            ]
          }
        ]
      }
    }
  }
]
```

2.3 Search Folders:

The following is the jQuery example for searching case folders using search filters employee_name=John where “employee_name” is a field name defined “EmployeeTraining” case type and “John” is the name of the employee. Comma should be used if more than one filters are used.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
     "url": "{{baseURL}}/CaseFolder",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "sectoken": "<sectoken>",
    "content-type": "application/json"
  },
"data": {
"employee_name": "John",
"employee_jobtype": "Manager"
  }
}
```

```
$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
[
  {
      "Id": 91,
      "FolderId": "2017-1234-00091",
      "Created": "2016-10-27T14:56:13.783",
      "Modified": "2016-10-27T14:56:13.84"
  },
 {
      "Id": 95,
      "FolderId": "2017-1234-00095",
      "Created": "2016-10-27T14:59:44.283",
      "Modified": "2016-10-27T14:59:44.333"
    }
]
```

2.4 Create Case Folder

The following is a jQuery example for creating new case folder, where case folder data is in JSON format with key value pair of field name and field value. POST the JSON data for creating case folder. The following is sample JSON data being assigned to data variable.

```
data =
{"Fields":[{"InternalName":"employee_name","Value":"John"},{"Inte
rnalName":"employee_id","Value":"12354"},{"InternalName":"date_of
_birth","Value":"2001-06-
06T00:00:00.0000000"},{"InternalName":"gender","Value":"Male"},{"
InternalName":"street_address","Value":"Test User
Uri"},{"InternalName":"city","Value":"TestCity"},{"InternalName":
"state","Value":"California"}],"RecordInfo":{"Certifications":[{"
RowNumber":1,"RowId":0,"Fields":[{"InternalName":"certification_d
```

```
ate","Value":"2015-11-
02T00:00:00.0000000"},{"InternalName":"certified_by","Value":"Mic
rosoft"},{"InternalName":"certificate_type","Value":"Expert"}]},{
"RowNumber":2,"RowId":0,"Fields":[{"InternalName":"certification_
date","Value":"2016-10-
11T00:00:00.0000000"},{"InternalName":"certified_by","Value":"JQu
ery"},{"InternalName":"certificate_type","Value":"Professional"}]
}]}}
```


In the above JSON data, the Certifications node is a repeating node. Please make sure “RowId” is always zero in create folder. “InternalName” is the internal field name for the case type, and “Value” is value of the field.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/CaseFolder/<CaseTypeInternalName>",
  "method": "POST",
  "headers": {
    "accept": "application/json",
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<sectoken>"
  },
  "data": {
    "data": JSON.stringify(<data>),
    "officeCode": "Ains"
  }
}

    $.ajax(settings).done(function (response) {
  console.log(response);
    });
```

Result: <Returns Folder Id of the created folder> "2017-1234-00096"

2.5 Update Case Folder

The following is the jQuery example for updating a case folder with data specified in JSON format. The folder will be uniquely identified by the “FolderNumber” field.

```
 data =
{"Fields":[{"InternalName":"employee_name","Value":"Micheal"},{"I
nternalName":"employee_id","Value":"12354"},{"InternalName":"date
_of_birth","Value":"2016-06-
06T00:00:00.0000000"},{"InternalName":"gender","Value":"Male"},{"
InternalName":"street_address","Value":"New Street
Address"},{"InternalName":"city","Value":"NewCityTest"},{"Interna
lName":"state","Value":"New
Jersey"}],"RecordInfo":{"Certifications":[{"RowNumber":1,"RowId":
104,"Fields":[{"InternalName":"certification_date","Value":"2015-
11-
02T00:00:00.0000000"},{"InternalName":"certified_by","Value":"Mic
rosoft"},{"InternalName":"certificate_type","Value":"Expert"}]},{
"RowNumber":2,"RowId":
105,"Fields":[{"InternalName":"certification_date","Value":"2016-
10-
11T00:00:00.0000000"},{"InternalName":"certified_by","Value":"JQu
ery"},{"InternalName":"certificate_type","Value":"Professional"}]
}]}}
```

In the above JSON data, the Certifications node is a repeating node. If “RowId” is preserved (non zero), it will update. If set to zero, it will create a new record entry.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/CaseFolder/<FolderNumber>",
  "method": "PUT",
  "headers": {
    "accept": "application/json",
```

```
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<sectoken>"
  },
  "data": {
    "data": JSON.stringify(<data>)
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

True

2.6 Delete Case Folder

The following is the jQuery example for deleting a case folder for the specified FolderId.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/CaseFolder/<folderNumber>",
  "method": "DELETE",
  "headers": {
    "accept": "application/json",
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<sectoken>"
  }
}
```

```
$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

None

3 Contact Services

3.1 GET Contact Model

The GET contact model REST Services method is used to get the contact model details.

The following is a jQuery example for getting the configured data model for a Contact type in eCASE. In this example, the data model for Contact Type “SENATOR” is queried:

ContactTypeName: SENATOR

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/Contact/Metadata/<contactTypeName>",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "sectoken": "<secToken>",
    "content-type": "application/json"
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
{
  "ContactTypeInternalName": "SENATOR",
  "Category": 0,
  "ContactFields": [
```

```
{
  "InternalName": "PREFIX",
  "FieldDataType": 1
},
{
  "InternalName": "FIRST_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "MIDDLE_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "LAST_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "CONTACT_FULL_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "SUFFIX",
  "FieldDataType": 1
},
{
  "InternalName": "TITLE",
  "FieldDataType": 1
},
{
  "InternalName": "WORK_PHONE",
  "FieldDataType": 1
},
{
```


```
  "InternalName": "FAX",
  "FieldDataType": 1
},
{
  "InternalName": "HOME_PHONE",
  "FieldDataType": 1
},
{
  "InternalName": "MOBILE",
  "FieldDataType": 1
},
{
  "InternalName": "ALT_PHONE",
  "FieldDataType": 1
},
{
  "InternalName": "EMAIL",
  "FieldDataType": 1
},
{
  "InternalName": "WEB_PAGE",
  "FieldDataType": 1
},
{
  "InternalName": "C_DESCRIPTION",
  "FieldDataType": 2
},
{
  "InternalName": "OFFICE_NAME",
  "FieldDataType": 1
},
{
  "InternalName": "DEPARTMENT",
```

```
  "FieldDataType": 1
},
{
  "InternalName": "ADDRESS_1",
  "FieldDataType": 1
},
{
  "InternalName": "ADDRESS_2",
  "FieldDataType": 1
},
{
  "InternalName": "CITY",
  "FieldDataType": 1
},
{
  "InternalName": "STATE_REGION",
  "FieldDataType": 12
},
{
  "InternalName": "OTHER",
  "FieldDataType": 1
},
{
  "InternalName": "COUNTRY",
  "FieldDataType": 12
},
{
  "InternalName": "ZIP_CODE",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_OFFICE_NAME",
  "FieldDataType": 1
```


```
},
{
  "InternalName": "SECONDARY_DEPARTMENT",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_ADDRESS_1",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_ADDRESS_2",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_CITY",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_STATE_REGION",
  "FieldDataType": 12
},
{
  "InternalName": "SECONDARY_OTHER",
  "FieldDataType": 1
},
{
  "InternalName": "SECONDARY_COUNTRY",
  "FieldDataType": 12
},
{
  "InternalName": "SECONDARY_ZIP_CODE",
  "FieldDataType": 1
},
```


```
  {
    "InternalName": "NOTES",
    "FieldDataType": 2
  },
  {
    "InternalName": "STATE",
    "FieldDataType": 12
  },
  {
    "InternalName": "PARTY",
    "FieldDataType": 12
  },
  {
    "InternalName": "CLASS",
    "FieldDataType": 12
  },
  {
    "InternalName": "SIGNIFICANT",
    "FieldDataType": 6
  },
  {
    "InternalName": "IN_SERVICE",
    "FieldDataType": 3
  },
  {
    "InternalName": "START_DATE",
    "FieldDataType": 4
  },
  {
    "InternalName": "END_DATE",
    "FieldDataType": 4
  }
]
```


```
}
```

3.2 GET Contact

The GET contact REST Services method is used to get the contact details for a specified contact. The Contact ID is used to retrieve the contact resource. Contact ID is the eCASE-generated unique identifier for a contact.

The following is a jQuery example for getting a contact information using contact identifier. In this example, we query contact type resource using Contact ID, as shown below:

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{BaseURI}}/Contact/<contactId)",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "content-type": "application/json",
    "sectoken": "<secToken>"
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
{
  "Id": 5,
  "Created": "2016-11-03T13:43:55.633",
  "Modified": null,
  "Fields": [
    {
```

```
  "InternalName": "STATE",
  "Value": "40;#New York"
},
{
  "InternalName": "PARTY",
  "Value": "27;#D"
},
{
  "InternalName": "CLASS",
  "Value": "30;#I"
},
{
  "InternalName": "SIGNIFICANT",
  "Value": "Y"
},
{
  "InternalName": "IN_SERVICE",
  "Value": "2015"
},
{
  "InternalName": "START_DATE",
  "Value": "2012-11-25T00:00:00.0000000"
},
{
  "InternalName": "END_DATE",
  "Value": "2016-11-30T00:00:00.0000000"
},
{
  "InternalName": "TITLE",
  "Value": null
},
{
  "InternalName": "FIRST_NAME",
```


```
  "Value": null
},
{
  "InternalName": "MIDDLE_NAME",
  "Value": null
},
{
  "InternalName": "LAST_NAME",
  "Value": "pointing"
},
{
  "InternalName": "PREFIX",
  "Value": null
},
{
  "InternalName": "SUFFIX",
  "Value": null
},
{
  "InternalName": "WORK_PHONE",
  "Value": null
},
{
  "InternalName": "HOME_PHONE",
  "Value": null
},
{
  "InternalName": "FAX",
  "Value": null
},
{
  "InternalName": "MOBILE",
  "Value": null
```

```
},
{
  "InternalName": "EMAIL",
  "Value": null
},
{
  "InternalName": "WEB_PAGE",
  "Value": null
},
{
  "InternalName": "NOTES",
  "Value": null
},
{
  "InternalName": "CONTACT_FULL_NAME",
  "Value": "pointing"
},
{
  "InternalName": "ALT_PHONE",
  "Value": null
},
{
  "InternalName": "C_DESCRIPTION",
  "Value": null
},
{
  "InternalName": "OFFICE_NAME",
  "Value": null
},
{
  "InternalName": "DEPARTMENT",
  "Value": null
},
```


```
{
  "InternalName": "ADDRESS_1",
  "Value": null
},
{
  "InternalName": "ADDRESS_2",
  "Value": null
},
{
  "InternalName": "CITY",
  "Value": null
},
{
  "InternalName": "STATE_REGION",
  "Value": ""
},
{
  "InternalName": "OTHER",
  "Value": null
},
{
  "InternalName": "COUNTRY",
  "Value": ""
},
{
  "InternalName": "ZIP_CODE",
  "Value": null
},
{
  "InternalName": "SECONDARY_OFFICE_NAME",
  "Value": null
},
{
```


```
      "InternalName": "SECONDARY_DEPARTMENT",
      "Value": null
    },
    {
      "InternalName": "SECONDARY_ADDRESS_1",
      "Value": null
    },
    {
      "InternalName": "SECONDARY_ADDRESS_2",
      "Value": null
    },
    {
      "InternalName": "SECONDARY_CITY",
      "Value": null
    },
    {
      "InternalName": "SECONDARY_STATE_REGION",
      "Value": ""
    },
    {
      "InternalName": "SECONDARY_OTHER",
      "Value": null
    },
    {
      "InternalName": "SECONDARY_COUNTRY",
      "Value": ""
    },
    {
      "InternalName": "SECONDARY_ZIP_CODE",
      "Value": null
    }
  ]
}
```

3.3 Search Contact

The following is the jQuery example for searching contacts using search filters CONTACT_FULL_NAME = smith where “CONTACT_FULL_NAME” is a field name defined in the contact and “smith” is the name of the contact. Comma should be added if more than one filters used.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/Contact/",
  "method": "GET",
  "headers": {
    "accept": "application/json",
    "sectoken": "{{secToken}}",
    "content-type": "application/json"
  },
      "data": {
"CONTACT_FULL_NAME": "John Smith",
"CITY": "New York"
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

```
[
  {
    "Id": 2,
```

```
    "Created": "2016-11-02T15:55:03.12",
    "FullName": "Smith",
    "Category": 0,
    "ContactTypeInternalName": "SENATOR",
    "PrimaryAddress": "xyz address , time, GAITHERSBURG MD
20878",
    "SecondaryAddress": null
  },
  {
    "Id": 3,
    "Created": "2016-11-03T13:40:59.753",
    "FullName": "Smith",
    "Category": 0,
    "ContactTypeInternalName": "SENATOR",
    "PrimaryAddress": null,
    "SecondaryAddress": null
  },
  {
    "Id": 4,
    "Created": "2016-11-03T13:41:30.567",
    "FullName": "Smith",
    "Category": 0,
    "ContactTypeInternalName": "SENATOR",
    "PrimaryAddress": null,
    "SecondaryAddress": null
  },
  {
    "Id": 21,
    "Created": "2016-11-04T11:31:54.42",
    "FullName": "Smith",
    "Category": 0,
    "ContactTypeInternalName": "SENATOR",
```

```
    "PrimaryAddress": "xyz address , time, GAITHERSBURG MD
20878",
    "SecondaryAddress": null
  }
]
```

3.4 Create Contact

The following is a jQuery example shown for creating new contact where contactData is in JSON format with key value pair of field name and field value. POST the JSON data for creating contact. The following is sample JSON data we are assigning to data variable.

```
contactData = [{"InternalName":"STATE","Value":"New
York"},{"InternalName":"PARTY","Value":"D"},{"InternalName":"CLAS
S","Value":"I"},{"InternalName":"SIGNIFICANT","Value":"Y"},{"Inte
rnalName":"IN_SERVICE","Value":"2015"},{"InternalName":"START_DAT
E","Value":"2012-11-
25T00:00:00.0000000"},{"InternalName":"END_DATE","Value":"2016-
11-30T00:00:00.0000000"},{"InternalName":"TITLE","Value":null}]
```

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/Contact/<contactTypName>",
  "method": "POST",
  "headers": {
    "accept": "application/json",
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<secToken>"
  },
  "data": {
    "data": "<contactData>"
```

```
  }
}

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result: <Returns Contact Id of the created contact>

"2"

3.5 Update Contact

The following is the jQuery example for updating a contact with contactData specified in JSON format. The contact will be uniquely identified by the ContactId field. The following is sample JSON data for updating a folder.

```
contactData = [{"InternalName":"STATE","Value":"New
York"},{"InternalName":"PARTY","Value":"D"},{"InternalName":"CLAS
S","Value":"I"},{"InternalName":"SIGNIFICANT","Value":"Y"},{"Inte
rnalName":"IN_SERVICE","Value":"2015"},{"InternalName":"START_DAT
E","Value":"2012-11-
25T00:00:00.0000000"},{"InternalName":"END_DATE","Value":"2016-
11-30T00:00:00.0000000"},{"InternalName":"TITLE","Value":null}]
```

In the above JSON data, the Certifications node is a repeating node. If “RowId” is preserved (non zero), it will update. If set to zero, then it creates a new record entry.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/Contact/<ContactId>",
  "method": "PUT",
  "headers": {
    "accept": "application/json",
```

```
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<sectoken>"
  },  "data": {
    "data": "<contactData>" }

$.ajax(settings).done(function (response) {
  console.log(response);
});
```

Result:

True

3.6 Delete Contact

The following is a jQuery example for deleting a contact for specified ContactId.

Sample JSON Request:

```
var settings = {
  "async": true,
  "crossDomain": true,
  "url": "{{baseURL}}/Contact/<ContactId>",
  "method": "DELETE",
  "headers": {
    "accept": "application/json",
    "content-type": "application/x-www-form-urlencoded",
    "sectoken": "<sectoken>"
  }

}

$.ajax(settings).done(function (response) {
  console.log(response);
```

```
});
```

Result:

None

4 System Data Services

The following methods provide access to system information.

4.1 GetAllLoggedInUsers

Sample HTTP GET Request:

```
http://qa-ecase-
app13/eCaseOData/systemodata/GetAllLoggedInUsers?fromDate=2019-
01-01&toDate=2019-09-01
```

Result:


```
[
  {
     "UserId": 1217,
     "LoginTime": "2018-07-17T10:53:04.013",
     "LogoutTime": "2018-07-17T21:13:06.5",
     "UserName": "ygelee",
     "Office": "AINS",
     "WorkStation": "192.168.0.11",
     "Duration": "10.33",
     "ModuleName": "ECASE"
  },
  {
     "UserId": 1221,
     "LoginTime": "2019-03-05T11:47:57.553",
     "LogoutTime": "2019-03-05T22:04:38.513",
     "UserName": "uladala",
     "Office": "AINS",
     "WorkStation": "192.168.0.52",
     "Duration": "10.28",
     "ModuleName": "ECASE"
  },
  {
     "UserId": 1,
```

```
        "LoginTime": "2018-04-25T15:46:06.523",
        "LogoutTime": "2018-04-25T16:01:23.08",
        "UserName": "smaeekh",
        "Office": "AINS",
        "WorkStation": "192.168.0.53",
        "Duration": "0.25",
       "ModuleName": "ECASE"
    }
]
```


4.2 GetAuditTrailReport

Sample HTTP GET Request:

```
http://qa-ecase-app13/eCaseOData/systemodata/
GetAuditTrailReport?fromDate=2018-01-01&toDate=2019-10-
01&auditEvents=Delete&appId=1
```

Result:


```
[{"TimeOfAction":"2019-09-26T15:11:32.07","Comments":"D
Patel","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-09-26T15:11:28.087","Comments":"D
Patel","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-09-24T11:25:56.757","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-09-18T09:21:59.747","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-29T11:00:19.513","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-27T15:34:01.927","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-27T15:29:54.28","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-27T15:29:48.92","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-27T09:20:51.07","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-21T13:34:35.313","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-08-20T09:41:54.253","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
```

```
019-08-12T13:47:31.407","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-07-05T09:37:15.683","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-05-29T10:52:11.71","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-05-08T13:41:03.143","Comments":"Chris2
Nieva","ActionBy":"CA","WorkStation":null},{"TimeOfAction":"201
9-05-08T13:41:02.673","Comments":"Chris2
Nieva","ActionBy":"CA","WorkStation":null},{"TimeOfAction":"201
9-05-08T13:41:02.143","Comments":"Chris2
Nieva","ActionBy":"CA","WorkStation":null},{"TimeOfAction":"201
9-04-12T14:33:24.48","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-04-12T11:03:50.92","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null},{"TimeOfAction":"2
019-04-12T10:46:48.787","Comments":"Admin
Admin","ActionBy":"AINS","WorkStation":null}
]
```


4.3 GetFailedLoginUsers

Sample HTTP GET Request:

http://qa-ecase-app13/eCaseOData/systemodata/GetFailedLoginUsers?fromDate=2018-01-01&toDate=2019-10-01

Result:


```
[
   {
      "UserId": 1217,
      "UserName": "jsmith",
      "FullName": "John Smith",
      "WorkStation": "192.168.0.11",
      "FailedTime": "2018-07-17T10:53:04.013",
      "ModuleName": "ECASE"
   },
   {
      "UserId": 1221,
      "UserName": "jcarter",
      "FullName": "John Carter",
```

```
        "WorkStation": "192.168.0.52",
        "FailedTime": "2019-03-05T11:47:57.553",
        "ModuleName": "ECASE"
    },
]
```