REST API: Guide for Developers

Version 1.03

SP-API-REST-PG-201805--R001.03

SP-API-REST-PG-201805--R001.03

About this Guide

This **REST API Guide for Developers** provides an introduction to the key characteristics and architecture of the Sage People REST API, using Sage People resources, and authentication.

To use this document, you need a basic familiarity with software development, Web services, and the Salesforce user interface.

Related Guides

REST API: Guide for Implementers

Coverage summary

A step by step guide to setting up a web service between a customer server and the Sage People database to enable REST API level access to Sage People hosted data.

Typical target audience

You have responsibility for configuring the Sage People system to support REST API access from a customer server.

Introduction

Sage People provides a REST API for external programs and systems to interact with the Sage People technologies. This document describes how to use the Sage People REST API.

Sage People is built on the Salesforce Force.com platform. The Force.com platform has its own REST API which can be used to interact with any of the Sage People data objects. However, such access is at the database layer of the Sage People system, bypassing the business logic and part of the security layers of Sage People. As such, use of the Sage People REST API is recommended.

Use this Guide to understand:

- The Force.com REST API, on which the Sage People REST API is based. The authentication and data formatting options are identical, and come from the same code base.
- The key characteristics and architecture of REST API. This will help you understand how your applications can best use the Sage People REST resources.
- How to set up your development environment so you can begin working with REST API immediately.
- How to use REST API by following a quick start that leads you step by step through a typical use case.

Using Sage People Resources

A REST resource is an abstraction of a piece of information, such as a single data record, a collection of records, or even dynamic real-time information. Each resource in the Force.com REST API is identified by a named URI, and is accessed using standard HTTP methods (HEAD, GET, POST, PATCH, DELETE). The Force.com REST API is based on the usage of resources, their URIs, and the links between them. You use a resource to interact with your Sage People or Force.com organization. For example, you can:

- Retrieve summary information about the API versions available to you.
- Obtain detailed information about a Sage People object such as a Team Member or a Training record.
- Update or delete records.

To retrieve information about a Team Member, submit a request for the Versions resource. This example uses cURL on the na1 server instance:

curl https://na1.cloudforce.com/services/apexrest/fHCM2/v1.0/teammember/12345

Sage People runs on multiple server instances, as does the underlying Force.com platform. The examples in this guide use the na1 instance. Your organization might use a different instance.

Important characteristics of the Sage People (and Force.com) REST API resources and architecture:

Stateless	Each request from client to server must contain all the information necessary to understand the request, and not use any stored context on the server. However, the representations of the resources are interconnected using URLs, which allow the client to progress between states.
Caching behavior	Responses are labeled as cacheable or non-cacheable
Uniform interface	All resources are accessed with a generic interface over HTTP.
Named resources	All resources are named using a base URI that follows your Sage People/Force.com URI.
Layered components	The Sage People REST API architecture allows for such intermediaries as proxy servers and gateways between the client and the resources.
Authentication	The Sage People REST API supports OAuth 2.0 (an open protocol to allow secure API authorization). Refer to Authentication (see page 8) for more details.
Support for JSON and XML	• JSON is the default. You can use the HTTP ACCEPT header to select either JSON or XML, or append .json or .xml to the URI - for example: `/teammember/001D000000INjVe.json` • JavaScript Object Notation (JSON) format is supported with UTF-8. • Date-time information is in ISO8601 format. • XML serialization is similar to SOAP API. • XML requests are supported in UTF-8 and UTF-16, and XML responses are provided in UTF-8.
Case Sensitivity	The urls and data field names are case sensitive. Use the case as defined within this document. /TeamMember/ is not the same as /teammember/

Using cURL in the REST Examples

The examples in this guide use the cURL command line tool to send HTTP requests to access, create, and manipulate REST resources on the Sage People/Force.com platform. cURL is pre-installed on many Linux and Mac systems. Windows users can download a version at curl.haxx.se/. When using HTTPS on Windows, ensure that your system meets the cURL requirements for SSL.

cURL is an open source tool and is not supported by Sage People.

Authentication

The Sage People REST API uses the same authentication options as the Force.com REST API. This document describes the main options in outline. There are several good coding examples of using these available on the web, and any that work with the Force.com REST API will also work with the Sage People REST API.

Please read the sections on authentication at the following site:

https://developer.salesforce.com/index.php?title=Getting_Started_with_the_Force.com_REST_API&oldid=35544

Understanding Authentication

Sage People uses authentication to allow users to securely access data without having to reveal username and password credentials.

Before making REST API calls, you must authenticate the user using OAuth 2.0. To do so, you'll need to:

- Set up a remote access application definition in Sage People.
- Determine the correct OAuth endpoint to use.
- Authenticate the user via one of several different OAuth 2.0 authentication flows. An OAuth authentication flow defines a series of steps used to coordinate the authentication process between your application and Sage People. Supported OAuth flows include:
  - Web server flow, where the server can securely protect the consumer secret.
  - User-agent flow, used by applications that cannot securely store the consumer secret.
  - Username-password flow, where the application has direct access to user credentials.

After successfully authenticating the user, you'll receive an access token which can be used to make authenticated REST API calls.

Defining Remote Access Applications

To authenticate using OAuth, you must define a remote access application in Sage People.

A **remote access application** is an application external to Sage People that uses the OAuth protocol to verify both the Sage People user and the external application. When you develop a new external application that needs to authenticate with Sage People, you need to define a new remote access application that informs Sage People of this new authentication entry point.

Use the following steps to create a new remote access application:

1. Got to **Setup** > **App Setup** > **Create** > **Apps** and in the **Connected Apps** Related List select **New**.
2. In the **Basic Information** section enter the name of your application, a contact email address, and any other information appropriate for your application.
3. Enter a URL for **Info URL**. This is where the user can go for more information about your application.
4. In the **API (Enable OAuth Settings)** section, check **Enable OAuth Settings** to display additional OAuth related fields.
5. Enter a **Callback URL**. Depending on which OAuth flow you use, this is typically the URL to which a user's browser is redirected after successful authentication. As this URL is used for some OAuth flows to pass an access token, the URL must use secure HTTP (HTTPS) or a custom URI scheme.
6. For set up and initial testing, select the **Full access (full)** OAuth Scope. Be aware that Full access enables the logged in user to access all permitted data, and includes all other scopes. Full access does not return a refresh token.

   When successfully tested, reset the OAuth Scope to provide a more restricted (safer) level of access appropriate to your needs.
7. Select **Save**.

   The summary page for the new Connected App is displayed, including the **Consumer Key** and a link to the **Consumer Secret**.
8. Select the link to reveal the Consumer Secret.
9. Select **Manage**.

   The Connected App Detail page is displayed.
10. Select **Edit**.
11. For setup and initial testing, in the **OAuth policies** section set **IP Relaxation** to **Relax IP restrictions**.

    When successfully tested, reset **IP Relaxation** to **Enforce IP restrictions**.
12. Select **Save**.

When you have defined a remote access application, you use the consumer key and consumer secret to authenticate your application.

OAuth Endpoints

OAuth endpoints are the URLs you use to make OAuth authentication requests to Sage People.

You need to use the correct Sage People OAuth endpoint when issuing authentication requests in your application. The primary OAuth endpoints are:

- For authorization:

  https://login.salesforce.com/services/oauth2/authorize

- For token requests:

  https://login.salesforce.com/services/oauth2/token

- For revoking OAuth tokens:

  https://login.salesforce.com/services/oauth2/revoke

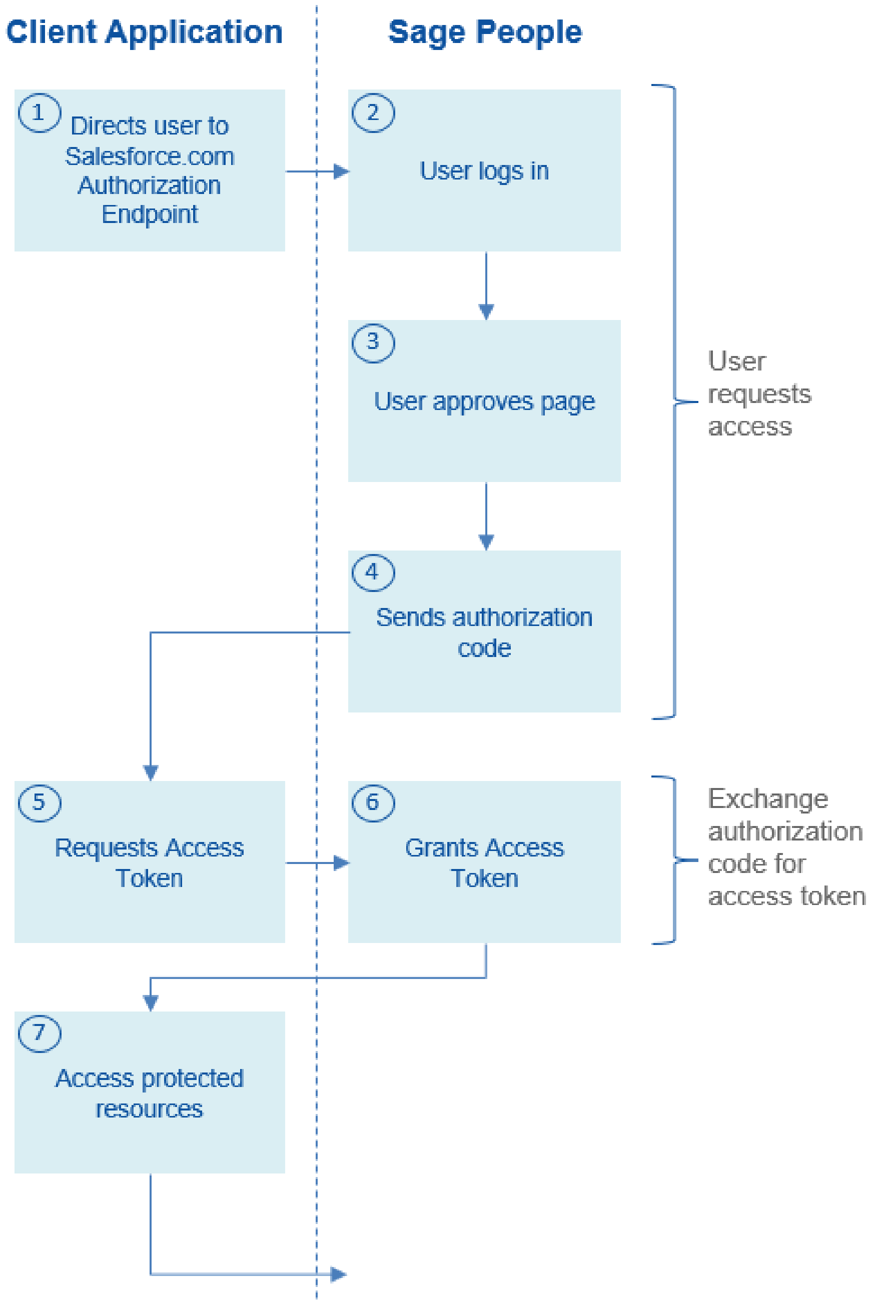
All endpoints require secure HTTP (HTTPS). Each OAuth flow defines which endpoints you need to use and what request data you need to provide.

If you’re verifying authentication on a sandbox organization, use "test.salesforce.com" instead of "login.salesforce.com" in all the OAuth endpoints listed above.

The Web Server OAuth Authentication Flow

The Web server authentication flow is used by applications that are hosted on a secure server. A critical aspect of the Web server flow is that the server must be able to protect the consumer secret.

In this flow, the client application requests the authorization server to redirect the user to another web server or resource that authorizes the user and sends the application an authorization code. The application uses the authorization code to request an access token.


Client Application
Sage People
1 Directs user to Salesforce.com Authorization Endpoint
2 User logs in
3 User approves page
4 Sends authorization code
User requests access
5 Requests Access Token
6 Grants Access Token
Exchange authorization code for access token
7 Access protected resources

1. The application redirects the user to the appropriate Salesforce authorization endpoint, such as:

   https://login.salesforce.com/services/oauth2/authorize

   The following parameters are required:

Parameter	Description
response_type	Must be **code** for this authentication flow.
client_id	The **Consumer Key** from the remote access application definition.
redirect_url	The **Callback URL** from the remote access application definition.

   The following parameters are optional:

Parameter	Description
display	Changes the login page's display type. Valid values are: • **page** Full-page authorization screen. The default value if none is specified. • **popup** Compact dialog optimized for modern Web browser popup windows. • **touch** Mobile-optimized dialog designed for modern smartphones such as Android and iPhone. • **mobile** Mobile optimized dialog designed for smartphones such as BlackBerry OS 5 that don't support touch screens.
immediate	Determines whether the user should be prompted for login and approval. Values are either **true** or **false**. Default is **false**. • If set to **true**, and if the user is currently logged in and has previously approved the application, the approval step is skipped. • If set to **true** and the user is not logged in or has not previously approved the application, the session is immediately terminated with the **immediate_unsuccessful** error code.
state	Specifies any additional URL-encoded state data to be returned in the callback URL after approval.
scope	Specifies what data your application can access. See **Scope Parameter Values** in the online help for more information.

   An example authorization URL might look something like:

   https://login.salesforce.com/services/oauth2/authorize?response_type=code&client_id=3MVG9lKcPoNINVBIPJjdw1J9LLM82HnFVVX19KY1uA5mu0QqEWhqKpoW3svG3XHrXDiCQjK1mdgAvhCscA9GE&redirect_uri=https%3A%2F%2Fwww.mysite.com%2Fcode_callback.jsp&state=mystate

2. The user logs into Sage People with their credentials. The user is interacting with the authorization endpoint directly, so the application never sees the user's credentials.

3. After successfully logging in, the user is asked to authorize the application. Note that if the user has already authorized the application, this step is skipped.

4. Once Sage People confirms that the client application is authorized, the end-user's Web browser is redirected to the callback URL specified by the **redirect_uri** parameter. Sage People appends authorization information to the redirect URL with the following values:

Parameter	Description
code	Authorization code the consumer must use to obtain the access and refresh tokens.
state	The state value that was passed in as part of the initial request, if applicable.

An example callback URL with authorization information might look something like:

https://www.mysite.com/authcode_callback?code=aWekysIEeqM9PiThEfm0Cnr6MoLIfwWyRJcqOqHdF8f9INokharAS09ia7UNP6RiVScerfhc4w%3D%3D

5. The application extracts the authorization code and passes it in a request to Sage People for an access token. This request is a POST request sent to the appropriate Sage People token request endpoint, such as:

https://login.salesforce.com/services/oauth2/token

The following parameters are required:

Parameter	Description
grant_type	Value must be **authorization_code** for this flow.
client_id	The **Consumer Key** from the remote access application definition.
client_secret	The **Consumer Secret** from the remote access application definition.
redirect_uri	The **Callback URL** from the remote access application definition.
code	Authorization code the consumer must use to obtain access and refresh tokens.

The following parameter is optional:

Parameter	Description
format	Expected return format. The default is **json.** Values are: • urlencoded • json • xml The return format can also be specified in the header of the request using one of the following: • Accept: application/x-www-form-urlencoded • Accept: application/json • Accept: application/xml

An example access token POST request might look something like:

POST /services/oauth2/token HTTP/1.1

Host: login.salesforce.com

grant_type=authorization_code&code=aPrxsmIEeqM9PiQroGEWx1UiMQd95_5JUZVEhsOFhS8EVvbfYBBJli2W5fn3zbo.8hojaNW_1g%3D%3D&client_id=3MVG9lKcPoNINVBIPJjdw1J9LLM82HnFVVX19KY1uA5mu0QqEWhqKpoW3svG3XHrXDiCQjK1mdgAvhCscA9GE&client_secret=1955279925675241571&

redirect_uri=https%3A%2F%2Fwww.mysite.com%2Fcode_callback.jsp

6. If this request is successful, the server returns a response body that contains the following:

Parameter	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
refresh_token	Token that can be used in the future to obtain new access tokens. **Warning:** This value is a secret. You should treat it like the user's password and use appropriate measures to protect it.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user and query for more information about the user. Can be used in an HTTP request to get more information about the end user.
issued_at	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970)
signature	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and **issued_at** value. The **signature** can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example JSON response body might look something like:

**"id":"https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
"issued_at":"1278448101416","refresh_token":"5Aep8614iLM.Dq661ePDmPEgaAW9**

**Oh_L3JKkDpB4xReb54_pZebnUG0h6Sb4KUVDpNtWEofWM39yg==","instance_url":
"https://na1.salesforce.com","signature":"CMJ4l+CCaPQiKjoOEwEig9H4wqhpuLSk**

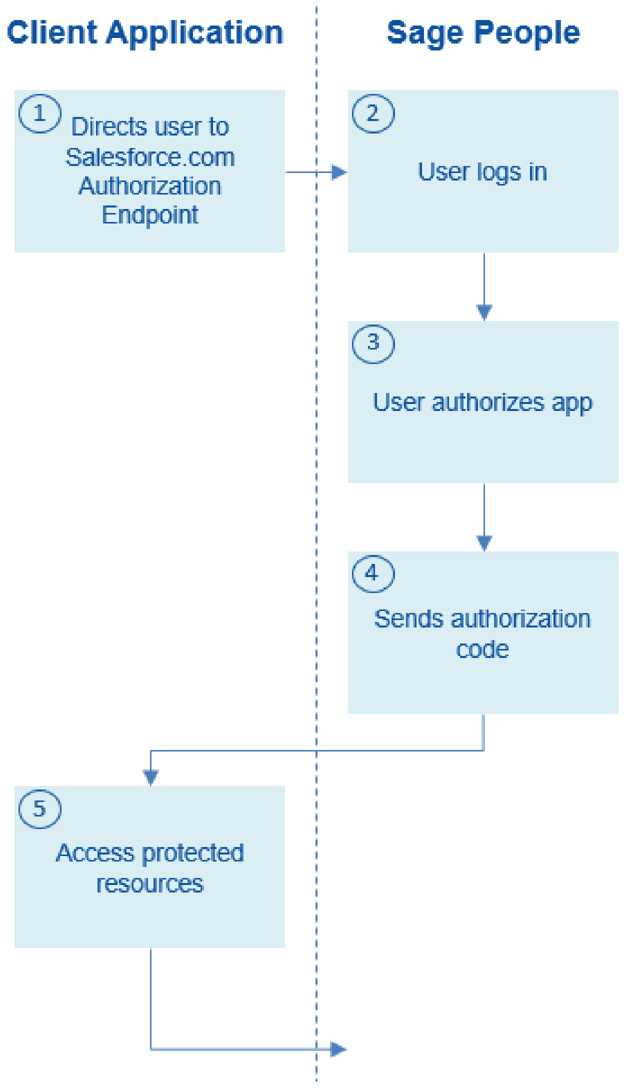
**4J2urAe+fVg=","access_token":"00Dx0000000BV7z!AR8AQP0jITN80ESEsj5EbaZTFG0R
NBaT1cyWk7TrqoDjoNIWQ2ME_sTZzBjfmOE6zMHq6y8PIW4eWze9JksNEkWUl.Cju7m4"}**

7. The application uses the provided access token and refresh token to access protected user data.

The User-Agent OAuth Authentication Flow

The user-agent authentication flow is used by client applications (consumers) residing in the user's device. This could be implemented in a browser using a scripting language such as JavaScript, or from a mobile device or a desktop application. These consumers cannot keep the client secret confidential.

In this flow, the client application requests the authorization server to redirect the user to another Web server or resource which is capable of extracting the access token and passing it back to the application.


Client Application
Sage People
1 Directs user to Salesforce.com Authorization Endpoint
2 User logs in
3 User authorizes app
4 Sends authorization code
5 Access protected resources

1. The application redirects the user to the appropriate Sage People authorization endpoint, such as:

   https://login.salesforce.com/services/oauth2/authorize

   The following parameters are required:

Parameter	Description
response_type	Must be **token** for this authentication flow.
client_id	The **Consumer Key** from the remote access application definition.
redirect_url	The **Callback URL** from the remote access application definition.

   The following parameters are optional:

Parameter	Description
display	Changes the login page's display type. Valid values are: • **page** Full-page authorization screen. The default value if none is specified. • **popup** Compact dialog optimized for modern Web browser popup windows. • **touch** Mobile-optimized dialog designed for modern smartphones such as Android and iPhone. • **mobile** Mobile optimized dialog designed for smartphones such as BlackBerry OS 5 that don't support touch screens.
scope	Specifies what data your application can access. See **Scope Parameter Values** in the online help for more information.
state	Specifies any additional URL-encoded state data to be returned in the callback URL after approval.

   An example authorization URL might look something like the following:

   https://login.salesforce.com/services/oauth2/authorize?response_type=token&client_id=3MVG9lKcPoNINVBIPJjdw1J9LLJbP_pqwoJYyuisjQhr_LLurNDv7AgQvDTZwCoZuDZrXcPCmBv4o.8ds.5iE&redirect_uri=https%3A%2F%2Fwww.mysite.com%2Fuser_callback.jsp& state=mystate

2. The user logs into Sage People with their credentials.
3. The user interacts with the authorization endpoint directly, so the application never sees the user's credentials.

4. When authorization is granted, the authorization endpoint redirects the user to the redirect URL. This URL is defined in the remote access application created for the application. Sage People appends access token information to the redirect URL with the following values:

Parameter	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
expires_in	Length of time in seconds for which the access token is valid.
refresh_token	Token that can be used in the future to obtain new access tokens. **Warning:** This value is a secret. You should treat it like the user's password and use appropriate measures to protect it. The refresh token is only returned if the redirect URI is: **https://login.salesforce.com/services/oauth2/success** or used with a custom protocol that is not HTTPS.
state	The state value that was passed in as part of the initial request, if applicable.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user and query for more information about the user. Can be used in an HTTP request to get more information about the end user.
issued_at	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
signature	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and **issued_at** value. The **signature** can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example callback URL with access information appended after the hash sign (#) might look something like:

https://www.mysite.com/user_callback.jsp#access_token=00Dx0000000BV7z%21AR8AQBM8J_xr9kLqmZIRyQxZgLcM4HVi41aGtW0qW3JCzf5xdTGGGSoVim8FfJkZEqxbjaFbberKGk8v8AnYrvChG4qJbQo8&refresh_token=5Aep8614iLM.Dq661ePDmPEgaAW9Oh_L3JKkDpB4xReb54_pZfVti1dPEk8aimw4Hr9ne7VXXVSIQ%3D%3D&expires_in=7200&state=mystate

5. The application uses the provided access token and refresh token to access protected user data.

Keep the following considerations in mind when using the user-agent OAuth flow:

- Because the access token is encoded into the redirection URI, it might be exposed to the end-user and other applications residing on the computer or device. If you're authenticating using JavaScript, call **window.location.replace();** to remove the callback from the browser's history.

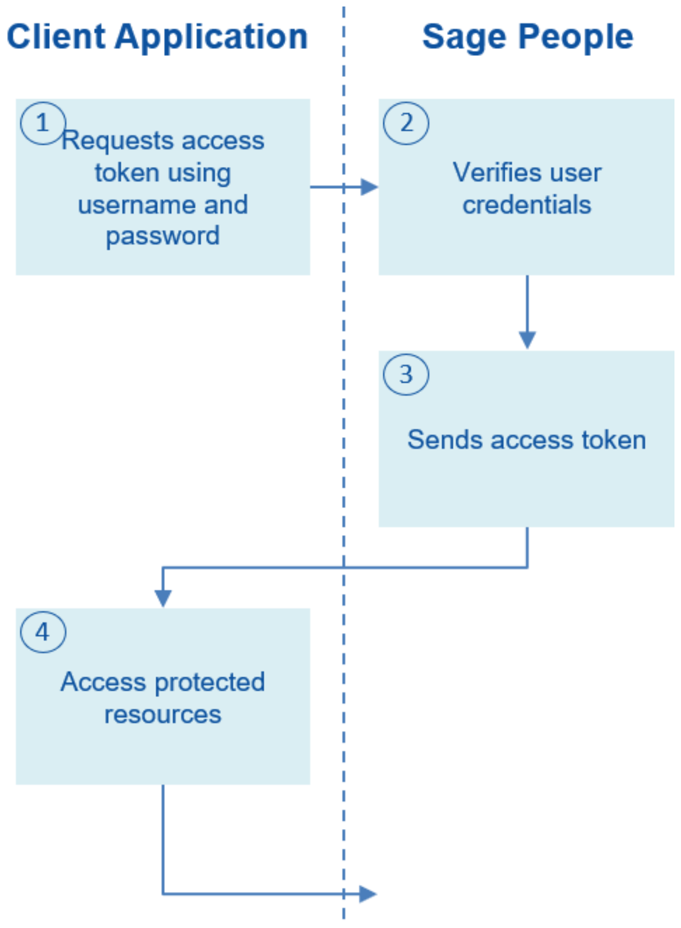
The Username-Password OAuth Authentication Flow

The username-password authentication flow can be used to authenticate when the consumer already has the user's credentials.

In this flow, the user's credentials are used by the application to request an access token as shown in the following steps.

Warning

This OAuth authentication flow involves passing the user's credentials back and forth. Use this authentication flow only when necessary. No refresh token will be issued.


Client Application
Sage People
1
Requests access token using username and password
2
Verifies user credentials
3
Sends access token
4
Access protected resources

1. The application uses the user's username and password to request an access token. This is done via an out-of-band POST request to the appropriate Salesforce token request endpoint, such as:

   https://login.salesforce.com/services/oauth2/token

   The following request fields are required:

Parameter	Description
grant_type	Must be **password** for this authentication flow.
client_id	The **Consumer Key** from the remote access application definition.
client_secret	The **Consumer Secret** from the remote access application definition.
username	End-user's username.
password	End-user's password. You must append the user's security token to their password A security token is an automatically-generated key from Sage People. For example, if a user's password is mypassword, and their security token is XXXXXXXXXX, then enter must be mypasswordXXXXXXXXXX.

   An example request body might look something like the following:

   grant_type=password&client_id=3MVG9lKcPoNINVBIPJjdw1J9LLM82Hn

   FVVX19KY1uA5mu0QqEWhqKpoW3svG3XHrXDiCQjK1mdgAvhCscA9GE&client_secret=

   1955279925675241571&username=testuser%40salesforce.com&password=mypassword123456

2. Sage People verifies the user credentials.
3. If successfully verified, Sage People sends a response to the application with the access token. This response contains the following values:

Parameter	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user as well as query for more information about the user. Can be used in an HTTP request to get more information about the end user.
issued_at	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970).
signature	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and **issued_at** value. The **signature** can be used to verify that the identity URL wasn't modified because it was sent by the server.

   An example response body might look something like:

   **{"id":"https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
   "issued_at":"1278448832702","instance_url":"https://na1.salesforce.com",
   "signature":"0CmxinZir53Yex7nE0TD+zMpvIWYGb/bdJh6XfOH6EQ=","access_token":
   "00Dx0000000BV7z!AR8AQAxo9UfVkh8AlV0Gomt9Czx9LjHnSSpwBMmbRcgKFmxOtvxjTrKW19ye
   6PE3Ds1eQz3z8jr3W7_VbWmEu4Q8TVGSTHxs"}**

4. The application uses the provided access token to access protected user data. Keep the following considerations in mind when using the user-agent OAuth flow:
   - Since the user is never redirected to login at Sage People in this flow, the user can't directly authorize the application, so no refresh tokens can be used. If your application requires refresh tokens, you should consider using the Web server or user-agent OAuth flow.

The OAuth Refresh Token Process

The Web server OAuth authentication flow and user-agent flow both provide a refresh token that can be used to obtain a new access token.

Access tokens have a limited lifetime specified by the session timeout in Salesforce. If an application uses an expired access token, a "Session expired or invalid" error is returned. If the application is using the Web server or user-agent OAuth authentication flows, a refresh token may be provided during authorization that can be used to get a new access token.

The client application obtains a new access token by sending a POST request to the token request endpoint with the following request parameters:

Parameter	Description
grant_type	Must be **refresh_token** for this authentication flow.
refresh_token	The refresh token the client application already received.
client_id	The **Consumer Key** from the remote access application definition.
client_secret	The **Consumer Secret** from the remote access application definition. This parameter is optional.
username	End-user’s username.
format	Expected return format. The default is **json**. Possible values are: • urlencoded • json • xml You can also specify the return format in the header of the request using one of the following: • **Accept: application/x-www-form-urlencoded** • **Accept: application/json** • **Accept: application/xml** This parameter is optional.

An example refresh token POST request might look something like:

```
POST /services/oauth2/token HTTP/1.1

Host: https://login.salesforce.com/

grant_type=refresh_token&client_id=3MVG9lKcPoNINVBIPJjdw1J9LLM82HnFVVX19KY1uA5mu0

QqEWhqKpoW3svG3XHrXDiCQjK1mdgAvhCscA9GE&client_secret=1955279925675241571

&refresh_token=your token here
```

When Salesforce verifies the refresh token request, it sends a response to the application with the following response body parameters:

Parameter	Description
access_token	Access token that acts as a session ID that the application uses for making requests. This token should be protected as though it were user credentials.
instance_url	Identifies the Salesforce instance to which API calls should be sent.
id	Identity URL that can be used to both identify the user and query for more information about the user. Can be used in an HTTP request to get more information about the end user.
issued_at	When the signature was created, represented as the number of seconds since the Unix epoch (00:00:00 UTC on 1 January 1970)
signature	Base64-encoded HMAC-SHA256 signature signed with the consumer's private key containing the concatenated ID and **issued_at** value. The **signature** can be used to verify that the identity URL wasn't modified because it was sent by the server.

An example JSON response body might look something like:

```
{ "id":"https://login.salesforce.com/id/00Dx0000000BV7z/005x00000012Q9P",
"issued_at":"1278448384422","instance_url":"https://na1.salesforce.com",
"signature":"SSSbLO/gBhmmyNUvN18ODBDFYHzakxOMgqYtu+hDPsc=",
"access_token":"00Dx0000000BV7z!AR8AQP0jITN80ESEsj5EbaZTFG0RNBaT1cyWk7T
rqoDjoNIWQ2ME_sTZzBjfmOE6zMHq6y8PIW4eWze9JksNEkWUl.Cju7m4"}
```

Keep in mind the following considerations when using the refresh token OAuth process:

- To configure the session timeout for an access token go to **Setup** > **Security Controls** > **Session Settings**.
- If the application uses the username-password OAuth authentication flow, no refresh token is issued, as the user cannot authorize the application in this flow. If the access token expires, the application using username-password OAuth flow must re-authenticate the user.

Working with Sage People HCM Resources

This section of the Sage People REST API concerns accessing the Team Member resources; including all the HR processes within Sage People WX.

Working with Sage People Team Member Resources

To view a Team Member use a GET to the Team Member URL:

Method	GET
URL	https://na1.salesforce.com/services/apexrest/fHCM2/v1.0/teammember/12345 The part of the URL after the /teammember/ is one of the identifiers of the team member. In the above example this is the unique id "12345".

The team member is identified by either:

- A data object id. This is either the 15 or 18 character long identifier allocated by the database layer. It is set when the team member is created and does not change.
- The team member unique id (or personnel number). This is the id allocated by HR to new team members following your organization's onboarding procedures.
- The single sign on federation id. This is allocated by IT to new team members following your organization's onboarding procedures.

Working with Training Resources

To add a new training record use a POST to the team member training URL:

Method	POST
URL	https://na1.salesforce.com/services/apexrest/fHCM2/v1.0/teammember/12345/training/

Posted Value	**{**
	"resource" : {
	{
	"item" : {
	"trainingName" : "Rosetta Stone: German Level 1, 2 and 3",
	"startDate" : "2011-09-20",
	"Outcome" : "Achieved"
	}
	}
	}

Result	**"a0d30000006dcnpAAA"**

To view a training record for a team member use a GET to the team member training URL. The training record is identified by adding the data object id to the end of the URL; this is the id returned from the POST.

Method	GET
URL	https://na1.salesforce.com/services/apexrest/fHCM2/v1.0/teammember/12345/training/a0d30000006dcnpAAA
Result	**{** **"url" : "/services/apexrest/fHCM2/v1.0/teammember/a0330000006SPgrAAG/training/a0d30000006dcnpAAA ",** **"item" : {** **"trainingName" : "Rosetta Stone: German Level 1, 2 and 3",** **"status" : "New",** **"startDate" : "2011-09-20",** **"Outcome" : " Achieved ",** **"needItem" : null,** **"needArea" : null,** **"managerAction" : null,** **"id" : " a0d30000006dcnp",** **"hasAction" : true,** **"endDate" : null,** **"action" : null** **},** **"fields" : {** **"fHCM2__Start_Date__c" : "2011-09-20",** **"fHCM2__Outcome__c" : null,** **"fHCM2__Result__c" : null** **}** **}**

Status Codes and Error Responses

When an error occurs or a response is successful the response header contains an HTTP code, and the response body usually contains:

- The HTTP response code
- The message accompanying the HTTP response code
- The field or object where the error occurred, if the response returns information about an error.

HTTP response code	Description
200	"OK" success code, for GET or HEAD request.
201	"Created" success code, for POST request.
204	"No Content" success code, for DELETE request.
300	The value returned when an external ID exists in more than one record. The response body contains the list of matching records.
400	The request couldn't be understood, usually because the JSON or XML body contains an error.
401	The session ID or OAuth token used has expired or is invalid. The response body contains the message and errorCode.
403	The request has been refused. Verify that the logged-in user has appropriate permissions.
404	The requested resource couldn't be found. Check the URI for errors, and verify that there are no sharing issues.
405	The method specified in the Request-Line isn't allowed for the resource specified in the URI.
415	The entity in the request is in a format that's not supported by the specified method.
500	An error has occurred within Sage People or Force.com, so the request couldn't be completed. Contact Sage People Customer Support.

Index