

ReachFive

CIAM integration for Salesforce Commerce Cloud Storefronts

Version 20.4.3



reachfive



commerce cloud

Table of Contents

1. Summary.....	4
2. Component Overview.....	5
2.1 Functional Overview.....	5
2.2 Use Cases.....	9
SSO:.....	9
Social Login:.....	9
ReachFive Profiles Synchronization:.....	9
Transition Mode:.....	10
2.3 Limitations, Constraints.....	10
2.4 Compatibility.....	10
2.5 Privacy.....	10
2.6 Security.....	10
Step 1: User authentication process.....	11
Step 2: Handle the Authorization Response.....	11
Step 3: Exchange authorization code for ID token.....	11
Step 4: Retrieve user's profile data from ID Token.....	12
3. Implementation Guide.....	13
3.1 Setup.....	13
3.1.1 Import Cartridge.....	13
3.1.2 Import Site Preferences and Custom Objects.....	13
3.2 Configuration.....	13
3.2.1 Add the Reach Five Cartridge to your Storefront Cartridge Path.....	13
3.2.2 Configure Reach Five Site Preferences.....	13
3.2.3 Configure Reach Five Services.....	15
3.2.4 Configure Reach Five Jobs.....	17
Set Scope:.....	17
Job Schedule (Job is scheduled to run every 5 minutes by default):.....	18
3.3 Custom Code.....	19
3.3.1 Custom Code. Controllers.....	19
3.3.2 Custom Code. Templates.....	23
3.4 External Interfaces.....	26
Authentication.....	26
Get an access token.....	26
Update User.....	26
3.5 Firewall Requirements.....	27
4. Data Storage.....	28
4.1 Availability.....	28
4.2 Support.....	28
5. User Guide.....	28
5.1 Roles, Responsibilities.....	28
5.2 Storefront Functionality.....	28
Case SSO Screen Shot.....	29
Register.....	29
Case Social Login.....	29
Login:.....	29
6. Known Issues.....	30
7. Failover and recovery process.....	30
8. Release History.....	30

1. Summary

ReachFive is a CIAM (Customer Identity and Access Management), providing social media and full authentication features.

With ReachFive, customers can simplify their authentication process and improve their customer identity management.

Currently, more than 30 social networks authentications are managed by ReachFive but the number of social networks supported is continuously growing.

As described in the ReachFive website (<https://www.reach5.co/>), ReachFive provide the following functionalities:

- Single Sign On: sets of API and widget to allow the user to signup, login, manage social account profiles
- Social Login: allows merchants to integrate OAuth providers registering and login functionalities, manage social account profiles

This component integrates SFCC with the ReachFive platform.

Using this cartridge, merchants and solution partners do not need to do anything to integrate new providers when they become available in the ReachFive platform (except activating these new providers in the ReachFive back office, as everything related to the connection is managed by ReachFive).

Merchants willing to use this component to connect to the ReachFive platform will be required to subscribe to the ReachFive service.

This component requires the generic **int_reachfive** and **int_reachfive_sg** cartridges to be included in the code source of the merchant Salesforce Commerce Cloud sites. The merchants will also have to set specific Site Preferences in order for the service to work properly, as described in this documentation.

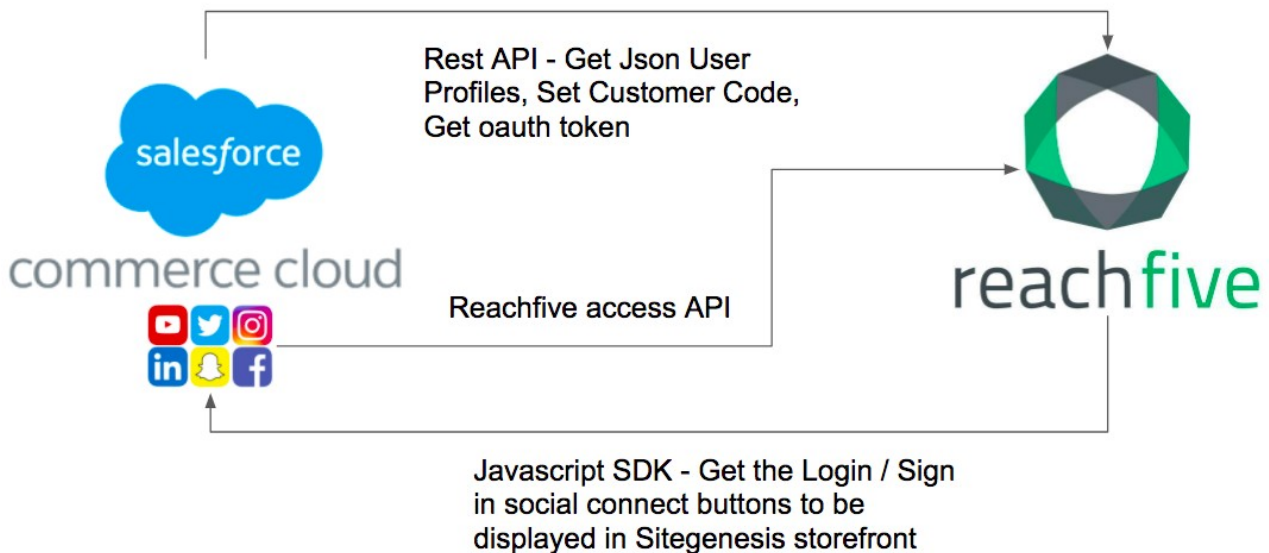
The merchant's storefronts will be slightly modified, to include the Reach Five SSO widget or social connect buttons, as described in this documentation.

Following the current LINK certification process guidelines, the **int_reachfive_sg** cartridge has been developed in the Javascript controller mode. The development of this cartridge is based on 18.3 SiteGenesis version.

2. Component Overview

2.1 Functional Overview

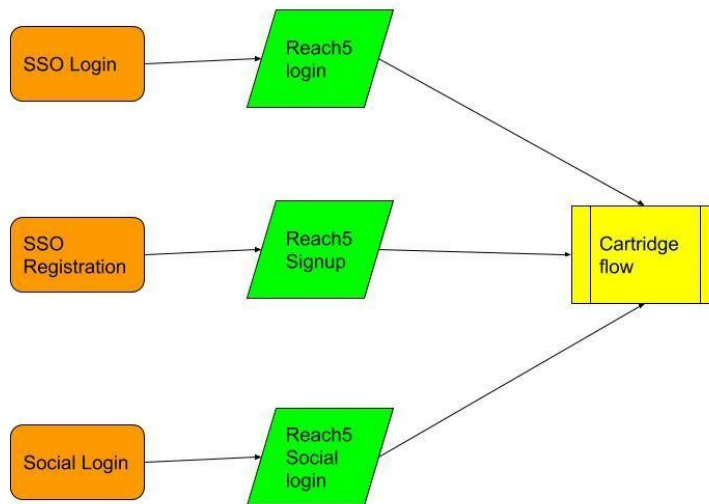
The `int_reachfive` and `int_reachfive_sg` cartridges make use of the Javascript SDK and the REST API provided by ReachFive to exchange data between the Salesforce Commerce Cloud instance and the ReachFive platform.



The Javascript SDK provides widget and API to integrate SFCC with ReachFive, in details:

- SSO: widget to manage login, signup, change password are included in the Sitegenesis login page, registration page, my account pages, checkout pages.
- Social Login: buttons for social login are added to the login page, registration page, my account page, checkout pages.

The following diagrams illustrate the Login and Registration Process



In the SSO scenario, a customer access to the SSO Login / Signup widget from:

- Login Page
- Registration Page
- Checkout Page

In these pages, a widget is presented instead of the standard sitegenesis related form.

In the SSO scenario and Social Login scenario the button for Social Login are present in the following pages:

- Login Page
- Registration Page
- Checkout Page

The SSO signup widget create a Profile in ReachFive server.

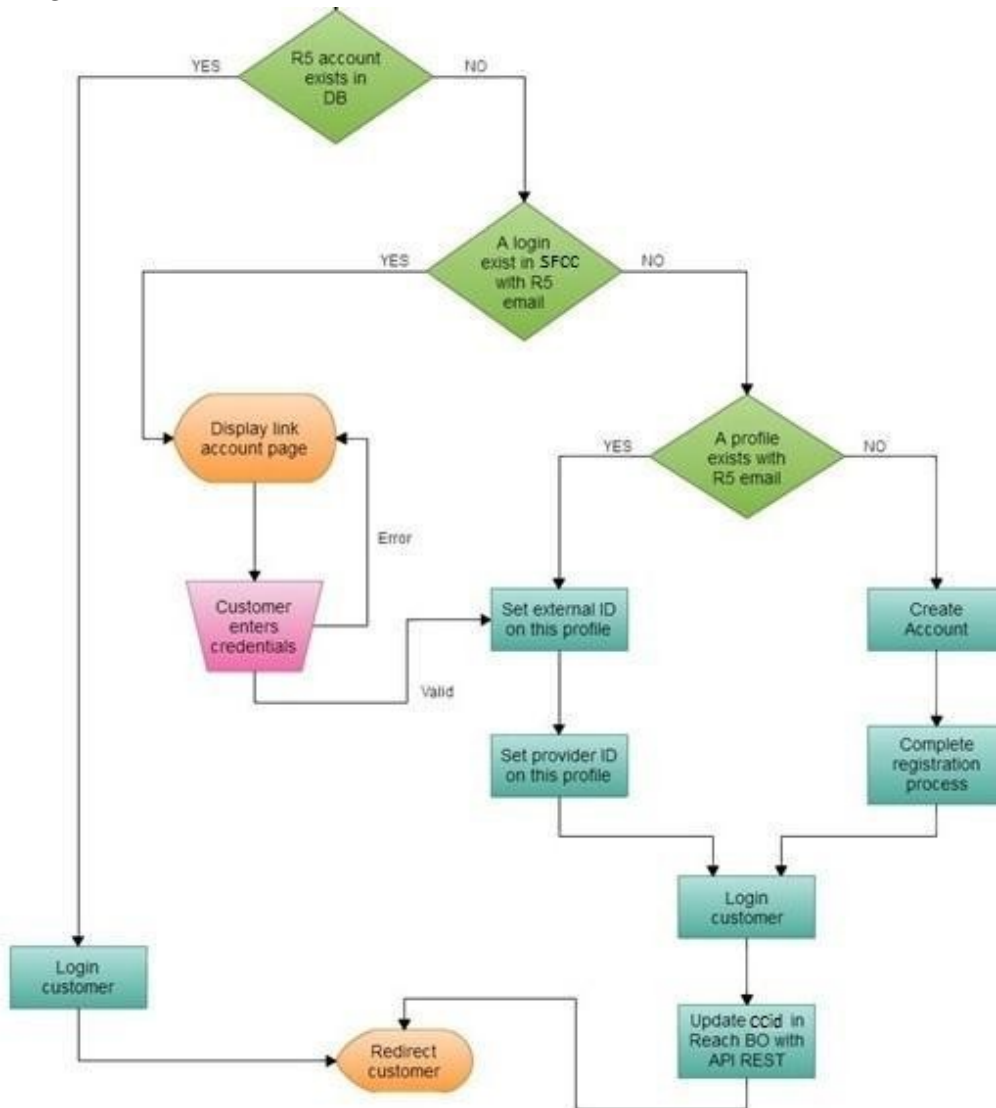
The SSO login widget search for a Profile present on ReachFive server.

The Social Login widget perform an oauth authentication with the social network account and pass the profile information to ReachFive server.

The cartridge flow process manages the link of ReachFive profile information with a customer in SFCC.

The business logic is described in the following diagram:

R5 = ReachFive



Details about ReachFive UI and Core SDK:

<https://www.npmjs.com/package/@reachfive/identity-ui>

<https://developer.reachfive.com/sdk-ui/index.html>


<https://www.npmjs.com/package/@reachfive/identity-core>

<https://developer.reachfive.com/sdk-core/index.html>

A transition phase is available in the cartridge since version 20.4.0 and allow to silently signup in Reachfive on login. It makes it possible to unplug the reachfive cartridge and keep the SFCC accounts working with the login and last password of the shoppers (password synchronized with reachfive and internal SFCC Profil).

ReachFive Profiles Synchronization

'ReachFive-Synchronization' job was created. The job will take all customers that have one of the Profile attributes (reachfiveSendVerificationEmail, reachfiveUpdateEmailAddress, reachfiveUpdateProfile) set to true. Then for each of those customers it will do the appropriate ReachFive calls and set the flags to false. If any of the calls fail the error will be logged in the error logs and the 'reachfiveError' attribute will be populated only with the last errors.



Sandbox - reachfive01
RefArch ▼

Merchant Tools ▼

Administration ▼

Storefront

Car Brand:

Phone Numbers

Home Phone:

Business Phone:

Mobile Phone:

Fax Number:

Loyalty Card Attribute

CardNo:

Expiration Date of Loyalty Card: ...
MM/dd/yyyy

ReachFive

ReachFive send verification email: ☒

ReachFive update email address: ☒

ReachFive update profile: ☒

ReachFive request error:

When syncing the profile the job uses the “reach5ProfileFieldsJSON” Site preference in ‘ReachFive’ group to determine which fields need to be synched and mapping between SFCC and ReachFive profile fields.

The job uses such services:

- reachfive.rest.auth – to get management access token which is used in other reachFive API calls
- reachfive.updateprofile.put – to update email and profile fields
- reachfive.verifyemail.post – to send the verification email for updated email address

This job is scheduled to run every 5 minutes by default.

All issues and exceptions which occur during job running are added to job log.

salesforce | Sandbox - reachfive01 | RefArch | Merchant Tools | Administration | Storefront | Toolkit | (oleh h)

ReachFive-Synchronization

General | Schedule and History | Resources | Job Steps | Failure Handling | Notification

☒ Enabled

Active

Trigger

Recurring Interval

From* 4/26/2020 12:00 am To

Run Time

Every

Amount* 5 Interval* Minutes

Run Only On These Days: ☐ Monday ☐ Tuesday ☐ Wednesday ☐ Thursday ☐ Friday ☐ Saturday ☐ Sunday

Job History

ID	Execution Scope	Status	Start Time	End Time	Duration	Log File
ReachFive-Synchronization	RefArch	OK	4/30/2020 8:30 am by yann	4/30/2020 8:30 am	0:00:00	Log File
ReachFiveSynchronization	RefArch	OK	4/30/2020 8:30 am	4/30/2020 8:30 am	0:00:00	

2.2 Use Cases

The following use cases are managed by the int_reachfive and int_reachfive_sg cartridges :

SSO:

- 1 Registration of a new customer through a ReachFive Signup widget
- 2 Login of an existing SFCC Customer through ReachFive. The SFCC Customer account has been created through the native SiteGenesis account creation form.
- 3 Login of an existing ReachFive customer. The SFCC customer is created and linked to the ReachFive Profile
- 4 Social Account Activation/Deactivation, in My Account the customer using a widget can connect more than one social account to his profile.

Social Login:

- 1 Registration of a new customer through a ReachFive supported social connect provider
- 2 Login of a customer who has previously created his SFCC Customer account through a ReachFive supported social connect provider
- 3 Login of an existing SFCC Customer through ReachFive. The SFCC Customer account has been created through the native SiteGenesis account creation form.
- 4 Login of an existing SFCC Customer through ReachFive. The SFCC Customer account has been created through a non ReachFive social connect provider (i.e through the social connect provider natively proposed by the SiteGenesis)
- 5 Social Account Activation/Deactivation, in My Account the customer using a widget can connect more than one social account to his profile.

ReachFive Profiles Synchronization:

- 1 Update the email address

- 2 Send verification email for updated email address
- 3 Update customer profile and addresses

Transition Mode:

- 1 Login on SFCC form and silently register in Reachfive
- 2 On first login after silent register, you can use the Reachfive login component
- 3 Reset Password and update profile allow a double update in SFCC and in Reachfive
- 4 Send verification email for updated email address
- 5 Update customer profile and addresses

The Customer Registration and login through ReachFive works whether the Customer logs in / registers while in the checkout process or in the standard account registration / login page.

2.3 Limitations, Constraints

There is currently no limitations nor constraints identified with this component at this time.

2.4 Compatibility

The latest release is compatible with SFCC 18.3.

2.5 Privacy

The `int_reachfive_sg` cartridge access the SFCC Customer.ID and the SFCC Site.ID properties values and stores this information within the ReachFive platform. Moreover, during the social connect process, ReachFive stores the social data (coming from the social provider), that the Customer has willingly accepted to share, and makes this data available to the merchant.

During the registration / login process :

If the data is available through the social provider and if there is no previous value stored within SFCC for these properties, the following properties are set on the Customer profile on the SFCC side:

- Profile.lastName
- Profile.firstName
- Profile.email
- Profile.birthday

The following Credentials properties will be set during the registration / login process through ReachFive :

- Credentials.authenticationProviderID
 - The value set is a custom value chosen by the merchant and manageable in the ReachFive Site Preferences of the SFCC Business Manager.
- Credentials.externalID
 - This property stores the ReachFive user ID.

2.6 Security

The social login process has to happen on the SFCC side, which implies a risk that a malicious attacker will attempt to tamper with the data sent from the client to the server. In the next

paragraph is the explanation of the authentication process and how this prevents from security issues

Step 1: User authentication process

The user authenticates via email/password or social login. This process must be at least partially implemented using the suitable ReachFive's SDK depending on the client platform (Web, mobile...).

ReachFive's Identity SDKs allow different authentication flows. For server-side authentication, you must use a code response type, and set your login callback URL in the `redirectUri` attribute (**The URI must be whitelisted in the "Allowed Callback URLs" field of your ReachFive's account settings**).

For security reasons, you have to list all callback URLs in the Allowed Callback URLs field of your ReachFive console.

You can find below an example using the Identity SDK for Web.

```
sdkUiClient.showAuth({
  container: 'auth-container',
  auth: {
    responseType: 'code', // This is the default value when "redirectUri" attribute is set.
    redirectUri: 'https://www.example.com/login/callback'
  }
});
```

Step 2: Handle the Authorization Response

After the authentication process, ReachFive responds to your application by redirecting the user to the callback URL specified previously (`redirectUri` attribute).

If the authentication succeeded, then the response contains an authorization code. If not (e.g. when the user does not approve access to his social data), the response contains an error message. The authorization code or error message that is returned to your web server appears in the query string, as shown below.

An authorization code response:

<https://www.example.com/login/callback?code=A8sLD49d-IPcKyUwBaSm4oThfjp4>

An error response:

https://www.example.com/login/callback?error=access_denied

Step 3: Exchange authorization code for ID token

After the web server receives the authorization code, it can exchange the authorization code for an ID token using an HTTP request to the ReachFive's token endpoint.

The string `YOUR_REACHFIVE_DOMAIN` should be replaced with the domain of your ReachFive account and the string `YOUR_REACHFIVE_CLIENT_ID` should be replaced with your client ID of your ReachFive account.

```
POST /oauth/token HTTP/1.1
Host: https://YOUR_REACHFIVE_DOMAIN
Content-Type: application/x-www-form-urlencoded

code=A8sLD49d-IPcKyUwBaSm4oThfjp4
&client_id=YOUR_REACHFIVE_CLIENT_ID
&redirect_uri=https://www.example.com/login/callback
&grant_type=authorization_code
```

Success response example:

```
{
  "id_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1Ni...",
  "access_token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1Ni...",
  "token_type": "Bearer",
  "expires_in": 3600
}
```

Step 4: Retrieve user's profile data from ID Token

After your application obtains an `id_token`, you can use the token to retrieve the user's profile data by parsing it. An `id_token` is a JWT (JSON Web Token), that is, a cryptographically signed Base64-encoded JSON object. As you are communicating directly with ReachFive over HTTPS channel, it is normally not critical that you validate your `id_token` before parsing it. However most API libraries combine the validation with the work of decoding the base64 and parsing the JSON.

Example:

```
var decodedToken = jwt.verify(idToken, clientSecret);
```

Success example:

```
{
  "sub": "AVPw-jHcQG5c_BvJk9e_",
  "given_name": "John",
  "family_name": "Doe",
  "email": "john.doe@example.com"
}
```

3. Implementation Guide

3.1 Setup

3.1.1 Import Cartridge

Import the **int_reachfive** and **int_reachfive_sg** cartridge into the UXStudio Workspace.

- Open UXStudio
- Click File -> Import -> General -> Existing Projects Into Workspace
- Browse to the directory where you saved the ":int_reachfive_sg:int_reachfive" cartridge.
- Click Finish.
- Click OK when prompted to link the cartridge to the sandbox.

3.1.2 Import Site Preferences and Custom Objects

Import the system object customizations into the SFCC Business Manager.

- Log into the SFCC Business Manager.
- Click Administration -> Sites Development -> Site Import & Export
- Click the upload link or button in the "Import" section.
- Use the upload control to browse for the site_template.zip file
- Note: if you are configuring the SiteGenesis site, you can import the site_template.zip like this, otherwise you should follow these steps:
 - Cd in the metadata downloaded from bitbucket
 - Cd /site_template/sites
 - Rename the folder. SiteGenesis with the Id of your site
 - zip -r site_template.zip site_template
- Click Upload
- Select the site_template.zip file that was just uploaded
- Click Import

3.2 Configuration

3.2.1 Add the Reach Five Cartridge to your Storefront Cartridge Path

Append int_reachfive_sg to the effective cartridge path

- Log into the SFCC Business Manager.
- Click Administration -> Sites -> Manage Sites
- Select the desired site
- Click on the Settings tab.
- Append ":int_reachfive " to the "Cartridges" field.
- Click Apply

3.2.2 Configure Reach Five Site Preferences

Configure Reach Five Custom Preferences using the SFCC Business Manager

- Log into the SFCC Business Manager
- Select the desired site from the tabs across the top of the page.
- Click Site Preferences -> Custom Preferences
- Fill in the Site Preferences

ENTITY	DESCRIPTION
--------	-------------

cartridgeControllersName	Name of the cartridge where are controllers (i.e. app_storefront_controllers)
isReachFiveEnabled	If set to true, ReachFive is activated
isReach5ThemeActive	If set to true, default theme is activated (actually is not used currently)
reach5Domain	The domain name of the ReachFive environment
reach5ApiKey	The API key required to access the ReachFive Environment
reach5ClientSecret	The client secret key required to access the ReachFive environment
reachFiveProviderId	The name of the provider that will be used to populate the Credentials.authenticationProviderID properties value for the Customers who will register or log in on the SFCC storefront through ReachFive
isReachFastRegister	If “No”, pending the Social Login or SSO, if the user profile does not exist in SFCC database, the user is redirected to a registration page, with pre-filled but editable fields. If “Yes”, pending the Social Login or SSO, if the user profile does not exist in SFCC Database, it is created using the information from the Social Login or SSO.
isReachFiveLoginAllowed	If “No” the Social Login scenario is enabled: the social button are added to login page, registration page, checkout page, personal data page. If “Yes” the Single Sign On scenario is enable: the reach five login / signup widget will be present in login page, registration page, checkout page. In Personal Data page, there will be the social account widget that permits to the customer to enable or disable social account related to his profile.
reach5ManagementScope	Space-delimited list of Management permissions. The default value is “manage:users read:users”
reach5ProfileFieldsJSON	ReachFive JSON allows the flexible configuration for updating profiles fields without code changing. It determines which fields need to be synched and set mapping between SFCC and ReachFive profile fields. It consists of such Objects: profile, address, consents, custom_fields. Each Object matches to appropriate ReachFive Profile Object. Here are used key-value pairs to map SFCC and ReachFive fields. Key is SFCC profile attribute and value is ReachFive field. Also is possible to use custom Profile attribute in key. It should contain ‘custom.’ prefix. To send new attribute you need to add it to appropriate Object in such format: (custom.)SfccAttribute: reach_five_field For ‘address’ JSON Object are used SFCC CustomerAddress attributes for other Objects are used SFCC Profile attributes

	<p>JSON Example:</p> <pre> { "profile": { "firstName": "given_name", "lastName": "family_name", "birthday": "birthdate", "gender": "gender", "companyName": "company", "phoneHome": "phone_number" }, "address": { "ID": "title", "fullName": "recipient", "address1": "street_address", "city": "locality", "postalCode": "postal_code", "stateCode": "region", "countryCode": "country", "phone": "phone_number" }, "consents": { "custom.isNewsletter": "newsletter" }, "custom_fields": { "custom.cardNo": "loyalty_card_number" } } </pre>
isReachFiveTransitionActive	<p>Enable the transition state of the Reach Five integration. It means that customers will login on SFCC form first (this login will create a new Reachfive profile and link it to customer profile).</p> <p>After successful login, next time a customer will have ReachFive login form on the same device.</p>
reachFiveTransitionCookieDuration	<p>Duration of the "Transition" cookie in days. Required to identify the user already migrated to ReachFive. It means how long the ReachFive login form will automatically display for this device during this option enable.</p>
reachFiveLoginCookieDuration	<p>Duration of the "Login" cookie in days. It defines how much time you'll check the reachfive cookie to login automatically again.</p>
isReachFiveSessionForcedAuth	<p>Automatic user authentication from any page. This enables automatic authentication from any page with a reachfive long session cookie.</p>

3.2.3 Configure Reach Five Services

Services should be imported from site_template.zip with other metadata.

Configure services credentials:

- 1 Log into the SFCC Business Manager.
- 2 Navigate to Administration -> Operations -> Services
- 3 Click on the Credentials tab.
- 4 If Credentials URL contains 'demandware.og4.me' domain then replace it to current reach5Domain. If URL contains '{reach5Domain}' domain then it will be replaced automatically with 'reach5Domain' Site Preference value.

Service Credentials

Select All	Name	URL	User
<input type="checkbox"/>	reachfive.gettoken.get	https://bbd.reach5.net/api/v1/access_token	admin
<input type="checkbox"/>	reachfive.logout.get.credentials	https://{reach5Domain}/identity/v1/logout	admin
<input type="checkbox"/>	reachfive.rest.auth.credentials	https://bbd.reach5.net/oauth/token	admin
<input type="checkbox"/>	reachfive.setcustomfields.post.credentials	https://bbd.reach5.net/api/v2/users/{user_id}?fields=external_id	admin
<input type="checkbox"/>	reachfive.signup.post.credentials	https://{reach5Domain}/identity/v1/signup	admin
<input type="checkbox"/>	reachfive.update.profile.post.credentials	https://{reach5Domain}/identity/v1/update-profile	admin
<input type="checkbox"/>	reachfive.updatepassword.post.credentials	https://{reach5Domain}/identity/v1/update-password	admin
<input type="checkbox"/>	reachfive.updateprofile.put.credentials	https://{reach5Domain}/api/v2/users/{user_id}	
<input type="checkbox"/>	reachfive.userinfo.get.credentials	https://{reach5Domain}/identity/v1/userinfo?fields=id,consents	
<input type="checkbox"/>	reachfive.users.get	/api/v2/users/{user_id}	admin
<input type="checkbox"/>	reachfive.verifyemail.post.credentials	https://{reach5Domain}/api/v2/users/{user_id}/verify-email	

Configure services logging.

- 1 Log into the SFCC Business Manager.
- 2 Navigate to Administration -> Operations -> Services
- 3 Click on the Services tab and open needed service.
- 4 Set 'Communication Log Enabled' flag
- 5 Set 'Log Name Prefix' to 'reachfive'. The service responses and requests will be logged in the files with 'service-reachfive-' prefix. To see them navigate to Business Manager -> Administration -> Site Development -> Development Setup -> Security Log Files

salesforce Sandbox - reachfive01 RefArch Merchant Tools Administration Store

Administration > Operations > Services > reachfive.updateprofile.put - Details

reachfive.updateprofile.put

Fields with a red asterisk (*) are mandatory. Click Apply to save the details. Click Reset to revert to the last saved state.

Name:*	reachfive.updateprofile.put
Type:	HTTP
Enabled:	<input checked="" type="checkbox"/>
Service Mode:	Live
Log Name Prefix:	reachfive
Communication Log Enabled:	<input checked="" type="checkbox"/>
Force PRD Behavior in Non-PRD Environments:	<input type="checkbox"/>
Profile:	reachfive.profile
Credentials:	reachfive.updateprofile.put.credentials

3.2.4 Configure Reach Five Jobs

Jobs should be imported from site_template.zip with other metadata.

Here is used 'ReachFive-Synchronization' job wich synchronizes profile data from SFCC to ReachFive.

Set Scope:

- 1 Log into the SFCC Business Manager.
- 2 Navigate to Administration -> Operations -> Jobs
- 3 Click on the 'Job Steps' tab.
- 4 Set needed site in the Scope

salesforce Sandbox - reachfive01 RefArch Merchant Tools Administration Storefront Toolkit

Administration / Operations / Jobs /

ReachFive-Synchronization

General Schedule and History Resources Job Steps Failure Handling Notification

Job Parameters 0

Scope: RefArch

ReachFiveSynchronization

+

Job Schedule (Job is scheduled to run every 5 minutes by default):

- 1 Log into the SFCC Business Manager.
- 2 Navigate to Administration -> Operations -> Jobs
- 3 Click on the 'Schedule and History' tab.
- 4 Set needed schedule or disable it

3.3 Custom Code

3.3.1 Custom Code. Controllers.

[app_storefront_controllers/cartridge/controllers/Account.js](#)

The code to add is:

```
80         var hasEditSucceeded = false;
81         var Customer = app.getModel('Customer');
82
83         if (!Customer.checkUserName()) {
84             app.getForm('profile.customer.email').invalidate();
85             isProfileUpdateValid = false;
86         }
87
88         if (app.getForm('profile.customer.email').value() !== app.getForm('profile.customer.emailconfirm').value()) {
89             app.getForm('profile.customer.emailconfirm').invalidate();
90             isProfileUpdateValid = false;
91         }
92
93         if (!app.getForm('profile.login.password').value()) {
94             app.getForm('profile.login.password').invalidate();
95             isProfileUpdateValid = false;
96         }
97
98         if (isProfileUpdateValid) {
99
100             hasEditSucceeded = Customer.editAccount(app.getForm('profile.customer.email').value(), app.getForm('profile.login.password').value())
101             if (!hasEditSucceeded) {
102                 app.getForm('profile.login.password').invalidate();
103                 isProfileUpdateValid = false;
104             }
105         }
106
107         var hasEditSucceeded = false;
108         var Customer = app.getModel('Customer');
109
110         if (customer.externallyAuthenticated) {
111             hasEditSucceeded = Form.get('profile.customer').copyTo(customer.profile);
112         } else {
113             if (!Customer.checkUserName()) {
114                 app.getForm('profile.customer.email').invalidate();
115                 isProfileUpdateValid = false;
116             }
117
118             if (app.getForm('profile.customer.email').value() !== app.getForm('profile.customer.emailconfirm').value()) {
119                 app.getForm('profile.customer.emailconfirm').invalidate();
120                 isProfileUpdateValid = false;
121             }
122
123             if (!app.getForm('profile.login.password').value()) {
124                 app.getForm('profile.login.password').invalidate();
125                 isProfileUpdateValid = false;
126             }
127
128             if (isProfileUpdateValid) {
129                 var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
130                 var profileRequestObj = reachFiveHelper.getProfileRequestObjFromForm(app.getForm('profile.customer'));
131
132                 hasEditSucceeded = Customer.editAccount(app.getForm('profile.customer.email').value(), app.getForm('profile.login.password').value())
133
134                 if (!hasEditSucceeded) {
135                     app.getForm('profile.login.password').invalidate();
136                     isProfileUpdateValid = false;
137                 } else if (reachFiveHelper.isReachFiveTransitionActive()) {
138                     reachFiveHelper.updateReachFiveProfile(profileRequestObj);
139                 }
140             }
141         }
142     }
143 }
```

```
-----
if(customer.externallyAuthenticated) {
    hasEditSucceeded = Form.get('profile.customer').copyTo(customer.profile);
} else {
    -----
    -----
    var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
    var profileRequestObj = reachFiveHelper.getProfileRequestObjFromForm(app.getForm('profile.customer'));
    -----
} else if (reachFiveHelper.isReachFiveTransitionActive()) {
    reachFiveHelper.updateReachFiveProfile(profileRequestObj);
}
```

```

114         var isProfileUpdateValid = true;
115         var hasEditSucceeded = false;
116         var Customer = app.getModel('Customer');

117
118         if (!Customer.checkUserName()) {
119             app.getForm('profile.customer.email').invalidate();
120         }
121     }
122 }
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999

```

The code to add is:

```

-----
var email = app.getForm('profile.customer.email').value();
var newPassword = app.getForm('profile.login.newpassword').value();
var oldPassword = app.getForm('profile.login.currentpassword').value();
-----
// Update R5 password if customer reachfive profile exist
var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
var reachFiveService = require('*/cartridge/scripts/interfaces/reachFiveInterface');
if (hasEditSucceeded) {
    var customerReachFiveProfile = reachFiveHelper.getCustomerReachFiveExtProfile(customer);
    if (customerReachFiveProfile) {
        var pwdChangeResult = reachFiveService.updatePassword(email, newPassword, oldPassword);
    }
}
-----
var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
reachFiveHelper.passwordUpdateManagementAPI(resettingCustomer.object.profile,
app.getForm('resetpassword.password').value());

```

app_storefront_controllers/cartridge/controllers/Address.js

```
70     success = false;
71   }
72
73   success = true;
74 },
75 edit: function () {
76   success = false;
77
78   // Creates R5 profile address if reachfive customer exist
79   var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
80   var reachFiveService = require('*/cartridge/scripts/interfaces/reachFiveInterface');
81   if (reachFiveHelper.isReachFiveTransitionActive()) {
82     var customerReachFiveProfile = reachFiveHelper.getCustomerReachFiveExtProfile(customer);
83     if (customerReachFiveProfile) {
84       var requestObj = {
85         'addresses': [
86           {
87             'id': session.forms.profile.address.addressid.value,
88             'address_type': 'billing',
89             'street_address': session.forms.profile.address.addressid.value,
90             'locality': session.forms.profile.address.city.value,
91             'postal_code': session.forms.profile.address.postal.value,
92             'region': session.forms.profile.address.states.state.value,
93             'country': session.forms.profile.address.country.value,
94             'recipient': session.forms.profile.address.firstname.value + ' ' + session.forms.profile.address.lastname.value,
95             'phone_number': session.forms.profile.address.phone.value
96           }
97         ]
98       };
99       var profileChangeResult = reachFiveService.updateProfileIdentityAPI(requestObj);
100     }
101   }
102   success = true;
103 },
104 edit: function () {
```

The code to add is:

```
-----
// Creates R5 profile address if reachfive customer exist
var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
var reachFiveService = require('*/cartridge/scripts/interfaces/reachFiveInterface');
if (reachFiveHelper.isReachFiveTransitionActive()) {
  var customerReachFiveProfile = reachFiveHelper.getCustomerReachFiveExtProfile(customer);
  if (customerReachFiveProfile) {
    var requestObj = {
      'addresses': [
        {
          'id': session.forms.profile.address.addressid.value,
          'address_type': 'billing',
          'street_address': session.forms.profile.address.addressid.value,
          'locality': session.forms.profile.address.city.value,
          'postal_code': session.forms.profile.address.postal.value,
          'region': session.forms.profile.address.states.state.value,
          'country': session.forms.profile.address.country.value,
          'recipient': session.forms.profile.address.firstname.value + ' ' +
            session.forms.profile.address.lastname.value,
          'phone_number': session.forms.profile.address.phone.value
        }
      ]
    };
    var profileChangeResult = reachFiveService.updateProfileIdentityAPI(requestObj);
  }
}
```

```

79     }
80     try {
81         Address.update(request.httpParameterMap.addressid.value, session.forms.profile.address);
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104     }
105     try {
106         Address.update(request.httpParameterMap.addressid.value, session.forms.profile.address);
107         // Update R5 profile address if reachfive customer exist
108         var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
109         var reachFiveService = require('*/cartridge/scripts/interfaces/reachFiveInterface');
110         if (reachFiveHelper.isReachFiveTransitionActive()) {
111             var customerReachFiveProfile = reachFiveHelper.getCustomerReachFiveExtProfile(customer);
112             if (customerReachFiveProfile) {
113                 var requestObj = {
114                     'addresses': [
115                         {
116                             'id': session.forms.profile.address.addressid.value,
117                             'address_type': 'billing',
118                             'street_address': session.forms.profile.address.addressid.value,
119                             'locality': session.forms.profile.address.city.value,
120                             'postal_code': session.forms.profile.address.postal.value,
121                             'region': session.forms.profile.address.states.state.value,
122                             'country': session.forms.profile.address.country.value,
123                             'recipient': session.forms.profile.address.firstname.value + ' ' + session.forms.profile.address.lastname.value,
124                             'phone_number': session.forms.profile.address.phone.value
125                         }
126                     ]
127                 };
128                 var profileChangeResult = reachFiveService.updateProfileIdentityAPI(requestObj);
129             }
130         }
131         success = true;
132     } catch (e) {
133         success = false;

```

The code to add is:

```

// Update R5 profile address if reachfive customer exist
var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
var reachFiveService = require('*/cartridge/scripts/interfaces/reachFiveInterface');
if (reachFiveHelper.isReachFiveTransitionActive()) {
    var customerReachFiveProfile = reachFiveHelper.getCustomerReachFiveExtProfile(customer);
    if (customerReachFiveProfile) {
        var requestObj = {
            'addresses': [
                {
                    'id': session.forms.profile.address.addressid.value,
                    'address_type': 'billing',
                    'street_address': session.forms.profile.address.addressid.value,
                    'locality': session.forms.profile.address.city.value,
                    'postal_code': session.forms.profile.address.postal.value,
                    'region': session.forms.profile.address.states.state.value,
                    'country': session.forms.profile.address.country.value,
                    'recipient': session.forms.profile.address.firstname.value + ' ' +
session.forms.profile.address.lastname.value,
                    'phone_number': session.forms.profile.address.phone.value
                }
            ]
        };
        var profileChangeResult = reachFiveService.updateProfileIdentityAPI(requestObj);
    }
}

```

app_storefront_controllers/cartridge/controllers/Login.js

32	var orderTrackForm = app.getForm('ordertrack');	32	var orderTrackForm = app.getForm('ordertrack');
33		33	
34		34	var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
35		35	var reachFiveConversion = reachFiveHelper.getReachFiveConversionMute();
36		36	
37	var loginView = app.getView('Login',{	37	var loginView = app.getView('Login',{
38	RegistrationStatus: false	38	RegistrationStatus: false,
39		39	ReachFiveConversion: reachFiveConversion
40	});	40	});
41		41	
42		42	if (!empty(request.httpParameterMap.verification_code.stringValue)) {
43		43	session.custom.passwordReset = true;
44		44	} else {
45		45	session.custom.passwordReset = false;
46		46	}
47		47	
48	loginForm.clear();	48	loginForm.clear();
49	oauthLoginForm.clear();	49	oauthLoginForm.clear();
50	orderTrackForm.clear();	50	orderTrackForm.clear();
51		51	
52	if (customer.registered) {	52	if (customer.registered) {
53	loginForm.setValue('username', customer.profile.credentials.login);	53	loginForm.setValue('username', customer.profile.credentials.login);
54	loginForm.setValue('rememberme', true);	54	loginForm.setValue('rememberme', true);
55	}	55	}
56		56	
57	if (content) {	57	if (content) {
58	pageMeta.update(content);	58	pageMeta.update(content);
59	}	59	}
60		60	
61		61	// Save return URL in session.
62		62	if (request.httpParameterMap.original.submitted) {
63		63	session.custom.TargetLocation = request.httpParameterMap.original.value;
64		64	}
65		65	

The code to add is:

```

-----
var reachFiveHelper = require('*/cartridge/scripts/helpers/reachFiveHelper');
var reachFiveConversion = reachFiveHelper.getReachFiveConversionMute();
-----
ReachFiveConversion: reachFiveConversion
-----
if (!empty(request.httpParameterMap.verification_code.stringValue)) {
    session.custom.passwordReset = true;
} else {
    session.custom.passwordReset = false;
}
-----
// Save return URL in session.
if (request.httpParameterMap.original.submitted) {
    session.custom.TargetLocation = request.httpParameterMap.original.value;
}

```

3.3.2 Custom Code. Templates.

app_storefront_core/cartridge/templates/default/account/accountoverview.isml

12		<iscontentasset aid="account-landing"/>
13		
14		
15	</isdecorate>	
16		
17		
18		
19	</isdecorate>	

12		<iscontentasset aid="account-landing"/>
13		
14		
15	<isif condition="\${dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveEnabled')}">	
16	<isinclude url="\${URLUtils.http('ReachFiveController-afterPrefillSave')}" />	
17	</isif>	
18		
19	</isdecorate>	

The code to add is:

```

-----
<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveEnabled')}">
    <isinclude url="${URLUtils.http('ReachFiveController-afterPrefillSave')}" />
</isif>

```

[app_storefront_core/cartridge/templates/default/account/login/logininclude.isml](#)

```

3 <div class="login-box login-account">
4
5 <!--${Resource.msg('account.login.logininclude.customersheader','account',null)}
6 <div class="dialog-required"> <span class="required-indicator">#9226; <em>${Resource.msg('global.requiredfield','locale',null)}</em></span></div>
7 </h2>
8
9 <div class="login-box-content returning-customers clearfix">
10
11 <!--if condition="${pdict.CurrentSession.customer.externallyAuthenticated}">
12
13 <p>${Resource.msg('account.login.logininclude.registered','account',null)}</p>
14
15 <form action="${URLUtils.httpsContinue()}" method="post" class="clearfix" id="${pdict.CurrentForms.login.htmlName}">
16
17 <!--comment>Login Unsuccessful Error Message</comment>
18 <!--if condition="${pdict.TempCustomer != null && pdict.TempCustomer.profile != null && pdict.TempCustomer.profile.credentials.locked}">
19 <div class="error-form">${Resource.msg('account.login.logininclude.locked','account',null)}</div>
20
21 </form>
22 </!--if>
23
24 <div class="login-oauth">
25 <!--include template="account/login/oauthlogininclude"/>
26
27 </div class="login-box login-account">
28
29 <!--if condition="${!empty(pdict.errorMessage)}">
30 <div class="error-form">${pdict.errorMessage}</div>
31 </!--if>
32
33 <!--${Resource.msg('account.login.logininclude.customersheader','account',null)}
34 <div class="dialog-required"> <span class="required-indicator">#9226; <em>${Resource.msg('global.requiredfield','locale',null)}</em></span></div>
35 </h2>
36
37 <div class="login-box-content returning-customers clearfix">
38
39 <!--if condition="${(dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveLoginAllowed') &&
40 pdict.ReachFiveConversion) ||
41 (pdict.ShowStandardLoginToLinkAccount != null && pdict.ShowStandardLoginToLinkAccount))}">
42 <p>${Resource.msg('account.login.logininclude.registered','account',null)}</p>
43
44 <form action="${URLUtils.httpsContinue()}" method="post" id="${pdict.CurrentForms.login.htmlName}"
45 class="clearfix" &{dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveTransitionActive')} ? 'reach5-login-ajax: ''">
46 <!--comment>Login Unsuccessful Error Message</comment>
47 <!--if condition="${pdict.TempCustomer != null && pdict.TempCustomer.profile != null && pdict.TempCustomer.profile.credentials.locked}">
48 <div class="error-form">${Resource.msg('account.login.logininclude.locked','account',null)}</div>
49
50 </form>
51 </!--if>
52
53 <!--include template="account/login/reachfiveincludebuttons" />
54
55 <div class="login-oauth">
56 <!--include template="account/login/oauthlogininclude"/>
57
58 </div>
59
60 </div>
61
62 </div>
63

```

The code to add is:

```
<isif condition="${!empty(pdct.errorMsg)}">
    <div class="error-form">${pdct.errorMsg}</div>
</isif>
-----
<isif condition="${!(dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveLoginAllowed') &&
pdct.ReachFiveConversion) ||
(pdct.ShowStandardLoginToLinkAccount != null && pdct.ShowStandardLoginToLinkAccount)}">
-----
<form action="${URLUtils.httpsContinue()}" method="post" id="${pdct.CurrentForms.login.htmlName}"
class="clearfix ${dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveTransitionActive') ?
'reach5-login-ajax': ''}">
-----
<isinclude template="account/login/reachfiveincludebuttons" />
```

[app_storefront_core/cartridge/templates/default/account/pt_account.isml](#)

```

62         <isinclude template="account/pt_account_VARS"/>
63     </div>
64
65 </body>
66 </html>
67
68 <isinclude template="account/pt_account_VARS"/>
69 </div>
70
71 <iscomment>Add Reachfive tag</iscomment>
72 <isinclude url="URLUtils.http('ReachFiveController-Init', 'disableSocialLogin', pdict.disableSocialLogin)"/>
73
74 </body>
75 </html>

```

The code to add is:

```
<iscomment>Add Reachfive tag</iscomment>
<include url="${URLUtils.http('ReachFiveController-Init', 'disableSocialLogin', pdict.disableSocialLogin)}"
/>
```


app_storefront_core/cartridge/templates/default/account/user/registration.isml

```
21         <h1>${Resource.msg('account.user.registration.createnew','account',null)}</h1>
22     </isif>
23
24     <form action="${URLUtils.httpsContinue()}" method="post" class="form-horizontal" id="RegistrationForm">
25
26         <fieldset>
102
103     </form>
104 </isif>
105 </isdecorate>
21
22     <h1>${Resource.msg('account.user.registration.createnew','account',null)}</h1>
23
24     <isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveEnabled')}">
25         <isinclude template="account/user/reachfiveregistrincluder" />
26     </iselse/>
27     <form action="${URLUtils.httpsContinue()}" method="post" class="form-horizontal" id="RegistrationForm">
28
29         <fieldset>
105
106     </form>
107 </isif>
108 </isif>
109 </isdecorate>
```

The code to add is:

```
-----
<isif condition="${dw.system.Site.getCurrent().getCustomPreferenceValue('isReachFiveEnabled')}">
    <isinclude template="account/user/reachfiveregistrincluder" />
</iselse/>
-----
</isif>
```

app_storefront_core/cartridge/templates/default/checkout/cart/pt_cart.isml

```
62         <isinclude template="account/pt_account_VARS"/>
63     </div>
64
65 </body>
66 </html>
62         <isinclude template="account/pt_account_VARS"/>
63     </div>
64
65 <iscomment>Add Reachfive tag</iscomment>
66 <isinclude url="${URLUtils.http('ReachFiveController-Init', 'disableSocialLogin', pdict.disableSocialLogin)}" />
67 </body>
68 </html>
69
```

The code to add is:

```
-----
<iscomment>Add Reachfive tag</iscomment>
<isinclude url="${URLUtils.http('ReachFiveController-Init', 'cart', '1')}" />
-----
```

app_storefront_core/cartridge/templates/default/components/footer/footer_UI.isml

```
32 var meta = "${pdict.CurrentPageMetaData.description}";
33 var keywords = "${pdict.CurrentPageMetaData.keywords}";
34 </script>
35
36 var meta = "${pdict.CurrentPageMetaData.description}";
37 var keywords = "${pdict.CurrentPageMetaData.keywords}";
38 </script>
39
40 <iscomment>Add Reachfive global tag</iscomment>
41 <isinclude url="${URLUtils.http('ReachFiveController-InitGlobal', 'state', request.httpURL.https().toString())}" />
```

The code to add is:

```
-----
<iscomment>Add Reachfive global tag</iscomment>
<isinclude url="${URLUtils.http('ReachFiveController-InitGlobal', 'state',
request.httpURL.https().toString())}" />
-----
```

app_storefront_core/cartridge/templates/default/components/header/htmlhead.isml

```
74 |  
75 <iscomment>Gather device-aware scripts</iscomment>  
76 <isinclude url="{URLUtils.url('Home-SetLayout')}" />  
77  
78 <!-- REACHFIVE -->  
79 <link rel="stylesheet" href="{URLUtils.staticURL('/css/reachfive.css')}" />
```

The code to add is:

```
-----  
<!-- REACHFIVE -->  
<link rel="stylesheet" href="{URLUtils.staticURL('/css/reachfive.css')}" />
```

The Int_reachfive_sg cartridge includes some english written text that may need to be translated depending on the storefront language. The text elements are stored in the reachfive.properties file.

3.4 External Interfaces

A couple of services will be called during the social registration / login process. The first call generates a token which is required by the second call.

Authentication

In order to access to the API, you will need to provide an access token to authenticate with the API server. That token will be required for all API requests.

You can acquire that token with the API endpoint described in the following section. Once you have acquired the API token, it may be provided preferably via an HTTP header.

Get an access token

Example Request

```
POST /oauth/token HTTP/1.1  
Host: https://YOUR_DOMAIN  
Content-Type: application/json  
{  
  "grant_type": "client_credentials",  
  "client_id": "YOUR_CLIENT_ID",  
  "client_secret": "YOUR_CLIENT_SECRET",  
  "scope": "read:users manage:users"  
}
```

Example Response

```
{  
  "access_token": "kGu...uLs",  
  "expires_in": 86400,  
  "token_type": "Bearer"  
}
```

Update User

Updates the specified user by setting the values of the object's properties passed. Any root properties (or custom fields) not provided will be left unchanged.

These are the user's attributes that can be updated at the root level:

```
external_id  
email  
email_verified  
phone_number  
custom_fields  
given_name  
middle_name  
family_name  
name  
nickname  
username  
birthdate  
gender
```

```
address
phone_number
picture
company
custom_fields
```

Example Request

```
PUT /api/v2/users/AVqvOB58Fg6nZfQ0ZqXt?fields=id,name,email,birthdate HTTP/1.1
Host: https://YOUR_DOMAIN
Authorization: Bearer YOUR_ACCESS_TOKEN
{
  "birthdate": "1981-10-13",
  "nickname": "Johnny"
}
```

Example Response

```
{
  "id": "AVqvOB58Fg6nZfQ0ZqXt",
  "name": "John Doe",
  "email": "johndoe@example.com",
  "birthdate": "1981-10-13"
}
```

3.5 Firewall Requirements

No firewall changes are required.

4. Data Storage

4.1 Availability

The ReachFive platform is designed to be up at any time.

4.2 Support

The following individuals should be contacted in case of defect fixes or if improvements are needed for this component :

NAME	ROLE	EMAIL
Guillaume Partenet	ReachFive Customer success Manager	guillaume@reach5.co
Gianluca Mirabelli	Salesforce Commerce Cloud Technical Architect	gmirabelli@salesforce.com
Aristide Okalla	Salesforce Commerce Cloud Technical Architect	aokalla@salesforce.com

5. User Guide

5.1 Roles, Responsibilities

The roles and responsibilities are shared among ReachFive and the merchant as described in the following table:

WHO	ROLE & RESPONSIBILITIES
ReachFive	<ul style="list-style-type: none">• Provide to the merchant the information required to connect to the ReachFive API : APISecret / APIKey• Provide and maintain the merchant's ReachFive platform.
Merchant	<ul style="list-style-type: none">• Integrate the Int_reachfive_sg cartridge in the code version of its site following the documentation.• Subscribe to the ReachFive service.

5.2 Storefront Functionality

The Int_reachfive_sg cartridge generates social connect buttons on the login and registration page of the SiteGenesis based site. If Providers are configured in reachFive console then their social connect buttons will be added to the form. The following screenshot illustrates what it looks like with the default ReachFive template :

Case SSO Screen Shot

My Account Login

RETURNING CUSTOMERS • REQUIRED

Log in



or

admin

[Don't remember your password?](#)

Log in

Do not have an account? [Sign up](#)

Register

Create Account

Sign up



or

First name

Last name

Email address

Password

Password confirmation

Sign up

Already have an account? [Log in](#)

Case Social Login

My Account Login

RETURNING CUSTOMERS • REQUIRED

If you are a registered user, please enter your email and password.

• Email Address

This field is required.

• Password

This field is required.

Login ☐ Remember Me

[Forgot Password?](#)



Login:

On the first register / login with a social connect provider, an OAuth provider popin window appears, asking for the user acceptance of data sharing.

On the acceptance of this terms, the user is logged in and then redirected to the My Account page or on the shipping address page, depending if the user were in the checkout process or not.

6. Known Issues

No known existing issues.

7. Failover and recovery process

During emergencies like platform or service down situations, merchants disable the cartridge features from custom site preferences. This will enable the users to connect or register through native SFCC functionalities.

8. Release History

VERSION	DATE	DATE CHANGES
16.1.0	August 4 2016	Initial Release
18.3.0	March 2018	ReachFive API version changed and add SSO functionalities
19.1.0	September 2019	<ul style="list-style-type: none">Fix security issues with the use of user management scope in API requestsSplitting cartridge into a core cartridge "int_reachfive" for core features and "int_reachfive_sg"
20.2.0	April 2020	Add ReachFive Profiles Synchronization Job
20.3.0	July 2020	Use synchronous web SDK: identity-ui@1.6.0, identity-core@1.15.0
20.4.1	February 2022	Transition mode added with all use cases to keep data up-to-date in both SFCC and Reachfive
20.4.2	February 2022	Add missing services & metaconfigurations
20.4.3	March 2022	Handle the native "remember me" feature for the R5 long session