accounts.verifyLogin REST

The accounts.verifyLogin API checks user profile information against the required schema fields and site policies to ensure that all the necessary data is in place. In site groups, this is done across multiple domains. If validation passes, the user's account information is returned. If not, an error is returned.

Note: Depending on your site package, this method may not return the user's account information, but will return an OK response instead.

Request URL

Where **<Data_Center>** is:

- `us1.gigya.com` - For the US data center.
- `eu1.gigya.com` - For the European data center.
- `au1.gigya.com` - For the Australian data center.
- `ru1.gigya.com` - For the Russian data center.
- `cn1.gigya-api.cn` - For the Chinese data center.

If you are not sure of your site's data center, see Finding Your Data Center.

> This method requires HTTPS.

Authorization Parameters

Each REST API request must contain identification and authorization parameters.

Some REST APIs may function without these authorization parameters, however, when that occurs, these calls are treated as **client-side** calls and all client-side rate limits will apply. In order to not reach client-side IP rate limits that may impact your implementation when using server-to-server REST calls, it is **Recommended Best Practice** to always sign the request or use a secret. A non-exhaustive list of REST APIs that this may apply to are as follows:

- accounts.login
- socialize.login
- accounts.notifyLogin
- socialize.notifyLogin
- accounts.finalizeRegistration
- accounts.linkAccounts

Please refer to the Authorization Parameters section for details.

Parameters

Required	Name	Type	Description
	uid	string	The unique user ID. This user ID should be used for login verification. See User.UID for more information.
	extraProfileFields	string	This parameter accepts a comma-separated list of additional social profile fields to retrieve, and return in the response. The current valid values are: **languages**, **address**, **phones**, **education**, **honors**, **publications**, **patents**, **certifications**, **professionalHeadline**, **bio**, **industry**, **specialties**, **work**, **skills**, **religion**, **politicalView**, **interestedIn**, **relationshipStatus**, **hometown**, **favorites**, **followersCount**, **followingCount**, **username**, **locale**, **verified**, **timezone**, **likes** and **samlData**.

Required	Name	Type	Description
	UID	string	The unique user ID. This user ID should be used for login verification. See User.UID for more information.
	include	string	A comma-separated list of fields to include in the response. The possible values are: **identities-active**, **identities-all**, **loginIDs**, **emails**, **profile**, **data**, **preferences**, **subscriptions**, **groups** and **irank**. The default is **profile** so if this parameter is not used, the response will return the Profile object.

	targetEnv	string	This parameter defines your client side environment, which in return determines the server response data fields. The default value of this parameter is "browser", which means that by default you receive cookie-related data in the response. If your client runs on a **mobile**: • If you are calling this method using a Mobile SDK since version 2.15.6, this parameter is automatically set to "*mobile*" (there is no need to set it manually). In any other case, you should set this parameter to be "*mobile*". • As a result of setting the parameter to "*mobile*" the server response data fields will include: *sessionToken* and *sessionSecret* (instead of cookie related data). In such case, you should send the *sessionToken* and *sessionSecret* to your mobile client. On your client side, call GSAPI.setSession (using the Mobile SDK) to save them in the app's storage.
	format	string	Determines the format of the response. The options are: • *json* (default) • *jsonp* - if the format is jsonp then you are required to define a *callback* method (see parameter below).
	callback	string	This parameter is *required* only when the *format* parameter is set to *jsonp* (see above). In such cases this parameter should define the name of the callback method to be called in the response, along with the jsonp response data.
	context	string/JSON	This parameter may be used to pass data through the current method and return it, unchanged, within the response.
	dontHandleScreenSet	Boolean	This parameter may be used in order to suppress the showing of screen-sets as a result of API calls. Default is **false**.
	httpStatusCodes	Boolean	The default value of this parameter is *false*, which means that the HTTP status code in Gigya's response is always 200 (OK), even if an error occurs. The error code and message is given within the response data (see below). If this parameter is set to *true*, the HTTP status code in Gigya's response would reflect an error, if one occurred.

Response Data

Field	Type	Description
errorCode	integer	The result code of the operation. Code '0' indicates success, any other number indicates failure. For a complete list of error codes, see the Error Codes table.
errorMessage	string	A short textual description of an error, associated with the errorCode, for logging purposes. This field will appear in the response only in case of an error.
errorDetails	string	This field will appear in the response only in case of an error and will contain the exception info, if available.
fullEventName	string	The full name of the event that triggered the response. This is an internally used parameter that is not always returned and **should not** be relied upon by your implementation.
callId	string	Unique identifier of the transaction, for debugging purposes.
time	string	The time of the response represented in ISO 8601 format, i.e., yyyy-mm-dd-Thh:MM:ss.SSSZ or
statusCode	integer	The HTTP response code of the operation. Code '200' indicates success. This property is deprecated and only returned for backward compatibility.
statusReason	string	A brief explanation of the status code. This property is deprecated and only returned for backward compatibility.
UID	string	The unique user ID. This user ID should be used for login verification. See User.UID for more information.
UIDSignature	string	The signature that should be used for login verification. See User.UID for more information.
signatureTimestamp	string	The GMT time of the response in UNIX time format, i.e., the number of seconds since Jan. 1st 1970. The timestamp should be used for login verification. See User.UID for more information.
created	string	The UTC time the account was created in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
createdTimestamp	integer	The UTC time the account was created in Unix time format including milliseconds (i.e., the number of seconds since Jan. 1st 1970 * 1000).
data	JSON object	Custom data. Any data that you want to store regarding the user that isn't part of the Profile obje ct.

emails	JSON object	The email addresses belonging to the user. This includes the following fields: • ***verified*** - an array of strings representing the user's verified email addresses • ***unverified*** - an array of strings representing the user's unverified email addresses. Note: ***emails*** must be specified explicitly in the ***include*** parameter in order to be included in the response.
identities	array	An array of Identity Objects, each object represents a user's social identity. Each Identity Object contains imported data from a social network that the user has connected to. **Note:** You must explicitly specify ***identities*** within the ***include*** parameter for them to be included in the response: ***identities-active*** , ***identities-all***, or ***identities-global*** to return only active identities, all identities of a site, or all identities of a site group, respectively. Be advised that if a user registers to your site using a **Social Identity**, then goes through the **Forgot Password** flow, a **Site Login** is added to their account, however, a **Site Identity** is *not*. A **Site Identity** can only be created when accounts.setAccountInfo is called on the user's account.
iRank	*integer*	*Influencer rank of the user. This property is deprecated and will always return 0.*
isActive	Boolean	Indicates whether the account is active. The account is active once the user creates it even without finalizing it. The account can be deactivated, but it will still be registered if the registration process has been finalized. If isActive==false the user cannot log in, however any currently active sessions remain valid.
isLockedOut	Boolean	Indicates whether the account is currently locked out. This parameter is not included in the response by default, and is not returned at all from accounts.search. If you wish to include it in a response, specify it as a value of the **include** parameter.
isRegistered	Boolean	Indicates whether the user is registered. The user is registered once his registration has been finalized.
isVerified	Boolean	Indicates whether the account email is verified.
lastLogin	string	The time of the last login of the user in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
lastLoginLocation	JSON object	The user's last login location. This includes the following fields: • **country** - a string representing the two-character country code. • **state** - a string representing the state, where available. • **city** - a string representing the city name. • **coordinates** - an object containing: • **lat** - a double representing the latitude of the center of the city. • **lon** - a double representing the longitude of the center of the city.
lastLoginTimestamp	integer	The UTC time of the last login of the user in Unix time format including milliseconds (i.e., the number of seconds since Jan. 1st 1970 * 1000).
lastUpdated	string	The UTC time when user profile, preferences, or subscriptions data was last updated (either full or partial update) in ISO 8601 format, e.g., "2017-07-16T19:20:30Z".
lastUpdatedTimestamp	integer	The UTC time when the last update of the object occurred (either full or partial update) in Unix time including milliseconds, based on when the 'lastUpdated', 'Report AccountsFirstLogin' or 'AccountsReturnedLogin' events are fired.
loginIDs	JSON object	The user's login identifiers. This includes the following fields: • ***username*** - a string representing the username • ***emails*** - an array of strings representing email addresses • ***unverifiedEmails*** - an array of strings representing email addresses that were not validated Note: ***loginIDs*** must be specified explicitly in the ***include*** parameter in order to be included in the response.
loginProvider	string	The name of the provider that the user used in order to login.
oldestDataUpdated	string	The UTC time when the oldest data of the object was refreshed in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
oldestDataUpdatedTimestamp	integer	The UTC time when the oldest data of the object was refreshed in Unix time format including milliseconds (i.e., the number of seconds since Jan. 1st 1970 * 1000).

password	JSON object	The user's **Site** account password details. Includes the following: • ***hash*** - the hashed password • ***hashSettings*** - object includes: ◦ **algorithm** - Represents the hash algorithm used to encrypt the password. ◦ **rounds** - Represents the number of iterations to perform the hashing. ◦ **salt** - Represents the BASE64 encoded value of the salt. ◦ **format** - Represents the template for merging clear-text passwords. This is only returned if the **pwHashFormat** parameter was set during account import and until the user's first login to Gigya (when the user's password is rehashed per the site's settings). See the RaaS Import Guide for additional information.
UIDSignature	*string*	***This property is deprecated in server to server REST calls!*** *The signature that should be used for login verification. See User.UID for more information.*
signatureTimestamp	*string*	***This property is deprecated in server to server REST calls!*** *The GMT time of the response in UNIX time format, i.e., the number of seconds since Jan. 1st 1970. The timestamp should be used for login verification. See User.UID for more information.*
phoneNumber	string	The Phone Number ID, if the account uses Phone Number Login.
preferences	Preferences object	The user's preferences information as described in the Preferences Object. To have this data returned in the response it must be specifically requested using the **include** parameter.
profile	Profile object	The user's profile information as described in the object. The ***profile*** is returned in the response by default, but if the ***include*** parameter is used to specify other fields that should be provided in the response, the ***profile*** must also be specified explicitly in the ***include*** parameter.
rbaPolicy	JSON object	The current RBA Policy defined for the specified user. Properties include: • **riskPolicy** - Determines the rule set from the defined rulesSets configured in accounts.rba.setPolicy or one of the default policies. • **riskPolicyLocked** - Determines whether the user can change their own riskPolicy. If true, only an admin can change the user's riskPolicy.
registered	string	The UTC time when the ***isRegistered*** parameter was set to true in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
registeredTimestamp	string	The GMT time when the ***isRegistered*** parameter was set to true in UNIX time format, including milliseconds.
regSource	string	A string representing the source of the registration. Can be used to set varying destination pages in accounts.setPolicies.
socialProviders	string	A comma-separated list of the names of the providers to which the user is connected/logged in.
subscriptions	Subscriptions Object	The user's subscription information.
userInfo	*User object*	*The Gigya User object. This property is deprecated and should not be relied upon.*
verified	string	The UTC time when the ***isVerified*** parameter was set to true in ISO 8601 format, e.g., "1997-07-16T19:20:30Z".
verifiedTimestamp	string	The GMT time when the ***isVerified*** parameter was set to true in Unix time format including milliseconds (i.e., the number of seconds since Jan. 1st 1970 * 1000).

A field that does not contain data will not appear in the response.

Response Example

```
{
  "sessionInfo": {
    "cookieName": "gac_2_ddxTpQZ_zGiuCsCePVKC6bZcBp_qD-pjql",
    "cookieValue": "VC1_739B3B4AD534B6F62AHNld3Knl98Q_vGL5_SxwA=="
  },
  "UID": "e862a450214c46b3973ff3c8368d1c7e",
  "UIDSignature": "iwPwRr3oDmbb8hhTeoO5JHTrc2Y=",
  "signatureTimestamp": "1344415327",
  "loginProvider": "site",
  "isRegistered": true,
  "registeredTimestamp": 1344415327000,
  "registered": "2012-08-08T08:42:07Z",
  "isActive": true,
  "isVerified": true,
  "verifiedTimestamp": 1344413279133,
  "verified": "2012-08-08T08:07:59.133Z",
  "socialProviders": "site",
  "profile": {
    "email": "Joe@hotmail.com",
    "firstName": "Joe",
    "lastName": "Smith",
    "age" : "31",
    "gender" : "m",
    "country" : "US"
  },
  "created": "2012-08-08T08:07:59.128Z",
  "createdTimestamp": 1344413279128,
  "lastLogin": "2012-08-08T08:42:07Z",
  "lastLoginTimestamp": 1344415327000,
  "lastUpdated": "2012-08-08T08:07:59.133Z",
  "lastUpdatedTimestamp": 1344413279133,
  "oldestDataUpdated": "2012-08-08T08:07:59.133Z",
  "oldestDataUpdatedTimestamp": 1344413279133,
  "statusCode": 200,
  "errorCode": 0,
  "statusReason": "OK",
  "callId": "8fb3eaf37a424cae8c3e6fe3f53cc177",
  "time": "2015-03-22T11:42:25.943Z"
}
```

```
{
  "UID": "e862a450214c46b3973ff3c8368d1c7e",
  "UIDSignature": "iwPwRr3oDmbb8hhTeoO5JHTrc2Y=",
  "signatureTimestamp": "1344415327",
  "loginProvider": "site",
  "isRegistered": true,
  "registeredTimestamp": 1344415327000,
  "registered": "2012-08-08T08:42:07Z",
  "isActive": true,
  "isVerified": true,
  "verifiedTimestamp": 1344413279133,
  "verified": "2012-08-08T08:07:59.133Z",
  "socialProviders": "site",
  "profile": {
    "email": "Joe@hotmail.com",
    "firstName": "Joe",
    "lastName": "Smith",
    "age" : "31",
    "gender" : "m",
    "country" : "US"
  },
  "created": "2012-08-08T08:07:59.128Z",
  "createdTimestamp": 1344413279128,
  "lastLogin": "2012-08-08T08:42:07Z",
  "lastLoginTimestamp": 1344415327000,
  "lastUpdated": "2012-08-08T08:07:59.133Z",
  "lastUpdatedTimestamp": 1344413279133,
  "oldestDataUpdated": "2012-08-08T08:07:59.133Z",
  "oldestDataUpdatedTimestamp": 1344413279133,
  "statusCode": 200,
  "errorCode": 0,
  "statusReason": "OK",
  "callId": "8fb3eaf37a424cae8c3e6fe3f53cc177",
  "time": "2015-03-22T11:42:25.943Z"
}
```

Errors

Gigya defines specific error codes and messages that are used with the Accounts API. These errors are returned with the APIs, indicating that some information is incorrect or missing.

This section describes the errors that are related to this API, the reasons for each error, and the expected next step.

- **Account pending registration** (error code 206001) - returned when the registration process has not been finalized, or the schema defines fields as required and one or more of these fields were not passed in the registration proccess. The expected next step is: if the schema defines fields that are required and one or more of these fields are missing from the user Profile or data, call accounts.setAccountInfo. If the registration process has not been finalized, call accounts.finalizeRegistration.
- **Account pending verification** (error code 206002) - returned when the account has already been verified, and a user tries to log in with a loginID (usually an email address) that we have not yet verified that actually belongs to this person. When the accountOptions policy states that ***verifyEmail*** is "true", the account must be validated by using the available email addresses. When the policy states that ***allowUnverifiedLogin*** is "false", users are not allowed to login before they have verified their emails. So, in this case, when a user tries to login, and his account has not been verified yet, and ***verifyEmail*** is "true" in the policy and ***allowUnverifiedLogin*** is "false" in the policy, the "Account pending verification" error is returned. The expected next step is: call accounts.resendVerificationCode to resend a

validation email to the unverified addresses associated with the account. The email format is according to the templates defined in the policy.

- **Login Failed Captcha Required** (error code 401020) - returned when login is attempted and the CAPTCHA threshold has been reached. The CAPTCHA threshold is set in the site Policies (*security .captcha.failedLoginThreshold* policy).
- **Login Failed Wrong Captcha** (error code 401021) - returned when login is attempted and the CAPTCHA threshold has been reached and the provided CAPTCHA text is wrong. The CAPTCHA threshold is set in the site Policies (*security .captcha.failedLoginThreshold* policy).
- **Old password used** (error code 401030) - returned when login is attempted with a password that doesn't match the current password but does match the previous one. The server will return this error with the message saying that "the password was modified on" the date when the current password was set.
- **Account disabled** (error code 403041) - returned when a user tries to login and the account is disabled.
- **Invalid loginID** (error code 403042) - returned when a user tries to perform an action that requires a login identifier (username or email) and the login ID doesn't exist in our accounts database. It is also returned if the password that is passed in the API is incorrect.
- **Login identifier exists** (error code 403043) - returned when email is defined as the ***loginIdentifier*** in the ***accountOptions*** policy, and the email address received from the provider exists in the system but is associated with a different user. The expected next step: call accounts.linkAccounts to merge between the account identified by the provided *UID* and the account identified by the provided login credentials (*loginID* + *password*).
- **Under User** (error code 403044) - returned when a user under the age of 13 tries to login.
- **Pending password change** (error code 403100) - returned when login is attempted and the password change interval has passed since the last password change. The interval is set in the security.passwordChangeInterval policy.
- **Account Pending TFA Verification/Registration** (error codes 403101/403102) - returned w hen a user calls this method and the policy (in the site Policies) requires 2-factor authentication, and the device is not in the verified device list for the account.
- **Account Temporarily Locked Out** (error code 403120) - returned when login is attempted and the account is locked out or the originating IP is locked out. This occurs after a set number of failed login attempts. The number is set in the site Policies - *security .accountLockout.failedLoginThreshold* policy and *security.ipLockout.hourlyFailedLoginThreshold* policy.