

# Package ‘firebase.auth.rest’

May 8, 2026

**Title** R Wrapper for 'Firebase Authentication REST API'

**Version** 1.0.1

**Description** A convenient and user-friendly interface to interact with the 'Firebase Authentication REST API': <https://firebase.google.com/docs/reference/rest/auth>. It enables R developers to integrate 'Firebase Authentication' services seamlessly into their projects, allowing for user authentication, account management, and other authentication-related tasks.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.2

**URL** <https://github.com/kennedymwavu/firebase.auth.rest>

**BugReports** <https://github.com/kennedymwavu/firebase.auth.rest/issues>

**Imports** httr2 (>= 0.2.3)

**NeedsCompilation** no

**Author** Kennedy Mwavu [aut, cre, cph] (Maintainer/developer of firebase.auth.rest since 2024, ORCID: <https://orcid.org/0009-0006-3157-7234>)

**Maintainer** Kennedy Mwavu <[mwavukennedy@gmail.com](mailto:mwavukennedy@gmail.com)>

**Repository** CRAN

**Date/Publication** 2025-07-18 02:20:02 UTC

## Contents

change_email . . . . .	2
change_password . . . . .	3
delete_account . . . . .	4
exchange_custom_token . . . . .	5
exchange_refresh_token . . . . .	6
get_user_data . . . . .	7
send_email_verification . . . . .	8

send_password_reset_email . . . . .	9
sign_in . . . . .	10
sign_in_anonymously . . . . .	11
sign_up . . . . .	12
update_profile . . . . .	13

<b>Index</b>	<b>15</b>
--------------	-----------

---

change_email	<i>Change email</i>
--------------	---------------------

---

## Description

Change email

## Usage

```
change_email(id_token, email)
```

## Arguments

id_token	String. A Firebase Auth ID token for the user.
email	String. User's new email.

## Details

**DISCLAIMER:** Changing a users's email requires that you disable email enumeration protection for your firebase project. This is NOT recommended.

- Learn about [email enumeration protection](#)
- Visit [Firebase Auth REST API docs](#) for more details

## Value

A named list with the following items:

- localId: The uid of the current user.
- email: User's email address.
- passwordHash: Hash version of password.
- providerUserInfo: A named list of of all linked provider objects which contain "providerId" and "federatedId".
- idToken: New Firebase Auth ID token for user.
- refreshToken: A Firebase Auth refresh token.
- expiresIn: string The number of seconds in which the ID token expires.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:
# first sign in user and get the 'id_token':
user <- sign_in(email = "user@gmail.com", password = "password")
id_token <- user$idToken

# change email:
response <- change_email(
  id_token = id_token,
  email = "new.email@mail.com"
)
response

## End(Not run)
```

---

change_password	<i>Change password</i>
-----------------	------------------------

---

**Description**

Change password

**Usage**

```
change_password(id_token, password)
```

**Arguments**

id_token	A Firebase Auth ID token for the user.
password	User's new password.

**Details**

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- localId: The uid of the current user.
- email: User's email address.
- passwordHash: Hash version of password.
- providerUserInfo: A named list of of all linked provider objects which contain "providerId" and "federatedId".
- idToken: New Firebase Auth ID token for user.
- refreshToken: A Firebase Auth refresh token.

- expiresIn: string The number of seconds in which the ID token expires.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

## Examples

```
## Not run:
# first sign in user and get the 'id_token':
user <- sign_in(email = "user@gmail.com", password = "password")
id_token <- user$idToken

# change password:
response <- change_password(
  id_token = id_token,
  password = "new-user-password"
)
response

## End(Not run)
```

---

delete_account	<i>Delete account</i>
----------------	-----------------------

---

## Description

Delete account

## Usage

```
delete_account(id_token)
```

## Arguments

id\_token      The Firebase ID token of the user to delete.

## Details

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:  
# first sign in user and get the 'id_token':  
user <- sign_in(email = "user@gmail.com", password = "password")  
id_token <- user$idToken  
  
# delete user account:  
response <- delete_account(id_token = id_token)  
response  
  
## End(Not run)
```

---

exchange\_custom\_token *Exchange custom token for an ID and refresh token*

---

**Description**

Exchanges a custom Auth token for an ID and refresh token

**Usage**

```
exchange_custom_token(token)
```

**Arguments**

token	String. A Firebase Auth custom token from which to create an ID and refresh token pair
-------	--

**Details**

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- idToken: A Firebase Auth ID token generated from the provided custom token.
- refreshToken: A Firebase Auth refresh token generated from the provided custom token.
- expiresIn: The number of seconds in which the ID token expires.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:  
  exchange_custom_token(token = "your-firebase-auth-custom-token")  
  
## End(Not run)
```

---

exchange\_refresh\_token

*Exchange a refresh token for an ID token*

---

**Description**

Refreshes a Firebase ID token

**Usage**

```
exchange_refresh_token(refresh_token)
```

**Arguments**

refresh\_token String. A Firebase Auth refresh token.

**Details**

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- expires\_in: The number of seconds in which the ID token expires.
- token\_type: The type of the refresh token, always "Bearer".
- refresh\_token: The Firebase Auth refresh token provided, or a new refresh token.
- id\_token: A Firebase Auth ID token.
- user\_id: The uid corresponding to the provided ID token.
- project\_id: Your Firebase project ID.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:  
# first sign in user and get the 'refresh_token':  
user <- sign_in(email = "user@gmail.com", password = "password")  
refresh_token <- user$refreshToken  
  
# exchange the refresh token:  
response <- exchange_refresh_token(refresh_token = refresh_token)  
response  
  
## End(Not run)
```

---

get\_user\_data

*Get user data from firebase*

---

**Description**

Get user data from firebase

**Usage**

```
get_user_data(id_token)
```

**Arguments**

id\_token           String. The Firebase ID token of the account.

**Details**

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- users: A list of length 1 which contains a nested named list with the following items:
  - localId: The uid of the current user.
  - email: The email of the account.
  - emailVerified: Whether or not the account's email has been verified.
  - displayName: The display name for the account.
  - providerUserInfo: Named list of provider objects which contain "providerId" and "federatedId".
  - photoUrl: The photo Url for the account.
  - passwordHash: Hash version of password.
  - passwordUpdatedAt: The timestamp, in milliseconds, that the account password was last changed.
  - validSince: The timestamp, in milliseconds, which marks a boundary, before which Firebase ID token are considered revoked.
  - disabled: Whether the account is disabled or not.
  - lastLoginAt: The timestamp, in milliseconds, that the account last logged in at.
  - createdAt: The timestamp, in milliseconds, that the account was created at.
  - customAuth: Whether the account is authenticated by the developer.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:
user_data <- get_user_data("<id_token>")
lapply(user_data, `[`, 1)

## End(Not run)
```

---

```
send_email_verification
```

*Send email verification*

---

**Description**

Send email verification

**Usage**

```
send_email_verification(id_token)
```

### Arguments

`id_token`            The Firebase ID token of the user to verify.

### Details

Visit [Firebase Auth REST API docs](#) for more details

### Value

A named list with the following items:

- `email`: The email of the account.
- `error`:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* `code`: Error code
    - \* `message`: Error message

### Examples

```
## Not run:  
send_email_verification("id-token-goes-here")  
  
## End(Not run)
```

---

```
send_password_reset_email  
                          Send password reset email
```

---

### Description

Send password reset email

### Usage

```
send_password_reset_email(email)
```

### Arguments

`email`                User's email address.

### Details

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- email: Users' email address.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

**Examples**

```
## Not run:  
send_password_reset_email("user-email-goes-here")  
  
## End(Not run)
```

---

sign_in	<i>Sign in a user with email &amp; password</i>
---------	---

---

**Description**

Sign in a user with email & password

**Usage**

```
sign_in(email, password)
```

**Arguments**

email	User email
password	User password

**Details**

Visit [Firebase Auth REST API docs](#) for more details

**Value**

A named list with the following items:

- idToken: A Firebase Auth ID token for the authenticated user.
- email: The email for the authenticated user.
- refreshToken: A Firebase Auth refresh token for the authenticated user.
- expiresIn: The number of seconds in which the ID token expires.
- localId: The uid of the authenticated user.

- registered: Whether the email is for an existing account.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

### Examples

```
## Not run:
  sign_in(email = "user-email", password = "strong-password")

## End(Not run)
```

---

sign\_in\_anonymously    *Sign in a user anonymously*

---

### Description

Sign in a user anonymously

### Usage

```
sign_in_anonymously()
```

### Details

To use sign in users anonymously, you must first enable the Anonymous sign in method in your firebase project.

Go to [Firebase console](#) and check your *Sign-in* providers under the *Sign-in Methods* tab in the *Authentication* service and make sure *Anonymous* is enabled.

Visit [Firebase Auth REST API docs](#) for more details.

### Value

A named list with the following items:

- idToken: A Firebase Auth ID token for the newly created user.
- email: Since the user is anonymous, this should be empty.
- refreshToken: A Firebase Auth refresh token for the newly created user.
- expiresIn: The number of seconds in which the ID token expires.
- localId: The uid of the newly created user.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

## Examples

```
## Not run:
  user <- sign_in_anonymously()
  user

## End(Not run)
```

---

sign\_up

*Sign up with email/password*

---

## Description

Sign up with email/password

## Usage

```
sign_up(email, password)
```

## Arguments

email	The email for the user to create.
password	The password for the user to create.

## Details

Visit [Firebase Auth REST API docs](#) for more details

## Value

A named list with the following items:

- idToken: A Firebase Auth ID token for the newly created user.
- email: The email for the newly created user.
- refreshToken: A Firebase Auth refresh token for the newly created user.
- expiresIn: The number of seconds in which the ID token expires.
- localId: The uid of the newly created user.
- error:
  - NULL if no error code in response
  - A list of 2 if response was an error:
    - \* code: Error code
    - \* message: Error message

## Examples

```
## Not run:
  sign_up(email = "new-user-email", password = "strong-password")

## End(Not run)
```

---

update_profile	<i>Update user profile</i>
----------------	----------------------------

---

### Description

Update a user's profile (display name / photo URL).

### Usage

```
update_profile(  
  id_token,  
  display_name = NULL,  
  photo_url = NULL,  
  delete_attribute = NULL  
)
```

### Arguments

id_token	String. A Firebase Auth ID token for the user.
display_name	String. User's new display name. Defaults to NULL.
photo_url	String. User's new photo url. Defaults to NULL.
delete_attribute	Character vector of attributes to delete. Either "DISPLAY_NAME" or "PHOTO_URL". This will nullify these values. Defaults to NULL.

### Details

Visit [Firebase Auth REST API docs](#) for more details

### Value

A named list with the following items:

- localId: The uid of the current user.
- email: User's email address.
- displayName: User's new display name.
- photoUrl: User's new photo url.
- passwordHash: Hash version of password.
- providerUserInfo: A named list of of all linked provider objects which contain "providerId" and "federatedId".
- idToken: New Firebase Auth ID token for user.
- refreshToken: A Firebase Auth refresh token.
- expiresIn: string The number of seconds in which the ID token expires.
- error:

- NULL if no error code in response
- A list of 2 if response was an error:
  - \* code: Error code
  - \* message: Error message

**Examples**

```
## Not run:
update_profile(
  id_token = "id-token-goes-here",
  display_name = "new-user-display-name",
  photo_url = "url-to-user-photo"
)

# to delete the display name attribute:
update_profile(
  delete_attribute = "DISPLAY_NAME"
)

## End(Not run)
```

# Index

`change_email`, [2](#)  
`change_password`, [3](#)  
  
`delete_account`, [4](#)  
  
`exchange_custom_token`, [5](#)  
`exchange_refresh_token`, [6](#)  
  
`get_user_data`, [7](#)  
  
`send_email_verification`, [8](#)  
`send_password_reset_email`, [9](#)  
`sign_in`, [10](#)  
`sign_in_anonymously`, [11](#)  
`sign_up`, [12](#)  
  
`update_profile`, [13](#)