Package: metricminer (via r-universe)

May 22, 2026

Type Package

Title Mine Metrics from Common Places on the Web

Version 1.0.1

Description Mine metrics on common places on the web through the power of their APIs (application programming interfaces). It also helps make the data in a format that is easily used for a dashboard or other purposes. There is an associated dashboard template and tutorials that are underdevelopment that help you fully utilize 'metricminer'.

License GPL-3

URL https://github.com/ottrproject/metricminer

BugReports https://github.com/ottrproject/metricminer/issues

Imports httr, jsonlite, assertthat, openssl, gh (>= 1.3.0), getPass, dplyr, lubridate, purrr, tidyr, googledrive, googlesheets4, janitor, stringr, methods, magrittr, rvest, rprojroot, yaml

Suggests knitr, rmarkdown, testthat (>= 3.0.0), withr

Config/testthat/edition 3

Encoding UTF-8

RoxygenNote 7.3.2

LazyData true

VignetteBuilder knitr

Config/pak/sysreqs cmake git make libicu-dev libuv1-dev libxml2-dev libssl-dev

Repository https://ottrproject.r-universe.dev

Date/Publication 2026-05-02 02:02:12 UTC

RemoteUrl https://github.com/ottrproject/metricminer

RemoteRef HEAD

RemoteSha 4eb22638ec5a7e47e5e6b995458a8a2036b6fce0

Contents

`app_set_up` *App Set Up*

Description

This is a function that sets up the app. It's generally called by another function

Usage

```
app_set_up(app_name = "google")
```

Arguments

`app_name`	app would you like to authorize? Supported apps are 'google' 'calendly' and 'github'

`auth_from_secret` *Use secrets to Authorize R package to access endpoints*

Description

This is a function to authorize metricminer to access calendly, github or google noninteractively from passing in a keys or tokens.

Usage

```
auth_from_secret(
  app_name,
  token,
  access_token,
  refresh_token,
  cache = FALSE,
  in_test = FALSE
)
```

Arguments

app_name	Which app are you trying to authorize? 'google', 'calendly' or 'github'?
token	For calendly or github, pass in the API key or Personal Access Token that you have set up from going to https://github.com/settings/tokens/new or https://calendly.com/integrations/api_ respectively.
access_token	For Google, access token can be obtained from running authorize interactively: token <-authorize(); token$credentials$access_token
refresh_token	For Google, refresh token can be obtained from running authorize interactively: token <-authorize(); token$credentials$refresh_token
cache	Should the credentials be cached? TRUE or FALSE?
in_test	If setting up auth in a test, set to TRUE so that way the authorization doesn't stick

Value

OAuth token saved to the environment so the package access the API data

Examples

```
## Not run:

# Example for authorizing Calendly
# You go to https://calendly.com/integrations/api_webhooks to get an api key
auth_from_secret("calendly", token = "A_calendly_token_here")

# Example for GitHub
# You go to https://github.com/settings/tokens/new to get a
# Personal Access Token
auth_from_secret("github", token = "ghp_a_github_pat_here")

# Example for authorizing for Google
token <- authorize("google")
auth_from_secret(
  app_name = "google",
  access_token = token$credentials$access_token,
  refresh_token = token$credentials$refresh_token
)

## End(Not run)
```

---

authorize *Authorize R package to access endpoints*

Description

This is a function to authorize the R package to access APIs interactively. To learn more about the privacy policy for metricminer [read here](https://www.metricminer.org/privacypolicy.html)

Usage

```
authorize(app_name = NULL, cache = FALSE, ...)
```

Arguments

app_name	app would you like to authorize? Supported apps are 'google' 'calendly' and 'github'
cache	Should the token be cached as an .httr-oauth file or API keys stored as global options?
...	Additional arguments to send to oauth2.0_token

Value

API token saved to the environment or the cache so it can be grabbed by functions

Examples

```
## Not run:

authorize()

authorize("github")

authorize("google")

authorize("calendly")

## End(Not run)
```

---

cache_secrets_folder *See where your cached secrets are being stored*

Description

This is a function to retrieve the file path of where your cached secrets are stored

Usage

```
cache_secrets_folder()
```

Value

an file path that shows where your cached secrets are stored

Examples

```
## Not run:

# You can see where your cached secrets are being stored by running:
cached_secrets_folder()

## End(Not run)
```

`calendly_get` *Handle Calendly GET requests*

Description

This is a function that handles Calendly GET requests

Usage

```
calendly_get(url, token = NULL, user = NULL, count = NULL, page_token = NULL)
```

Arguments

`url`	The endpoint URL for this API request
`token`	You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the ‘authorize("calendly")‘ function
`user`	The user param for Calendly. Usually looks like "https://api.calendly.com/users/c208a750-9214-4c62-9ee6-a1a9507c7b43"
`count`	For paginated GETs, you can specify how many things you’d like returned
`page_token`	For a paginated GET, what page are we on?

Value

Calendly REST API response as a list

Examples

```
## Not run:

authorize("calendly")
token <- get_token(app_name = "calendly")

result_list <- calendly_get(
  url = "https://api.calendly.com/users/me",
  token = token
)

## End(Not run)
```

check_check *Check the testthat check log file and print out how many errors*

Description

if testthat's tests have been run, this will look for the check to see if anything truly broke It will return a TRUE/FALSE for whether or not there were errors based on the check/testthat.Rout file produced.

Usage

```
check_check(report_warning = TRUE)
```

Arguments

report_warning	Should the number include warnings in addition errors? Default is both will be reported but if you'd like to ignore warnings set this to FALSE.

Value

a how many errors/warnings were found

clean_ga_metrics *Handle Google Analytics Lists*

Description

These functions are to clean metric and dimension data from Google Analytics 'get_ga_stats()' function.

Usage

```
clean_ga_metrics(metrics = NULL, type = NULL)
```

Arguments

metrics	a metrics object from 'get_ga_stats()' function
type	If type == "pages" then treat the data frame for in the instance that the dimensions of the subpages were collected

Value

a data frame of cleaned metrics from Google Analytics

clean_repo_metrics *Summarizing metrics from GitHub*

Description

This is a function to get metrics for all the repositories underneath an organization

Usage

```
clean_repo_metrics(repo_name, repo_metric_list)
```

Arguments

repo_name	The repository name. So for 'https://github.com/ottrproject/metricminer', it would be 'metricminer'
repo_metric_list	a list containing the metrics

Value

Metrics for a repository on GitHub

default_creds_path *Default Credentials path*

Description

Default Credentials path

Usage

```
default_creds_path(app_name)
```

Arguments

app_name	What app set up are you looking for? Supported apps are 'google' 'calendly' and 'github' Get file path to an default credentials RDS

delete_creds *Delete cached metricminer credentials*

Description

This is a function to delete cached creds and creds in the current environment that were set by metricminer

Usage

```
delete_creds(app_name = "all")
```

Arguments

app_name	which app would you like to delete the creds for? Default is to delete the creds for all.

Value

Cached credentials are deleted and report is given back

Examples

```
## Not run:

delete_creds("google")

## End(Not run)
```

encrypt_creds_path *Default creds path*

Description

Default creds path

Usage

```
encrypt_creds_path(app_name)
```

Arguments

app_name	What app set up are you looking for? Supported apps are 'google' 'calendly' and 'github'

`example_data_folder` *Default Credentials path Get file path to an default credentials RDS*

Description

Default Credentials path Get file path to an default credentials RDS

Usage

```
example_data_folder()
```

Value

Returns the file path to folder where the example data is stored

`extract_answers` *Google Form handling functions – extracting answers*

Description

This is a function to get extract answers from a Google Form. It is used by the ‘get_google_form()‘ function if dataformat = "dataframe"

Usage

```
extract_answers(form_info)
```

Arguments

`form_info`	The return form_info list that is extracted in ‘get_google_form()‘

Value

This returns answers from a google form

get_calendly_user *Get Calendly API user*

Description

This is a function to get the Calendly API user info

Usage

```
get_calendly_user(token = NULL)
```

Arguments

token	You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the 'authorize("calendly")' function

Value

Calendly API user info as a list

Examples

```
## Not run:

authorize("calendly")
get_calendly_user()

## End(Not run)
```

get_citation_count *Get a list of papers that cite your paper*

Description

This is a function to retrieve a list of papers that cite your papers

Usage

```
get_citation_count(paper_cite_link)
```

Arguments

paper_cite_link	This is not a google citation page. 1. Go to: https://scholar.google.com/scholar 2. Search for the paper we are looking for the citation count. 3. Then click the Cited by ___ button below the title of the paper 4. Copy and paste this url and put it in this get_citation_count() function

Value

A list of the example datasets available in this package

Examples

```
## Not run:

paper_cite_link <- "https://scholar.google.com/scholar?cites=6140457238337460780"

papers_cited_df <- get_citation_count(paper_cite_link)

## End(Not run)
```

---

get_config_file *Get config file*

Description

Get the _config_automation.yml file to set up a metricminer repo

Usage

```
get_config_file(overwrite = FALSE)
```

Arguments

overwrite	Should a _config_automation.yml file in the current directory be overwritten? Default is false.

Value

Copies a config_automation.yml file to your current working directory

---

get_example_data *Get retrieve an example dataset*

Description

This is a function to retrieve a list of the example datasets included with metricminer

Usage

```
get_example_data(dataset_name, envir = 1)
```

Arguments

dataset_name	the name of the example dataset to be retrieved from the metricminer package.
envir	By default the example data is saved in the global environment but this parameter allows you to change that if desired.

Value

an object in the environment of the same example dataset name that was requested.

Examples

```
## Not run:

# You can see the list of example datasets by running:
list_example_data()

# Then use the datasetes of your interest by calling it with this function
get_example_data("gform_info")

# Then if you check your global environment you will see "gform_info" included
ls()

## End(Not run)
```

---

get_ga_metadata *Get metadata associated Google Analytics property*

Description

This is a function to get the Google Analytics accounts that this user has access to. The scope it uses is the 'See and download your Google Analytics data' If you don't this check this box on the OAuth screen this won't work.

Usage

```
get_ga_metadata(property_id, token = NULL)
```

Arguments

property_id	a GA property. Looks like '123456789' Can be obtained from running 'get_ga_properties()'
token	credentials for access to Google using OAuth. 'authorize("google")'

Value

A list showing the metadata types available for the Google Analytics property. This can be used to craft an API request.

Examples

```
## Not run:

authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])

property_id <- gsub("properties/", "", properties_list$name[1])
property_metadata <- get_ga_metadata(property_id = property_id)

## End(Not run)
```

`get_ga_properties` *Get all property ids for all Google Analytics associated with an account id*

Description

This retrieves all the property ids associated with a Google Analytics Account. The scope it uses is the 'See and download your Google Analytics data' If you don't this check this box on the OAuth screen this won't work.

Usage

```
get_ga_properties(account_id, token = NULL)
```

Arguments

`account_id`	the account id of the properties you are trying to retrieve
`token`	credentials for access to Google using OAuth. 'authorize("google")'

Value

All the property ids and information about them for a Google Analytics account.

Examples

```
## Not run:

authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])

## End(Not run)
```

get_ga_property_info *Get all property information for a particular property id*

Description

This is a function to get the Google Analytics accounts that this user has access to. The scope it uses is the 'See and download your Google Analytics data' If you don't this check this box on the OAuth screen this won't work.

Usage

```
get_ga_property_info(property_id, token = NULL)
```

Arguments

property_id	the property id you want information about.
token	credentials for access to Google using OAuth. 'authorize("google")'

Value

All the property ids and information about them for a Google Analytics account.

Examples

```
## Not run:

authorize("google")
accounts <- get_ga_user()
properties_list <- get_ga_properties(account_id = accounts$id[1])
property_id <- gsub("properties\\/", "", properties_list$name[1])

property_info <- get_ga_property_info(property_id = property_id)

## End(Not run)
```

get_ga_stats *Get stats for an associated Google Analytics property*

Description

This is a function to get the Google Analytics accounts that this user has access to. The scope it uses is the 'See and download your Google Analytics data' If you don't this check this box on the OAuth screen this won't work.

Usage

```
get_ga_stats(
  property_id,
  start_date = "2015-08-14",
  token = NULL,
  body_params = NULL,
  end_date = NULL,
  stats_type = "metrics",
  dataformat = "dataframe"
)
```

Arguments

property_id	a GA property. Looks like '123456789' Can be obtained from running 'get_ga_properties()'
start_date	YYYY-MM-DD format of what metric you'd like to collect metrics from to start. Default is the earliest date Google Analytics were collected.
token	credentials for access to Google using OAuth. 'authorize("google")'
body_params	The body parameters for the request
end_date	YYYY-MM-DD format of what metric you'd like to collect metrics from to end. Default is today.
stats_type	Do you want to retrieve metrics or dimensions?
dataformat	How would you like the data returned to you? Default is a "dataframe" but if you'd like to see the original API list result, put "raw".

Value

Metrics dimensions for a GA returned from the Google Analytics API. It can be returned as a curated data.frame or the raw version which is the API response as a list

Examples

```
## Not run:

authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[2])

property_id <- gsub("properties/", "", properties_list$name[1])
metrics <- get_ga_stats(property_id, stats_type = "metrics")
dimensions <- get_ga_stats(property_id, stats_type = "dimensions")
pages <- get_ga_stats(property_id, stats_type = "pages")

## End(Not run)
```

get_ga_user *Get Google Analytics Accounts*

Description

This is a function to get the Google Analytics accounts that this user has access to. The scope it uses is the 'See and download your Google Analytics data' If you don't this check this box on the OAuth screen this won't work.

Usage

```
get_ga_user(token = NULL, request_type = "GET")
```

Arguments

token	credentials for access to Google using OAuth. 'authorize("google")'
request_type	Is this a GET or a POST?

Value

Information about what accounts Google Analytics credentials has access to

Examples

```
## Not run:

authorize("google")
get_ga_user()

## End(Not run)
```

get_github *Handler function for GET requests from GitHub*

Description

This is a function to get the GitHub user's info

Usage

```
get_github(token = NULL, url)
```

Arguments

token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
url	What is the URL endpoint we are attempting to grab here?

Value

Information regarding a Github account

`get_github_metrics` *Get the repository summary or time course metrics*

Description

This is a function to get the information about a repository

Usage

```
get_github_metrics(
  repo,
  token = NULL,
  count = 1e+05,
  data_format = "dataframe",
  github_stats = "all",
  time_course = FALSE
)
```

Arguments

`repo`	The repository name. So for 'https://github.com/ottrproject/metricminer', it would be 'ottrproject/metricminer'
`token`	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
`count`	How many items would you like to receive? default is 100000
`data_format`	Default is to return a curated data frame. However if you'd like to see the raw information returned from GitHub set format to "raw".
`github_stats`	Which stats would you like to collect from the GitHub API? Argument should be a vector of the names of the stats to be collected. This differs whether time_course is TRUE/FALSE. If time_course = FALSE should be a vector that can include: "repo_activity", "stars", "forks", "contributors", "community" If time_course = TRUE should be a vector that can include: "clones" and "views". By default "all" will be collected.
`time_course`	Should the time course data be collected or only the summary metrics?

Value

Repository summary or time course metrics for a particular GitHub repository as a dataframe

Examples

```
## Not run:

authorize("github")
metrics <- get_github_metrics(repo = "ottrproject/metricminer")

# If you only want some of the stats you can choose which ones with the
# github_stats argument
metrics <- get_github_metrics(
  repo = "ottrproject/metricminer",
  github_stats = c("repo_activity", "stars"))

summary_metrics <- get_github_repo_summary(repo = "ottrproject/metricminer")
timecourse_metrics <- get_github_repo_timecourse(repo = "ottrproject/metricminer")

## End(Not run)
```

---

get_github_repo_summary

Collect repository summary metrics

Description

This is a wrapper for get_github_metrics that has 'time_course = FALSE' so that summary metrics are collected

This is a function to get the information about a repository

Usage

```
get_github_repo_summary(
  repo,
  token = NULL,
  count = 1e+05,
  data_format = "dataframe",
  github_stats = "all"
)
```

Arguments

repo	The repository name. So for 'https://github.com/ottrproject/metricminer', it would be 'ottrproject/metricminer'
token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
count	How many items would you like to receive? default is 100000
data_format	Default is to return a curated data frame. However if you'd like to see the raw information returned from GitHub set format to "raw".

`github_stats` Which stats would you like to collect from the GitHub API? Argument should be a vector of the names of the stats to be collected. This differs whether time_course is TRUE/FALSE. If time_course = FALSE should be a vector that can include: "repo_activity", "stars", "forks", "contributors", "community" If time_course = TRUE should be a vector that can include: "clones" and "views". By default "all" will be collected.

Value

GitHub repository summary metrics

Examples

```
## Not run:

authorize("github")

summary_metrics <- get_github_repo_summary(repo = "ottrproject/metricminer")

## End(Not run)
```

`get_github_repo_timecourse`
Collect repository timecourse metrics

Description

This is a wrapper for `get_github_metrics` that has ‘time_course = TRUE‘ so that timecourse metrics are collected

This is a function to get the information about a repository

Usage

```
get_github_repo_timecourse(
  repo,
  token = NULL,
  count = 1e+05,
  data_format = "dataframe",
  github_stats = "all"
)
```

Arguments

`repo` The repository name. So for ‘https://github.com/ottrproject/metricminer‘, it would be ‘ottrproject/metricminer‘

`token` You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")‘ function

count	How many items would you like to receive? default is 100000
data_format	Default is to return a curated data frame. However if you'd like to see the raw information returned from GitHub set format to "raw".
github_stats	Which stats would you like to collect from the GitHub API? Argument should be a vector of the names of the stats to be collected. This differs whether time_course is TRUE/FALSE. If time_course = FALSE should be a vector that can include: "repo_activity", "stars", "forks", "contributors", "community" If time_course = TRUE should be a vector that can include: "clones" and "views". By default "all" will be collected.

Value

GitHub repository timecourse metrics for views and clones

Examples

```
## Not run:

authorize("github")

timecourse_metrics <- get_github_repo_timecourse(repo = "ottrproject/metricminer")

## End(Not run)
```

---

get_github_user *Get the GitHub User's info*

Description

This is a function to get the GitHub user's info

Usage

```
get_github_user(token = NULL)
```

Arguments

token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function

Value

Information regarding a Github account

Examples

```
## Not run:

authorize("github")
get_github_user()

## End(Not run)
```

`get_google_form` *Get Google Forms*

Description

This is a function to get Google Form info and responses from the API. The scopes it uses are the ‘See all your Google Forms forms.‘ and ‘See all responses to your Google Forms forms.‘ If you don’t check this box on the OAuth screen this function won’t work.

Usage

```
get_google_form(form_id, token = NULL, dataformat = "dataframe")
```

Arguments

`form_id`	The form ID we need to get
`token`	credentials for access to Google using OAuth. ‘authorize("google")‘
`dataformat`	What format would you like the data? Options are "raw" or "dataframe". "dataframe" is the default.

Value

This returns a list of the form info and responses to the google form. Default is to make this a list of nicely formatted dataframes.

Examples

```
## Not run:

authorize("google")
form_info <- get_google_form(
  "https://docs.google.com/forms/d/1Neyj7wwNpn8wC7NzQND8kQ30cnbbETSpT0lKhX7uaQY/edit"
)
form_id <- "https://docs.google.com/forms/d/1Neyj7wwNpn8wC7NzQND8kQ30cnbbETSpT0lKhX7uaQY/edit"

### OR You can give it a direct form id

form_info <- get_google_form("1Neyj7wwNpn8wC7NzQND8kQ30cnbbETSpT0lKhX7uaQY")

## End(Not run)
```

`get_multiple_forms` *Get multiple Google forms*

Description

This is a wrapper function for returning google form info and responses for multiple forms at once. The scopes it uses are the ‘See all your Google Forms forms.‘ and ‘See all responses to your Google Forms forms.‘ If you don’t check this box on the OAuth screen this function won’t work.

Usage

```
get_multiple_forms(form_ids = NULL, token = NULL, dataformat = "dataframe")
```

Arguments

`form_ids`	a vector of form ids you’d like to retrieve information for
`token`	credentials for access to Google using OAuth. ‘authorize("google")‘
`dataformat`	What format would you like the data? Options are "raw" or "dataframe". "dataframe" is the default.

Value

This returns a list of API information for google forms

Examples

```
## Not run:

authorize("google")
form_list <- googledrive::drive_find(
  shared_drive = googledrive::as_id("0AJb5Zemj0AAkUk9PVA"),
  type = "form"
)

multiple_forms <- get_multiple_forms(form_ids = form_list$id)

## End(Not run)
```

`get_multiple_ga_metrics` *Get all metrics for all properties associated with an account*

Description

This is a function to gets metrics and dimensions for all properties associated with an account. The scope it uses is the ‘See and download your Google Analytics data‘ If you don’t this check this box on the OAuth screen this won’t work.

Usage

```
get_multiple_ga_metrics(
  account_id = NULL,
  property_ids = NULL,
  token = NULL,
  start_date = "2015-08-14",
  end_date = NULL,
  dataformat = "dataframe",
  stats_type = c("metrics", "dimensions", "link_clicks")
)
```

Arguments

`account_id`	the account id that you’d like to retrieve stats for all properties associated with it.
`property_ids`	A vector of property ids you’d like to retrieve metrics for.
`token`	credentials for access to Google using OAuth. ‘authorize("google")‘
`start_date`	YYYY-MM-DD format of what metric you’d like to collect metrics from to start. Default is the earliest date Google Analytics were collected.
`end_date`	YYYY-MM-DD format of what metric you’d like to collect metrics from to end. Default is today.
`dataformat`	How would you like the data returned to you? Default is a "dataframe" but if you’d like to see the original API list result, put "raw".
`stats_type`	Do you want to retrieve metrics or dimensions? List all you want to collect as a vector

Value

Either a list of dataframes where ‘metrics‘, ‘dimensions‘ and ‘link clicks‘ are reported. But if ‘format‘ is set to "raw" then the original raw API results will be returned

A list of metrics, dimensions, and link clicks for a for all properties underneath a Google Analytics account. It can be returned as a curated data.frame or the raw version which is the API response as a list

Examples

```
## Not run:

authorize("google")
accounts <- get_ga_user()

properties_list <- get_ga_properties(account_id = accounts$id[1])
property_ids <- gsub("properties/", "", properties_list$name[1:2])

all_properties <- get_multiple_ga_metrics(account_id = accounts$id[1])

some_properties <- get_multiple_ga_metrics(property_ids = property_ids)


## End(Not run)
```

`get_multiple_repos_metrics` *Retrieve metrics for a list of repos*

Description

This is a function to get metrics for a list of repos. You can provide an owner and attempt retrieve all repositories from a particular organization, or you can provide a character vector of repositories.

Usage

```
get_multiple_repos_metrics(
  repo_names = NULL,
  token = NULL,
  data_format = "dataframe",
  time_course = FALSE,
  github_stats = "all"
)
```

Arguments

`repo_names`	a character vector of repositories you'd like to collect metrics from.
`token`	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the 'authorize("github")' function
`data_format`	Default is to return a curated data frame. However if you'd like to see the raw information returned from GitHub set format to "raw".
`time_course`	Should the time course data be collected or only the summary metrics?
`github_stats`	Which stats would you like to collect from the GitHub API? Argument should be a vector of the names of the stats to be collected. This differs whether time_course is TRUE/FALSE. If time_course = FALSE should be a vector that

can include: "repo_activity", "stars", "forks", "contributors", "community" If time_course = TRUE should be a vector that can include: "clones" and "views". By default "all" will be collected.

Value

A list of metrics for a list of repos in a dataframe format

Examples

```
## Not run:

authorize("github")

repo_names <- c("ottrproject/metricminer", "ottrproject/OTTR_Template")
some_repos_metrics <- get_multiple_repos_metrics(repo_names = repo_names)

stars_and_forks <- get_multiple_repos_metrics(repo_names = repo_names,
github_stats = c("stars", "forks"))

some_repos_metrics <- get_multiple_repos_metrics(repo_names = repo_names,
 time_course = TRUE)

gh_timecourse <- get_multiple_repos_metrics(repo_names = repo_names,
 github_stats = c("repo_activity", "stars", "forks", "contributors"))

## End(Not run)
```

---

get_org_repo_list *Retrieve list of repositories for an organization*

Description

This is a function to get the information about a repository

Usage

```
get_org_repo_list(
  owner,
  count = 1e+05,
  data_format = "dataframe",
  token = NULL
)
```

Arguments

owner	The owner of the repository. So for ‘https://github.com/ottrproject/metricminer‘, it would be ‘fhdsl‘
count	The number of responses that should be returned. Default is 100000
data_format	Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".
token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")‘ function

Value

a list of repositories that an organization has

Examples

```
## Not run:

authorize("github")
get_org_repo_list(owner = "fhdsl")

## End(Not run)
```

---

get_question_metadata *Google Form handling functions*

Description

This is a function to get metadata about a Google Form. It is used by the ‘get_google_form()‘ function if dataformat = "dataframe".

Usage

```
get_question_metadata(form_info)
```

Arguments

form_info	The return form_info list that is extracted in ‘get_google_form()‘

Value

This returns metadata from a google form

`get_slido_files` *Get Slido Files*

Description

This is a function to get slido response output files. The slido files must be saved as googlesheets and cannot be xlsx. Authentication for this function is handled by the 'googledrive' package. However, the user will need to authorize their google account twice (once for googledrive and once for googlesheets). If you don't check these boxes on the OAuth screens, this function won't work.

Usage

```
get_slido_files(
  shared_drive_name,
  tags_to_find = "^Polls-per|^JoinedParticipants-",
  file_type = "spreadsheet",
  keep_duplicates = FALSE
)
```

Arguments

`shared_drive_name`	a name of a shared drive (not a URL or subpart of a URL) (will recursively search for files by default).
`tags_to_find`	pattern or character that's a regular expression to look for in file names. Default is "^Polls-per|^JoinedParticipants-" which will search for files starting with either of those patterns.
`file_type`	which file type to search for. Default is "spreadsheet"
`keep_duplicates`	By default we won't keep duplicated files if a two files have the same name. But if you set this to true, duplicates will be returned.

Value

A list of the slido files and their content in a Googledrive location.

Examples

```
## Not run:

shared_drive_name <- "ITCR"
slido_data <- get_slido_files(shared_drive_name)

## End(Not run)
```

get_timestamp_repo_metrics *Get timestamp repository metrics*

Description

Get timestamp repository metrics

Usage

```
get_timestamp_repo_metrics(results, column)
```

Arguments

results	An API result from GitHub typically the views or clones for a repo
column	name of the column being extracted. Typically "views" or "clones"

Value

Extracted timestamp metrics from the API response

get_user_repo_list *Retrieve list of repositories for an organization*

Description

This is a function to get the information about a repository

Usage

```
get_user_repo_list(
  owner,
  count = 1e+05,
  data_format = "dataframe",
  token = NULL
)
```

Arguments

owner	The owner of the repository. So for ‘https://github.com/ottrproject/metricminer‘, it would be ‘fhdsl‘
count	The number of responses that should be returned. Default is 100000
data_format	Default is to return a curated data frame. However if you’d like to see the raw information returned from GitHub set format to "raw".
token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")‘ function

Value

a list of repositories that an organization has

Examples

```
## Not run:

authorize("github")
get_user_repo_list(owner = "metricminer")

## End(Not run)
```

`get_youtube_channel_stats` *Get Youtube channel stats*

Description

This is a function to retrieve statistics for a Youtube channel

Usage

```
get_youtube_channel_stats(channel_id, token = NULL, dataformat = "dataframe")
```

Arguments

`channel_id`	ID of the Youtube channel to retrieve stats from.
`token`	OAuth token from Google login.
`dataformat`	How would you like the data returned to you? Default is a "dataframe" but if you'd like to see the original API list result, put "raw".

Value

A data frame of the channel stats from a Youtube channel.

Examples

```
## Not run:

authorize("google")
youtube_channel_stats <- get_youtube_channel_stats("UCr73I9ZEPbn-3_1CBM57QgQ")

## End(Not run)
```

get_youtube_video_stats *Get Youtube video stats*

Description

This is a function to get a statistics on a Youtube video

Usage

```
get_youtube_video_stats(video_id, token = NULL, dataformat = "dataframe")
```

Arguments

video_id	ID of the Youtube video to retrieve stats from.
token	OAuth token from Google login. https://www.youtube.com/watch?v=YkYnni-WuaQor just the "YkYnni-WuaQor" part that comes after the 'v=' bit.
dataformat	How would you like the data returned to you? Default is a "dataframe" but if you'd like to see the original API list result, put "raw".

Value

A data frame of the Youtube video stats.

Examples

```
## Not run:

authorize("google")
youtube_video_stats <- get_youtube_video_stats("YkYnni-WuaQ")

## End(Not run)
```

gh_repo_wrapper *Wrapper function for gh repository calls*

Description

This is a function that wraps up gh calls for us

Usage

```
gh_repo_wrapper(api_call, owner, repo, token = NULL, count = 1e+05)
```

Arguments

api_call	an API call and endpoint. That has ‘owner‘ and ‘user‘.
owner	The repository name. So for ‘https://github.com/ottrproject/metricminer‘, it would be ‘fhdsl‘
repo	The repository name. So for ‘https://github.com/ottrproject/metricminer‘, it would be ‘metricminer‘
token	You can provide the Personal Access Token key directly or this function will attempt to grab a PAT that was stored using the ‘authorize("github")‘ function
count	How many items would you like to receive? default is 100000

Value

Metrics for a repository on GitHub

---

key_encrypt_creds_path

Get file path to an key encryption RDS

Description

Get file path to an key encryption RDS

Usage

```
key_encrypt_creds_path()
```

---

list_calendly_events *Get Calendly Event Lists*

Description

This is a function to get a list of scheduled events from a Calendly user.

Usage

```
list_calendly_events(token = NULL, user, count = 100)
```

Arguments

token	You can provide the API key directly using this argument or this function will attempt to grab an API key that was stored using the ‘authorize("calendly")‘ function
user	You need to retrieve the Calendly user’s URI. You can do this by doing ‘user <- get_calendly_user()‘ and ‘user$resource$uri‘
count	The number of responses that should be returned. Default is 20 or you can say "all" to retrieve all.

Value

Calendly REST API response as a list

Examples

```
## Not run:

authorize("calendly")
user <- get_calendly_user()
list_calendly_events(user = user$resource$uri)

## End(Not run)
```

`list_example_data` *Get list of example datasets*

Description

This is a function to retrieve a list of the example datasets included with metricminer

Usage

```
list_example_data()
```

Value

A list of the example datasets available in this package

Examples

```
## Not run:

list_example_data()

# Now you could use any of these example datasets that are printed out

get_example_data("calendly_events")

## End(Not run)
```

request_ga *Handler for API requests from Google Analytics*

Description

This is a function that handles requests from Google Analytics. The scope it uses is the ‘See and download your Google Analytics data‘ If you don’t this check this box on the OAuth screen this won’t work.

Usage

```
request_ga(token, url, query = NULL, body_params = NULL, request_type)
```

Arguments

token	credentials for access to Google using OAuth. ‘authorize("google")‘
url	The endpoint URL for the request
query	A list to be passed to query
body_params	The body parameters for the request
request_type	Is this a GET or a POST?

Value

An API response in the form of a list

request_google_forms *Get Google Forms*

Description

This is a function to get the Google Forms API requests. The scopes it uses are the ‘See all your Google Forms forms.‘ and ‘See all responses to your Google Forms forms.‘ If you don’t check this box on the OAuth screen this function won’t work.

Usage

```
request_google_forms(
  token,
  url,
  body_params = NULL,
  query_params = NULL,
  return_request = TRUE
)
```

Arguments

token	credentials for access to Google using OAuth. ‘authorize("google")‘
url	The endpoint URL for the request
body_params	The body parameters for the request
query_params	The body parameters for the request
return_request	Should a list of the request be returned as well?

Value

This function returns a list from a API response JSON file

---

setup_folders *Setups folder structure for metricminer*

Description

This is a function to setup a folder structure for metricminer data to be saved to. It depends on and reads Scope used for this function is the ‘See, edit, create, and delete only the specific Google Drive files you use with this app.‘

Usage

```
setup_folders(
  config_file = file.path(rprojroot::find_root(rprojroot::has_dir(".git")),
    "_config_automation.yml"),
  token = NULL
)
```

Arguments

config_file	The file path to the _config_automation.yml file
token	OAuth token from Google login.

Value

The googlesheet URL where the data has been written

Examples

```
## Not run:

authorize("google")

setup_folders(
  config_file =  "_config_automation.yml"
)
```

```
## End(Not run)
```

`supported_endpoints` *Supported endpoints*

Description

This is function stores endpoints and supported app names

Usage

```
supported_endpoints()
```

`write_playlist_details`
Write playlist details from YouTube

Description

Write playlist details from YouTube

Usage

```
write_playlist_details(playlist_id, token = NULL, outfile = NULL)
```

Arguments

`playlist_id`	string, playlist ID on YouTube
`token`	OAuth token from Google login.
`outfile`	string, a filename to which to write results in the 'resources' folder

Value

writes a file containing the dataframe of cleaned results

Examples

```
## Not run:
# Not run
write_playlist_details(
  playlist_id = shorts_playlist_id,
  outfile = "youtube_shorts_data.tsv"
)
write_playlist_details(
  playlist_id = "PL6aYJ_0zJ4uCABkMngSYjPo_3c-nUUmio",
  outfile = "youtube_shorts_data.tsv"
)

## End(Not run)
```

`write_to_gsheet` *Writes data to a Googlesheet*

Description

This is a function to write metricminer data to a Googlesheet. Scope used for this function is the 'See, edit, create, and delete only the specific Google Drive files you use with this app.' When you get to the OAuth consent screen. If you do not check this box, this function won't work.

Usage

```
write_to_gsheet(
  input,
  token = NULL,
  gsheet = NULL,
  overwrite = FALSE,
  append_rows = FALSE,
  sheet = 1,
  new_sheet = FALSE,
  ...
)
```

Arguments

`input`	input data to write to a googlesheet
`token`	OAuth token from Google login.
`gsheet`	Optionally a googlesheet to write to
`overwrite`	TRUE/FALSE overwrite if there is data at the destination
`append_rows`	TRUE/FALSE should the data be appended to the data?
`sheet`	Index or name of the worksheet you want to write to. Forwarded to googlesheets4::write_sheet or googlesheets4::append_sheet to indicate what sheet it should be written to.
`new_sheet`	default is FALSE. But if it is anything else will be used as the name for a new worksheet that will be made and written to.
`...`	these parameters are sent to googlesheets4::write_sheet.

Value

The googlesheet URL where the data has been written

Examples

```
## Not run:

authorize("github")
repo_list <- get_user_repo_list(owner = "metricminer")
gsheet <- paste0(
  "https://docs.google.com/spreadsheets/d/",
  "166MV4_1pfATB3Hes2HbdZCpkMc8JTT3u3eJes6Wu7Rk/edit#gid=0"
)
write_to_gsheet(repo_list)

datasheet <- write_to_gsheet(
  gsheet = gsheet,
  input = repo_list, append_rows = TRUE,
  sheet = 1
)

datasheet <- write_to_gsheet(
  gsheet = gsheet,
  input = repo_list,
  new_sheet = "github_data"
)

## End(Not run)
```

Index