

# An automated data pipeline using R and GitHub Actions



Anne Treasure  
Talarify  
anne@talarify.co.za

afrimapr Community Meetup  
20 April 2022

# Overview

---

- The requirement
- Overview of the data pipeline (input, processing, output)
- Data processing
  - data import and authorisations
  - data manipulation
  - automation using GitHub Actions



# The requirement

# SADiLaR and ESCALATOR

---

South African Centre for Digital Language Resources ([SADiLaR](#)) - a national centre supported by the Department of Science and Innovation as part of the South African Research Infrastructure Roadmap.

- has an enabling function, with a focus on all official languages of South Africa, supporting research and development in the domains of language technologies and language-related studies in the humanities and social sciences
- has a mandate to develop digital humanities capacity in South Africa
- to bring large scale adoption of digital research methodologies and practices to the social sciences and humanities, SADiLaR established the [ESCALATOR](#) project, which consists of a national digital champions programme in combination with an orchestrated capacity development and awareness raising initiative



# The requirement

---

- [Stakeholder map project](#): aims to collect and share data on Digital Humanities, Computational Social Sciences, and related activities and initiatives in South Africa (projects, people, publications, datasets, training courses, learning materials, tools, archives, unclassified, etc)
  - aim: to provide deeper insight into the breadth of activities in this area, to facilitate enhanced networking and collaboration, and support the optimal use of resources (e.g. researchers looking for collaborators, help potential students to identify training programmes, and highlight gaps and opportunities to funders and institutions, etc.)

**Data collected using Google Forms**

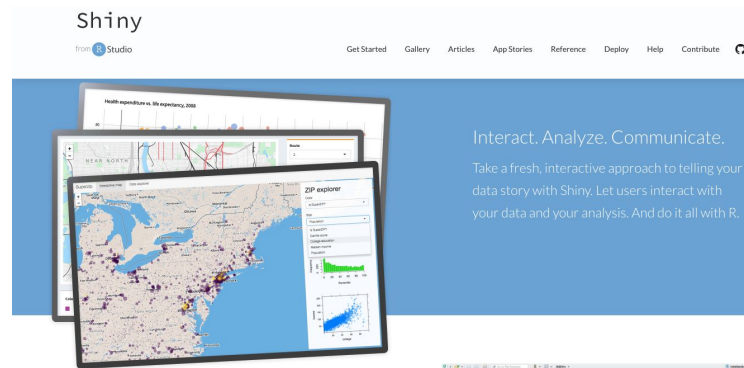


**Data visualisations using Shiny and Kumu**



# Required data visualisation tools

- Shiny is an R package that makes it easy to build interactive web apps straight from R



Source: <https://shiny.rstudio.com/>

- Kumu is a tool that makes it easy to organise complex data into relationship maps



**Make sense of your messy world.**

Kumu makes it easy to organize complex data into relationship maps that are beautiful to look at and a pleasure to use.



Source: <https://kumu.io/>



# Data pipeline: overview

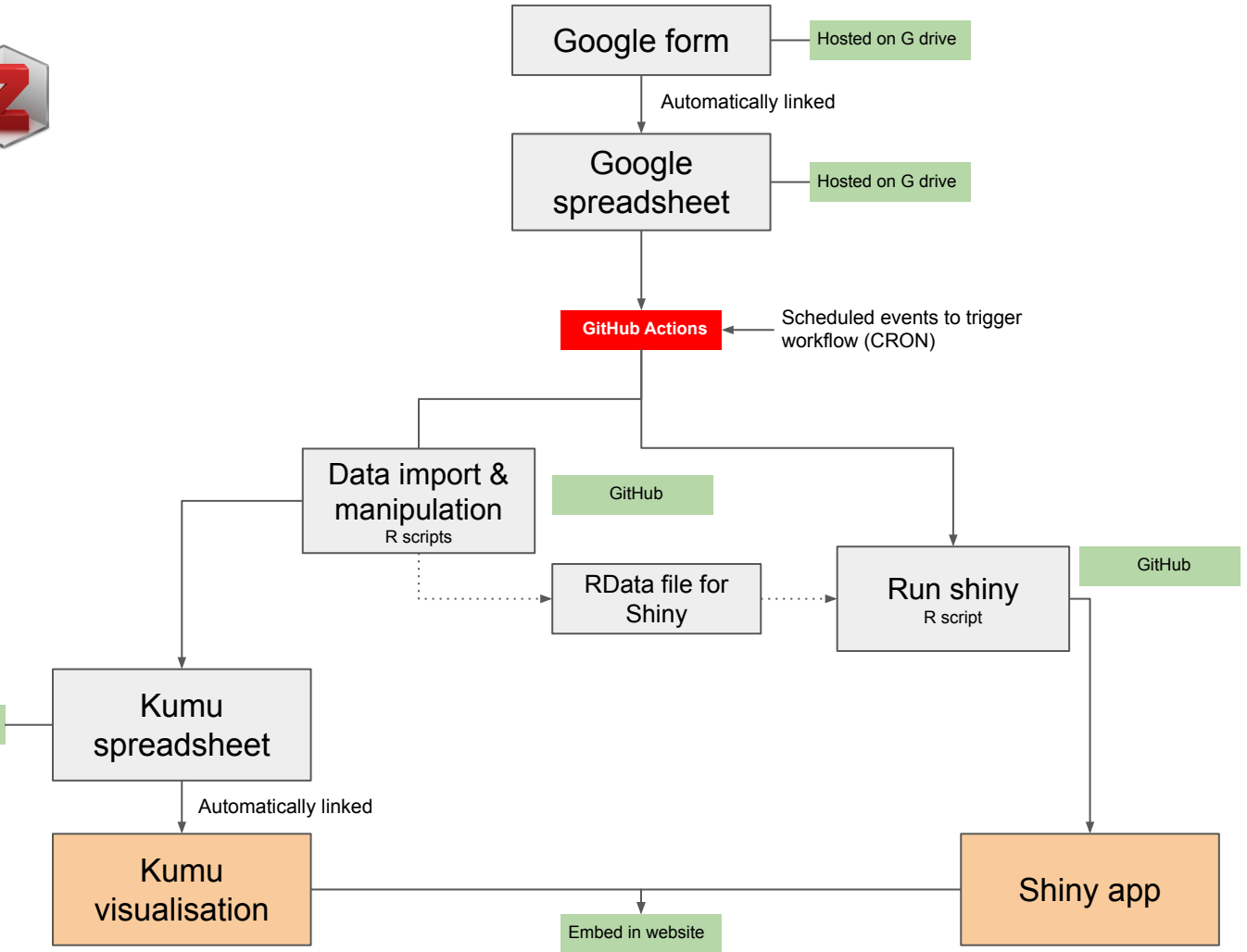
# Input / Source



# Data processing

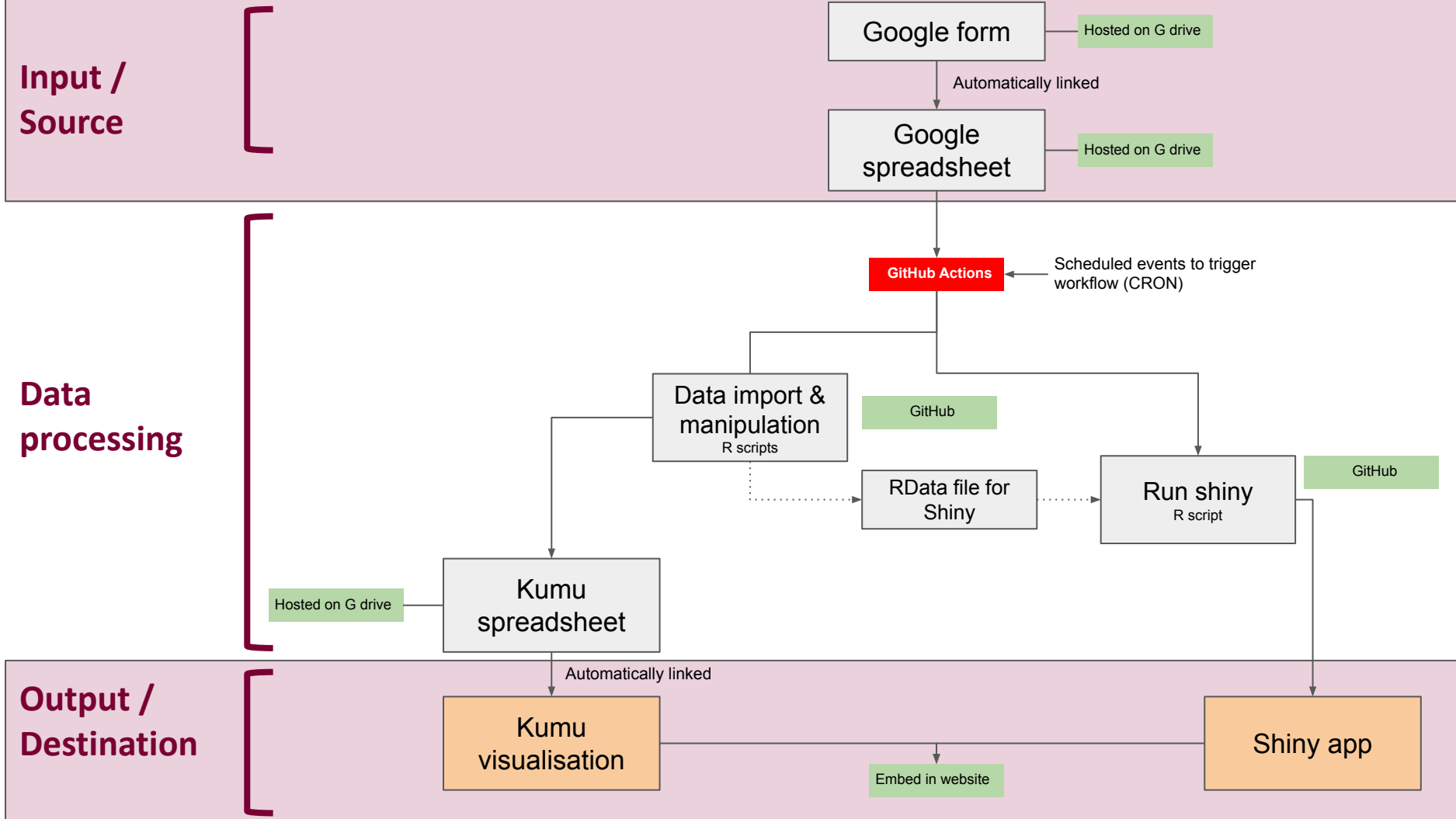


# Output / Destination





**The input and the output**





# Output: data visualisations



## Digital Humanities and Computational Social Sciences landscape in South Africa

The South African Centre for Digital Language Resources (SADiLaR) is a national centre supported by the Department of Science and Innovation (DSI) as part of the South African Research Infrastructure Roadmap (SARIR). SADiLaR has an enabling function, with a focus on all official languages of South Africa, supporting research and development in the domains of language technologies and language-related studies in the humanities and social sciences. Furthermore the centre has a mandate to develop digital humanities capacity in South Africa.

- Activities Map
  - Projects
  - People
  - Datasets
  - Tools
  - Publications
  - Training
  - Learning materials
- Archives    Unclassified records

Choose which record type to view

Person

Reset map view

This map shows data on Digital Humanities (DH) and Computational Social Sciences (CSS) activities and initiatives in South Africa. Records from locations close to each other are grouped together. To ungroup, click on a green circle and see individual locations, click again to see individual beneficiaries at one location. Click on **Reset map view** to get back to the original view.



### Project 2

PROJECT Imported

Project 2 description

Librarianship, Information and Museum Studies

Design and modelling

EMAIL sue.someone@email.com

FUNDERS Funding organisation

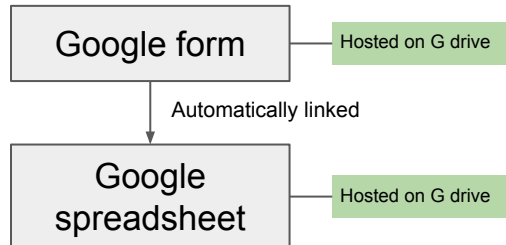
ORGANISATION University of Cape Town (UCT)

Legend

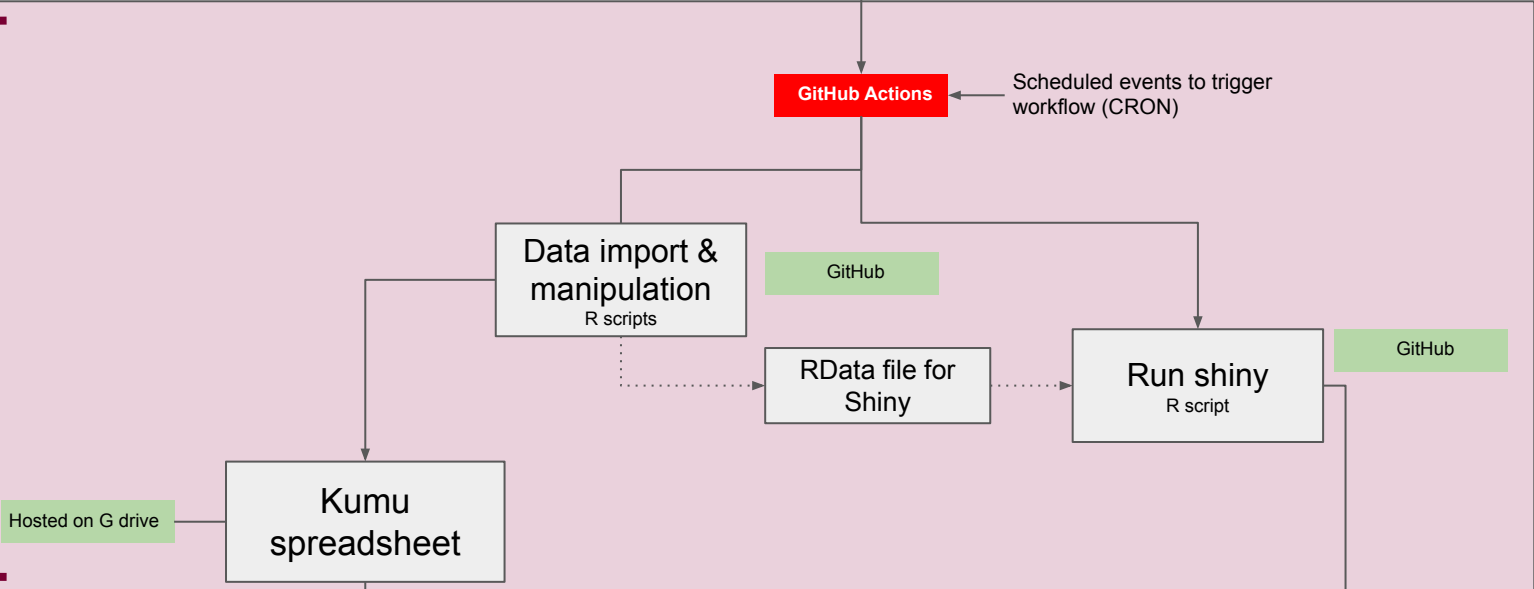
- Person
- Organization
- Project
- Organisation

# Data processing

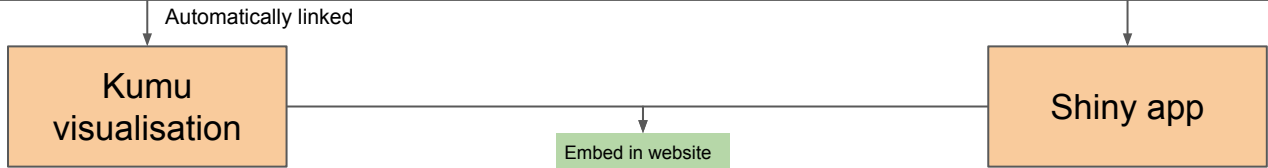
# Input / Source



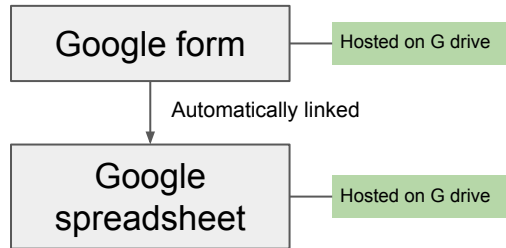
# Data processing



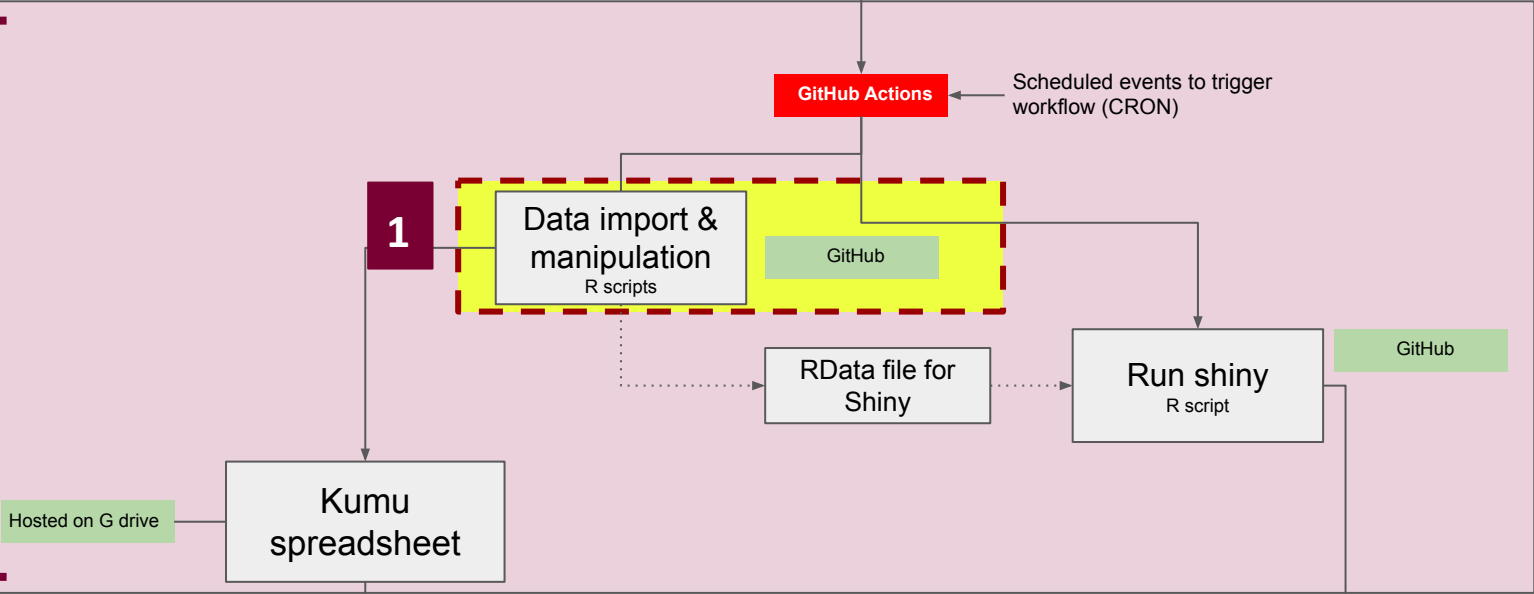
# Output / Destination



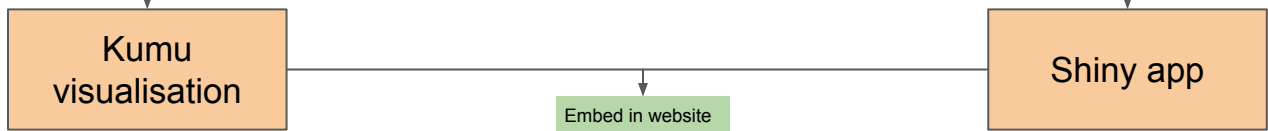
# Input / Source



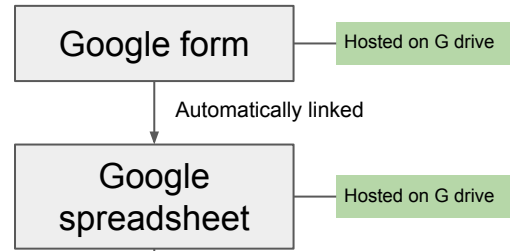
# Data processing



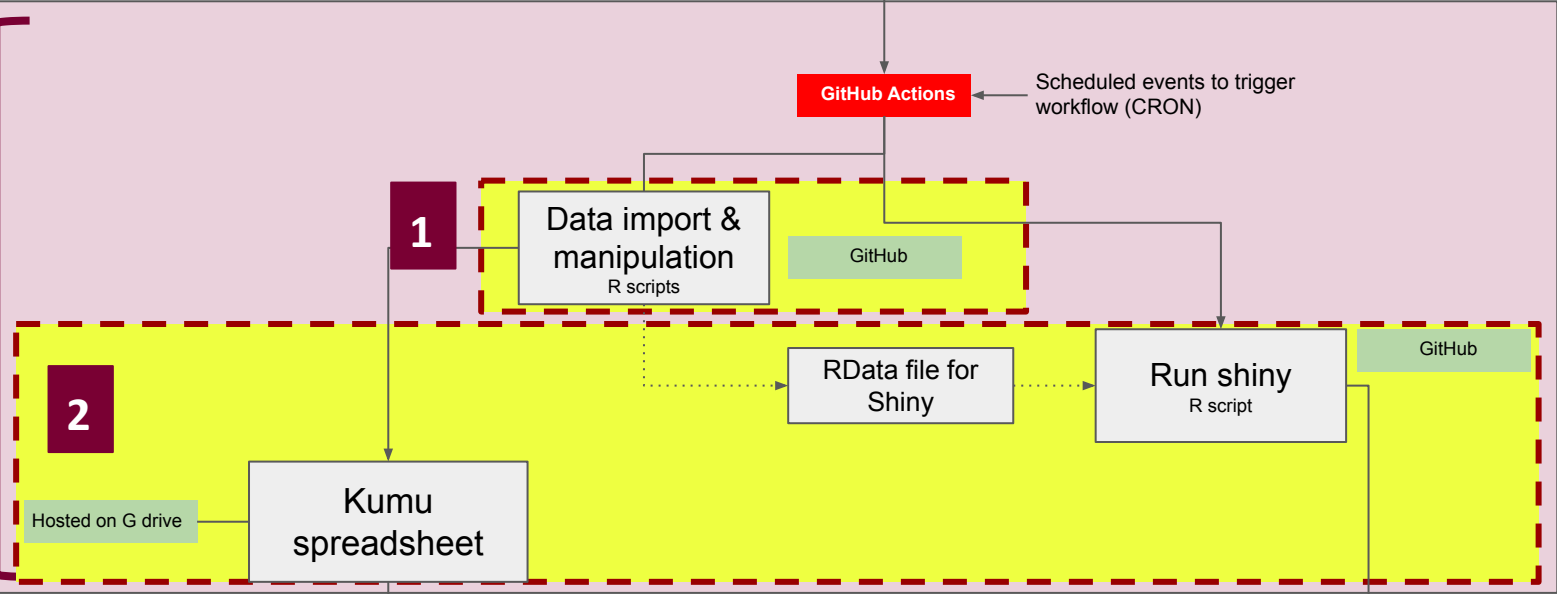
# Output / Destination



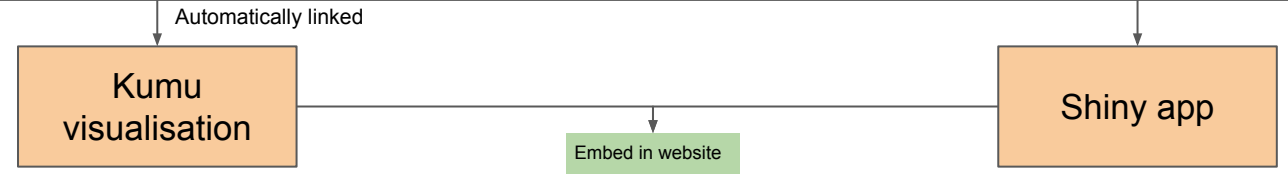
# Input / Source



# Data processing

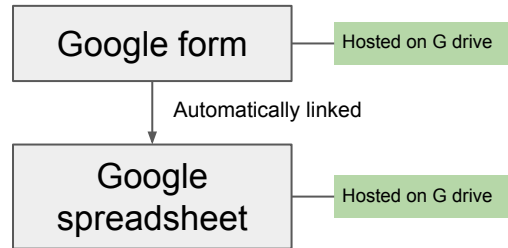


# Output / Destination

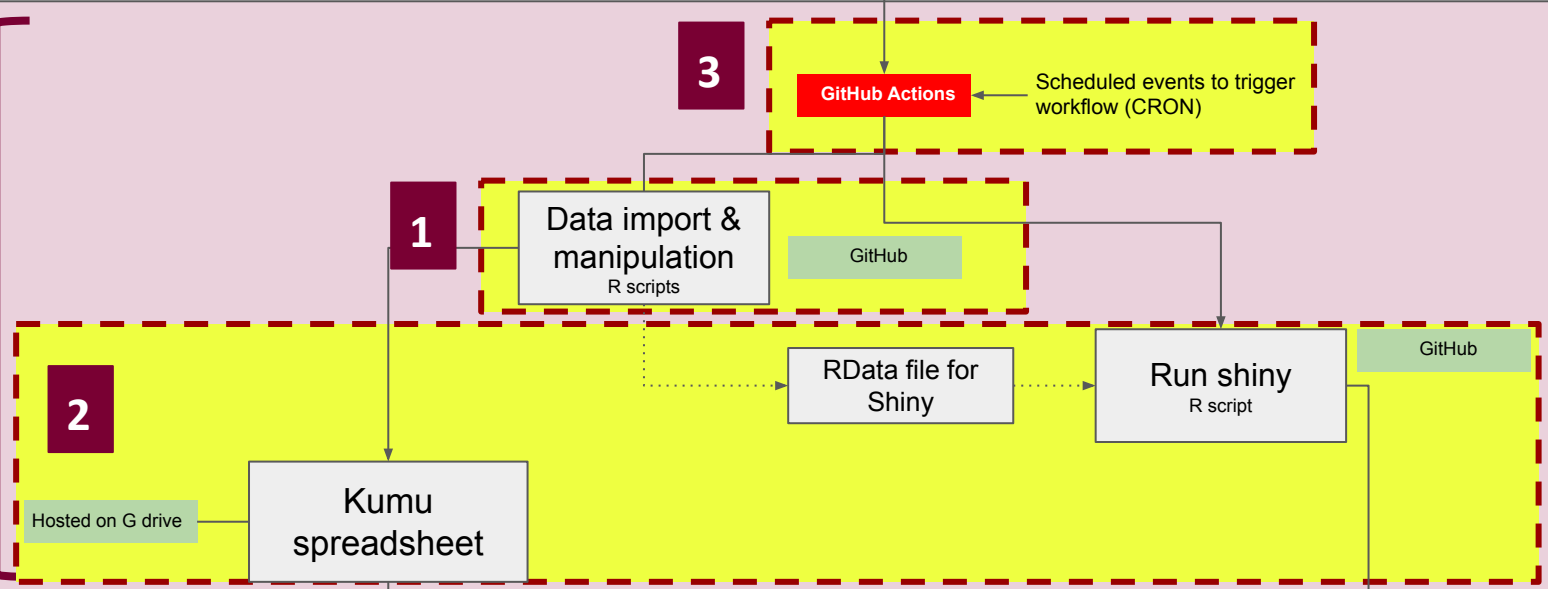




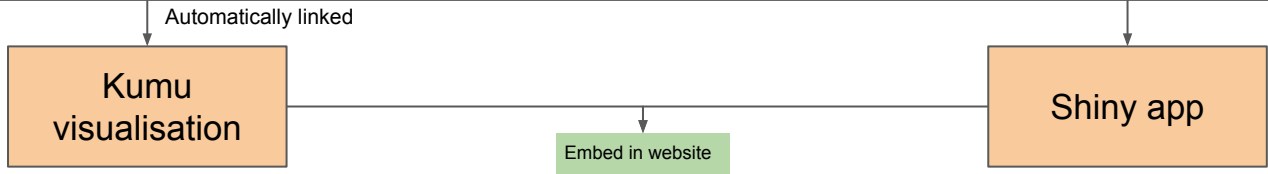
# Input / Source



# Data processing



# Output / Destination



# 1) Data import and authorisations

---

- R package to read data from a Google Sheet
  - `googlesheets4`
- Authorisations for import
  - a. script runs locally, but needs interaction (demo)
  - b. script runs locally, non-interactive (demo)
  - c. automate the non-interactive process (see 3. GitHub Action; demo)
- Data manipulation using R

## 2) Write to sheet for Kumu; save RData file for Shiny



# 1) Data import and authorisations

---

- R package to read data from a Google Sheet
  - `googlesheets4`
- Authorisations
  - a. script runs locally, but needs interaction (**demo**)
  - b. script runs locally, non-interactive (see next slide; demo)
  - c. automate the non-interactive process (see 3. GitHub Action; demo)
- Data manipulation using R

## 2) Write to sheet for Kumu; save RData file for Shiny



# 1) Data import and authorisations

---

- R package to read data from a Google Sheet
  - `googlesheets4`
- Authorisations
  - a. script runs locally, but needs interaction (**demo**)
  - b. script runs locally, non-interactive (**see next slide; demo**)
  - c. automate the non-interactive process (see 3. GitHub Action; demo)
- Data manipulation using R

## 2) Write to sheet for Kumu; save RData file for Shiny



## b. Non-interactive authorisations

---

- From this issue, Jenny Bryan's advice about using a service account for non-interactive authorisations: <https://github.com/tidyverse/googledrive/issues/327>
  1. Create a Google Cloud Platform account
  2. New project and create a service account
  3. Create a key and download the .json file
  4. Make the service account email address an editor to your google sheet
  5. Point `gs4_auth()` to the .json from step 3

See Appendices for further information.



## b. Non-interactive authorisations

---

### Demo:

- non-interactive authorisations
- data manipulation using R
- **2) Write to sheet for Kumu; save RData file for Shiny**



# 1) Data import and authorisations

---

- R package to read data from a Google Sheet
  - `googlesheets4`
- Authorisations
  - a. script runs locally, but needs interaction (demo)
  - b. script runs locally, non-interactive (see next slide; demo)
  - c. automate the non-interactive process (**see 3. GitHub Action; demo**)
- Data manipulation using R
- 2) Write to sheet for Kumu; save RData file for Shiny



# 3) Automation using a GitHub Action

- “GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline”  
(<https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions>)”

- see references below for understanding GitHub Actions

- GitHub Action

- where to put the .json file and how to keep it secret?

- how to set up the non-interactive authorisations to read and write to Google Sheets using `googlesheets4` within a GitHub Action?

- Reached out using the R for Data Science Slack channel



<https://www.rfordatasci.com/>

Join the Slack channel !

- R package: `tokencodr` : demo





# GitHub Action: demo

---

See: <https://github.com/jdtrat/tokencodr-google-demo>,  
[https://github.com/AnneMTreasure/stakeholder\\_map\\_project](https://github.com/AnneMTreasure/stakeholder_map_project), [https://github.com/DHCSSza/stakeholder\\_map](https://github.com/DHCSSza/stakeholder_map)

- 1) In your R Project - GitHub repo
  - a) Make sure you have `functions/` and `scripts/` directories
  - b) Add a `DESCRIPTION` file to your R Project (similar to an R package)
- 2) Encode .json file, and create a GitHub repository secret
- 3) R scripts
  - a) Add function for authorisation using `tokencodr`
  - b) Edit data import / manipulation scripts
- 4) Create your .yml file for your GitHub Action



# GitHub Action: demo

---

- **Demo**

- GitHub repo: where and how to add repository secret
- `.github/workflows/` directory with `.yml` file
  - Add file -> create new file: type `.github/workflows/` and `[filename].yml`
- The `.yml` file
- Viewing the workflow's activity (actions tab)
  - In the left sidebar, click the workflow you want to see
  - Under "Workflow runs", click the name of the run you want to see
  - Under Jobs or in the visualization graph, click the job you want to see
  - View the results of each step



# GitHub Action for Shiny: get token & secret from shinyapps.io

1



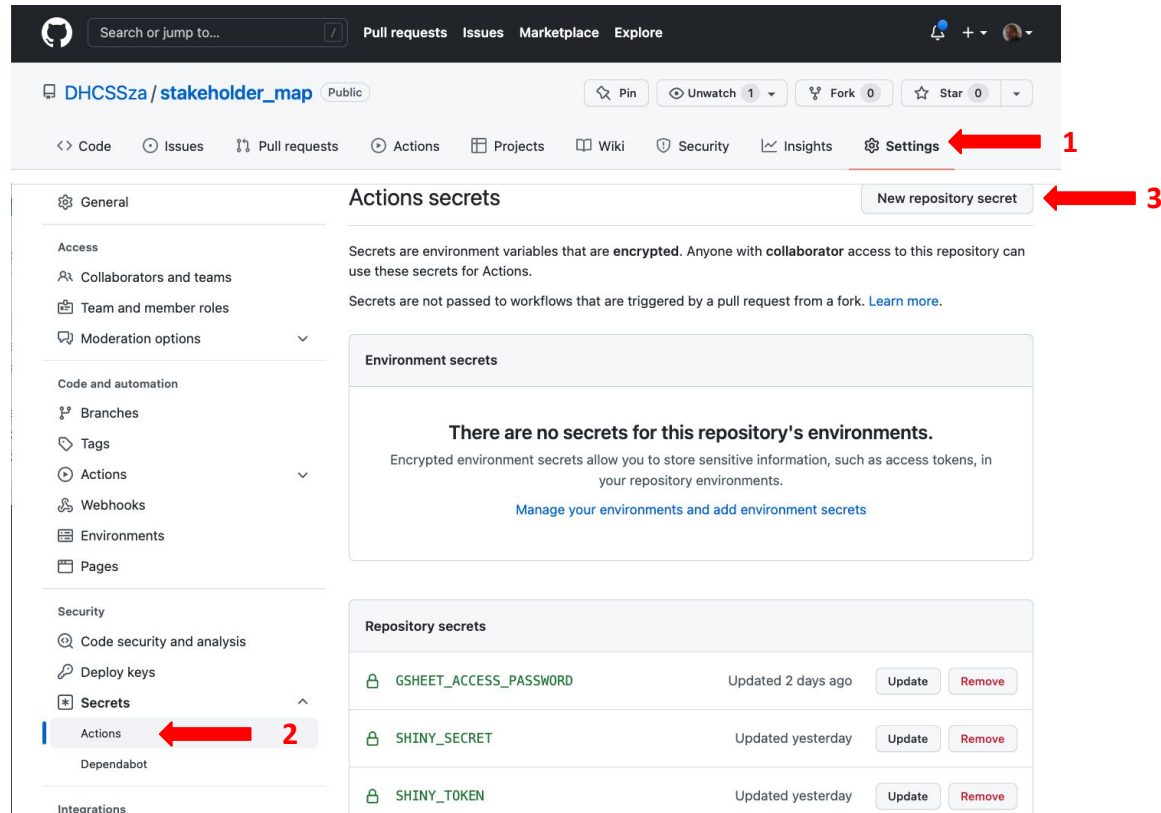
2



**Add to GitHub secrets, and use in the GitHub Action .yml - demo**



# GitHub Action for Shiny: add token & secret to GitHub repo secrets



The screenshot shows the GitHub repository settings for 'DHCSSza / stakeholder\_map'. The 'Settings' tab is selected, indicated by a red arrow labeled '1'. In the left sidebar, the 'Secrets' menu item is highlighted with a red arrow labeled '2'. The main content area is titled 'Actions secrets' and features a 'New repository secret' button, also highlighted with a red arrow labeled '3'. Below this, there are sections for 'Environment secrets' (which is empty) and 'Repository secrets' (which contains three entries: GSHEET\_ACCESS\_PASSWORD, SHINY\_SECRET, and SHINY\_TOKEN).

Search of jump to... 7 Pull requests Issues Marketplace Explore

DHCSSza / stakeholder\_map Public Pin Unwatch 1 Fork 0 Star 0

Code Issues Pull requests Actions Projects Wiki Security Insights **Settings** 1

General

Access

- Collaborators and teams
- Team and member roles
- Moderation options

Code and automation

- Branches
- Tags
- Actions 2
- Webhooks
- Environments
- Pages

Security

- Code security and analysis
- Deploy keys
- Secrets**
- Dependabot

Integrations

### Actions secrets

New repository secret 3

Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions.

Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#).

#### Environment secrets

**There are no secrets for this repository's environments.**

Encrypted environment secrets allow you to store sensitive information, such as access tokens, in your repository environments.

[Manage your environments and add environment secrets](#)

#### Repository secrets

GSHEET_ACCESS_PASSWORD	Updated 2 days ago	Update	Remove
SHINY_SECRET	Updated yesterday	Update	Remove
SHINY_TOKEN	Updated yesterday	Update	Remove



# GitHub Action for shiny: demo if time

---

- **Demo if time**
  - Secret & token
  - The .yml file



# Data visualisations updated daily

## Digital Humanities and Computational Social Sciences landscape in South Africa

The South African Centre for Digital Language Resources (SADiLaR) is a national centre supported by the Department of Science and Innovation (DSI) as part of the South African Research Infrastructure Roadmap (SARIR). SADiLaR has an enabling function, with a focus on all official languages of South Africa, supporting research and development in the domains of language technologies and language-related studies in the humanities and social sciences. Furthermore the centre has a mandate to develop digital humanities capacity in South Africa.

- Activities Map
- Projects
- People
- Datasets
- Tools
- Publications
- Training
- Learning materials

Archives    Unclassified records

Choose which record type to view

Person

Reset map view

This map shows data on Digital Humanities (DH) and Computational Social Sciences (CSS) activities and initiatives in South Africa.

Records from locations close to each other are grouped together. To ungroup, click on a green circle and see individual locations, click again to see individual beneficiaries at one location. Click on **Reset map view** to get back to the original view.



stakeholder\_map\_2    Untitled map    Untitled view

### Project 2

PROJECT

Imported

#### Project 2 description

Librarianship, Information and Museum Studies

Design and modelling

EMAIL: [sue.someone@email.com](mailto:sue.someone@email.com)

FUNDERS: Funding organisation

ORGANISATION: University of Cape Town (UCT)

#untitled-map/project-2

Legend

- Person
- Organization
- Project
- Organisation

Help    Feedback

# Resources

# Resources

---

- **Service account tokens, non-interactive, workflows for this**
  - <https://github.com/tidyverse/googledrive/issues/327> - used this for workflow
  - <https://github.com/tidyverse/googlesheets4/issues/170>
  - <https://github.com/marketplace/actions/google-sheets-secrets-action> - useful bit on workflow for service account, token, etc
  - [Creating a data pipeline with Github Actions & the {googledrive} package for the Canadian Premier League soccer data initiative!](#) - setting up GCP service account, etc
  
- **Non-interactive authorisations**
  - <https://github.com/tidyverse/googledrive/issues/239>
  - <https://cran.r-project.org/web/packages/gargle/vignettes/non-interactive-auth.html> - often recommended, especially by Jenny Bryan
  - <https://gargle.r-lib.org/articles/non-interactive-auth.html>





# Resources

---

- **Github Actions**

- Read this to understand GitHub Actions:  
<https://docs.github.com/en/actions/learn-github-actions/understanding-github-actions>

- **GitHub Actions Yaml's**

- [Running R Scripts on a Schedule with GitHub Actions](#) - really good blog post, very useful for understanding GitHub Actions and YAMLS, for a package environment though
- Events to trigger the workflow: GitHub documentation on this can be found [here](#)
- <https://github.com/simonpcouch/scheduled-commit-action/blob/master/.github/workflows/schedule-commit.yaml>
- [Running an R Script on a Schedule: Gh-Actions](#) - some useful info here
- [Creating a data pipeline with Github Actions & the {googledrive} package for the Canadian Premier League soccer data initiative!](#) - useful for GitHub Action workflow
- <https://www.rforseo.com/ressources/launch-an-r-script-using-github-actions> - simple e.g. of R script and .yaml
- [Automatic Rendering of a Plot with GitHub Actions](#) - some useful info on setting up the .yaml
- [GitHub Action with R book](#) - good, got some pointers from here about how to set up GitHub Action for R, whats going on in the .yaml file
- Look at the r-lib example YAMLS, e.g.:  
<https://github.com/r-lib/actions/blob/master/.github/workflows/check-standard.yaml>



# Resources

---

- **gargle documentation**

- [Managing tokens securely](#) - recommended by Jenny Bryan, applies to packages, but `tokenodr` works largely on the same principles for a non-package environment
- <https://gargle.r-lib.org/articles/get-api-credentials.html#service-account-token-1>

- **Secrets**

- Managing [Secrets](#) vignette by Hadley Wickam
- Packages to deal with secrets
- <https://github.com/gaborcsardi/secret>
- <https://github.com/ropensci/cyphr>

- **GitHub Actions and Shiny**

- <https://towardsdatascience.com/automating-a-covid19-report-update-and-publishing-with-github-actions-a3d64315e515#1dc4>
- <https://github.com/lucharo/COVID19/blob/master/.github/workflows/automate.yml>
- <https://stackoverflow.com/questions/67040654/r-shinyapps-deployment-error-when-doing-it-manually>
- <https://github.com/MattCowgill/djprlabourdash/blob/main/.github/workflows/deploy-shiny.yaml>
- <https://mirai-solutions.ch/techguides/cicd-pipelines-for-automatic-deployment-of-a-r-shiny-web-app.html>
- <https://mirai-solutions.ch/techguides/github-actions.html>



# Contact information

---

Twitter: @DHCSSza

Website: <https://escalator.sadilar.org/stakeholder-map/>

Email: stakeholder-map@talarify.co.za

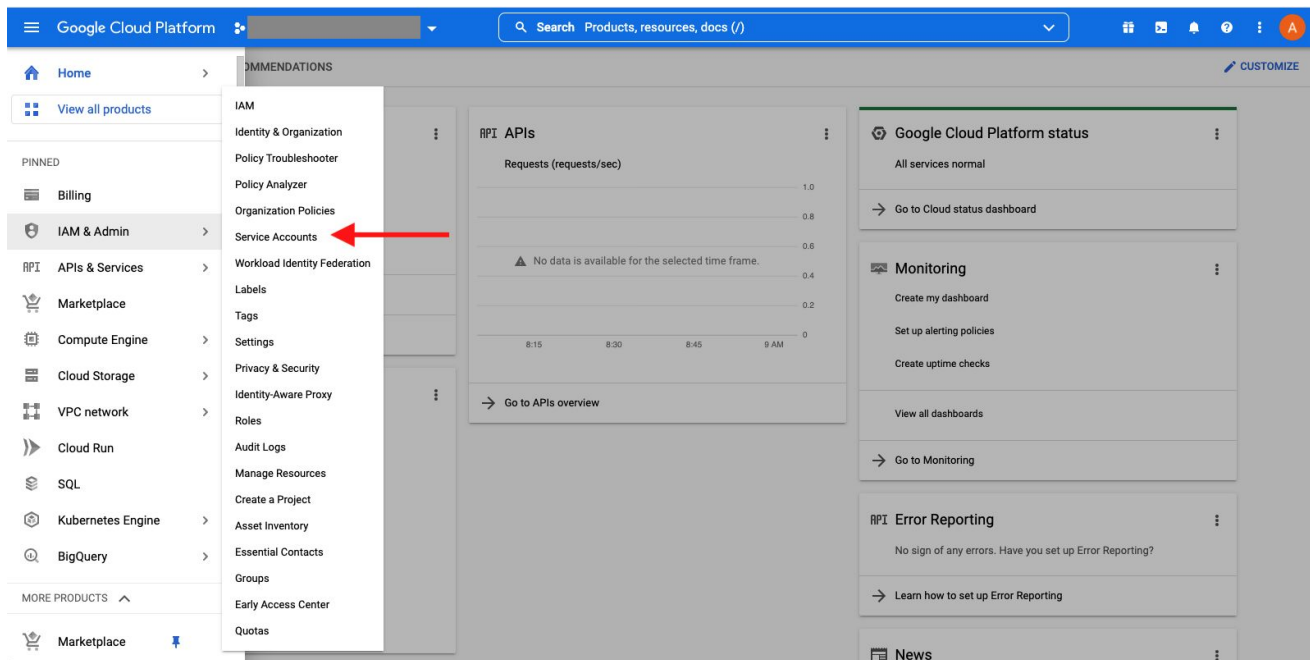
ESCALATOR: escalator@talarify.co.za



# Appendices

## b. Non-interactive authorisations

1. Create a Google Cloud Platform account
2. Create a new project, and a service account for this project



The screenshot displays the Google Cloud Platform (GCP) console interface. On the left, a navigation menu is open, showing the 'IAM & Admin' section highlighted. A red arrow points to the 'Service Accounts' option within this menu. The main content area shows a dashboard with several widgets: 'RPI APIs' (Requests (requests/sec)), 'Google Cloud Platform status' (All services normal), 'Monitoring' (Create my dashboard, Set up alerting policies, Create uptime checks), and 'RPI Error Reporting' (No sign of any errors. Have you set up Error Reporting?).



# b. Non-interactive authorisations

## 2. Create a service account

The screenshot displays the Google Cloud Platform IAM & Admin console. The left sidebar shows the navigation menu with 'Service Accounts' selected. The main content area is titled 'Create service account' and contains a three-step wizard:

- 1 Service account details**
  - Service account name:
  - Service account ID:  (with X and refresh icons)
  - Email address: `<id>@stakeholder-map-gsheets-access.iam.gserviceaccount.com` (with a copy icon)
  - Service account description:
  - [CREATE AND CONTINUE](#)
- 2 Grant this service account access to project (optional)**
- 3 Grant users access to this service account (optional)**

At the bottom of the wizard, there are two buttons: [DONE](#) and [CANCEL](#).



# b. Non-interactive authorisations

## 3. Create a key and download the .json file

Google Cloud Platform

IAM & Admin

Service accounts

Service accounts for project "[redacted]"

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more about service accounts.](#)

Organization policies can be used to secure service accounts and block risky service account features, such as automatic IAM Grants, key creation/upload, or the creation of service accounts entirely. [Learn more about service account organization policies.](#)

Filter	Email	Status	Name	Description	Key ID	Actions
	[redacted]	✔	afri-mapr-presentation-temp	A temporary service account for the afri-mapr-presentation -	No keys	<ul style="list-style-type: none"><li>Manage details</li><li>Manage permissions</li><li>Manage keys</li></ul>

Cloud Platform

Keys

Service account keys could pose a security risk if compromised. We recommend you avoid downloading service account keys and instead use the [Workload Identity Federation](#). You can learn more about the best way to authenticate service accounts on Google Cloud [here](#).

Add a new key pair

Block service account keys

ADD KEY

Key type

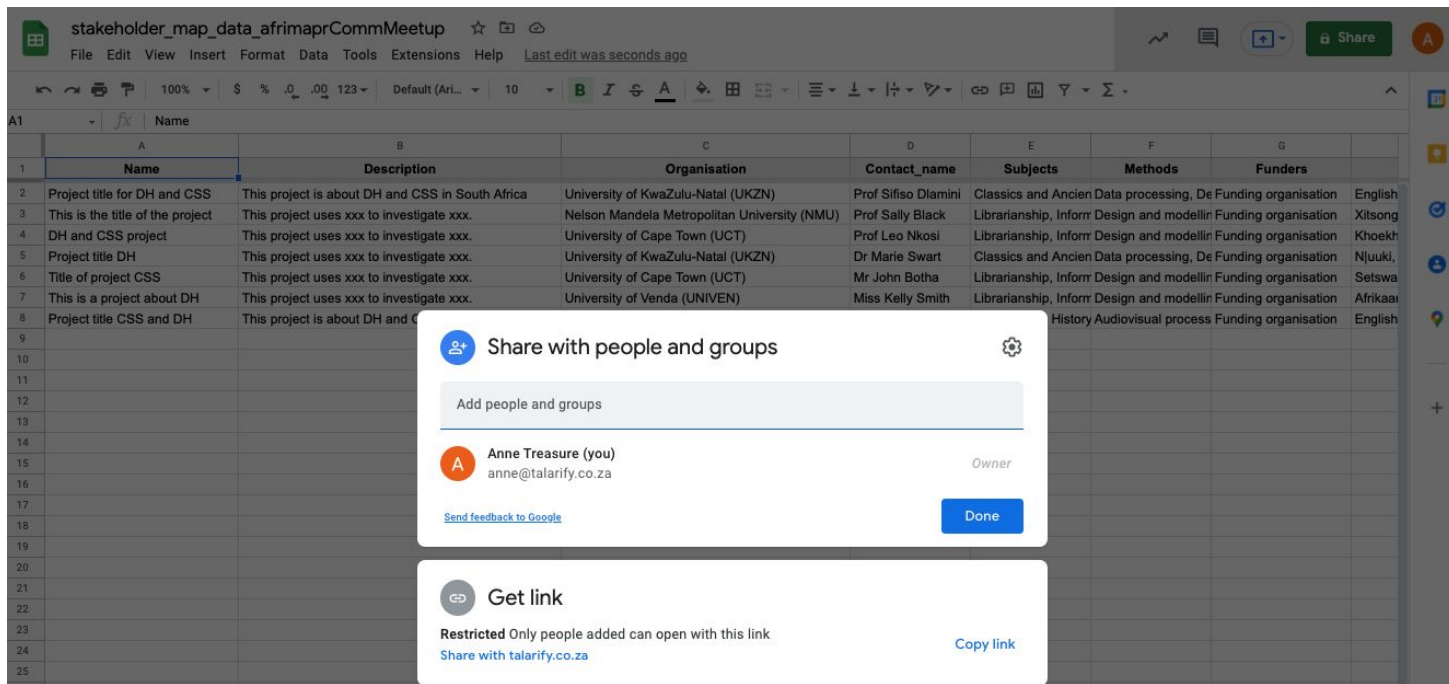
- JSON  
Recommended
- P12  
For backward compatibility with code using the P12 format

CANCEL CREATE

## b. Non-interactive authorisations

4. Make the service account email address an editor to your google sheet

- service account email address: find under 'Details' on GCP site, or in the .json file



The screenshot shows a Google Sheet with the following data:

Name	Description	Organisation	Contact_name	Subjects	Methods	Funders
Project title for DH and CSS	This project is about DH and CSS in South Africa	University of KwaZulu-Natal (UKZN)	Prof Sifiso Dlamini	Classics and Ancien Data processing, De	Funding organisation	English
This is the title of the project	This project uses xxx to investigate xxx.	Nelson Mandela Metropolitan University (NMU)	Prof Sally Black	Librarianship, Inform Design and modellir	Funding organisation	Xitsong
DH and CSS project	This project uses xxx to investigate xxx.	University of Cape Town (UCT)	Prof Leo Nkosi	Librarianship, Inform Design and modellir	Funding organisation	Khoekht
Project title DH	This project uses xxx to investigate xxx.	University of KwaZulu-Natal (UKZN)	Dr Marie Swart	Classics and Ancien Data processing, De	Funding organisation	Njuuki,
Title of project CSS	This project uses xxx to investigate xxx.	University of Cape Town (UCT)	Mr John Botha	Librarianship, Inform Design and modellir	Funding organisation	Setswa
This is a project about DH	This project uses xxx to investigate xxx.	University of Venda (UNIVEN)	Miss Kelly Smith	Librarianship, Inform Design and modellir	Funding organisation	Afrikaai
Project title CSS and DH	This project is about DH and C			History Audiovisual process	Funding organisation	English

The 'Share with people and groups' dialog box is open, showing the user 'Anne Treasure (you)' as the owner. The 'Get link' section is visible, indicating the link is 'Restricted' and only people added can open it. The link is 'Share with talarify.co.za'.





## b. Non-interactive authorisations

---

5. Point `gs4_auth()` to the `.json` from step 3

- ```
gs4_auth(email = "[your email address]",  
path = "~/[path to .json file]/[filename].json")
```



# GitHub Action & tokencoder:

## 1) In your R project

Make sure you have `functions/` and `scripts/` directories

Add a `DESCRIPTION` file; make sure the this has a valid package name, e.g. 'stakeholder.map.afrimapr'

Packages needed

The screenshot shows the RStudio interface for a project named 'stakeholder\_map\_afrimapr'. The main editor displays the content of the 'DESCRIPTION' file, which includes package metadata and dependencies. A red arrow points from the 'DESCRIPTION' text box to the first line of the file. Another red arrow points from the 'Packages needed' text box to the 'Imports' section of the file. On the right side, the 'Files' pane shows the project's directory structure, including folders for 'functions', 'scripts', and 'stakeholder\_map', and files like 'DESCRIPTION', 'README.md', and 'stakeholder\_map\_afrimapr.Rproj'. A red arrow points from the 'functions/scripts/directories' text box to the 'functions' and 'scripts' folders in the file explorer.

```
DESCRIPTION
1 Package: stakeholder.map.afrimapr
2 Title: Demo of 'tokencodr' with Google Sheets & Drive on Github Action:
3 Version: 0.0.0.9000
4 Authors@R:
5   person("Anne", "Treasure", , "anne@talarify.co.za", role = c("aut"
6     comment = c(ORCID = "0000-0002-8345-4811"))
7 Description: What the package does (one paragraph).
8 License: MIT + file LICENSE
9 Encoding: UTF-8
10 Language: es
11 LazyData: true
12 Roxygen: list(markdown = TRUE)
13 RoxygenNote: 7.1.1.9000
14 Imports:
15   googlesheets4,
16   tokencodr (>= 0.0.0.9000)
17 Remotes:
18   jdtrat/tokencodr
19
```

# GitHub Action & tokencodR:

## 2) Encode .json file, create a repository secret

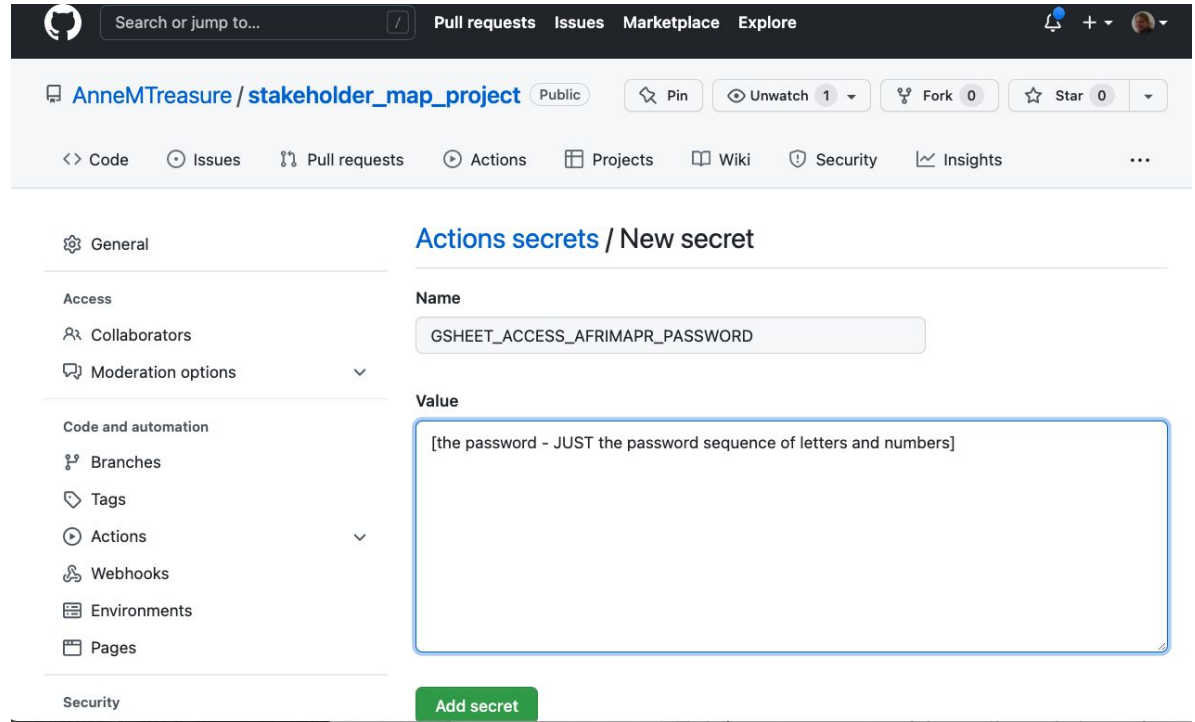
- Install the `tokencodR` package
- To encrypt a file, you call `tokencodR::create_env_pw()`. For example:
  - `create_env_pw("GSHEET_ACCESS_AFRIMAPR")`
- Copy password to `.Renviron:`
  - `usethis::edit_r_environ()`
  - paste password, insert new line, close, restart R
- Then, to encrypt the .json file and put it in a secret directory (you specify the location):
  - `encrypt_token(service = "MY_GOOGLE",  
input = "[filename].json",  
destination = "~/[path to where you want the file]/")`
- Then, for using **locally**, in your R script, set the authorisations in the `googlesheets4` package:
  - `gs4_auth(email = "[your email address]", path = "~/[path to the  
file]/.secret/GSHEET_ACCESS_AFRIMAPR")`



# GitHub Action & tokencodeR:

## 2) Encode .json file, create a repository secret

- For the **GitHub Action**: copy the password from `create_env_pw()` to your GitHub repository's secrets (e.g. if you call `create_env_pw("GSHEET_ACCESS_AFRIMAPR")`, you should create a repository secret with the name `GSHEET_ACCESS_AFRIMAPR_PASSWORD`)



The screenshot shows the GitHub repository settings page for 'stakeholder\_map\_project'. The left sidebar contains a navigation menu with the following items: General, Access, Collaborators, Moderation options, Code and automation (with sub-items: Branches, Tags, Actions, Webhooks, Environments, Pages), and Security. The main content area is titled 'Actions secrets / New secret' and contains a form with the following fields:

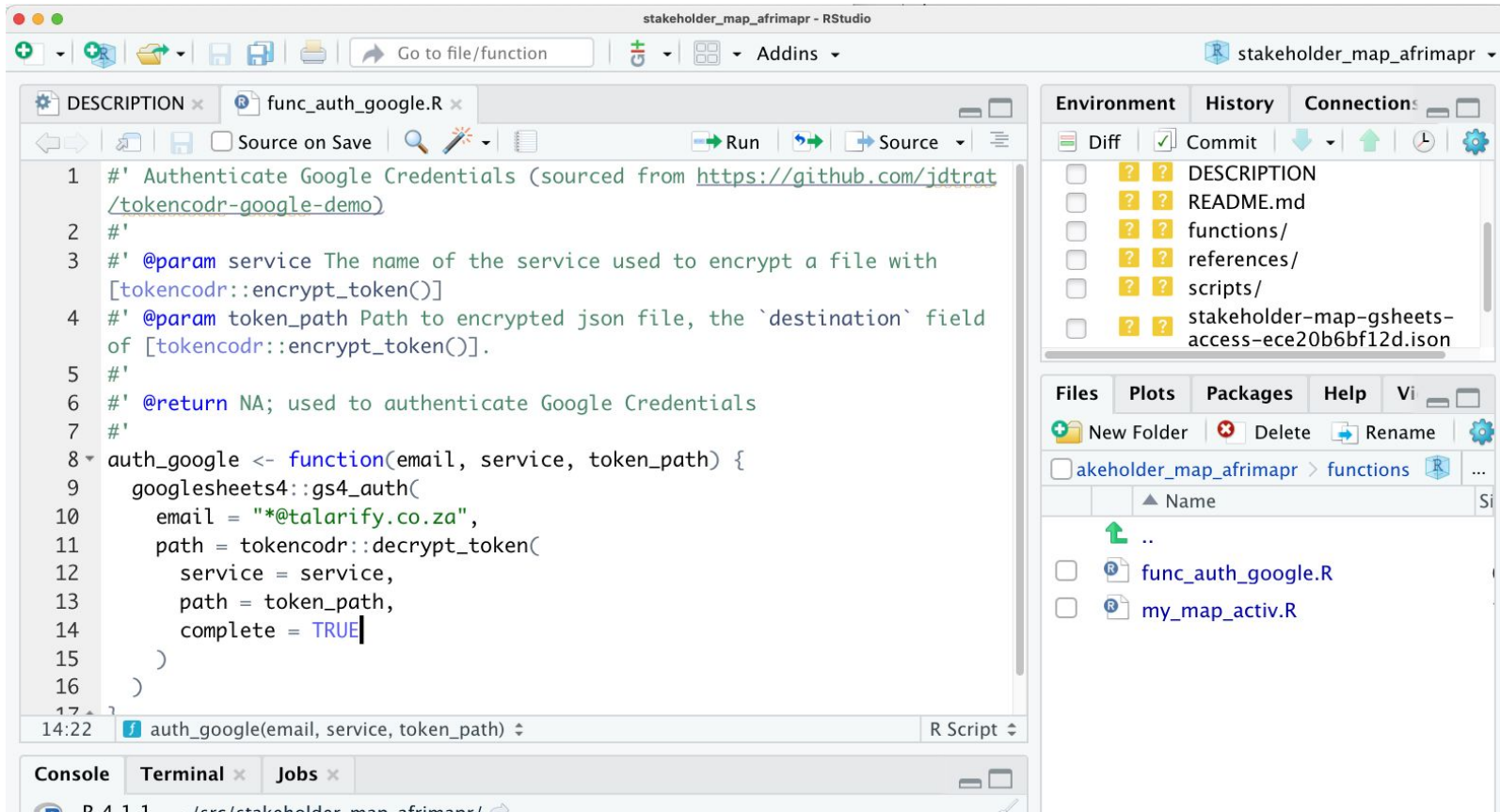
- Name:** A text input field containing the text 'GSHEET\_ACCESS\_AFRIMAPR\_PASSWORD'.
- Value:** A large text area containing the text '[the password - JUST the password sequence of letters and numbers]'.

At the bottom right of the form, there is a green button labeled 'Add secret'.

- GitHub repo:
  - > Settings
  - > Secrets
  - > Actions
  - > New repository secret

# GitHub Action: 3) R scripts: add function for authorisation using `tokencodr`

- Function: copy `func_auth_google.R` into `functions/` directory; edit as necessary



The screenshot shows the RStudio interface for a project named 'stakeholder\_map\_afrimapr'. The main editor displays the R script 'func\_auth\_google.R' with the following code:

```
1 #' Authenticate Google Credentials (sourced from https://github.com/jdtrat/tokencodr-demo)
2 #'
3 #' @param service The name of the service used to encrypt a file with
4 #' @param token_path Path to encrypted json file, the `destination` field
5 #' of [tokencodr::encrypt_token()].
6 #' @return NA; used to authenticate Google Credentials
7 #'
8 auth_google <- function(email, service, token_path) {
9   googlesheets4::gs4_auth(
10     email = "*@talarify.co.za",
11     path = tokencodr::decrypt_token(
12       service = service,
13       path = token_path,
14       complete = TRUE
15     )
16   )
17 }
```

The right-hand side of the interface shows the Environment pane with a list of files and folders, including 'DESCRIPTION', 'README.md', 'functions/', 'references/', 'scripts/', and 'stakeholder-map-gsheets-access-ece20b6bf12d.ison'. Below this is the Files pane, which shows the current directory structure, including a folder named 'functions' containing the files 'func\_auth\_google.R' and 'my\_map\_activ.R'.



# GitHub Action & tokenodeR:

## 3) R scripts: edit your scripts for authorisation (function)

Add authorisation function code to your scripts (adapted from <https://github.com/jdtrat/tokenodr-google-demo>)

# load the function

```
source("functions/func_auth_google.R")
```

# authenticate Google Service Account

```
auth_google(email = "*@talarify.co.za",  
            service = "GSHEET_ACCESS_AFRIMAPR",  
            token_path = ".secret/GSHEET_ACCESS_AFRIMAPR")
```



# GitHub Action: 4) Create your .yaml file for your GitHub Action

---

- The GitHub Action workflow is defined by the YAML file (.yaml) and is triggered by an event in your repository, manually, or at a defined schedule
  - Events to trigger the workflow
    - GitHub documentation on this can be found [here](#)
    - I chose a scheduled event for my needs - you can schedule a workflow to run at specific UTC times using POSIX cron syntax. You set this in the GitHub Action .yaml file
- In your GitHub repo, create the `.github/workflows/` directory to store your workflow files
- In the `.github/workflows/` directory, create a new file `[filename].yaml`
  - my .yaml defines a workflow that runs my R script



# GitHub Action for Shiny: .yml

---

## Add to jobs:

```
env:  
  # set as environment variables  
  SHINY_TOKEN: ${ secrets.SHINY_TOKEN }  
  SHINY_SECRET: ${ secrets.SHINY_SECRET }  
  
- name: Connect to Shiny  
  run: |  
    shiny_token = Sys.getenv("SHINY_TOKEN")  
    shiny_secret = Sys.getenv("SHINY_SECRET")  
    rsconnect::setAccountInfo(name='anne-treasure', token=shiny_token, secret=shiny_secret)  
  shell: Rscript {0}  
  
- name: Uploading to shinyapps.io  
  run: rsconnect::deployApp(appDir = "shiny_stakeholder_map",  
    appFiles=c('app.R', 'shiny_data.RData', 'my_map_activ.R'),  
    account = 'anne-treasure', server = 'shinyapps.io',  
    getOption("rsconnect.force.update.apps", TRUE))  
  shell: Rscript {0}
```





**Thank you!**