SHINY APP DEVELOPMENT WITH GOLEM

Handré Williams

NPCartel.ai

AGENDA

Introduction

Should I use Shiny?

Pitfalls with Shiny

Golem

NPCartel.ai

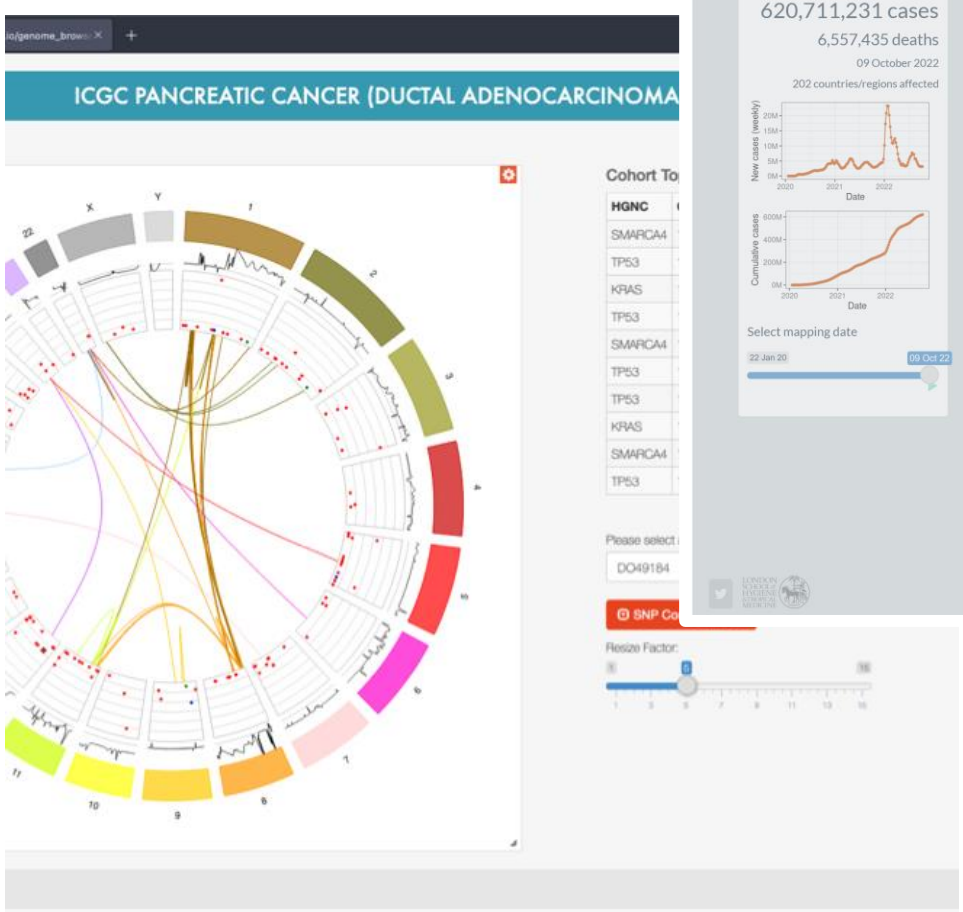
INTRODUCTION

NPCartel.ai

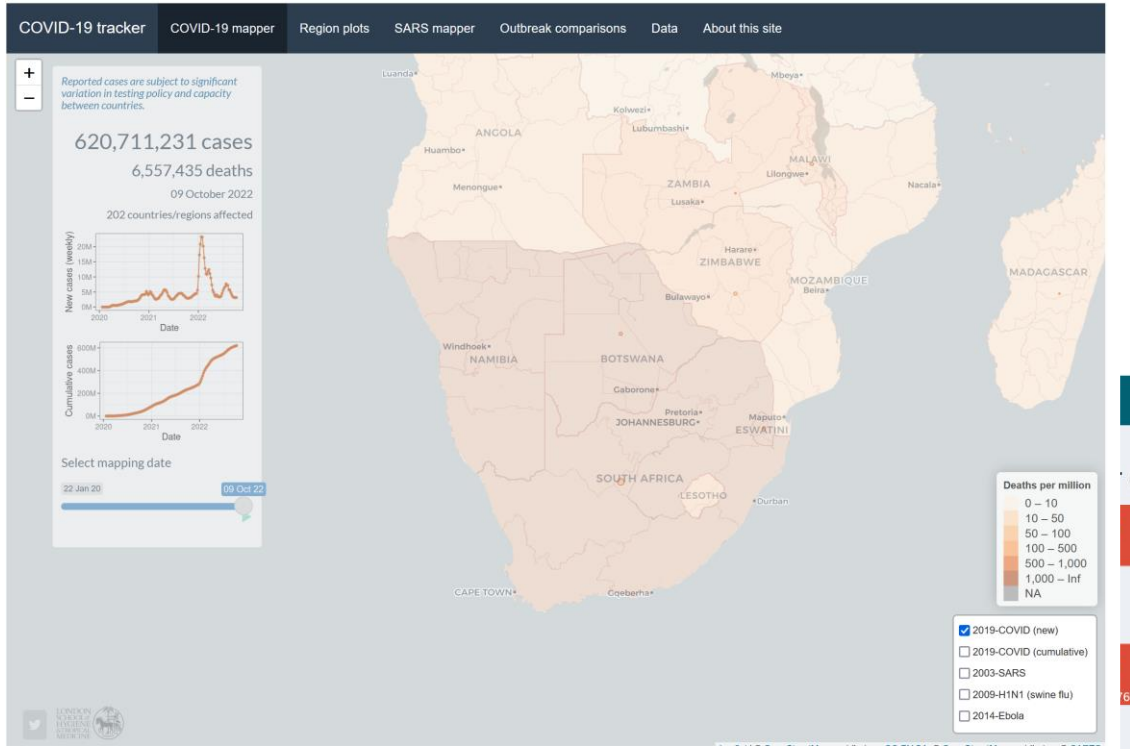
SHINY

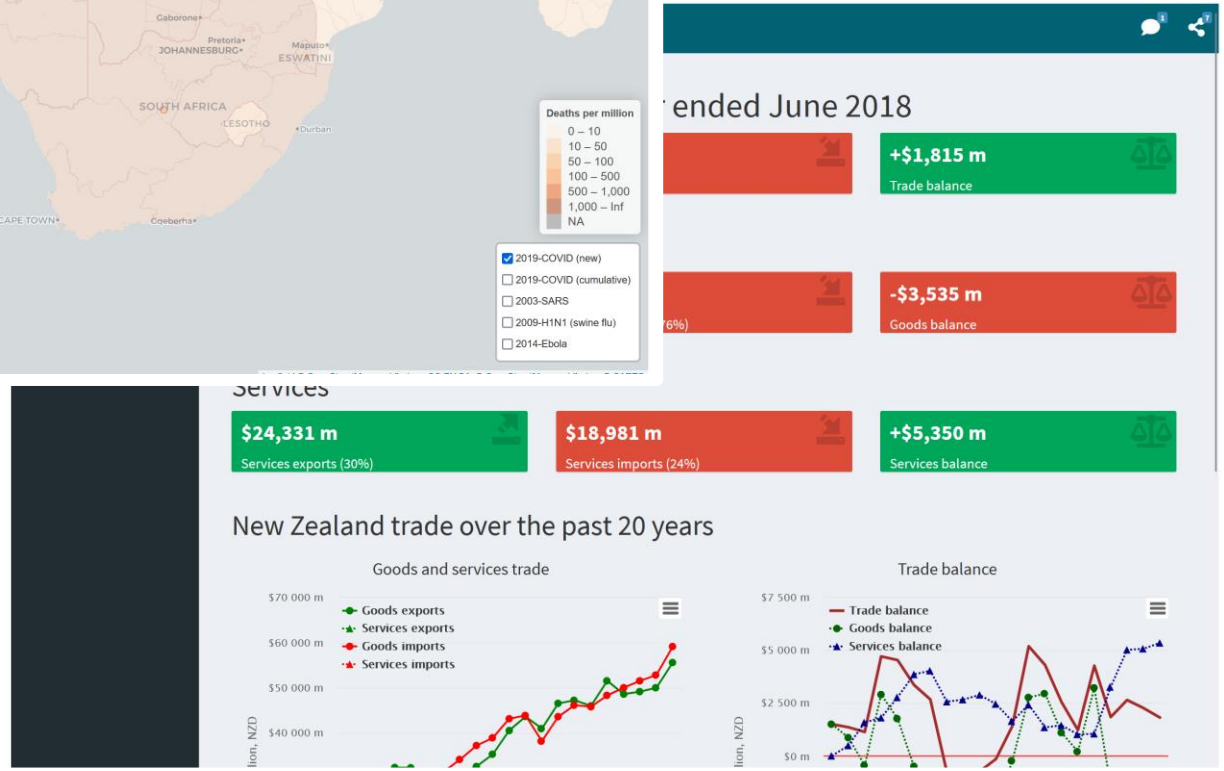
Shiny is an R package that makes it easy to build interactive web apps straight from R. You can host standalone apps on a webpage or embed them in R Markdown documents or build dashboards

https://shiny.rstudio.com/



https://shiny.rstudio.com/





https://shiny.rstudio.com/gallery/covid19-tracker.html

https://shiny.rstudio.com/gallery/nz-trade-dash.html

NPCartel.ai

SHOULD I USE SHINY?

Yes	No
Clean, dynamic dashboard to visualise your data that can be refreshed easily	You are building a massive site that will get a lot of traffic
You are building a tool to interact with R code. Some end users might not be R savvy, so you need a UI	You are linking to large databases
You are building dashboards and reports for your business, to communicate data to leadership	Have access to a full stack dev team
	Power BI or Tableau would be sufficient

PITFALLS WITH SHINY

Code structure doesn’t need to be optimised

Out of memory crashes

Dependencies are difficult to manage. Updates can break the application

Doesn’t fail gracefully in general

GOLEM

{golem} is an opinionated framework for building production-grade shiny applications.

It is a toolkit for simplifying the creation, development and deployment of a shiny application

NPCartel.ai

https://github.com/ThinkR-open/golem

GOLEM

A shiny app should be an R package

Metadata and dependency management should be present since the start

File and folder structures are strict and enforced

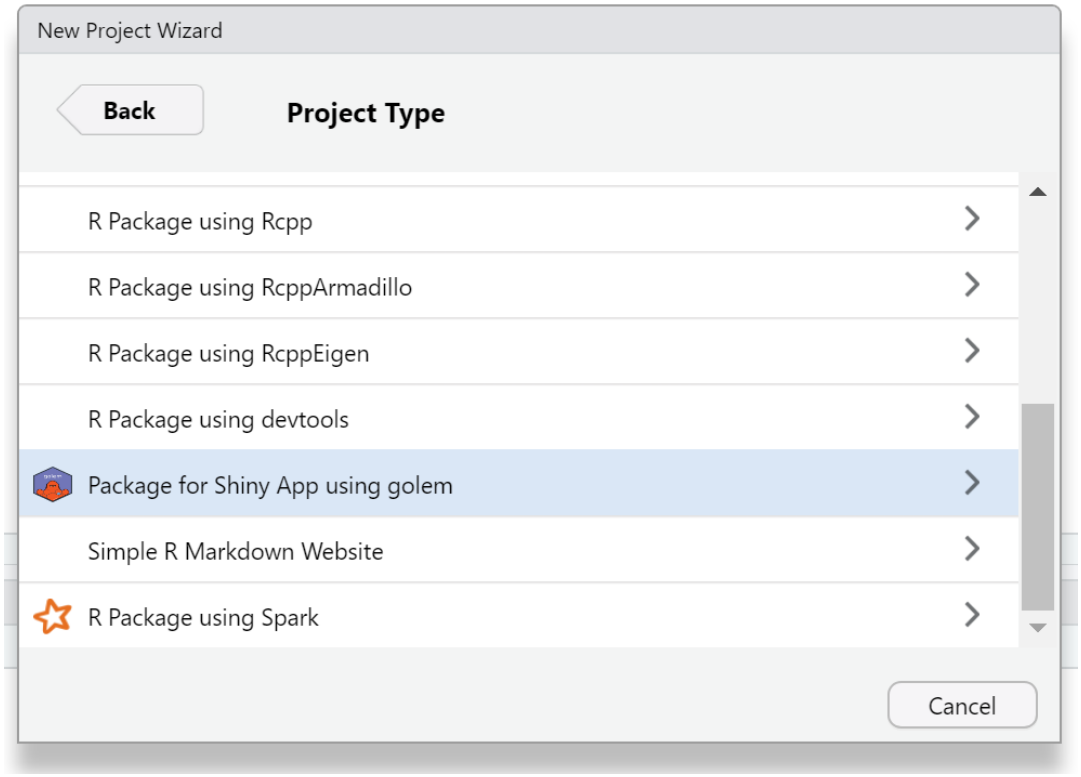
Testing are important and should be integrated

Documentation is important

NPCartel.ai

https://github.com/ThinkR-open/golem

CREATE A GOLEM PACKAGE



NPCartel.ai

CREATE A GOLEM PACKAGE

Console | Terminal | Background Jobs

R 4.2.1 · ~/npc.demo/

```
> fs::dir_tree(".")
.
├── DESCRIPTION
├── dev
│   ├── 01_start.R
│   ├── 02_dev.R
│   ├── 03_deploy.R
│   └── run_dev.R
├── inst
│   ├── app
│   │   └── www
│   │       └── favicon.ico
│   └── golem-config.yml
├── man
│   └── run_app.Rd
├── NAMESPACE
├── npc.demo.Rproj
└── R
    ├── app_config.R
    ├── app_server.R
    ├── app_ui.R
    └── run_app.R
>
```

NPCartel.ai

ADD METADATA

01_start.R

```r
# Building a Prod-Ready, Robust Shiny Application.
#
# README: each step of the dev files is optional, and you don't have to
# fill every dev scripts before getting started.
# 01_start.R should be filled at start.
# 02_dev.R should be used to keep track of your development during the project.
# 03_deploy.R should be used once you need to deploy your app.
#
#
########################################
#### CURRENT FILE: ON START SCRIPT #####
########################################

## Fill the DESCRIPTION ----
## Add meta data about your application
##
## /!\ Note: if you want to change the name of your app during development,
## either re-run this function, call golem::set_golem_name(), or don't forget
## to change the name in the app_sys() function in app_config.R /!\
##
golem::fill_desc(
  pkg_name = "npc.demo", # The Name of the package containing the App
  pkg_title = "PKG_TITLE", # The Title of the package containing the App
  pkg_description = "PKG_DESC.", # The Description of the package containing the App
  author_first_name = "AUTHOR_FIRST", # Your First Name
  author_last_name = "AUTHOR_LAST", # Your Last Name
  author_email = "AUTHOR@MAIL.COM", # Your Email
  repo_url = NULL # The URL of the GitHub Repo (optional)
)
```

9:2 (Top Level) R Script

NPCartel.ai

ADD METADATA

01_start.R

```r
## Set {golem} options ----
golem::set_golem_options()

## Create Common Files ----
## See ?usethis for more information
usethis::use_mit_license("Golem User") # You can set another license here
usethis::use_readme_rmd(open = FALSE)
# Note that `contact` is required since usethis version 2.1.5
# If your {usethis} version is older, you can remove that param
usethis::use_code_of_conduct(contact = "Golem User")
usethis::use_lifecycle_badge("Experimental")
usethis::use_news_md(open = FALSE)

## Use git ----
usethis::use_git()

## Init Testing Infrastructure ----
## Create a template for tests
golem::use_recommended_tests()

## Favicon ----
# If you want to change the favicon (default is golem's one)
golem::use_favicon() # path = "path/to/ico". Can be an online file.
# golem::remove_favicon() # Uncomment to remove the default favicon

## Add helper functions ----
golem::use_utils_ui(with_test = TRUE)
golem::use_utils_server(with_test = TRUE)

# You're now set! ----
```

9:2 (Top Level) R Script

NPCartel.ai

DEVELOPMENT

01_start.R | 02_dev.R

Source on Save | Run | Source

```r
# Building a Prod-Ready, Robust Shiny Application.
#
# README: each step of the dev files is optional, and you don't have to
# fill every dev scripts before getting started.
# 01_start.R should be filled at start.
# 02_dev.R should be used to keep track of your development during the project.
# 03_deploy.R should be used once you need to deploy your app.
#
#
##################################
#### CURRENT FILE: DEV SCRIPT #####
##################################

# Engineering

## Dependencies ----
## Amend DESCRIPTION with dependencies read from package code parsing
## install.package('attachment') # if needed.
attachment::att_amend_desc()

## Add modules ----
## Create a module infrastructure in R/
golem::add_module(name = "name_of_module1", with_test = TRUE) # Name of the module
golem::add_module(name = "name_of_module2", with_test = TRUE) # Name of the module

## Add helper functions ----
## Creates fct_* and utils_*
golem::add_fct("helpers", with_test = TRUE)
golem::add_utils("helpers", with_test = TRUE)

```

10:36 (Untitled) R Script

NPCartel.ai

DEVELOPMENT

01_start.R | 02_dev.R

Source on Save | Run | Source

```r
## External resources
## Creates .js and .css files at inst/app/www
golem::add_js_file("script")
golem::add_js_handler("handlers")
golem::add_css_file("custom")
golem::add_sass_file("custom")

## Add internal datasets ----
## If you have data in your package
usethis::use_data_raw(name = "my_dataset", open = FALSE)

## Tests ----
## Add one line by test you want to create
usethis::use_test("app")

# Documentation

## Vignette ----
usethis::use_vignette("npc.demo")
devtools::build_vignettes()

## Code Coverage----
## Set the code coverage service ("codecov" or "coveralls")
usethis::use_coverage()

# Create a summary readme for the testthat subdirectory
covrpage::covrpage()

## CI ----
```

10:36 (Untitled) R Script

NPCartel.ai

DEVELOPMENT

01_start.R | 02_dev.R

Source on Save | Run | Source

```r
## Use this part of the script if you need to set up a CI
## service for your application
##
## (You'll need GitHub there)
usethis::use_github()

# GitHub Actions
usethis::use_github_action()
# Chose one of the three
# See https://usethis.r-lib.org/reference/use_github_action.html
usethis::use_github_action_check_release()
usethis::use_github_action_check_standard()
usethis::use_github_action_check_full()
# Add action for PR
usethis::use_github_action_pr_commands()

# Travis CI
usethis::use_travis()
usethis::use_travis_badge()
# AppVeyor
usethis::use_appveyor()
usethis::use_appveyor_badge()
# Circle CI
usethis::use_circleci()
usethis::use_circleci_badge()
# Jenkins
usethis::use_jenkins()
# GitLab CI
usethis::use_gitlab_ci()
```

10:36 (Untitled) R Script

NPCartel.ai

DEPLOY

01_start.R | 02_dev.R | 03_deploy.R

Source on Save | Run | Source

```r
#####################################
#### CURRENT FILE: DEPLOY SCRIPT #####
#####################################

# Test your app

## Run checks ----
## Check the package before sending to prod
devtools::check()
rhub::check_for_cran()

# Deploy

## Local, CRAN or Package Manager ----
## This will build a tar.gz that can be installed locally,
## sent to CRAN, or to a package manager
devtools::build()

## RStudio ----
## If you want to deploy on RStudio related platforms
golem::add_rstudioconnect_file()
golem::add_shinyappsio_file()
golem::add_shinyserver_file()

## Docker ----
## If you want to deploy via a generic Dockerfile
golem::add_dockerfile_with_renv()

## If you want to deploy to ShinyProxy
golem::add_dockerfile_with_renv_shinyproxy()
```

5:40 (Top Level) R Script

NPCartel.ai

CREATE A GOLEM PACKAGE

Console | Terminal | Background Jobs

R 4.2.1 · ~/npc.demo/

```
> fs::dir_tree(".")
.
├── DESCRIPTION
├── dev
│   ├── 01_start.R
│   ├── 02_dev.R
│   ├── 03_deploy.R
│   └── run_dev.R
├── inst
│   ├── app
│   │   └── www
│   │       └── favicon.ico
│   └── golem-config.yml
├── man
│   └── run_app.Rd
├── NAMESPACE
├── npc.demo.Rproj
└── R
    ├── app_config.R
    ├── app_server.R
    ├── app_ui.R
    └── run_app.R
>
```

Meta Data

man: app documentation, will be automatically generated.

app_server.R and app_ui.R – Shiny UI and Server files

NPCartel.ai

YML

01_start.R | 02_dev.R | 03_deploy.R | golem-config.yml | NAMESPACE

```yml
default:
  golem_name: npc.demo
  golem_version: 0.0.0.9000
  app_prod: no
production:
  app_prod: yes
dev:
  golem_wd: !expr here::here()

```

9:1 YAML

NPCartel.ai

CREATE A GOLEM PACKAGE

Console | Terminal | Background Jobs

R 4.2.1 · ~/npc.demo/

```
> fs::dir_tree(".")
.
├── DESCRIPTION
├── dev
│   ├── 01_start.R
│   ├── 02_dev.R
│   ├── 03_deploy.R
│   └── run_dev.R
├── inst
│   ├── app
│   │   └── www
│   │       └── favicon.ico
│   └── golem-config.yml
├── man
│   └── run_app.Rd
├── NAMESPACE
├── npc.demo.Rproj
└── R
    ├── app_config.R
    ├── app_server.R
    ├── app_ui.R
    └── run_app.R
>
```

Meta Data

man: app documentation, will be automatically generated.

app_server.R and app_ui.R – Shiny UI and Server files

NPCartel.ai

NAMESPACE

01_start.R | 02_dev.R | 03_deploy.R | golem-config.yml | NAMESPACE

This document is read only.

```
# Generated by roxygen2: do not edit by hand

export(run_app)
import(shiny)
importFrom(golem,activate_js)
importFrom(golem,add_resource_path)
importFrom(golem,bundle_resources)
importFrom(golem,favicon)
importFrom(golem,with_golem_options)
importFrom(shiny,shinyApp)
```

11:1 NAMESPACE

NPCartel.ai

CREATE A GOLEM PACKAGE

Console | Terminal | Background Jobs

R 4.2.1 · ~/npc.demo/

```
> fs::dir_tree("")
.
├── DESCRIPTION
├── dev
│   ├── 01_start.R
│   ├── 02_dev.R
│   ├── 03_deploy.R
│   └── run_dev.R
├── inst
│   ├── app
│   │   └── www
│   │       └── favicon.ico
│   └── golem-config.yml
├── man
│   └── run_app.Rd
├── NAMESPACE
├── npc.demo.Rproj
└── R
    ├── app_config.R
    ├── app_server.R
    ├── app_ui.R
    └── run_app.R
>
```

Meta Data

man: app documentation, will be automatically generated.

app_server.R and app_ui.R – Shiny UI and Server files

NPCartel.ai

MODULES

A module is a part of a Shiny App, which will be included in the bigger application.

It's used to split your app in smaller pieces

- Maintainability
- Add some order to the app

It can be reused

MODULES AND SUB-MODULES

Modules and sub-modules will be created in the R folder

```
golem::add_module("login")
```

Will create a R/mod_login.R file

MODULES AND SUB-MODULES

There are 2 types in Golem. They are are utils_* and fct_* files:

- utils_* files contain small functions that might be used several times in the application.
- fct_* files contain larger functions that are more central to the application.

```
# Adding fct_ and utils_ files to the project
golem::add_fct( “connect" )
golem::add_utils( "helpers" )
```

The first will create a R/fct_connect.R file.

The second will create a R/utils_helpers.R file.

MODULES AND SUB-MODULES

Sub-modules can be used in Modules

```
# Creating the fct_ and utils_ file along the module creation if they are only applicable to the one module
golem::add_module( name = “login", fct = "connect", utils = "wrapper“ )
```

Will create:

R/mod_login.R

R/mod_login_fct_connect.R

R/mod_login_utils_wrapper.R

TESTING

```
## Init Testing Infrastructure ----
## Create a template for tests
golem::use_recommended_tests()

## Custom Tests ----
## Add one line by test you want to create
usethis::use_test( "app" )
```

NPCartel.ai

TESTING

With test_that:

```
test_that( “test set 1", {
test1a
test1b
}
)
```

```
test_that(“test set 2", {
test2a
test2b
}
)
```

NPCartel.ai

SUMMARY

Golem helps us with:

- Standardised framework to build Shiny applications
- Packaging the application for easier deployment
- Integrated testing
- Integration with CI frameworks
- Documentation & Maintainability

NPCartel.ai

RESOURCES

https://golemverse.org/

https://engineering-shiny.org

https://github.com/ThinkR-open/golem

NPCartel.ai

THANK YOU

Handre Williams

E: handre@npcartel.ai

C: 082 765 4749

https://www.linkedin.com/in/handre/

NPCartel.ai