Shiny is a framework for creating web applications using R code. It is a powerful tool for data exploration, visualisation, analysis and the development of research questions. At time of writing (February 2023), the current version of Shiny is 1.7.4 (<https://cran.r-project.org/web/packages/shiny/index.html>) and can be installed just as any other R package:

install.packages("shiny")

library(shiny)

The simplest way to start working on a Shiny app is to create a new directory for your app and put a single file called app.R in it. This app.R file will be used to tell Shiny both how the app should look, and how it should behave.

The two key components of every Shiny app are the user interface (UI) which defines how the app looks, and the server function which defines how the app works. The framework for every app is:

ui <- fluidPage() { } #the user interface object

server <- function(input, output, session) { } #the server function

shinyApp(ui = ui, server = server) #the shinyApp function to launch the app

The the shinyApp function actually creates Shiny app objects from an explicit UI/server pair. This framework can be created by typing *shinyapp* and then pressing Shift+Tab.

The UI will contain various inputs (\*Input(inputId = "nameofinput", ...)) and corresponding outputs (\*Output()) whose Id’s must be unique. In the server, these inputs and outputs are referred to as input$nameofinputId and "outputname" (outputs often in a render\*() function). Shiny uses reactive programming to automatically update outputs when inputs change:

* observe({}): side-effects and no output
* reactive({}): output and no side-effects
  + observeEvent(input$nameofinputId, {}) / eventReactive(...) to run code only if a specific input changes (or use isolate())
    - keep reactive values separate to ensure only relevant code runs when changed by user
  + ", priority = N" option to ensure order each value will be updated
* render\*({}): meaningful data-related outputs

Note: Prior to version 0.10.2, Shiny did not support single-file apps and the UI object and server function needed to be contained in separate scripts called ui.R and server.R, respectively. This functionality is still supported in Shiny.

You can run the app by:

* clicking the Run App button in the document toolbar
* typing Cmd/Ctrl + Shift + Enter
* If you’re not using RStudio, you can source() the whole script, or call shiny::runApp() with the path to the directory containing app.R

You can stop the app and return access to the console by:

* clicking the stop sign icon on the R console toolbar
* clicking on the console, then pressing Esc (or press Ctrl + C if you’re not using RStudio).
* closing the Shiny app window.

Tips:

* Other packages that are commonly used in Shiny apps: shinyjs, shinyWidgets, shinycssloaders, dplyr, tidyr, ggplot2, plotly, leaflet, sf, rgdal, data.table, RColorBrewer.
* It is good practice to keep data cleaning and other preparation in a separate script to app.R, improving readability.
* To increase iteration speed consider turning on autoreload and run the app in a background job (https://github.com/sol-eng/background-jobs/tree/master/shiny-job). As soon as you save a file your app will re-launch (no need to close and restart). The chief disadvantage of this technique is that it’s considerably harder to debug because the app is running in a separate process. Example workflow:
  + Write some code and press Cmd/Ctrl + S to save the file.
  + Interactively experiment.
  + Repeat.
* Shiny documentation and resources:
  + [shiny-cheatsheet copy (rstudio.com)](https://shiny.rstudio.com/images/shiny-cheatsheet.pdf)
  + <https://mastering-shiny.org/>
  + <https://shiny.rstudio.com/tutorial/>
  + <http://zevross.com/blog/2016/04/19/r-powered-web-applications-with-shiny-a-tutorial-and-cheat-sheet-with-40-example-apps/>
  + <https://github.com/rstudio/shiny-examples>
  + <https://shiny.rstudio.com/articles/layout-guide.html> (templates)
  + <https://shiny.rstudio.com/gallery/widget-gallery.html> (widget gallery)

Shiny apps related to the ADILA project:

* [wcuningham/MIDAS\_AMU: Visualisations of the IQVIA MIDAS global antimicrobial use data (github.com)](https://github.com/wcuningham/MIDAS_AMU)