Introduction to Shiny day 3

Posit Implementation Programme
05/10/2023

Learning Outcomes – Day 3

- CSS style script
- Publishing a Shiny dashboard
  - PRA password protection
- Making your dashboard more efficient
  - profvis package

Recap styles

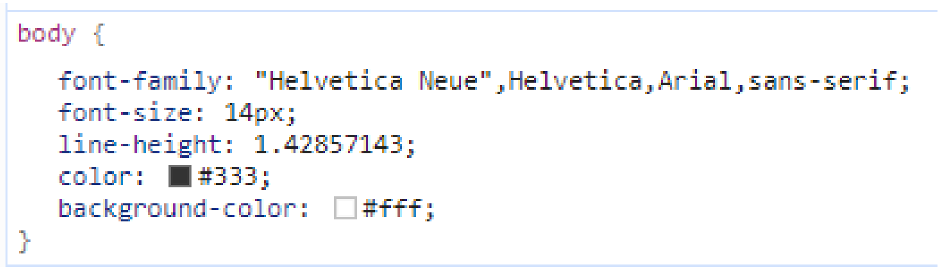
- There are several ready made Shiny themes that you can use in your dashboard, see https://shiny.rstudio.com/gallery/shiny-theme-selector.html for full list.

- You can use tags, e.g. **h1** and **tags$i**, to change your titles and text

- You can use **icon()** to add any of the icons found on Font Awesome and Glyphicon: https://fontawesome.com/v5.15/icons?d=gallery&p=2 https://getbootstrap.com/docs/3.3/components/#glyphicons

CSS style script

- CSS code can be included as part of the UI however, most keep it in a separate script (style.css) that is sourced into the app.
- The style script uses CSS code to change the look of the dashboard. Some things you can change are:
  - Colours of navbar, titlePanel, widgets and drop-downs etc.
  - Font, colour, size of text and tags
  - Image and icon sizes and positions
- CSS code uses HTML tags followed by a curly bracket and commands within the curly bracket.

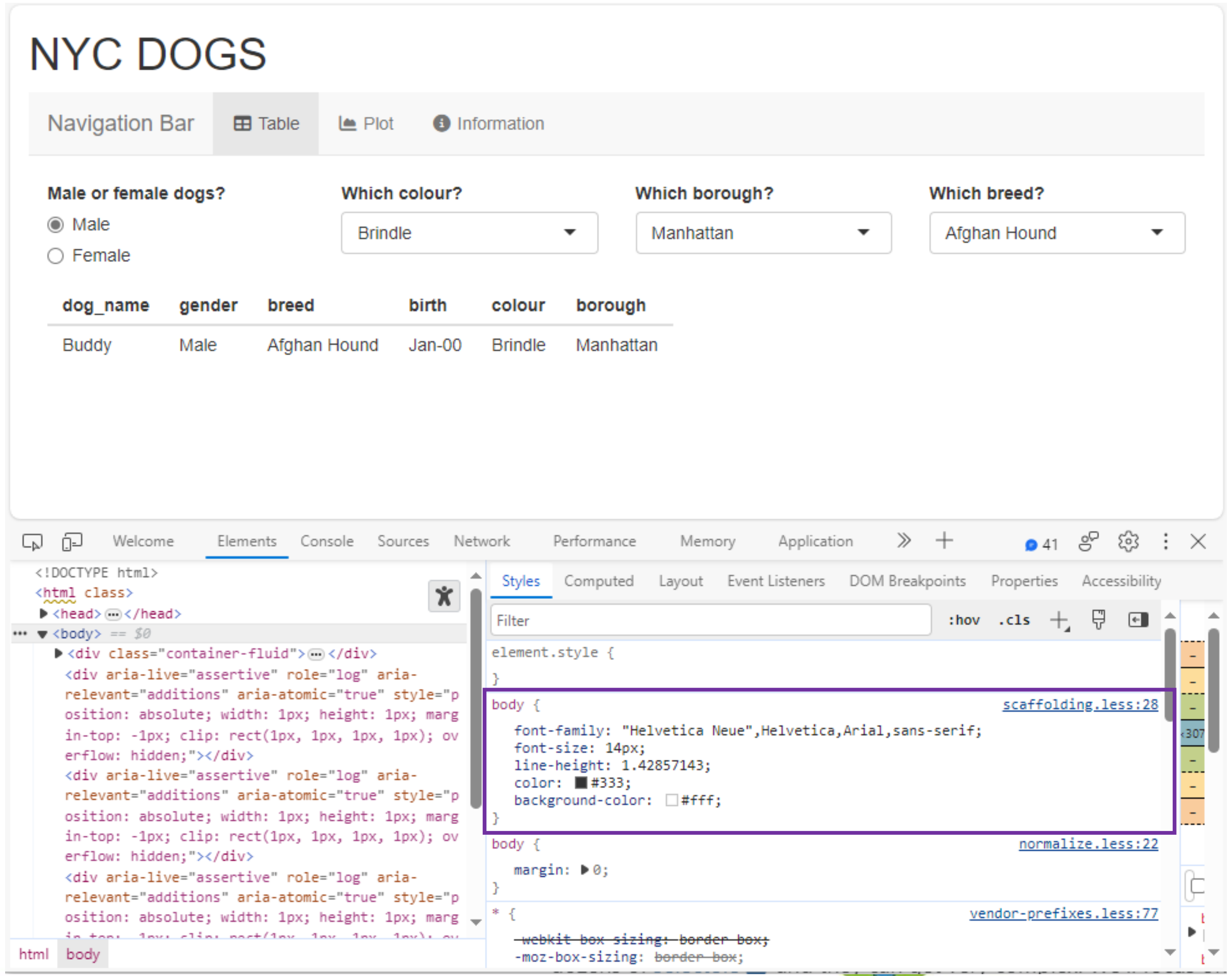


```
body {
   font-family: "Helvetica Neue",Helvetica,Arial,sans-serif;
   font-size: 14px;
   line-height: 1.42857143;
   color: ■#333;
   background-color: □#fff;
}
```

NOTE: color in CSS refers to text, and background-color refers to layout

CSS style script

- You can change items in the dashboard by finding their HTML tag.
  - **Run app -> Open in Browser -> Right click -> Inspect**
  - “Elements” show the apps html code and “Styles” show the CSS style code for each element
- Expand the <body> section of the HTML code in “Elements”.
  - Hovering over code will highlight the element in the app and clicking on it will show its CSS style code which you can copy into the CSS style script and change in Posit.



Male or female dogs? [✓ Male] Female

Which colour? Brindle

Which borough? Manhattan

Which breed? Afghan Hound

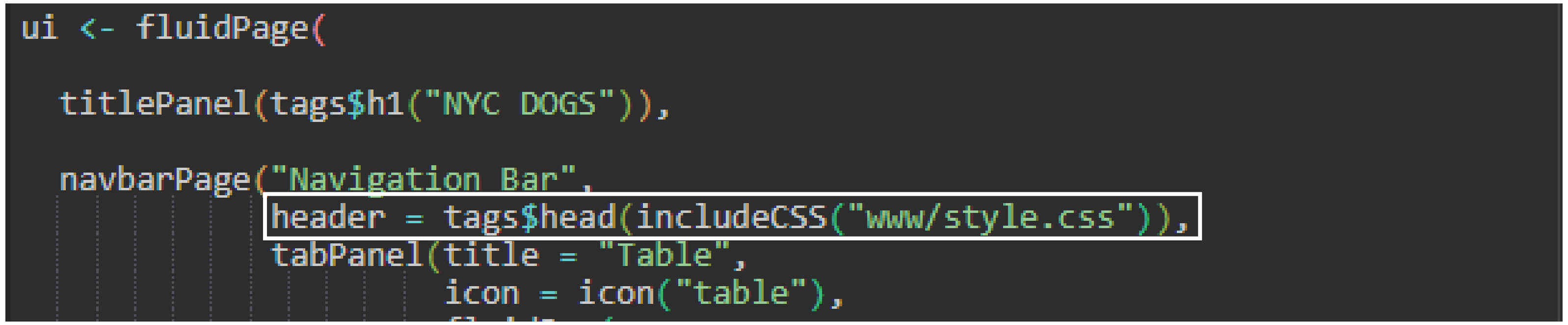
dog_name	gender	breed	birth	colour	borough
Buddy	Male	Afghan Hound	Jan-00	Brindle	Manhattan

```
<!DOCTYPE html>
<html class>
▶<head>…</head>
▼<body> == $0
  ▶<div class="container-fluid">…</div>
   <div aria-live="assertive" role="log" aria-
   relevant="additions" aria-atomic="true" style="p
   osition: absolute; width: 1px; height: 1px; marg
   in-top: -1px; clip: rect(1px, 1px, 1px, 1px); ov
   erflow: hidden;"></div>
   <div aria-live="assertive" role="log" aria-
   relevant="additions" aria-atomic="true" style="p
   osition: absolute; width: 1px; height: 1px; marg
   in-top: -1px; clip: rect(1px, 1px, 1px, 1px); ov
   erflow: hidden;"></div>
   <div aria-live="assertive" role="log" aria-
   relevant="additions" aria-atomic="true" style="p
   osition: absolute; width: 1px; height: 1px; marg
```

```
element.style {
}
body {                                                    scaffolding.less:28
  font-family: "Helvetica Neue",Helvetica,Arial,sans-serif;
  font-size: 14px;
  line-height: 1.42857143;
  color: ■#333;
  background-color: □#fff;
}
body {                                                    normalize.less:22
  margin: ▶0;
}
* {                                                       vendor-prefixes.less:77
  ~~-webkit-box-sizing: border-box;~~
  -moz-box-sizing: ~~border-box~~;
```

CSS style script Exercise

- Create a folder in the same folder as your app from day one, name it **“www”**.
- Copy the style.css script into your new “www” folder.
- Add code to UI to source css file.



```
ui <- fluidPage(

  titlePanel(tags$h1("NYC DOGS")),

  navbarPage("Navigation Bar",
             header = tags$head(includeCSS("www/style.css")),
             tabPanel(title = "Table",
                      icon = icon("table"),
```

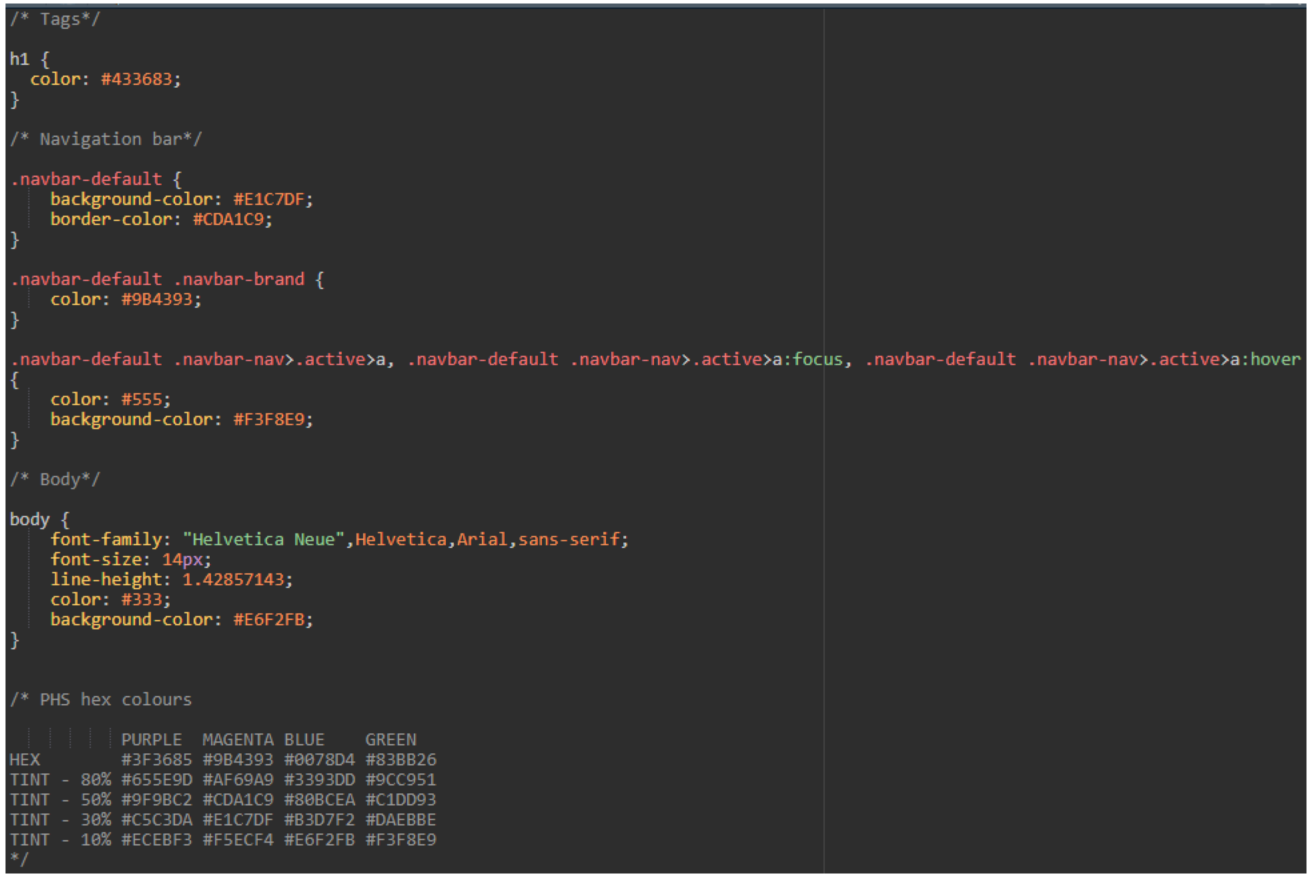
CSS style script Exercise

- **"Run app",** open in browser, inspect, and complete the exercises

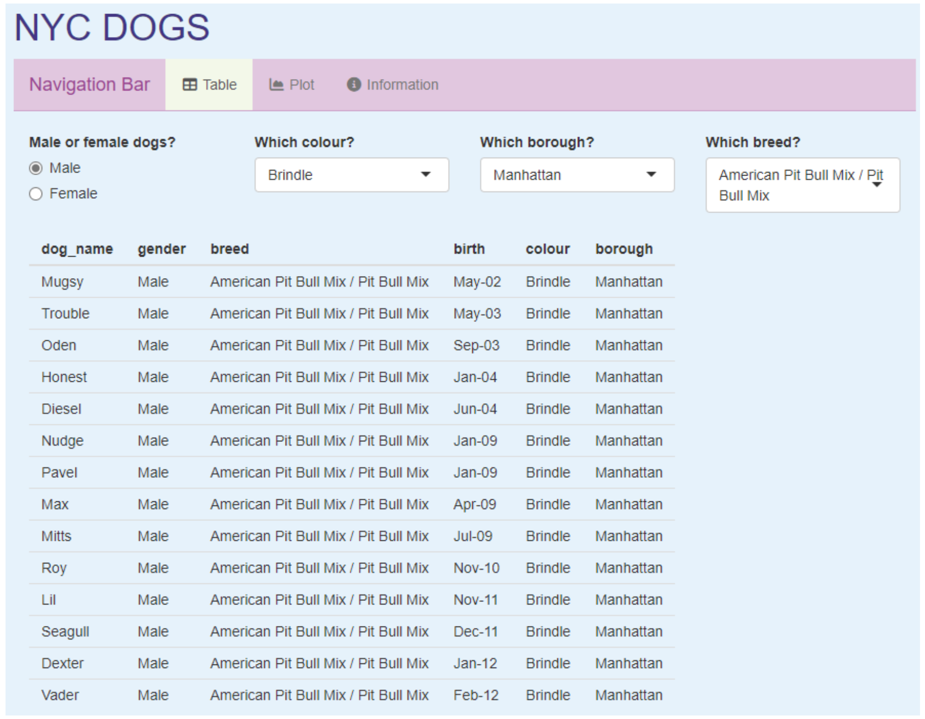
- Open the CSS style script and add code to change the following
  - Change colour of the h1 tag
  - Change the navbar title colour, and navbar colour
  - Change the body colour
  - **Extra**: Change the active tab colour

- **Remember: CSS uses "{}", ":" instead of "()" for functions and ";" instead of "," for new functions in the curly brackets.**

CSS style script Answer



```
/* Tags*/

h1 {
  color: #433683;
}

/* Navigation bar*/

.navbar-default {
    background-color: #E1C7DF;
    border-color: #CDA1C9;
}

.navbar-default .navbar-brand {
    color: #9B4393;
}

.navbar-default .navbar-nav>.active>a, .navbar-default .navbar-nav>.active>a:focus, .navbar-default .navbar-nav>.active>a:hover
{
    color: #555;
    background-color: #F3F8E9;
}

/* Body*/

body {
    font-family: "Helvetica Neue",Helvetica,Arial,sans-serif;
    font-size: 14px;
    line-height: 1.42857143;
    color: #333;
    background-color: #E6F2FB;
}


/* PHS hex colours

           PURPLE  MAGENTA BLUE    GREEN
HEX        #3F3685 #9B4393 #0078D4 #83BB26
TINT - 80% #655E9D #AF69A9 #3393DD #9CC951
TINT - 50% #9F9BC2 #CDA1C9 #80BCEA #C1DD93
TINT - 30% #C5C3DA #E1C7DF #B3D7F2 #DAEBBE
TINT - 10% #ECEBF3 #F5ECF4 #E6F2FB #F3F8E9
*/
```



NYC DOGS

Male or female dogs? [✓ Male] Female

Which colour? Brindle

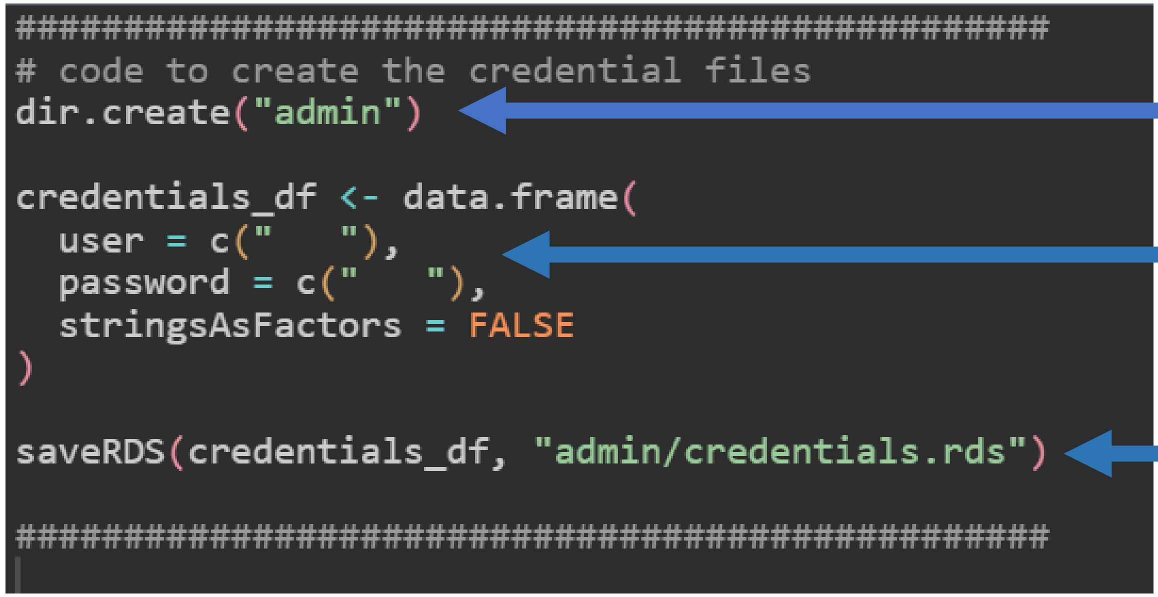
Which borough? Manhattan

Which breed? American Pit Bull Mix / Pit Bull Mix

dog_name	gender	breed	birth	colour	borough
Mugsy	Male	American Pit Bull Mix / Pit Bull Mix	May-02	Brindle	Manhattan
Trouble	Male	American Pit Bull Mix / Pit Bull Mix	May-03	Brindle	Manhattan
Oden	Male	American Pit Bull Mix / Pit Bull Mix	Sep-03	Brindle	Manhattan
Honest	Male	American Pit Bull Mix / Pit Bull Mix	Jan-04	Brindle	Manhattan
Diesel	Male	American Pit Bull Mix / Pit Bull Mix	Jun-04	Brindle	Manhattan
Nudge	Male	American Pit Bull Mix / Pit Bull Mix	Jan-09	Brindle	Manhattan
Pavel	Male	American Pit Bull Mix / Pit Bull Mix	Jan-09	Brindle	Manhattan
Max	Male	American Pit Bull Mix / Pit Bull Mix	Apr-09	Brindle	Manhattan
Mitts	Male	American Pit Bull Mix / Pit Bull Mix	Jul-09	Brindle	Manhattan
Roy	Male	American Pit Bull Mix / Pit Bull Mix	Nov-10	Brindle	Manhattan
Lil	Male	American Pit Bull Mix / Pit Bull Mix	Nov-11	Brindle	Manhattan
Seagull	Male	American Pit Bull Mix / Pit Bull Mix	Dec-11	Brindle	Manhattan
Dexter	Male	American Pit Bull Mix / Pit Bull Mix	Jan-12	Brindle	Manhattan
Vader	Male	American Pit Bull Mix / Pit Bull Mix	Feb-12	Brindle	Manhattan

Password Protection

- Required when publishing an app for PRA.
- We need to create a Create Credentials Script.



```r
################################################
# code to create the credential files
dir.create("admin")

credentials_df <- data.frame(
  user = c("   "),
  password = c("   "),
  stringsAsFactors = FALSE
)

saveRDS(credentials_df, "admin/credentials.rds")

################################################
```

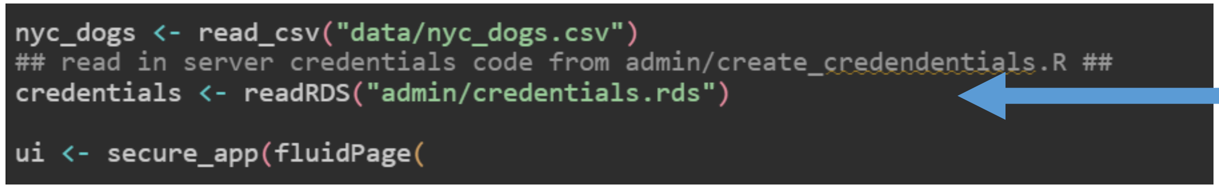
This creates an admin folder within our current working directory

This is where we insert our chosen username and password

Reads out a .rds file with our credentials

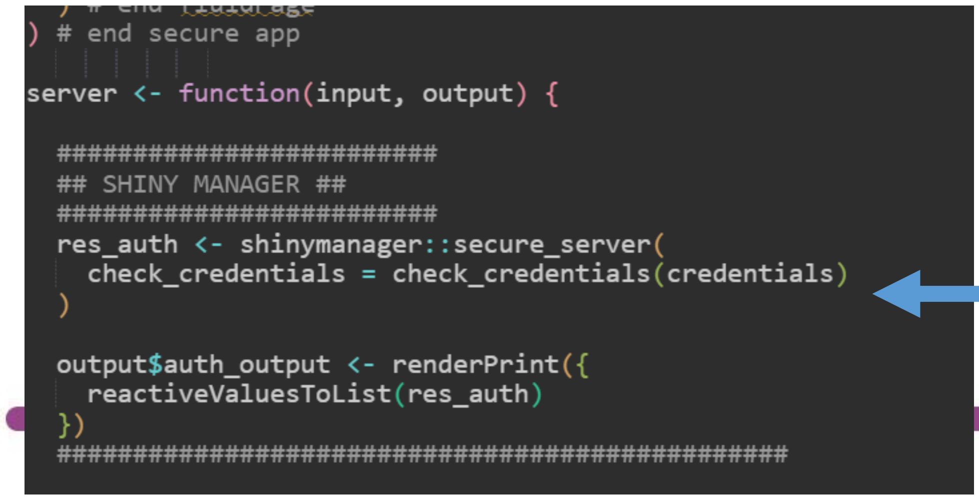
Password Protection: UI and Server

- To add password protection to our dashboard we need to wrap our UI within the function **shinymanager::secure_app()**. **Note: We need to install the shiny manager package.**
- We then need to add the below code into our server.



```r
nyc_dogs <- read_csv("data/nyc_dogs.csv")
## read in server credentials code from admin/create_credendentials.R ##
credentials <- readRDS("admin/credentials.rds")

ui <- secure_app(fluidPage(
```

Reads in our credentials.rds file at start of script and wrap UI in secure_app

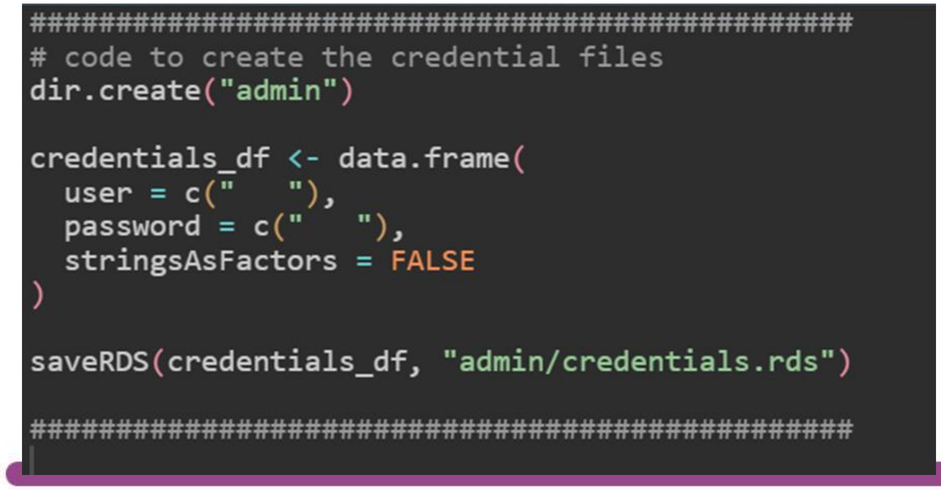


```r
  ) # end fluidPage
) # end secure app

server <- function(input, output) {

  #########################
  ## SHINY MANAGER ##
  #########################
  res_auth <- shinymanager::secure_server(
    check_credentials = check_credentials(credentials)
  )

  output$auth_output <- renderPrint({
    reactiveValuesToList(res_auth)
  })
  ################################################
```

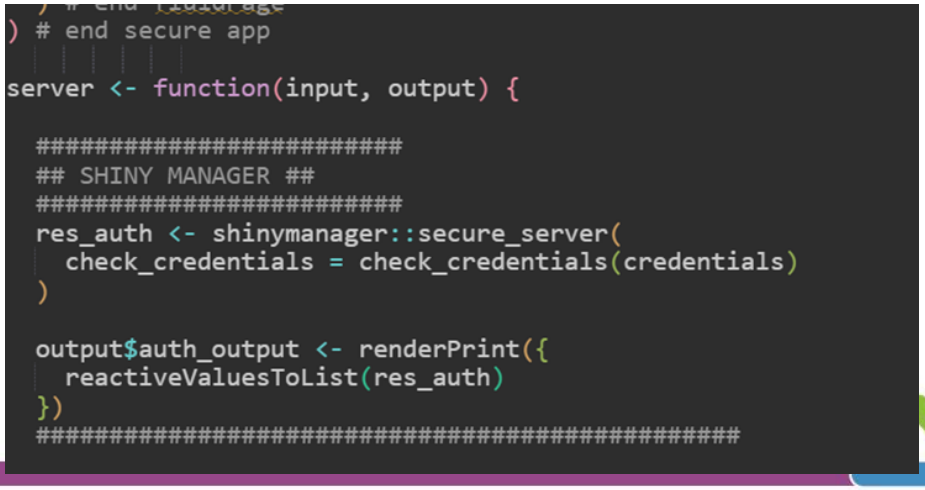
Checks what the user enters against the credentials we have set.

Exercise: Password Protection

- Create a username and password for your Shiny app.
- Wrap the UI within the **secure_app** function.
- Read in your credentials script.
- Add the required lines of code to your server.
- Run your app – Does your username and password work?



```
################################################
# code to create the credential files
dir.create("admin")

credentials_df <- data.frame(
  user = c("    "),
  password = c("    "),
  stringsAsFactors = FALSE
)

saveRDS(credentials_df, "admin/credentials.rds")

################################################
```

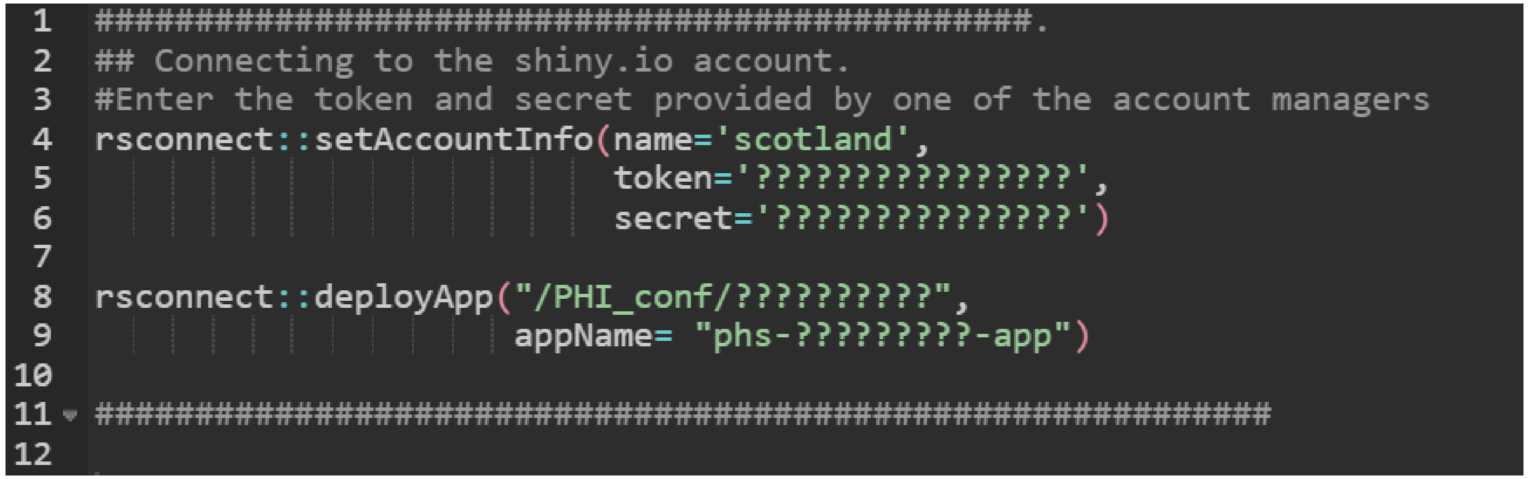


```
) # end secure app

server <- function(input, output) {

  #########################
  ## SHINY MANAGER ##
  #########################
  res_auth <- shinymanager::secure_server(
    check_credentials = check_credentials(credentials)
  )

  output$auth_output <- renderPrint({
    reactiveValuesToList(res_auth)
  })
  ################################################
```

Publishing a Shiny App

- For published apps, we use the **scotland.shinyapps.io** server.
  - This is shared with the Scottish Government
- To deploy an app on this server we need both a **Token** and a **Secret.**
  - These can be obtained from the R Shiny user group by asking who the current administrators are.
  - Important that you don’t share these codes with others.
- The app name should ideally be **phs-<nameofapp>-app** so that PHS dashboards can be easily identified.
- **Note:** All file paths within the shiny dashboard should be relative and not absolute. I.e. Paths should not be /PHI_conf/… but should be pathways related to the working directory that the Shiny app is in

Publishing a Shiny App - Continued



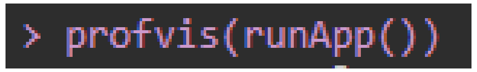
```r
###############################################.
## Connecting to the shiny.io account.
#Enter the token and secret provided by one of the account managers
rsconnect::setAccountInfo(name='scotland',
                          token='????????????????',
                          secret='????????????????')

rsconnect::deployApp("/PHI_conf/??????????",
                     appName= "phs-?????????-app")

############################################################

```

- This is the code that we are required to run to deploy our app.
- This requires the **rsconnect package** and the token and secret that we have obtained.
- This will publish our app to the web address **https://scotland.shinyapps.io/<appName>**

Shiny and memory

- Each new app is allocated 1GB of memory on the Shiny server and works for most of our apps.
- If your app is slow or crashes, before requesting more memory on the server, it is a good idea to profile your Shiny code.
  - Profiling will show how much memory each function in your script uses

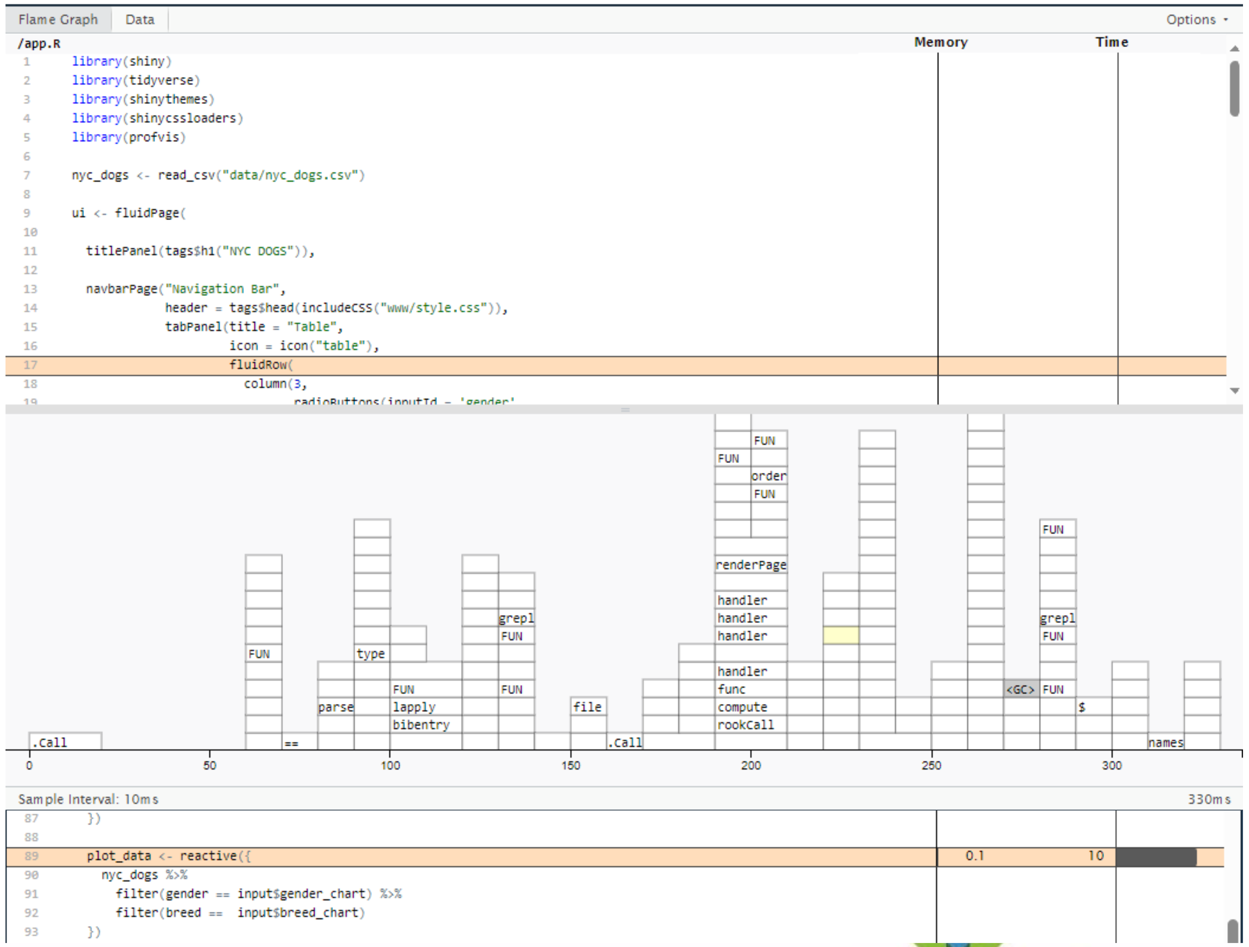
- Use **profvis()** to see how memory intensive your code is. More info here: [Profvis — Interactive Visualizations for Profiling R Code (rstudio.github.io)]()
- Running the app in **profvis()** in the console will, after the app is closed, produce an interactive report of your code.



```
> profvis(runApp())
```

Shiny and memory: profvis profile Flame graph

- Shows the full code, and memory and time allocated to each function
- Graph shows what other functions are called (call stack) and how long each function run for.
- E.g. in our app the most memory intensive function is creating the reactive plot_data



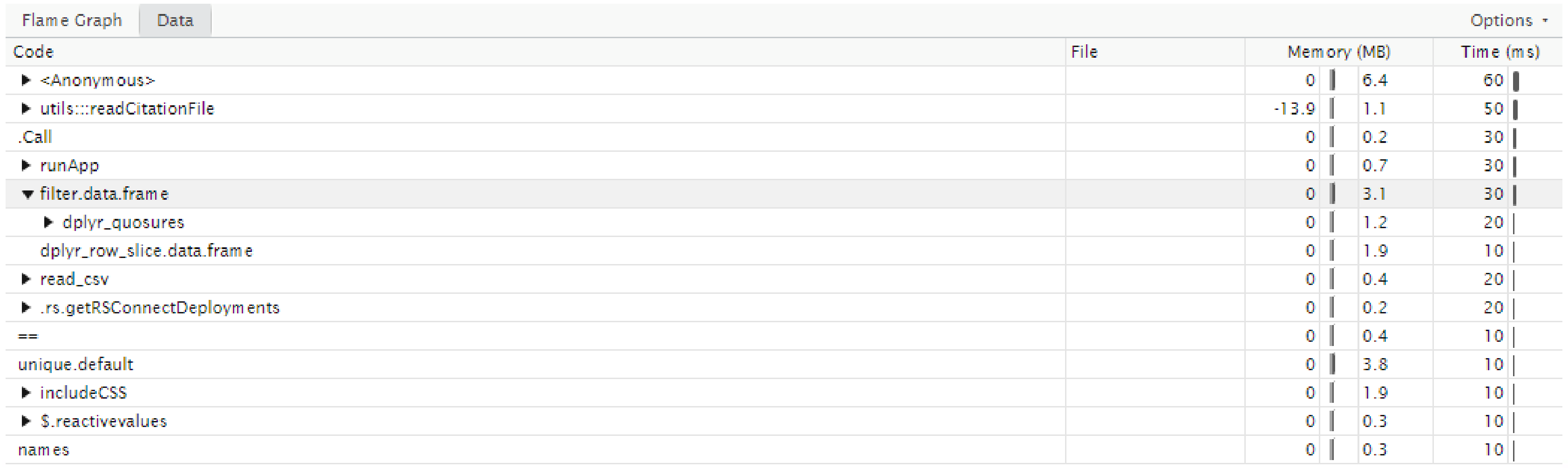
/app.R

```r
library(shiny)
library(tidyverse)
library(shinythemes)
library(shinycssloaders)
library(profvis)

nyc_dogs <- read_csv("data/nyc_dogs.csv")

ui <- fluidPage(

  titlePanel(tags$h1("NYC DOGS")),

  navbarPage("Navigation Bar",
             header = tags$head(includeCSS("www/style.css")),
             tabPanel(title = "Table",
                      icon = icon("table"),
                      fluidRow(
                        column(3,
```

```r
  })

  plot_data <- reactive({                                   0.1        10
    nyc_dogs %>%
      filter(gender == input$gender_chart) %>%
      filter(breed ==  input$breed_chart)
  })
```

Shiny and memory: profvis profile Data

- Shows each function and memory allocation and time for each function.
- This is a good place to look when you want to make your app faster and less memory intensive.



Flame Graph | Data | Options

Code	File	Memory (MB)		Time (ms)
▶ <Anonymous>		0	6.4	60
▶ utils:::readCitationFile		-13.9	1.1	50
.Call		0	0.2	30
▶ runApp		0	0.7	30
▼ filter.data.frame		0	3.1	30
▶ dplyr_quosures		0	1.2	20
dplyr_row_slice.data.frame		0	1.9	10
▶ read_csv		0	0.4	20
▶ .rs.getRSConnectDeployments		0	0.2	20
==		0	0.4	10
unique.default		0	3.8	10
▶ includeCSS		0	1.9	10
▶ $.reactivevalues		0	0.3	10
names		0	0.3	10

Recap Shiny course

- You have created a dashboard from scratch with reactive data to user input, a chart, table and information text.
- You have created a PHS dashboard displaying Allergic conditions and Asthma data with interactive plots, and data download functionality.
- You have learned to change the appearance of your dashboard using CSS.
- You have learned to deploy an app and to protect it with a username and password.
- You have learned to profile your code to make your app faster.

End of day 3 – any questions?