

Dashboards con RShiny

III Congreso & XIV Jornadas de Usuarios de R

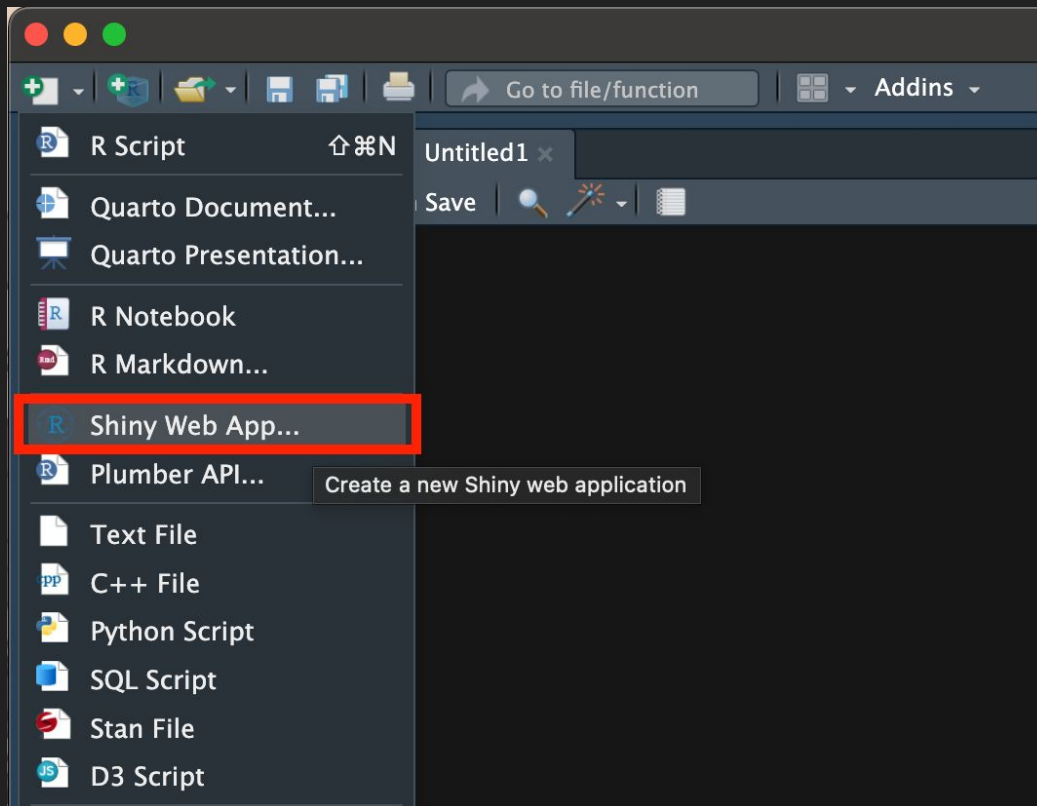


Importante:

- ❖ Instalar el paquete {Shiny} antes de empezar.
- ❖ Es recomendable trabajar en proyectos de R.
- ❖ Tenéis un **repositorio** en el que podéis seguir el taller paso a paso.




Nuestra primera ShinyApp





Nuestra primera ShinyApp


New Shiny Web Application



Application name:

Application type: ☒ Single File (app.R)
☐ Multiple File (ui.R/server.R)

Create within directory:

 [Shiny Web Applications](#)



```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                 "Number of bins:",
23                 min = 1,
24                 max = 50,
25                 value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',
45          xlab = 'Waiting time to next eruption (in mins)',
46          main = 'Histogram of waiting times')
47   })
48 }
49
50 # Run the application
51 shinyApp(ui = ui, server = server)
52
```



```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14   # Application title
15   titlePanel("Old Faithful Geyser Data"),
16   # Sidebar with a slider input for number of bins
17   sidebarLayout(
18     sidebarPanel(
19       sliderInput("bins",
20         "Number of bins:",
21         min = 1,
22         max = 50,
23         value = 30)
24     ),
25     # Show a plot of the generated distribution
26     mainPanel(
27       plotOutput("distPlot")
28     )
29   )
30 )
31
32 # Define server logic required to draw a histogram
33 server <- function(input, output) {
34   output$distPlot <- renderPlot({
35     # generate bins based on input$bins from ui.R
36     x <- faithful[, 2]
37     bins <- seq(min(x), max(x), length.out = input$bins + 1)
38
39     # draw the histogram with the specified number of bins
40     hist(x, breaks = bins, col = 'darkgray', border = 'white',
41         xlab = 'Waiting time to next eruption (in mins)',
42         main = 'Histogram of waiting times')
43   })
44 }
45
46 # Run the application
47 shinyApp(ui = ui, server = server)
```



```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                 "Number of bins:",
23                 min = 1,
24                 max = 50,
25                 value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',
45          xlab = 'Waiting time to next eruption (in mins)',
46          main = 'Histogram of waiting times')
47   })
48 }
49
50 # Run the application
51 shinyApp(ui = ui, server = server)
52
```



```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                  "Number of bins:",
23                  min = 1,
24                  max = 50,
25                  value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',
45          xlab = 'Waiting time to next eruption (in mins)',
46          main = 'Histogram of waiting times')
47   })
48 }
49
50 # Run the application
51 shinyApp(ui = ui, server = server)
52
```




```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                   "Number of bins:",
23                   min = 1, max = 50, value = 10),
24
25       # Show a plot of the distribution
26     ),
27     mainPanel(
28       plotOutput("distPlot")
29     )
30   )
31 )
32
33 # Define server logic to draw a histogram
34 server <- function(input, output, session) {
35
36   # Draw a histogram with the specified number of bins
37   output$distPlot <- plot({
38     # get the data from the dataset
39     x <- rnorm(1000)
40     bins <- seq(min(x), max(x), length.out = input$bins)
41
42     # draw the histogram with the specified number of bins
43     hist(x, breaks = bins, col = 'darkgray', border = 'white',
44          xlab = 'Waiting time to next eruption (in mins)',
45          main = 'Histogram of waiting times')
46   })
47 }
48
49 # Run the application
50 shinyApp(ui = ui, server = server)
51
52
```



1



Las librerías, siempre al principio

```
1
2  ## Cargamos librerías
3
4  library(shiny)
5  library(shinydashboard)
6  #library(shinyWidgets)
7  library(tidyverse)
8  library(plotly)
9  library(leaflet)
10 library(sf)
11 library(reactable)
12 #library(rsconnect) #En caso de querer subir el Shiny a un servidor.
13
```



¿Dónde cargamos los datasets?

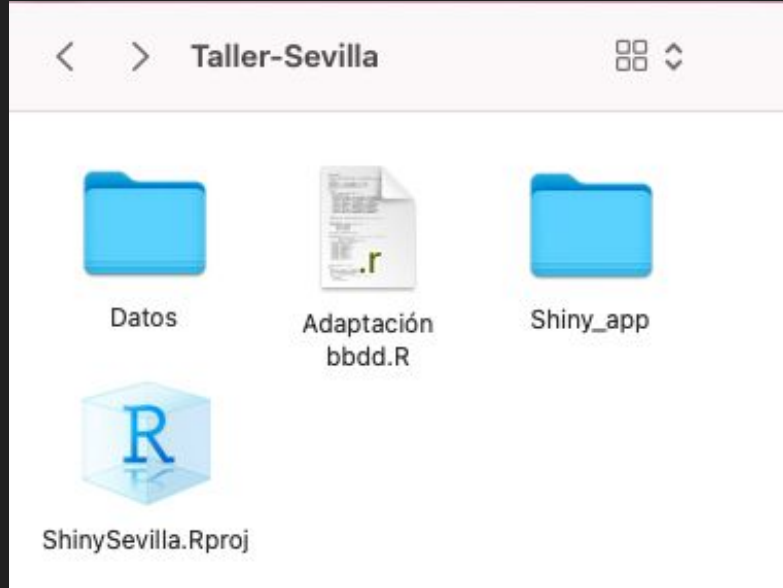
```
1
2  ## Cargamos librerías
3
4  library(shiny)
5  library(shinydashboard)
6  #library(shinyWidgets)
7  library(tidyverse)
8  library(plotly)
9  library(leaflet)
10 library(sf)
11 library(reactable)
12 #library(rsconnect) #En caso de querer subir el Shiny a un servidor.
13
14 ## Cargamos dataset y lo adaptamos a lo que necesitará la app.
15 datos_contaminacion <- read_csv("Contaminantes_Sevilla.csv") %>%
16   st_as_sf(coords = c("y", "x")) %>%
17   st_set_crs(4326)
18
```



⚠ No user setuid() ⚠



¿Dónde cargamos los datasets?





¿Dónde cargamos los datasets?

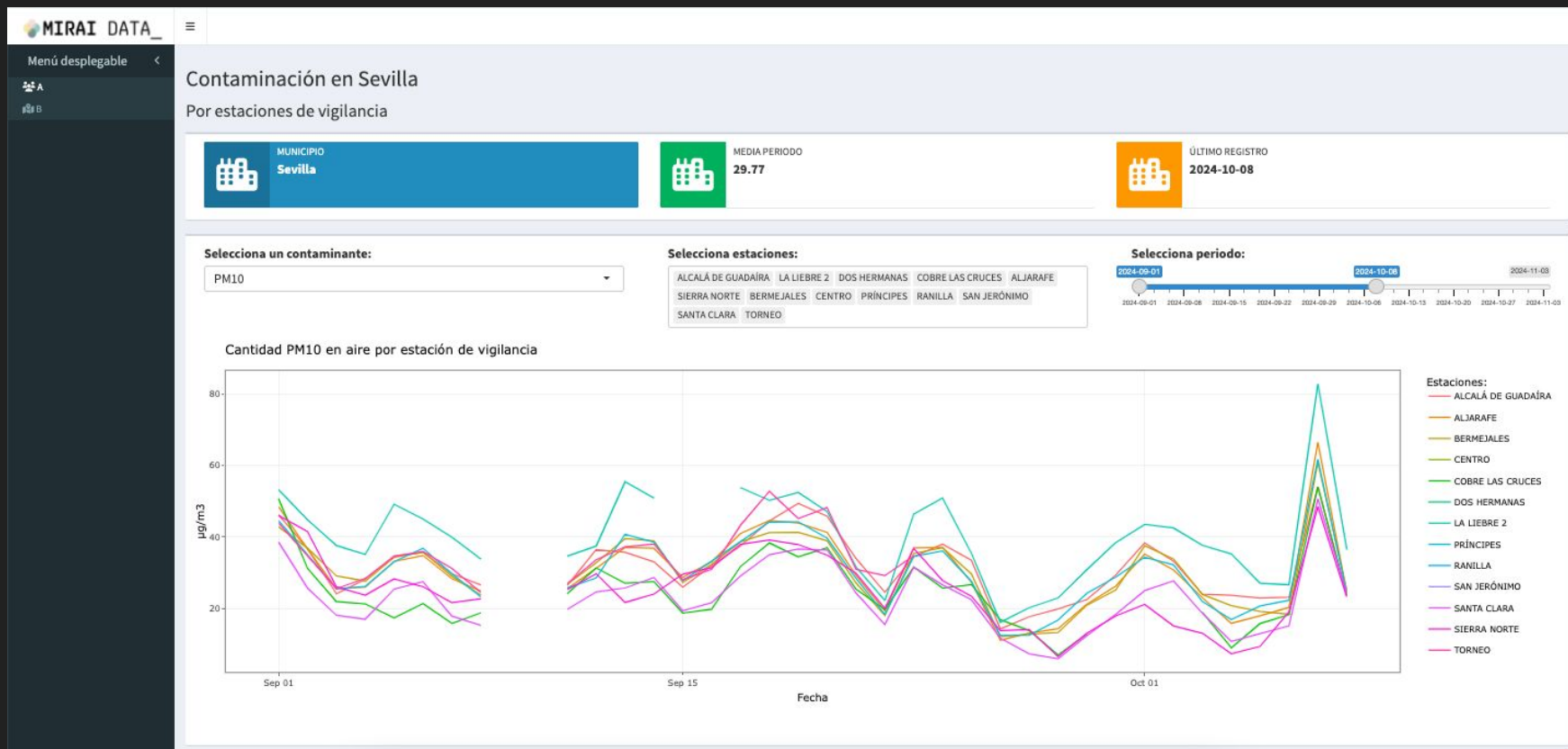
```
1
2 ## Cargamos librerías
3
4 library(shiny)
5 library(shinydashboard)
6 #library(shinyWidgets)
7 library(tidyverse)
8 library(plotly)
9 library(leaflet)
10 library(sf)
11 library(reactable)
12 #library(rsconnect) #En caso de querer subir el Shiny a un servidor.
13
14 ## Cargamos dataset y lo adaptamos a lo que necesitará la app.
15 datos_contaminacion <- read_csv("Contaminantes_Sevilla.csv") %>%
16   st_as_sf(coords = c("y", "x")) %>%
17   st_set_crs(4326)
18
```



El dataset

| | D_MUNICIPIO | D_ESTACION | F_FECHA | Contaminante | Registro_contaminante | id | geometry |
|----|--------------------|--------------------|------------|--------------|-----------------------|----|----------------------------|
| 1 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-01 | PM10 | 46.132542 | 1 | POINT (-5.830987 37.34082) |
| 2 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-01 | PM25 | N/A | 1 | POINT (-5.830987 37.34082) |
| 3 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-01 | NO2 | 4.291667 | 1 | POINT (-5.830987 37.34082) |
| 4 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-01 | O3 | 79.651042 | 1 | POINT (-5.830987 37.34082) |
| 5 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-01 | SO2 | 4.375000 | 1 | POINT (-5.830987 37.34082) |
| 6 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-02 | PM10 | 36.782458 | 1 | POINT (-5.830987 37.34082) |
| 7 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-02 | PM25 | N/A | 1 | POINT (-5.830987 37.34082) |
| 8 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-02 | NO2 | 6.750000 | 1 | POINT (-5.830987 37.34082) |
| 9 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-02 | O3 | 70.880208 | 1 | POINT (-5.830987 37.34082) |
| 10 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-02 | SO2 | 4.541667 | 1 | POINT (-5.830987 37.34082) |
| 11 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-03 | PM10 | 24.092458 | 1 | POINT (-5.830987 37.34082) |
| 12 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-03 | PM25 | N/A | 1 | POINT (-5.830987 37.34082) |
| 13 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-03 | NO2 | 6.708333 | 1 | POINT (-5.830987 37.34082) |
| 14 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-03 | O3 | 60.317708 | 1 | POINT (-5.830987 37.34082) |
| 15 | ALCALÁ DE GUADAÍRA | ALCALÁ DE GUADAÍRA | 2024-09-03 | SO2 | 4.291667 | 1 | POINT (-5.830987 37.34082) |

El paquete {Shinydashboard}



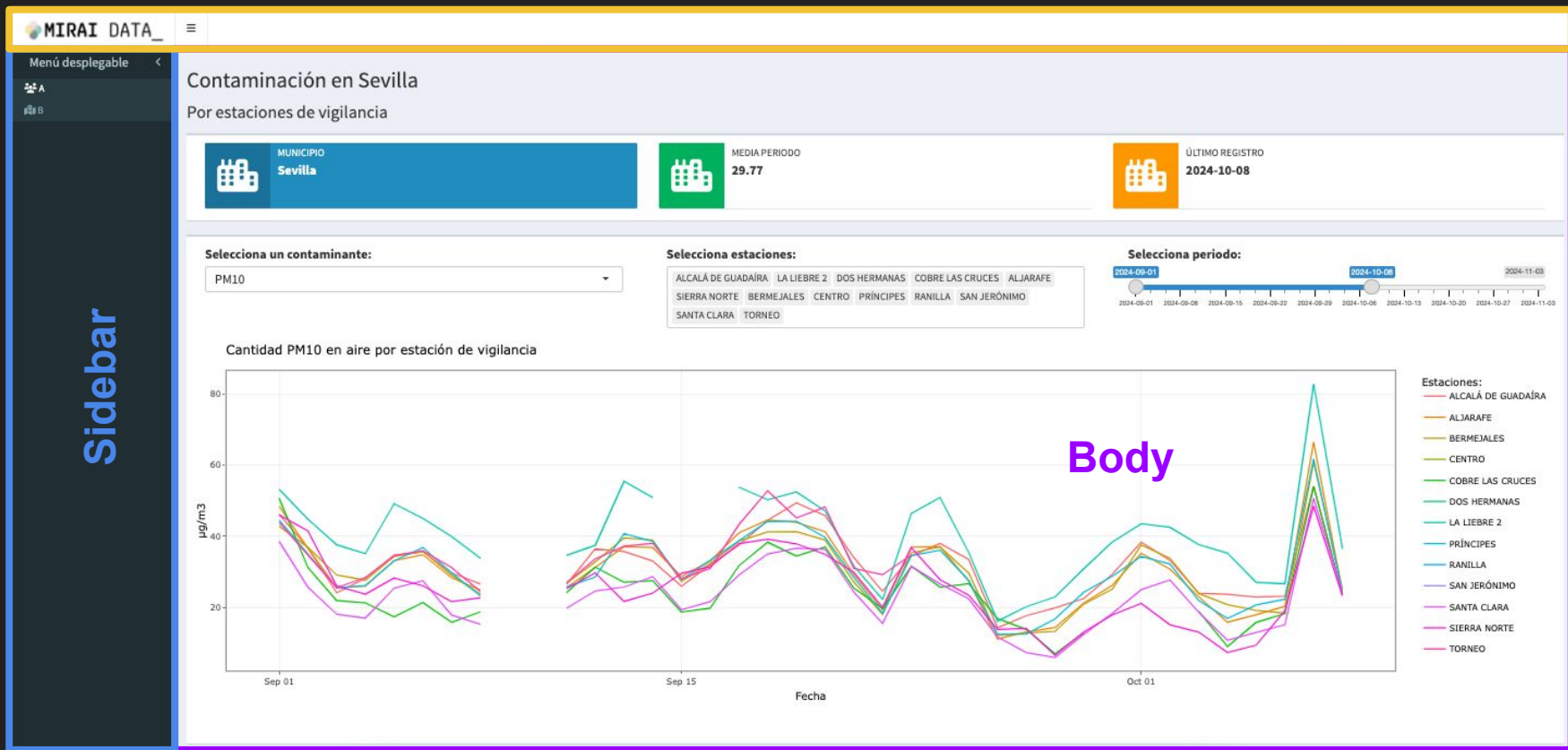


El paquete {Shinydashboard}



El paquete {Shinydashboard}

Header





Header, sidebar y body

```
24  ## Siguiendo la lógica de {shinydashboard}, necesitamos definir 3 objetos:  
25  ## header, sidebar y body.  
26  
27  header <- shinydashboard::dashboardHeader()  
28  
29  sidebar <- shinydashboard::dashboardSidebar()  
30  
31  body <- shinydashboard::dashboardBody()  
32
```



2



Header

Si quisiéramos poner solamente texto en el título del dashboard basta con
añadir texto al argumento 'title'.

```
header <- shinydashboard::dashboardHeader(title = "Contaminación Sevilla")
```



Header

*## Si quisiéramos poner solamente texto en el título del dashboard basta con
añadir texto al argumento 'title'.*

```
header <- shinydashboard::dashboardHeader(title = "Contaminación Sevilla")
```

Contaminación Sevilla





Header

```
## Si en lugar de texto nos interesa añadir una imagen (por ejemplo un logo),  
## tendremos que usar etiquetas de html, en este caso la etiqueta <img>.  
  
## Para evitar deformaciones y adaptar la imagen al espacio del dashboard,  
## definimos los argumentos 'height' y 'width'.  
header <- shinydashboard::dashboardHeader(title = tags$img(src='https://raw.githubusercontent.com/DataMirai/'  
height='40', width='210'))
```





Sidebar

```
sidebar <- shinydashboard::dashboardSidebar(  
  shinydashboard::sidebarMenu(  
    shinydashboard::menuItem(h4("Menú desplegable", style="margin:0px; margin-left:10px; padding: 0px;"),  
      tabName = "menu1", # Es necesario un ID que relacione el elemento con el resto de la app.  
      startExpanded = TRUE,  
      shinydashboard::menuSubItem("A", tabName = "A_id", icon = icon("users")),  
      # Al añadir Subitems, el menú se vuelve desplegable  
      shinydashboard::menuSubItem("B", tabName = "B_id", icon = icon("map-location-dot"))  
    )  
  )  
)
```

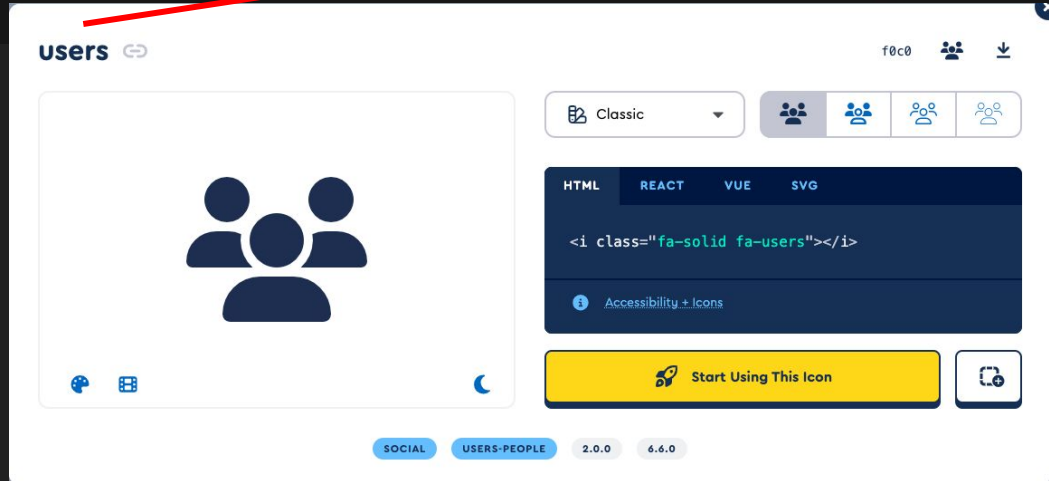




Sidebar

```
sidebar <- shinydashboard::dashboardSidebar(  
  shinydashboard::sidebarMenu(  
    shinydashboard::menuItem(h4("Menú desplegable", style="margin:0px; margin-left:10px; padding: 0px;"),  
      tabName = "menu1", # Es necesario un ID que relacione el elemento con el resto de la app.  
      startExpanded = TRUE,  
      shinydashboard::menuSubItem("A", tabName = "A_id", icon = icon("users")),  
      # Al añadir Subitems, el menú se vuelve desplegable  
      shinydashboard::menuSubItem("B", tabName = "B_id", icon = icon("map-location-dot"))  
    )  
  )  
)
```

Fontawesome.com





Sidebar

```
sidebar <- shinydashboard::dashboardSidebar(  
  shinydashboard::sidebarMenu(  
    shinydashboard::menuItem(h4("Menú desplegable", style="margin:0px; margin-left:10px; padding: 0px;"),  
      tabName = "menu1", # Es necesario un ID que relacione el elemento con el resto de la app.  
      startExpanded = TRUE,  
      shinydashboard::menuSubItem("A", tabName = "A_id", icon = icon("users")),  
      # Al añadir Subitems, el menú se vuelve desplegable  
      shinydashboard::menuSubItem("B", tabName = "B_id", icon = icon("map-location-dot"))  
    )  
  )  
)
```





Body: vamos a montar la estructura base

```
body <- shinydashboard::dashboardBody(  
  fluidRow(  
  
    shinydashboard::tabItems(  
  
      shinydashboard::tabItem("A_id", ## Id que relacione la pestaña con la barra lateral (sidebar)  
        h2("Contaminación en Sevilla", style="margin:15px;"),  
  
        h3("Por estaciones de vigilancia", style="margin:15px;"),  
  
        ## Primer recuadro de la estructura del dashboard (Siguiendo el ppt):  
  
        shinydashboard::box( width = 12, height = 120, style="font-size: 18px;"),  
  
        ## Segundo recuadro:  
  
        shinydashboard::box( width = 12, height = 700, style="font-size: 18px;"),  
  
        ## Tercer recuadro:  
  
        shinydashboard::box( width = 12,  
                              h3("Haz click en una estación del mapa:")  
        ),  
      shinydashboard::tabItem("B_id", ## Id que relacione la pestaña con la barra lateral (sidebar)  
        h2("Esta es la pestaña B", style="margin:15px;"),  
  
        h3("Está vacía pero puedes probar a añadirle lo que quieras:)", style="margin:15px;"))  
    )  
  )  
)
```

Contaminación en Sevilla

Por estaciones de vigilancia

MUNICIPIO
Sevilla

MEDIA PERIODO
24.83



ÚLTIMO REGISTRO
2024-11-03

Selecciona un contaminante:

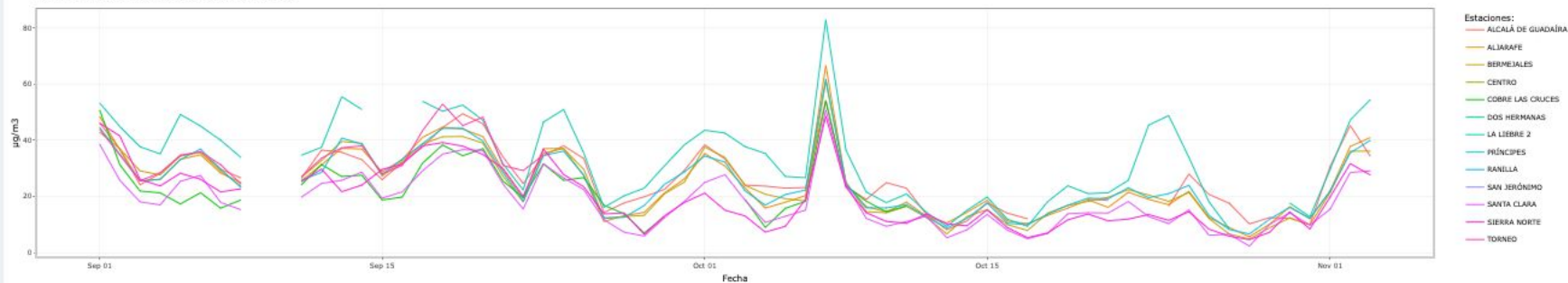
PM10

Selecciona estaciones:

ALCALÁ DE GUADARA LA LIEBRE 2 DOS HERMANAS COBRE LAS CRUCES ALJARAFE SIERRA NORTE BERMEJALES CENTRO
PRÍNCIPES RANILLA SAN JERÓNIMO SANTA CLARA TORNEO

Selecciona periodo:

Cantidad PM10 en aire por estación de vigilancia



Haz click en una estación del mapa:



| Estación | Fecha | Contaminante | Valor Registro |
|---------------------|------------|--------------|----------------|
| ALCALÁ DE GUADAJIRA | 2024-09-01 | PM10 | 46.13 |
| ALCALÁ DE GUADAJIRA | 2024-09-01 | PM25 | |
| ALCALÁ DE GUADAJIRA | 2024-09-01 | NO2 | 4.29 |
| ALCALÁ DE GUADAJIRA | 2024-09-01 | O3 | 79.65 |
| ALCALÁ DE GUADAJIRA | 2024-09-01 | SO2 | 4.38 |
| ALCALÁ DE GUADAJIRA | 2024-09-02 | PM10 | 36.78 |
| ALCALÁ DE GUADAJIRA | 2024-09-02 | PM25 | |
| ALCALÁ DE GUADAJIRA | 2024-09-02 | NO2 | 6.75 |
| ALCALÁ DE GUADAJIRA | 2024-09-02 | O3 | 70.88 |
| ALCALÁ DE GUADAJIRA | 2024-09-02 | SO2 | 4.54 |

1-10 of 4095 rows

Previous 1 2 3 4 5 ... 410 Next



Body: la pestaña B

 MIRAI DATA_

☰

Menú desplegable <

 A

 B

Esta es la pestaña B

Está vacía pero puedes probar a añadirle lo que quieras:)



3



Body: el contenido (I)

```
## Primer recuadro de la estructura del dashboard (Siguiendo el ppt):  
  
shinydashboard::box( width = 12, height = 120, style="font-size: 18px;",  
  infoBoxOutput("infobox_municipio", width = 4),  
  # Se asigna un id a la caja de información y una anchura (3x4)  
  infoBoxOutput("infobox_media_periodo", width = 4),  
  infoBoxOutput("infobox_ultimo_registro", width = 4)),
```





Body: el contenido (II)

Segundo recuadro:

```
shinydashboard::box( width = 12, height = 700, style="font-size: 18px;",
  fluidRow(
    column(width = 4, style="padding:0px 30px;",
      selectInput("select_contaminante",
        label = "Selecciona un contaminante:",
        choices = unique(datos_contaminacion$Contaminante),
        selected = "PM10"),
      multiple = FALSE),

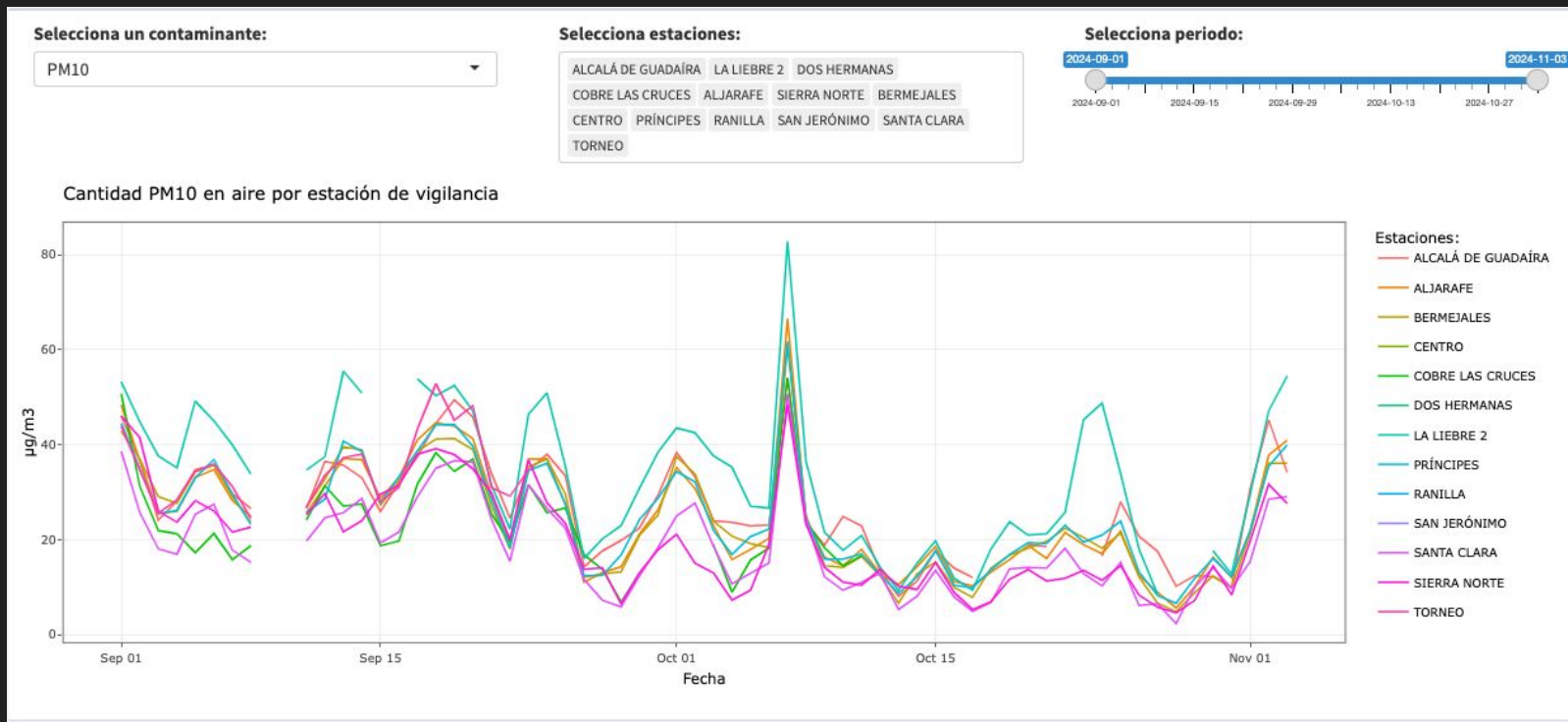
    column(width = 4, style="padding:0px 30px;",
      selectizeInput("select_estaciones",
        label = "Selecciona estaciones:",
        choices = unique(datos_contaminacion$D_ESTACION),
        selected = c(unique(datos_contaminacion$D_ESTACION)),
        multiple = TRUE)),

    column(width = 4, style="padding:0px 30px;",
      sliderInput("slider_fecha",
        label = "Selecciona periodo:",
        min = min(datos_contaminacion$F_FECHA),
        max = max(datos_contaminacion$F_FECHA),
        value = c(min(datos_contaminacion$F_FECHA), max(datos_contaminacion$F_FECHA))),
    ),

    fluidRow(
      column(width = 12, style="padding:0px 30px;",
        plotlyOutput("grafico_lineas", height = 500)))
  ),
```



Body: el contenido (II)





Body: el contenido (II)

Segundo recuadro:

```
shinydashboard::box( width = 12, height = 700, style="font-size: 18px;",
  fluidRow(
    column(width = 4, style="padding:0px 30px;",
      selectInput("select_contaminante",
        label = "Selecciona un contaminante:",
        choices = unique(datos_contaminacion$Contaminante),
        selected = "PM10"),
      multiple = FALSE),

    column(width = 4, style="padding:0px 30px;",
      selectizeInput("select_estaciones",
        label = "Selecciona estaciones:",
        choices = unique(datos_contaminacion$D_ESTACION),
        selected = c(unique(datos_contaminacion$D_ESTACION)),
        multiple = TRUE)),

    column(width = 4, style="padding:0px 30px;",
      sliderInput("slider_fecha",
        label = "Selecciona periodo:",
        min = min(datos_contaminacion$F_FECHA),
        max = max(datos_contaminacion$F_FECHA),
        value = c(min(datos_contaminacion$F_FECHA), max(datos_contaminacion$F_FECHA))),
      ),
    fluidRow(
      column(width = 12, style="padding:0px 30px;",
        plotlyOutput("grafico_lineas", height = 500)))
  ),
```



Body: el contenido (II)

1

Selecciona un contaminante:

PM10

Selecciona estaciones:

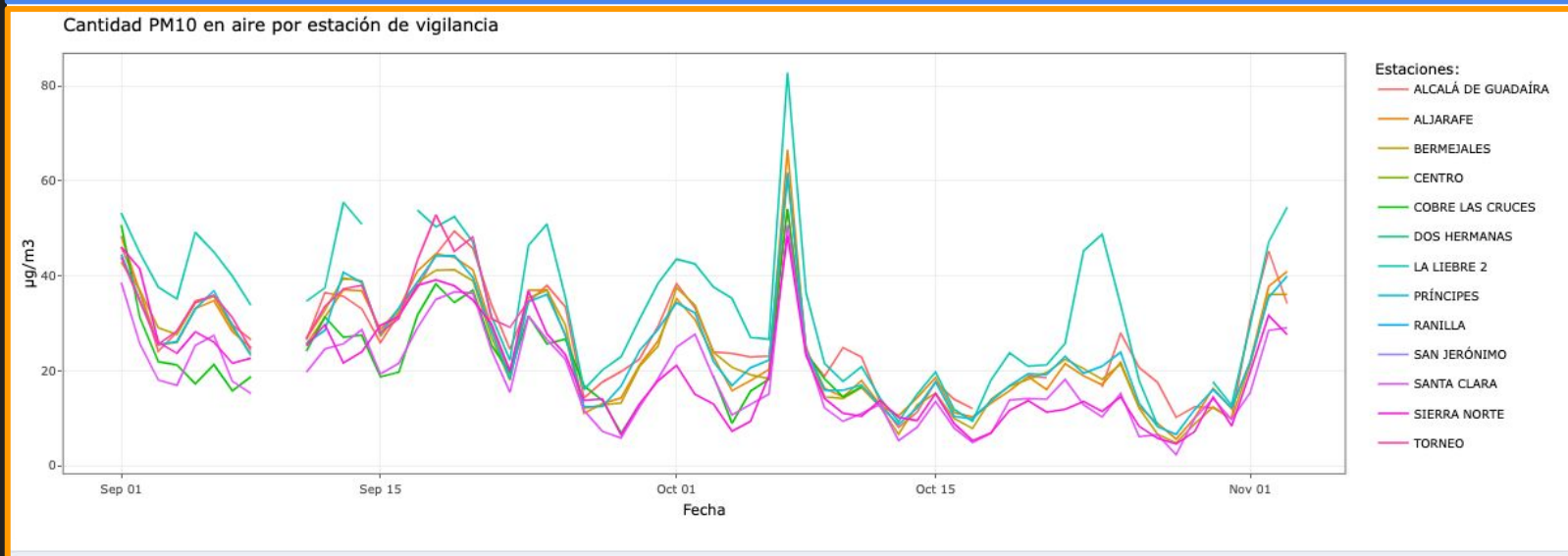
ALCALÁ DE GUADAÍRA LA LIEBRE 2 DOS HERMANAS
COBRE LAS CRUCES ALJARAFA SIERRA NORTE BERMEJALES
CENTRO PRÍNCIPES RANILLA SAN JERÓNIMO SANTA CLARA
TORNEO

Selecciona periodo:

2024-09-01 2024-11-03

2024-09-01 2024-09-15 2024-09-29 2024-10-13 2024-10-27

2





Body: el contenido (II)

Segundo recuadro:

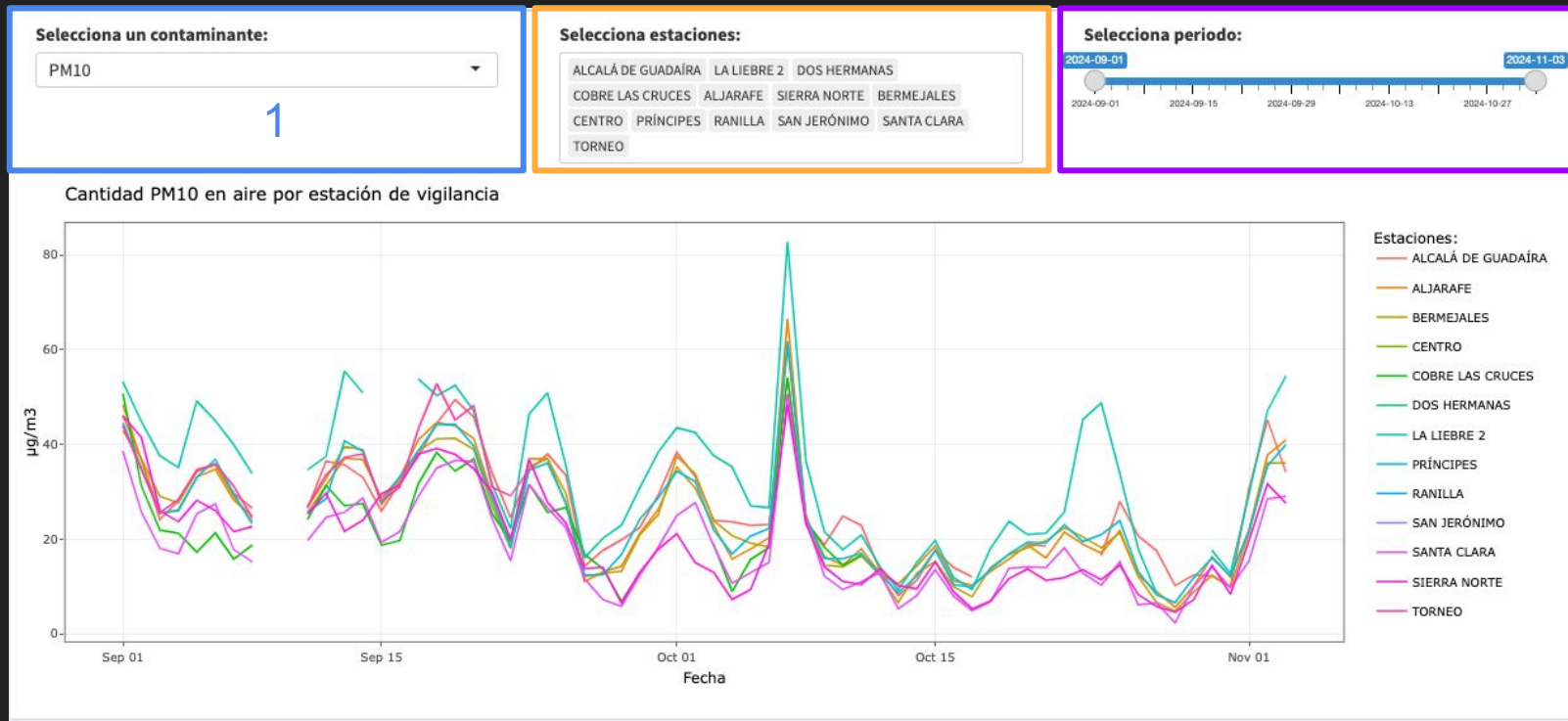
```
shinydashboard::box( width = 12, height = 700, style="font-size: 18px;",
  fluidRow(
    column( width = 4, style="padding:0px 30px;",
      selectInput("select_contaminante",
        label = "Selecciona un contaminante:",
        choices = unique(datos_contaminacion$Contaminante),
        selected = "PM10"),
      multiple = FALSE),
    column( width = 4, style="padding:0px 30px;",
      selectizeInput("select_estaciones",
        label = "Selecciona estaciones:",
        choices = unique(datos_contaminacion$D_ESTACION),
        selected = c(unique(datos_contaminacion$D_ESTACION)),
        multiple = TRUE)),
    column( width = 4, style="padding:0px 30px;",
      sliderInput("slider_fecha",
        label = "Selecciona periodo:",
        min = min(datos_contaminacion$F_FECHA),
        max = max(datos_contaminacion$F_FECHA),
        value = c(min(datos_contaminacion$F_FECHA), max(datos_contaminacion$F_FECHA))),
      ),
    fluidRow(
      column( width = 12, style="padding:0px 30px;",
        plotlyOutput("grafico_lineas", height = 500)))
  ),
```



Body: el contenido (II)

2

3





Body: el contenido (II)

```
column(width = 4, style="padding:0px 30px;",  
  selectInput("select_contaminante",  
    label = "Selecciona un contaminante:",  
    choices = unique(datos_contaminacion$Contaminante),  
    selected = "PM10"),  
  multiple = FALSE),
```

Selecciona un contaminante:

PM10





Body: el contenido (II)

```
column(width = 4, style="padding:0px 30px;",
        selectizeInput("select_estaciones",
                        label = "Selecciona estaciones:",
                        choices = unique(datos_contaminacion$D_ESTACION),
                        selected = c(unique(datos_contaminacion$D_ESTACION)),
                        multiple = TRUE)),
```

Selecciona estaciones:

ALCALÁ DE GUADAÍRA

LA LIEBRE 2

DOS HERMANAS

COBRE LAS CRUCES

ALJARAFE

SIERRA NORTE

BERMEJALES

CENTRO

PRÍNCIPES

RANILLA

SAN JERÓNIMO

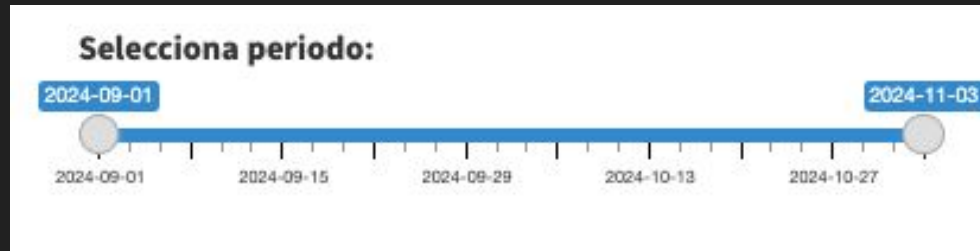
SANTA CLARA

TORNEO



Body: el contenido (II)

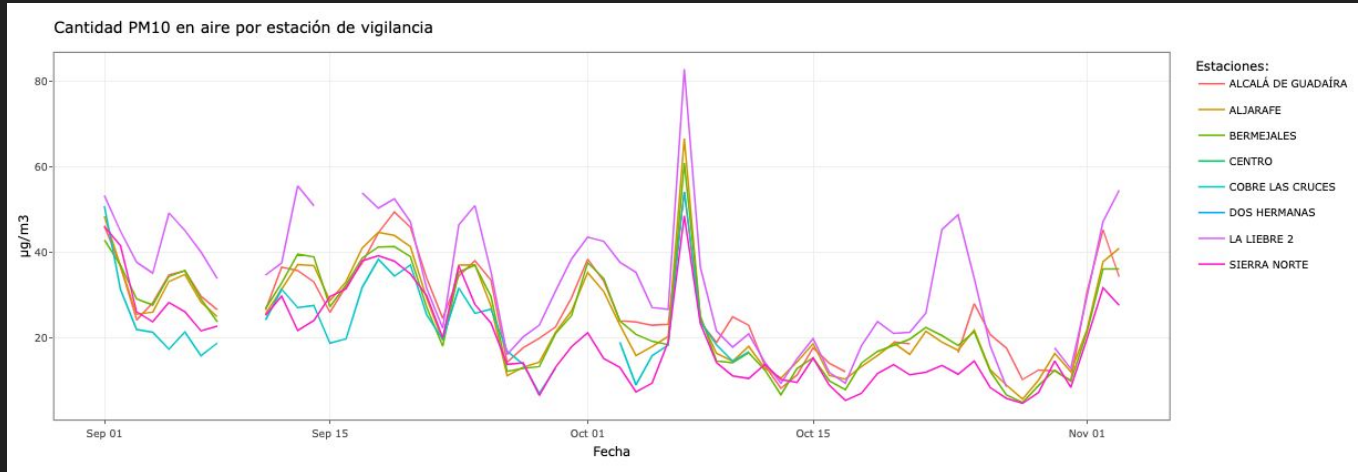
```
column(width = 4, style="padding:0px 30px;",  
  sliderInput("slider_fecha",  
    label = "Selecciona periodo:",  
    min = min(datos_contaminacion$F_FECHA),  
    max = max(datos_contaminacion$F_FECHA),  
    value = c(min(datos_contaminacion$F_FECHA), max(datos_contaminacion$F_FECHA))),  
),
```





Body: el contenido (II)

```
fluidRow(  
  column(width = 12, style="padding:0px 30px;",  
    plotlyOutput("grafico_lineas", height = 500)))  
)
```

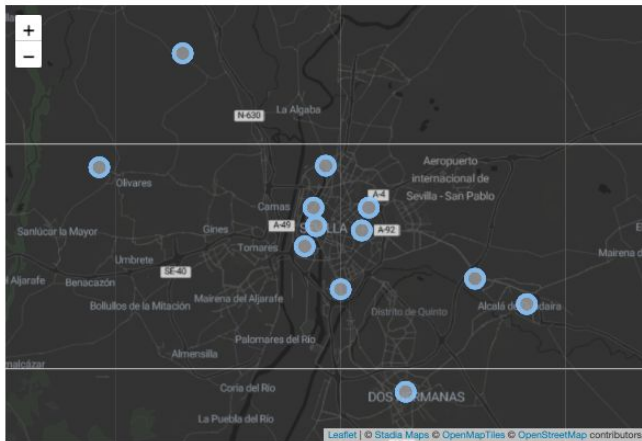




Body: el contenido (II)

```
shinydashboard::box( width = 12,  
  h3("Haz click en una estación del mapa:"),  
  column(width = 6,  
    leaflet::leafletOutput("mapa_leaflet", height = 500)),  
  column(width = 6,  
    reactableOutput("tabla_reactable", height = 500))  
)
```

Haz click en una estación del mapa:



| Estación | Fecha | Contaminante | Valor Registro |
|--------------------|------------|--------------|----------------|
| ALCALÁ DE GUADAÍRA | 2024-09-01 | PM10 | 46.13 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | PM25 | |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | NO2 | 4.29 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | O3 | 79.65 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | SO2 | 4.38 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | PM10 | 36.78 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | PM25 | |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | NO2 | 6.75 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | O3 | 70.88 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | SO2 | 4.54 |

1-10 of 4095 rows

Previous 1 2 3 4 ... 410 Next

¿IU y Server?



```
8 #
9
10 library(shiny)
11
12 # Define UI for application that draws a histogram
13 ui <- fluidPage(
14
15   # Application title
16   titlePanel("Old Faithful Geyser Data"),
17
18   # Sidebar with a slider input for number of bins
19   sidebarLayout(
20     sidebarPanel(
21       sliderInput("bins",
22                 "Number of bins:",
23                 min = 1,
24                 max = 50,
25                 value = 30)
26     ),
27
28     # Show a plot of the generated distribution
29     mainPanel(
30       plotOutput("distPlot")
31     )
32   )
33 )
34
35 # Define server logic required to draw a histogram
36 server <- function(input, output) {
37
38   output$distPlot <- renderPlot({
39     # generate bins based on input$bins from ui.R
40     x <- faithful[, 2]
41     bins <- seq(min(x), max(x), length.out = input$bins + 1)
42
43     # draw the histogram with the specified number of bins
44     hist(x, breaks = bins, col = 'darkgray', border = 'white',
45          xlab = 'Waiting time to next eruption (in mins)',
46          main = 'Histogram of waiting times')
47   })
48 }
49
50 # Run the application
51 shinyApp(ui = ui, server = server)
52
```



¿IU y Server?

```
ui <- dashboardPage(title = "Shiny Sevilla", skin = "black", header, sidebar, body)
```



¿IU y Server?



```
ui <- dashboardPage(title = "Shiny Sevilla", skin = "black", header, sidebar, body)
```



4



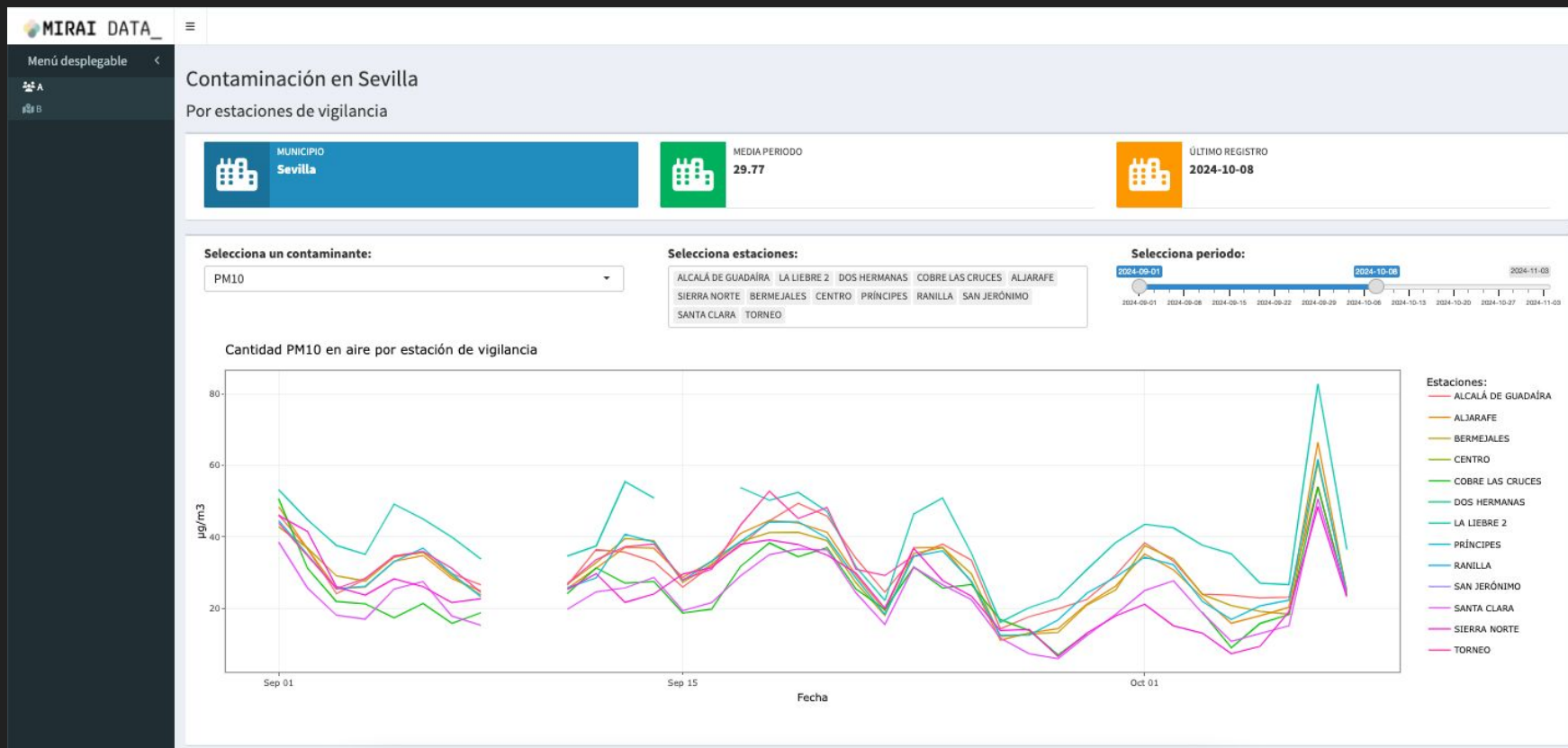
El Server



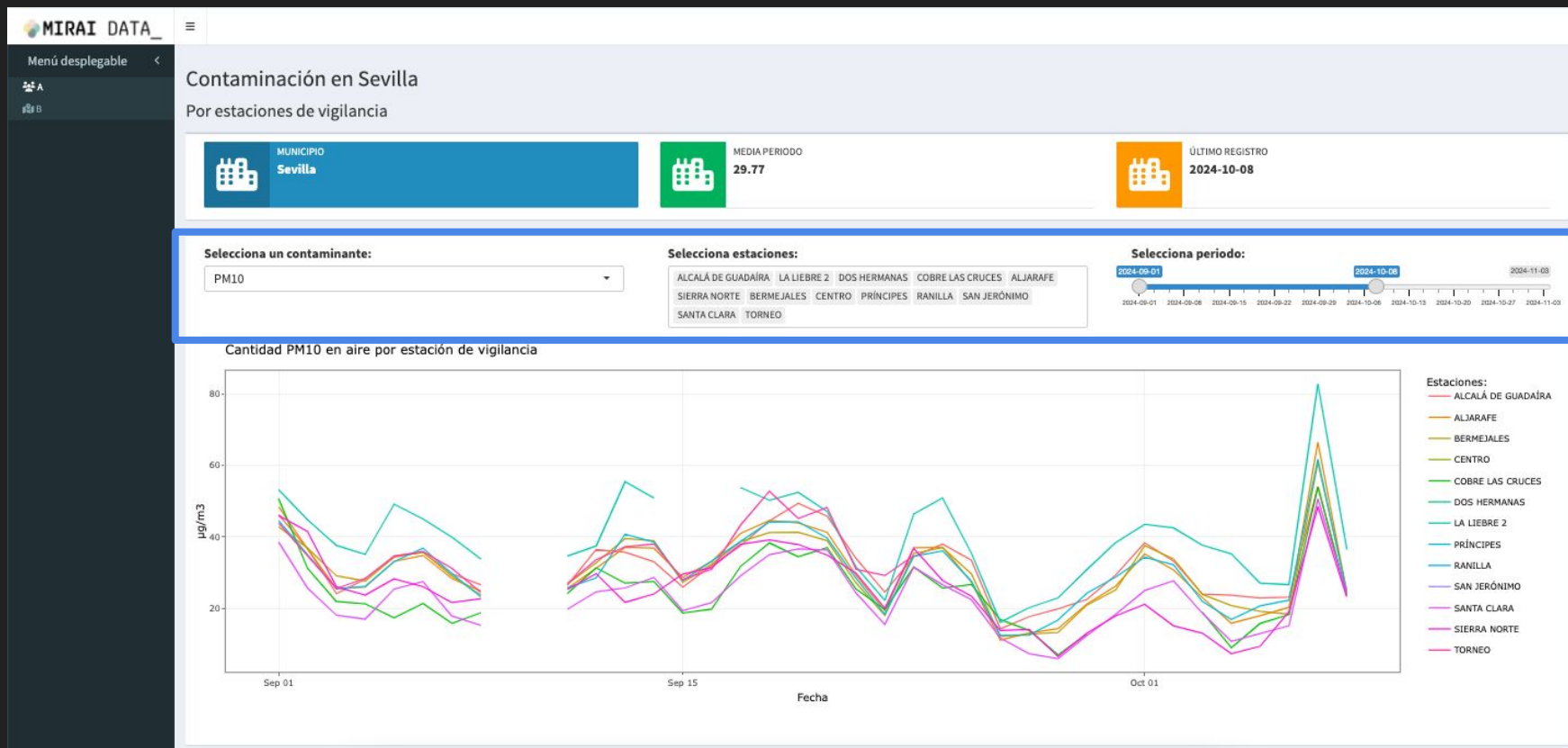


```
server <- function(input, output, session) {  
  ## TODO EL CÓDIGO VA AQUÍ  
}
```

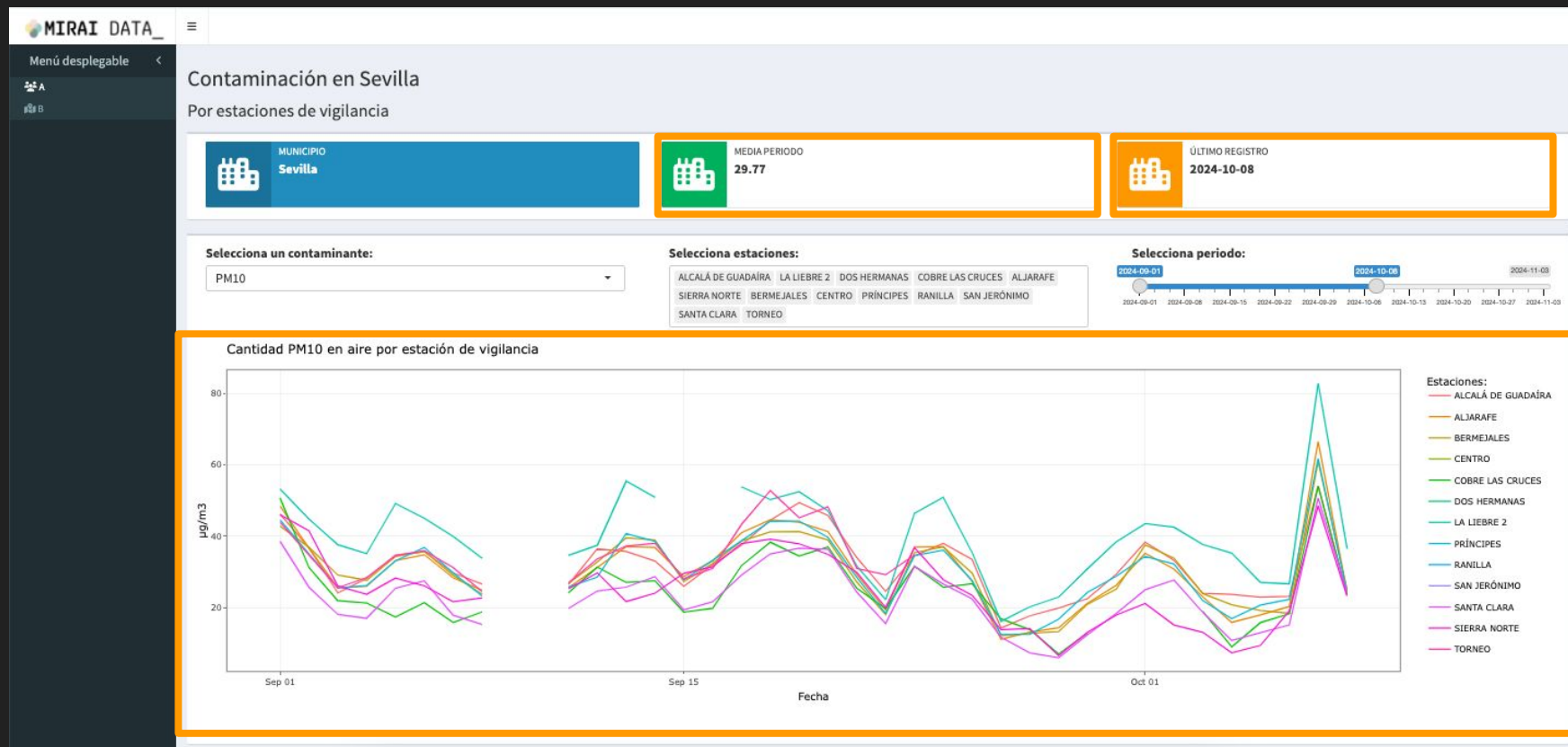
Filtrando con los filtros



Filtrando con los filtros



Filtrando con los filtros





Filtrando con los filtros

```
dataset_filtrado <- reactive({  
  datos_contaminacion %>%  
    filter(datos_contaminacion$F_FECHA >= min(input$slider_fecha) & datos_contaminacion$F_FECHA <= max(input$slider_fecha),  
           datos_contaminacion$Contaminante == input$select_contaminante,  
           datos_contaminacion$D_ESTACION %in% c(input$select_estaciones))  
})
```



Filtrando con los filtros

```
dataset_filtrado <- reactive({  
  datos_contaminacion %>%  
    filter(datos_contaminacion$F_FECHA >= min(input$slider_fecha) & datos_contaminacion$F_FECHA <= max(input$slider_fecha),  
           datos_contaminacion$Contaminante == input$select_contaminante,  
           datos_contaminacion$D_ESTACION %in% c(input$select_estaciones))  
})
```



Filtrando con los filtros

```
dataset_filtrado <- reactive({  
  datos_contaminacion %>%  
    filter(datos_contaminacion$F_FECHA >= min(input$slider_fecha) & datos_contaminacion$F_FECHA <= max(input$slider_fecha),  
           datos_contaminacion$Contaminante == input$select_contaminante,  
           datos_contaminacion$D_ESTACION %in% c(input$select_estaciones))  
})
```

```
column(width = 4, style="padding: 0px 30px:",  
  selectInput("select_contaminante",  
    label = "Selecciona un contaminante:",  
    choices = unique(datos_contaminacion$Contaminante),  
    selected = "PM10"),  
  multiple = FALSE),
```




Filtrando con los filtros

```
dataset_filtrado <- reactive({  
  datos_contaminacion %>%  
    filter(datos_contaminacion$F_FECHA >= min(input$slider_fecha) & datos_contaminacion$F_FECHA <= max(input$slider_fecha),  
           datos_contaminacion$Contaminante == input$select_contaminante,  
           datos_contaminacion$D_ESTACION %in% c(input$select_estaciones))  
})
```

```
column(width = 4, style="padding:0px 30px:",  
  selectInput("select_contaminante",  
    label = "Selecciona un contaminante",  
    choices = unique(datos_contaminacion$Contaminante),  
    selected = "PM10"),  
  multiple = FALSE),
```

```
column(width = 4, style="padding:0px 30px:",  
  selectizeInput("select_estaciones",  
    label = "Selecciona estaciones:",  
    choices = unique(datos_contaminacion$D_ESTACION),  
    selected = c(unique(datos_contaminacion$D_ESTACION)),  
    multiple = TRUE)),
```



Filtrando con los filtros

```
dataset_filtrado <- reactive({  
  datos_contaminacion %>%  
    filter(datos_contaminacion$F_FECHA >= min(input$slider_fecha) & datos_contaminacion$F_FECHA <= max(input$slider_fecha),  
           datos_contaminacion$Contaminante == input$select_contaminante,  
           datos_contaminacion$D_ESTACION %in% c(input$select_estaciones))  
})
```



Añadiendo datos reactivos

```
output$infobox_municipio <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Municipio", "Sevilla", icon = icon("city"), color="light-blue", fill = TRUE)  
})  
  
output$infobox_media_periodo <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Media periodo", round(mean(dataset_filtrado()$Registro_contaminante, na.rm = TRUE),2),  
    icon = icon("city"), color="green")  
})  
  
output$infobox_ultimo_registro <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Último registro", max(dataset_filtrado()$F_FECHA), icon = icon("city"), color="yellow")  
})
```



MUNICIPIO
Sevilla



MEDIA PERIODO
24.83



ÚLTIMO REGISTRO
2024-11-03



Añadiendo datos reactivos

```
output$infobox_municipio <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Municipio", "Sevilla", icon = icon("city"), color="light-blue", fill = TRUE)  
})
```

```
output$infobox_media_periodo <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Media periodo", "24.83", icon = icon("calendar"), color="green", fill = TRUE)  
})
```

```
output$infobox_ultimo_registro <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Último registro", "2024-11-03", icon = icon("calendar"), color="orange", fill = TRUE)  
})
```

BODY (UI)

Primer recuadro de la estructura del dashboard (Siguiendo el ppt):

```
shinydashboard::box( width = 12, height = 120, style="font-size: 18px;",  
  infoBoxOutput("infobox_municipio", width = 4),  
  # Se asigna un id a la caja de información y una anchura (3x4)  
  infoBoxOutput("infobox_media_periodo", width = 4),  
  infoBoxOutput("infobox_ultimo_registro", width = 4)),
```



MUNICIPIO
Sevilla



MEDIA PERIODO
24.83



ÚLTIMO REGISTRO
2024-11-03



Añadiendo datos reactivos

```
output$infobox_municipio <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Municipio", "Sevilla", icon = icon("city"), color="light-blue", fill = TRUE)  
})
```

```
output$infobox_media_periodo <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Media periodo", "24.83", icon = icon("calendar"), color="green", fill = TRUE)  
})
```

```
output$infobox_ultimo_registro <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Último registro", "2024-11-03", icon = icon("calendar"), color="orange", fill = TRUE)  
})
```

BODY (UI)

Primer recuadro de la estructura del dashboard (Siguiendo el ppt):

```
shinydashboard::box( width = 12, height = 120, style="font-size: 18px;",  
  infoBoxOutput("infobox_municipio", width = 4),  
  # Se asigna un id a la caja de información y una anchura (3x4)  
  infoBoxOutput("infobox_media_periodo", width = 4),  
  infoBoxOutput("infobox_ultimo_registro", width = 4)),
```



MUNICIPIO
Sevilla



MEDIA PERIODO
24.83



ÚLTIMO REGISTRO
2024-11-03



Añadiendo datos reactivos

```
output$infobox_municipio <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Municipio", "Sevilla", icon = icon("city"), color="light-blue", fill = TRUE)  
})  
  
output$infobox_media_periodo <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Media periodo", round(mean(dataset_filtrado()$Registro_contaminante, na.rm = TRUE),2),  
    icon = icon("city"), color="green")  
})  
  
output$infobox_ultimo_registro <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Último registro", max(dataset_filtrado()$F_FECHA), icon = icon("city"), color="yellow")  
})
```



MUNICIPIO
Sevilla



MEDIA PERIODO
24.83



ÚLTIMO REGISTRO
2024-11-03



Añadiendo datos reactivos

```
output$infobox_municipio <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Municipio", "Sevilla", icon = icon("city"), color="light-blue", fill = TRUE)  
})  
  
output$infobox_media_periodo <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Media periodo", round(mean(dataset_filtrado()$Registro_contaminante na.rm = TRUE),2),  
    icon = icon("city"), color="green")  
})  
  
output$infobox_ultimo_registro <- shinydashboard::renderInfoBox({  
  shinydashboard::infoBox("Último registro", max(dataset_filtrado()$F_FECHA), icon = icon("city"), color="yellow")  
})
```



MUNICIPIO
Sevilla



MEDIA PERIODO
24.83

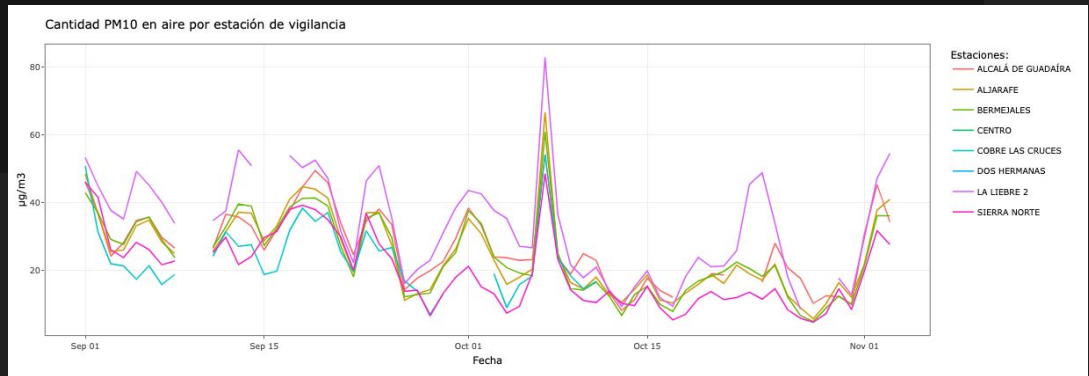


ÚLTIMO REGISTRO
2024-11-03



Añadiendo datos reactivos

```
output$grafico_lineas <- renderPlotly({  
  
  plot <- dataset_filtrado() %>%  
    ggplot(aes(x = F_FECHA, y = Registro_contaminante, color = D_ESTACION), group = D_ESTACION)+  
    geom_line()+  
    theme_bw()+  
    labs(title = paste("Cantidad", input$select_contaminante, "en aire por estación de vigilancia"),  
         x = "Fecha",  
         y = "µg/m3",  
         color = "Estaciones: ") +  
    theme(legend.position = "bottom")  
  
  ggplotly(plot)  
  
})
```





Añadiendo datos reactivos

```
output$grafico_lineas <- renderPlotly({  
  
  plot <- dataset_filtrado() %>%  
    ggplot(aes(x = F_FECHA, y = Registro_contaminante, color = D_ESTACION), group = D_ESTACION)+  
    geom_line()+  
    theme_bw()+  
    labs(title = paste("Cantidad", input$select_contaminante, "en aire por estación de vigilancia"),  
         x = "Fecha",  
         y = "µg/m3",  
         color = "Estaciones: ") +  
    theme(legend.position = "bottom")  
  
  ggplotly(plot)  
  
})
```

BODY (UI)

```
fluidRow(  
  column(width = 12, style="padding:0px 30px;",  
    plotlyOutput("grafico_lineas", height = 500)))  
),
```



5

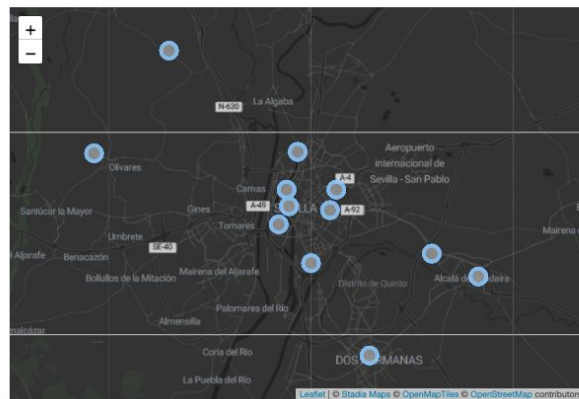


¿Y si queremos interactuar sin widgets?

```
output$mapa_leaflet <- renderLeaflet({  
  
  leaflet() %>%  
    addProviderTiles("Stadia.AlidadeSmoothDark") %>%  
    setView(-5.98818, 37.3905, zoom = 11) %>%  
    addCircleMarkers(data = datos_contaminacion,  
      radius = 10,  
      color = "lightblue",  
      opacity = 0.02,  
      fillOpacity = 0.01,  
      popup = ~datos_contaminacion$ID_ESTACION,  
      layerId = ~id) ## Es mejor si el ID es un número  
})
```

```
output$tabla_reactable <- renderReactable({  
  datos_tabla_sin_filtros <- datos_contaminacion %>%  
    select(Estación = D_ESTACION, Fecha = F_FECHA, Contaminante,  
      `Valor Registro` = Registro_contaminante) %>%  
    mutate(`Valor Registro` = round(`Valor Registro`, 2))  
  
  reactable(datos_tabla_sin_filtros,  
    searchable = TRUE,  
    striped = TRUE,  
    columns = list(  
      geometry = colDef(show = F))  
  )  
})
```

Haz click en una estación del mapa.



| Estación | Fecha | Contaminante | Valor Registro |
|--------------------|------------|--------------|----------------|
| ALCALÁ DE GUADAÍRA | 2024-09-01 | PM10 | 46.13 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | PM25 | |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | NO2 | 4.29 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | O3 | 79.65 |
| ALCALÁ DE GUADAÍRA | 2024-09-01 | SO2 | 4.38 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | PM10 | 36.78 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | PM25 | |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | NO2 | 6.75 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | O3 | 70.88 |
| ALCALÁ DE GUADAÍRA | 2024-09-02 | SO2 | 4.54 |

1-10 of 4095 rows

Previous 1 2 3 4 5 ... 410 Next



¿Y si queremos interactuar sin widgets?

- ❖ **observe({})**: Más general. “Observa” lo que hace el usuario y re-renderiza uno o varios elementos automáticamente cuando se cumplen ciertas condiciones.
- ❖ **ObserveEvent({})**: Más específico. Espera a que el usuario haga una opción específica (por ej. apretar un botón) para actuar.



```
observe({
  event <- input$mapa_leaflet_marker_click
  if (is.null(event))
    return()
  output$tabla_reactable <- renderReactable({
    datos_tabla_sin_filtros <- datos_contaminacion %>%
      select(Estación = D_ESTACION, Fecha = F_FECHA, Contaminante,
        `Valor Registro` = Registro_contaminante) %>%
      mutate(`Valor Registro` = round(`Valor Registro`, 2))

    reactable(datos_tabla_sin_filtros,
      searchable = TRUE,
      striped = TRUE,
      columns = list(
        geometry = colDef(show = F))
  )
})

datos_tabla_con_filtros <- datos_contaminacion %>%
  filter(id == event$id) %>%
  select(Estación = D_ESTACION, Fecha = F_FECHA, Contaminante,
    `Valor Registro` = Registro_contaminante) %>%
  mutate(`Valor Registro` = round(`Valor Registro`, 2))

output$tabla_reactable <- renderReactable({
  reactable(datos_tabla_con_filtros,
    searchable = TRUE,
    striped = TRUE,
    columns = list(
      geometry = colDef(show = F))
  )
})
})
```



¡Todo listo para renderizar!

```
226 shinyApp(ui, server)
```



¡Felicitades, has creado un Shiny totalmente funcional! 🎉



@Mirai_data



@MiraiData



Muchas gracias



MIRAI DATA_



data.mirai@gmail.com

Mireia Camacho