Components of a shinydashboard

BUILDING DASHBOARDS WITH SHINYDASHBOARD

Png Kee Seng
Researcher

What is a shinydashboard?

- A shinydashboard is just another shinyApp with a dashboard output
  - A UI is required
  - A server is required
- A `shinyApp()` is also needed to glue the UI and server together
- See that the UI is defined by `dashboardPage()` and not `fluidPage()`

```
ui <- dashboardPage(header, sidebar, body)
```

```
server <- function(input, output){
...
}
```

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

The user interface (UI)

- The UI now contains more parts
  1. A header
  2. A side bar
  3. A body
- A shinydashboard UI is put together with `dashboardPage()`

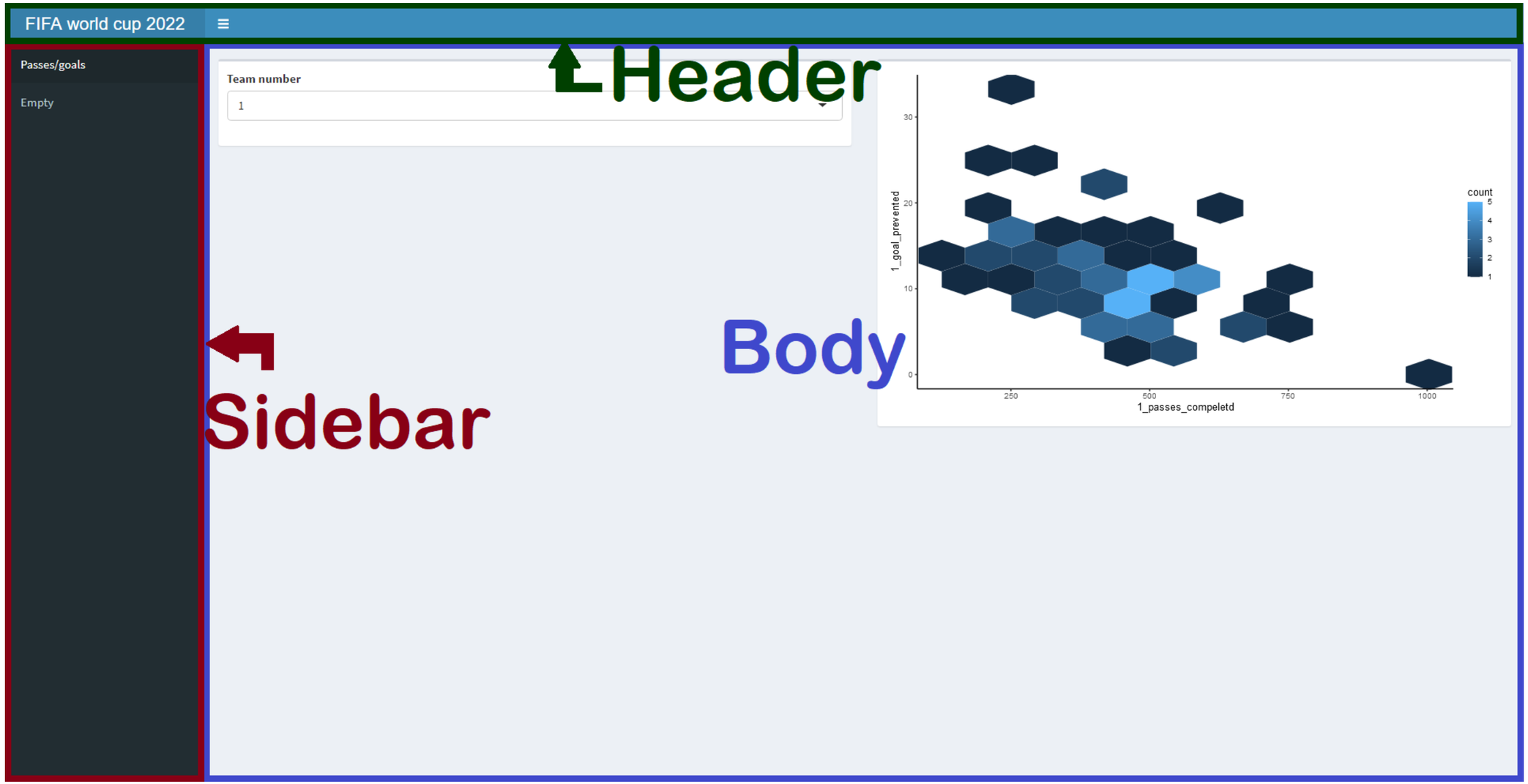
```
header <- dashboardHeader(...)
```

```
sidebar <- dashboardSidebar(...)
```

```
body <- dashboardBody(...)
```

```
# a shinydashboard ui
ui <- dashboardPage(header, sidebar, body)
```

A preview of the UI



Revisiting the restaurant analogy

- The UI is like the restaurant
- The server is like the waiter
- In particular, the UI can be further broken down into smaller components

[1] Image by macrovector on Freepik

The UI as a multi-course meal

- In the UI:
  - The header is like the signboard
  - The side bar is like the list of courses
  - The outputs and inputs are like the dishes and orders
- One course is presented at a time
- Instructions are given for each course
- Each course is like a shinyApp
  - We can have multiple sets of inputs and outputs

[1] Image by brgfx on Freepik

An empty shinydashboard

```
library(shinydashboard)
library(shiny)
```

```
header <- dashboardHeader(title = "My first dashboard")
```

```
sidebar <- dashboardSidebar()
```

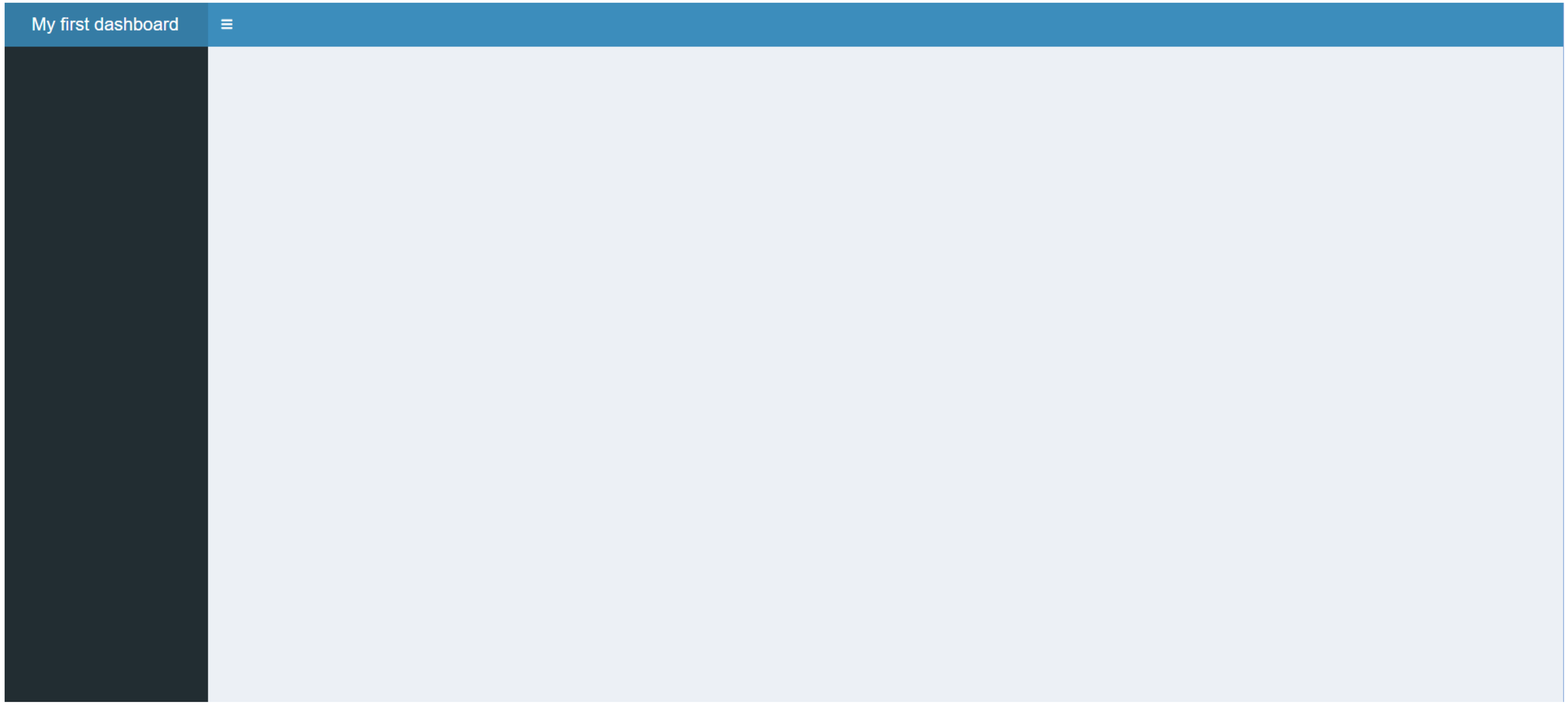
```
body <- dashboardBody()
```

```
ui <- dashboardPage(header, sidebar, body)
```

Rendering the shinydashboard

```
header <- dashboardHeader(title = "My first dashboard")

sidebar <- dashboardSidebar()

body <- dashboardBody()

ui <- dashboardPage(header, sidebar, body)

server <- function(input, output) {
}

shinyApp(ui, server)
```

Rendering the shinydashboard



Let's practice!

BUILDING DASHBOARDS WITH SHINYDASHBOARD

The header and sidebar

BUILDING DASHBOARDS WITH SHINYDASHBOARD

Png Kee Seng
Researcher

datacamp

The header

- The header is like the signboard of the restaurant
- It can contain the dashboard title
- It can contain dropdown menus
  - E.g., messages, notifications, tasks

[1] Image by upklyak on Freepik

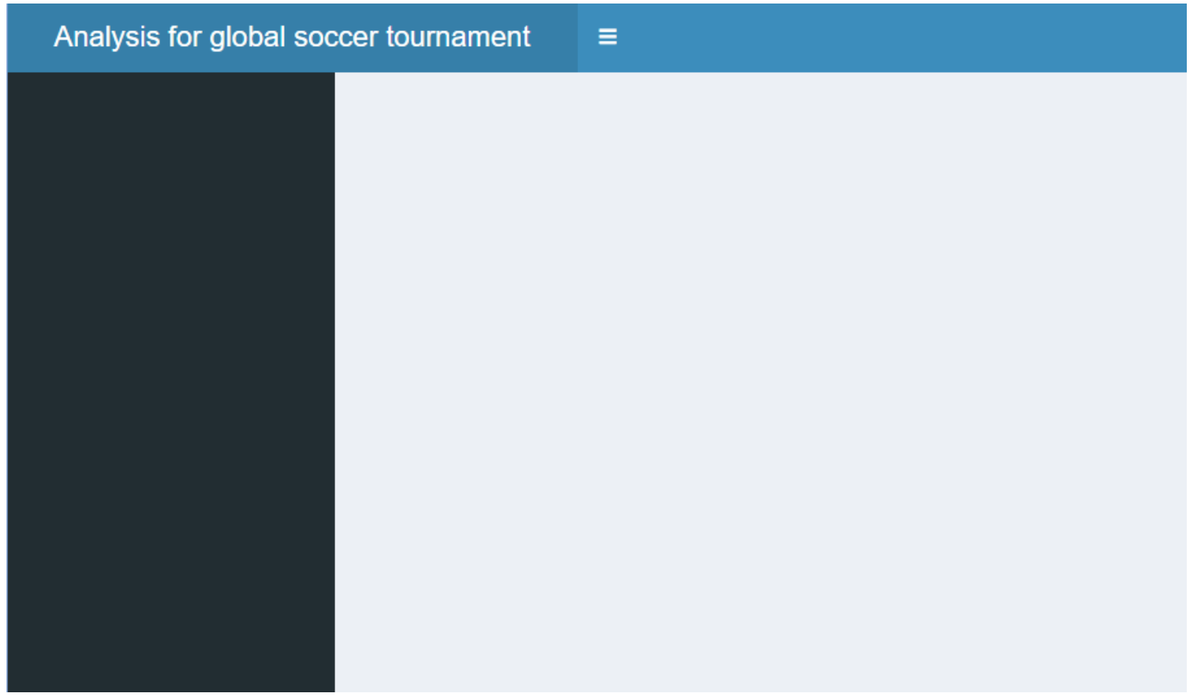
The shinydashboard title

- The header is defined by `dashboardHeader()`
- What if the title is too long?
  - Set the `titleWidth` argument
  - Default is 230 pixels

```
header <- dashboardHeader(title = "Soccer tournament")
```

```
header <- dashboardHeader(title = "Analysis for global soccer tournament")
```

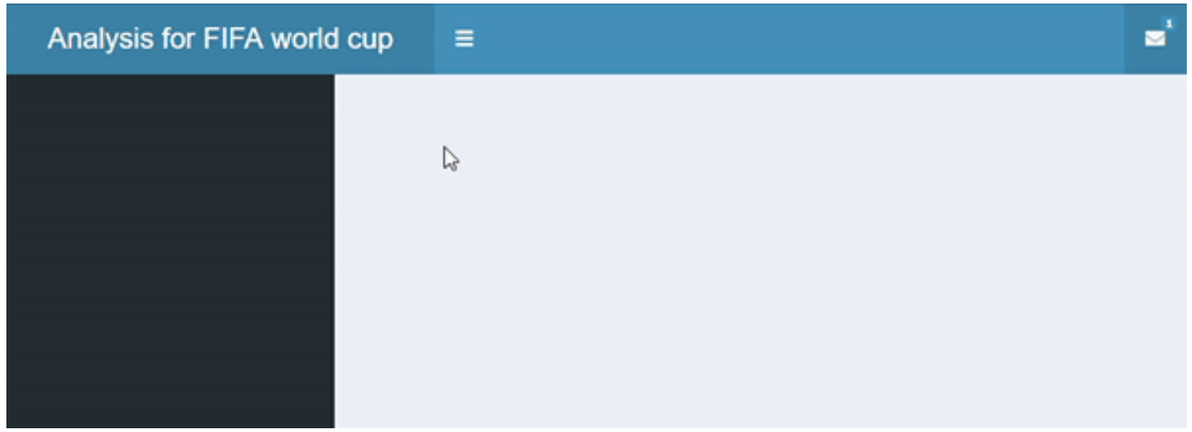
```
header <- dashboardHeader(title = "Analysis for global soccer tournament")
                          titleWidth = 400)
```



Adding a dropdown menu with a message

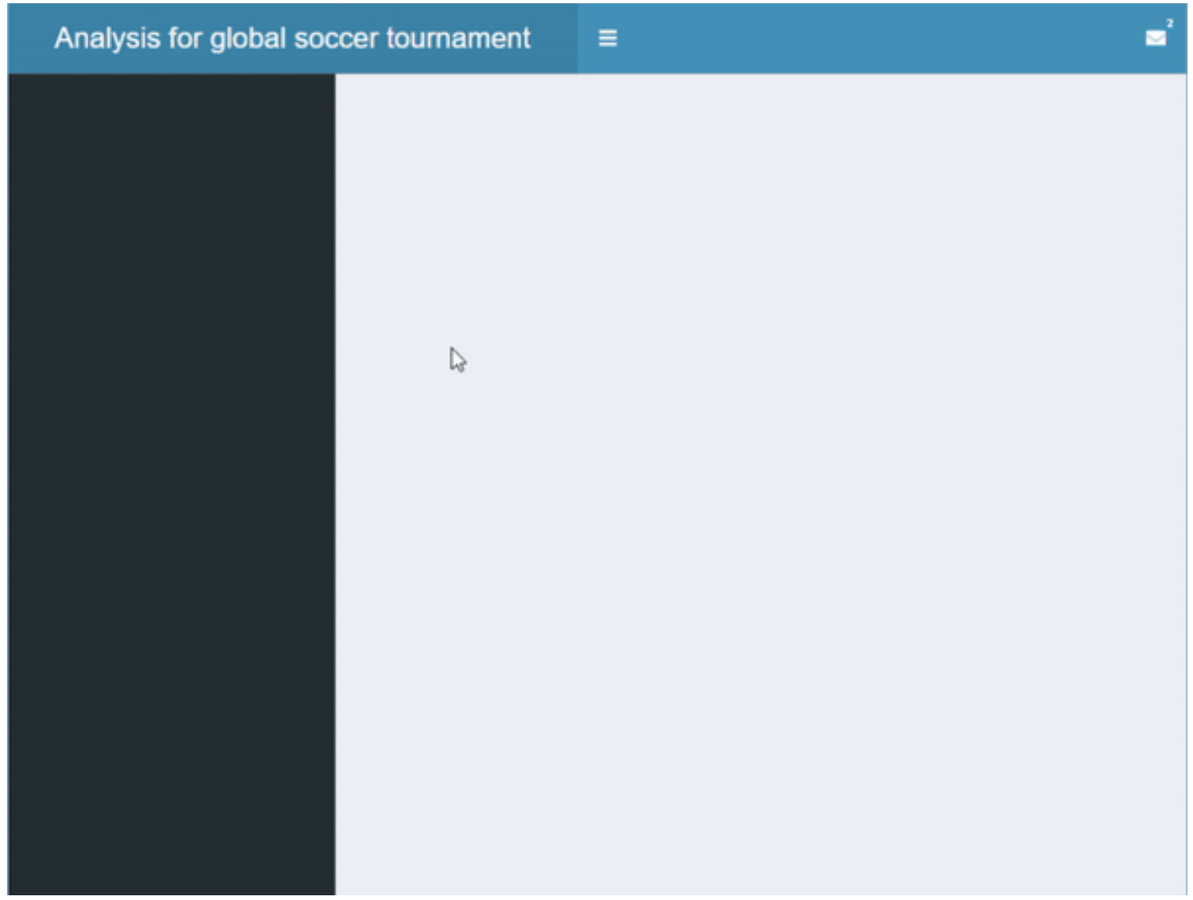
- Dropdown menus can be added to the header
- Three types of dropdown menus
  - Messages
  - Notifications
  - Tasks
- Add `dropdownMenu()` of `type` "messages"
  - Add a `messageItem()`
  - The icon is that of `icon("user")` by default
  - We can also set `time`

```
header <- dashboardHeader(
  title = "Analysis for FIFA world cup",
  titleWidth = 300,
  dropdownMenu(type = "messages",
    messageItem("Data division",
      "Keep up the good work!",
      time = "5 mins"
)))
```



Adding a dropdown menu with multiple messages

- Add another `messageItem()` within `dropdownMenu()` called "Twitter"



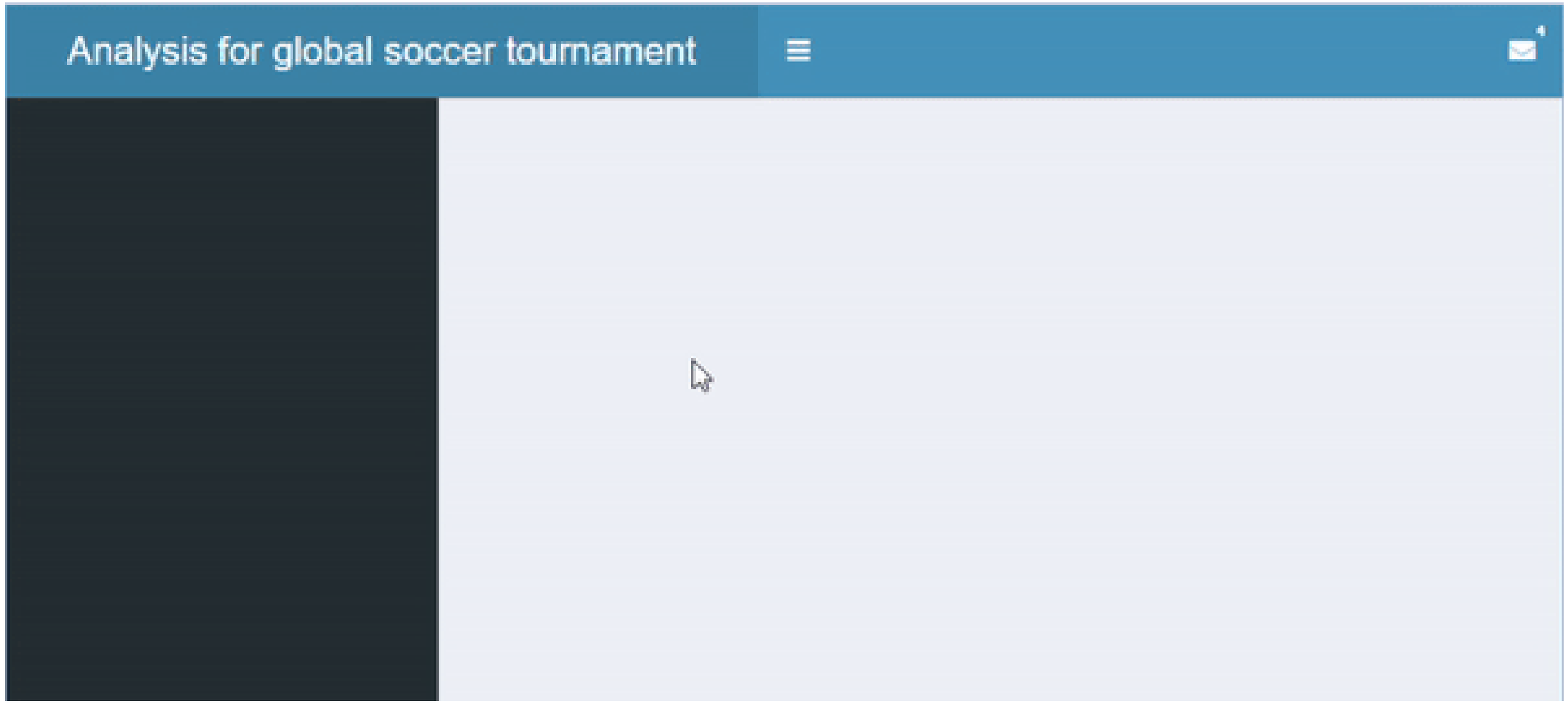

```
header <- dashboardHeader(
  title = "Analysis for global soccer tournament",
  titleWidth = 400,
  dropdownMenu(type = "messages",
    messageItem("Data division",
      "Keep up the good work!",
      time = "5 mins"
    ),
    messageItem("Twitter",
      "You have a Tweet!",
      time = "1 hour", icon=icon("twitter")
    )
  )
)
```

Adding a dropdown menu with multiple items

- We can also add other types of items
  - `messageItem()`
  - `notificationItem()`
  - `taskItem()`
- There are appearance differences
  - `messageItem()` : Default icon is `icon("user")`
  - `notificationItem()` : Default icon is `icon("exclamation-triangle")`
  - `taskItem()` : Contains a progress bar

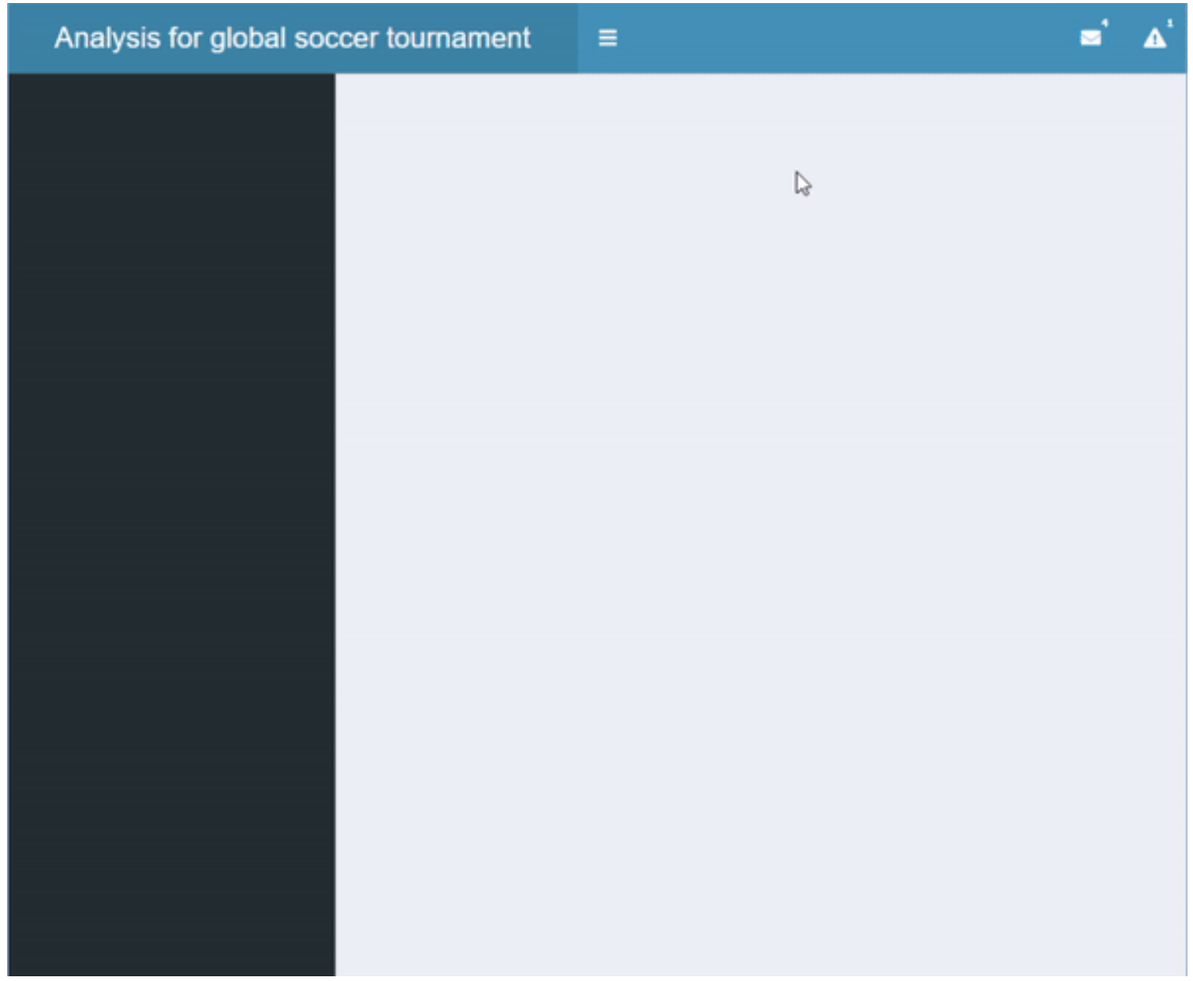
```
header <- dashboardHeader(
  title = "Analysis for global soccer tournament",
  titleWidth = 400,
  dropdownMenu(type = "messages",
    messageItem("Data division",
      "Keep up the good work!",
      time = "5 mins"),
    messageItem("Twitter",
      "You have a Tweet!",
      time = "1 hour", icon=icon("twitter")),
    notificationItem("This is a notification."
    ),
    taskItem(value = 30, color = "blue",
      "Dashboard construction")
))
```

Adding a dropdown menu with multiple items



Adding more dropdown menus

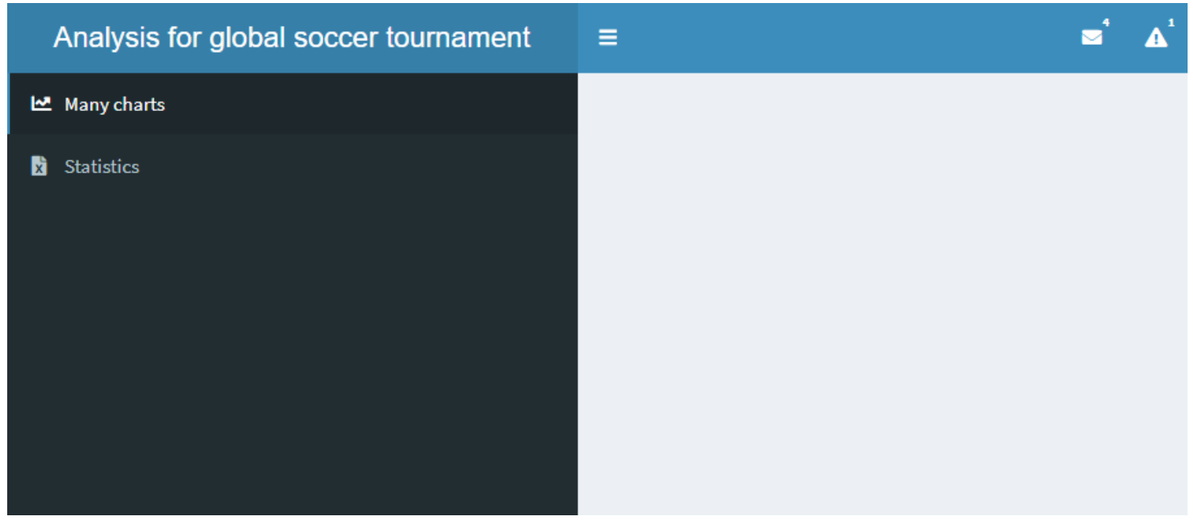
- We can also add another dropdown menu




```
header <- dashboardHeader(
  title = "Analysis for global soccer tournament",
  titleWidth = 400,
  dropdownMenu(type = "messages",
    messageItem("Data division",
      "Keep up the good work!",
      time = "5 mins"),
    messageItem("Twitter",
      "You have a Tweet!",
      time = "1 hour", icon=icon("twitter")),
    notificationItem("This is a notification."),
    taskItem(value = 30, color = "blue",
      "Dashboard construction")),
  dropdownMenu(type = "notifications",
    notificationItem(icon = icon("users"),
      "This is another notification."
)))
```

The sidebar

- The sidebar is defined by `dashboardSidebar()`
- The width of the sidebar can be adjusted by setting `width`.
- `sidebarMenu()` allows us to place several pages in a shinydashboard




```
sidebar <- dashboardSidebar()
```

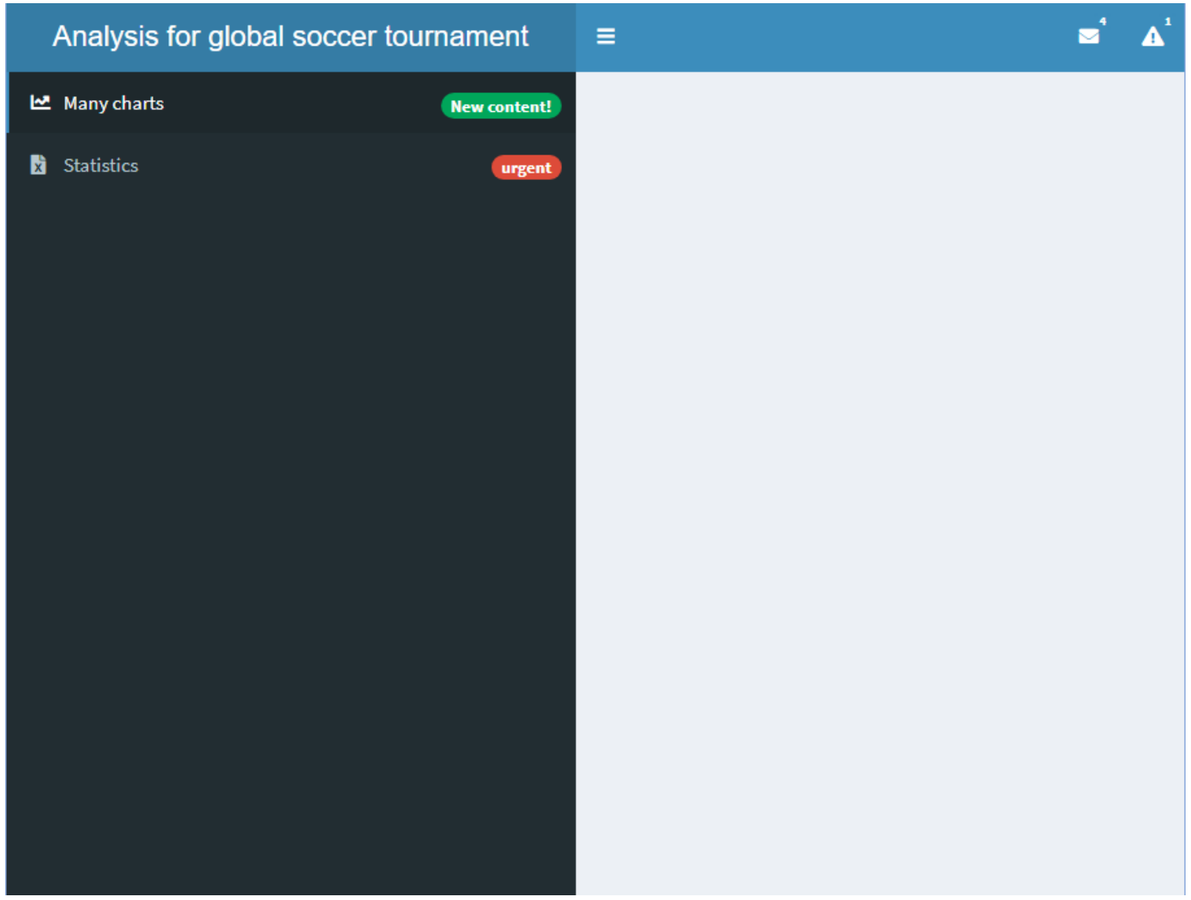
```
sidebar <- dashboardSidebar(width=400)
```

```
sidebar <- dashboardSidebar(width=400,
  sidebarMenu(
    id = "pages",
    menuItem("Many charts", tabName = "charts",
        icon = icon("chart-line")),
    menuItem("Statistics", icon = icon("file-excel"),
        tabName = "stats")
  )
)
```

```
sidebar <- dashboardSidebar(disable = TRUE)
```

Adding a badge

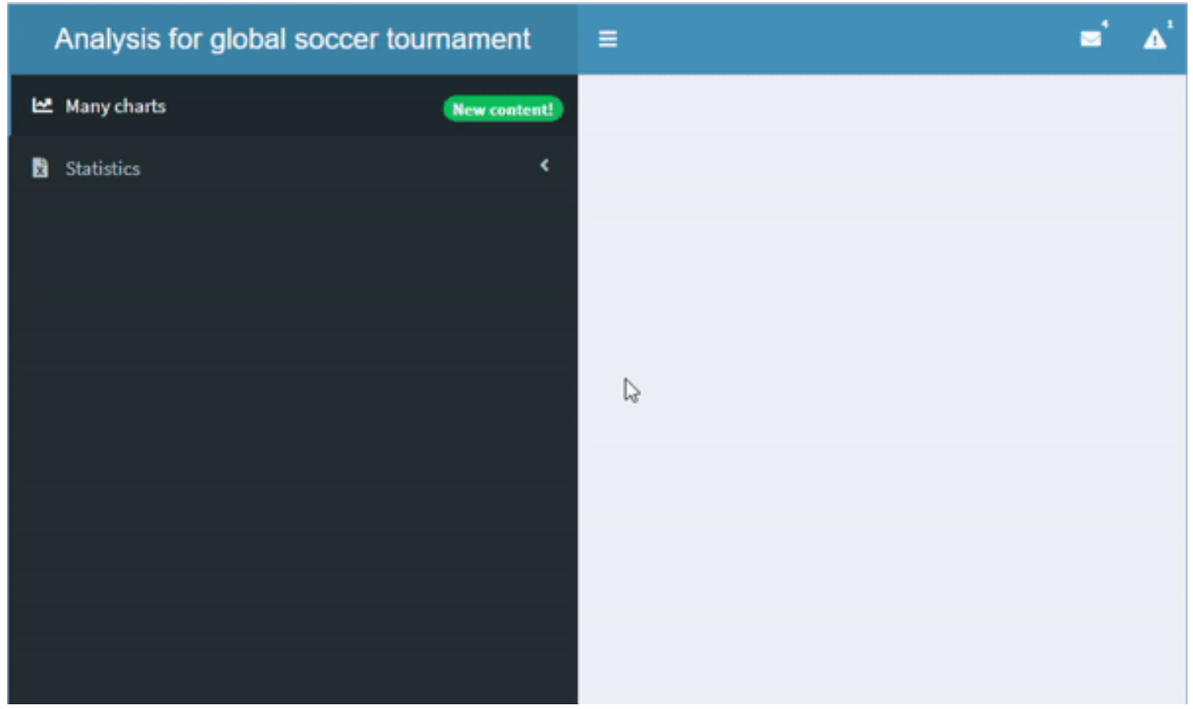
- A badge can also be added
  - Set `badgeLabel` and `badgeColor`




```
sidebar <- dashboardSidebar(width=400,
  sidebarMenu(
    id = "pages",
    menuItem("Many charts", tabName = "charts",
             icon = icon("chart-line"),
             badgeLabel = "New content!",
             badgeColor = "green"),
    menuItem("Statistics", icon = icon("file-excel"),
             tabName = "stats",
             badgeLabel = "urgent",
             badgeColor = "red")
  )
)
```

Adding subtabs

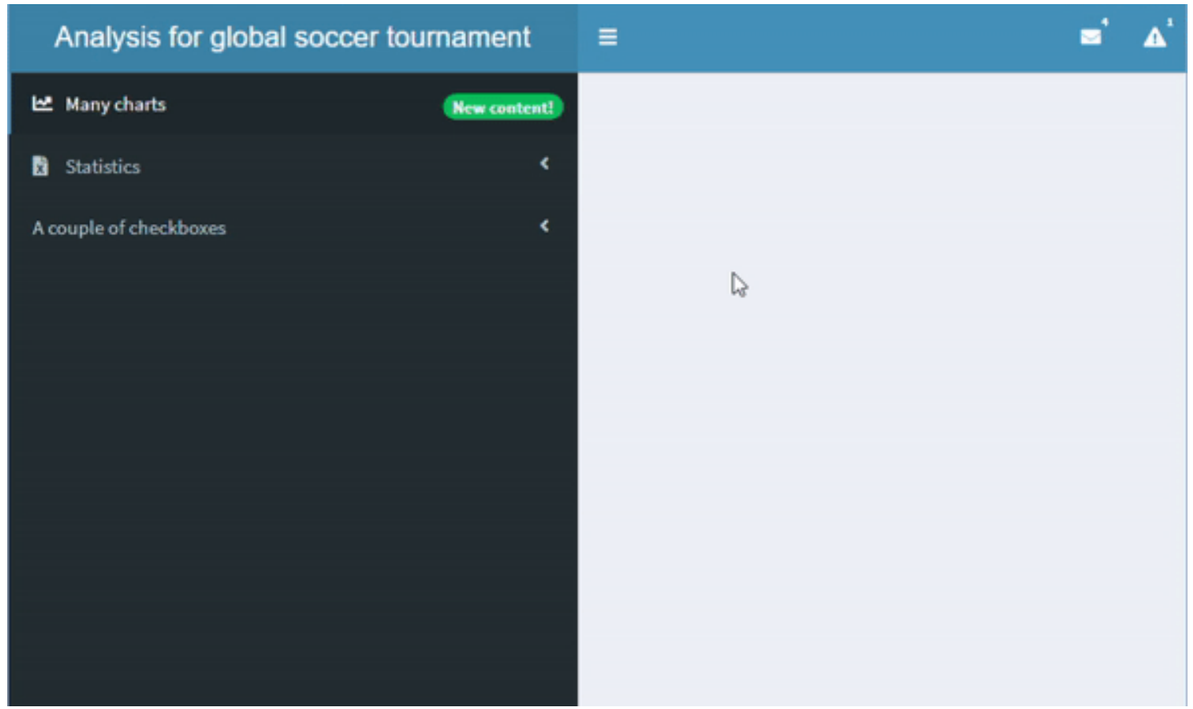
- Subtabs can be added by adding `menuSubItem()`
- Note that badges cannot be added to a tab containing subtabs




```
sidebar <- dashboardSidebar(width=400,
  sidebarMenu(
    id = "pages",
    menuItem("Many charts", tabName = "charts",
             icon = icon("chart-line"),
             badgeLabel = "New content!",
             badgeColor = "green"),
    menuItem("Statistics", icon = icon("file-excel"),
            tabName = "stats",
            menuSubItem("Team 1", tabName = "team1",
                     icon=icon("user")),
            menuSubItem("Team 2", tabName = "team2",
                     icon=icon("user")))
  )
)
```

Adding inputs and outputs in the sidebar

- Inputs and outputs can be added to the sidebar
- We also have to define inputs and outputs in `server`




```
sidebar <- dashboardSidebar(width=400,
  sidebarMenu(
    id = "pages",
    menuItem("Many charts", tabName = "charts",
             icon = icon("chart-line"),
             badgeLabel = "New content!",
             badgeColor = "green"),
    menuItem("Statistics", icon = icon("file-excel"),
            tabName = "stats",
            menuSubItem("Team 1", tabName = "team1",
            icon=icon("user")),
            menuSubItem("Team 2", tabName = "team2",
            icon=icon("user"))),
    menuItem("A couple of checkboxes",
             checkboxGroupInput("checkboxes",
              "Day of the week",
              choices =  c("Mon","Tue","Wed","Thu","Fri","Sat","Sun"),
              selected = c("Mon","Tue","Wed","Thu","Fri")))
  )
)
```

Let's practice!

BUILDING DASHBOARDS WITH SHINYDASHBOARD

The body

BUILDING DASHBOARDS WITH SHINYDASHBOARD

Png Kee Seng
Researcher

datacamp

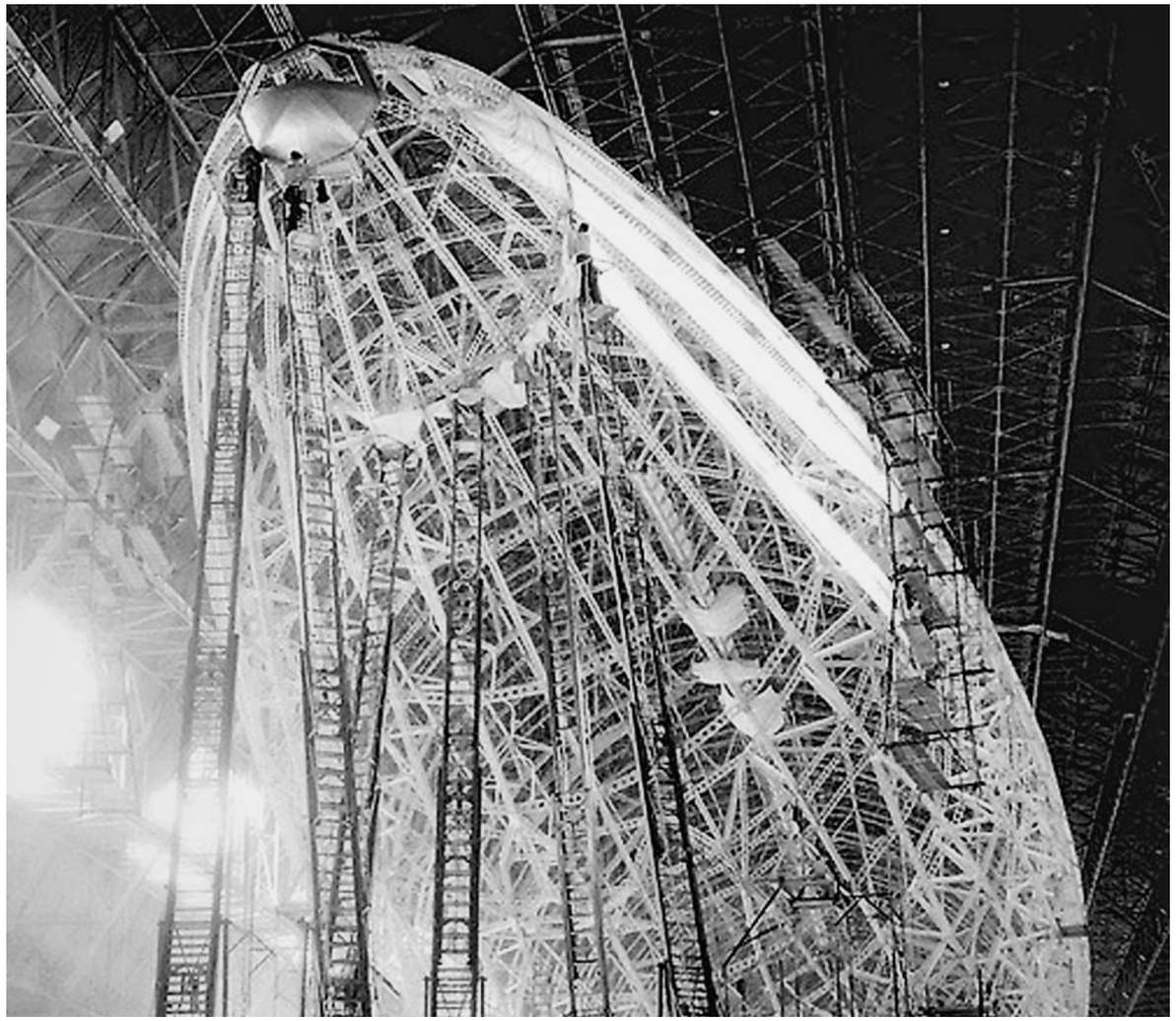
An overview

- Bulk of the shinydashboard UI
- Typically, inputs and outputs are placed here
  - These are like the dishes and orders at a restaurant
  - The server helps with communication
- The body is defined by `dashboardBody()` and is conventionally stored as `body`

```
body <- dashboardBody(...)
```

Wireframing

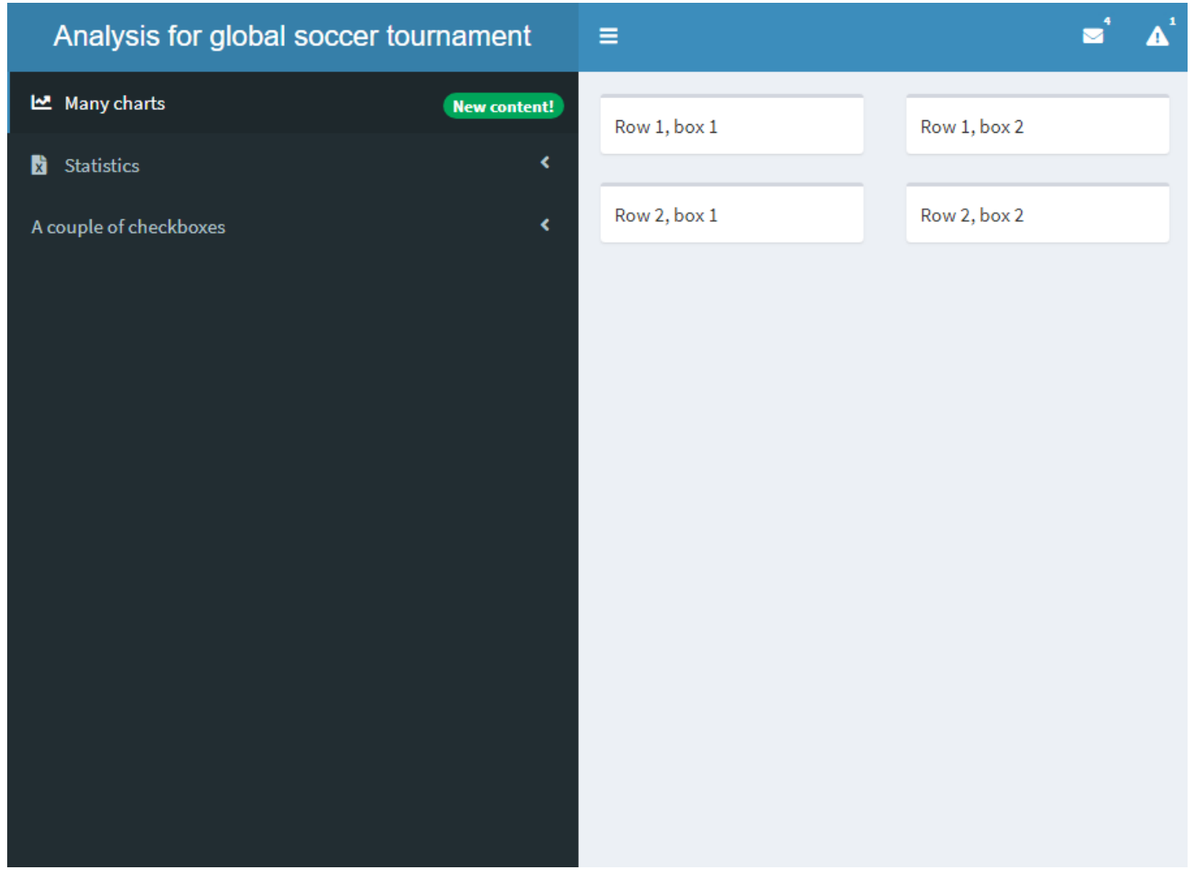
- Carry out wire-framing
  - Set up the structure before adding in the contents
- In a shinydashboard, first decide positions of objects



[1] Brett Jordan, Flickr

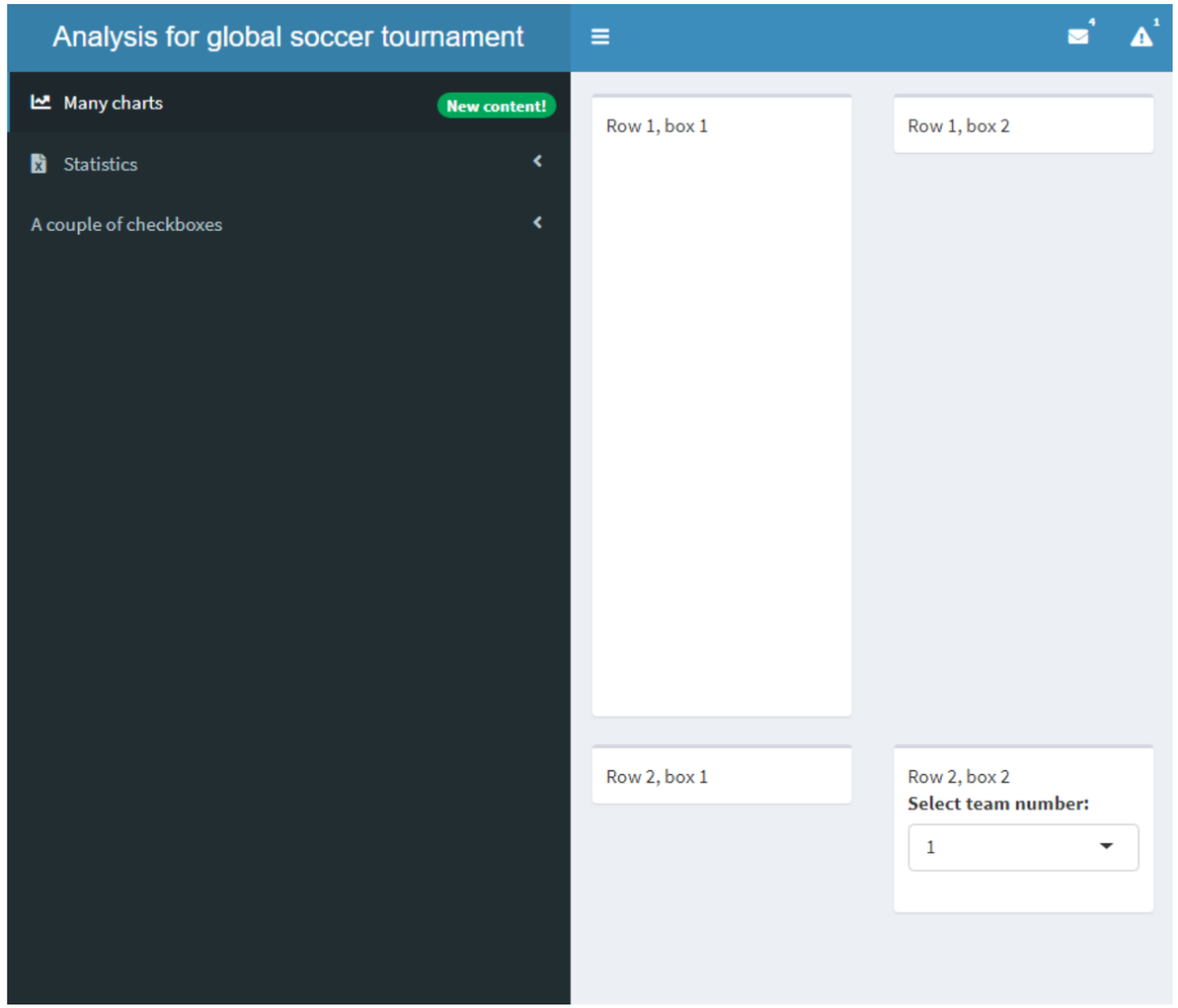
Adding rows and boxes: empty boxes

- Add rows with `fluidRow()`
- And add boxes with `box()`




```
body <- dashboardBody(
  fluidRow(
           box("Row 1, box 1"),
           box("Row 1, box 2")
           ),
  fluidRow(
           box("Row 2, box 1"),
           box("Row 2, box 2")
           )
)
```

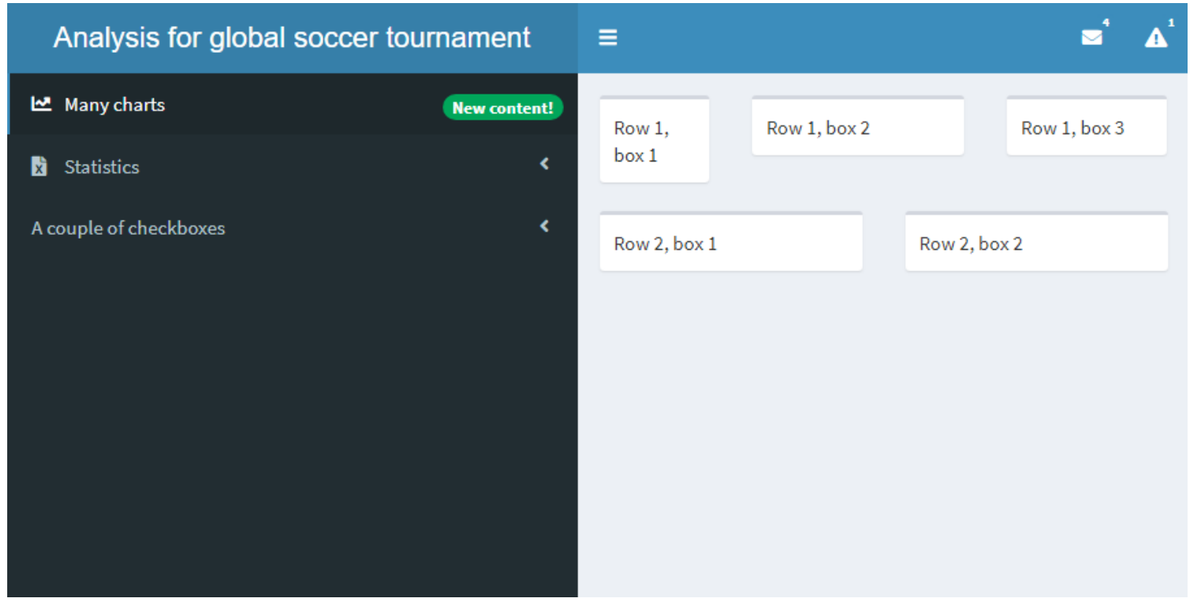
Adding rows and boxes: content in boxes




```
body <- dashboardBody(
  fluidRow(box("Row 1, box 1",
               plotOutput('plot')),
           box("Row 1, box 2")),
  fluidRow(box("Row 2, box 1"),
           box("Row 2, box 2",
               selectInput("select",
                           "Select team number:",
                           choices = c(1,2))))
)
```

Prevent overflowing boxes

- Each box takes up 6 out of 12 units of the total horizontal space by default
- To prevent overflowing, set `width`
  - In our example, set widths to 3, 5 and 4
- Adding `box()` 's is highly recommended

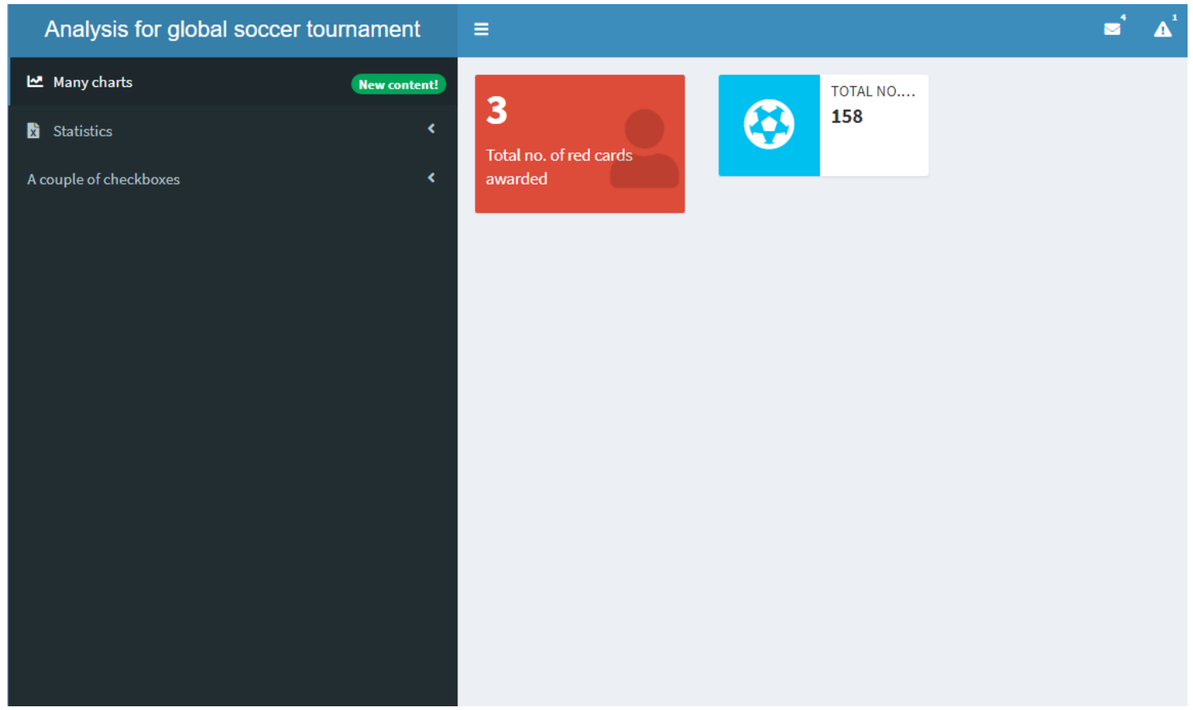



```
body <- dashboardBody(
  fluidRow(box("Row 1, box 1"),
           box("Row 1, box 2"),
           box("Row 1, box 3")),
  fluidRow(box("Row 2, box 1"),
           box("Row 2, box 2"))
)
```

```
body <- dashboardBody(
  fluidRow(box("Row 1, box 1", width = 3),
           box("Row 1, box 2", width = 5),
           box("Row 1, box 3", width = 4)),
  fluidRow(box("Row 2, box 1"),
           box("Row 2, box 2"))
)
```

valueBox and infoBox

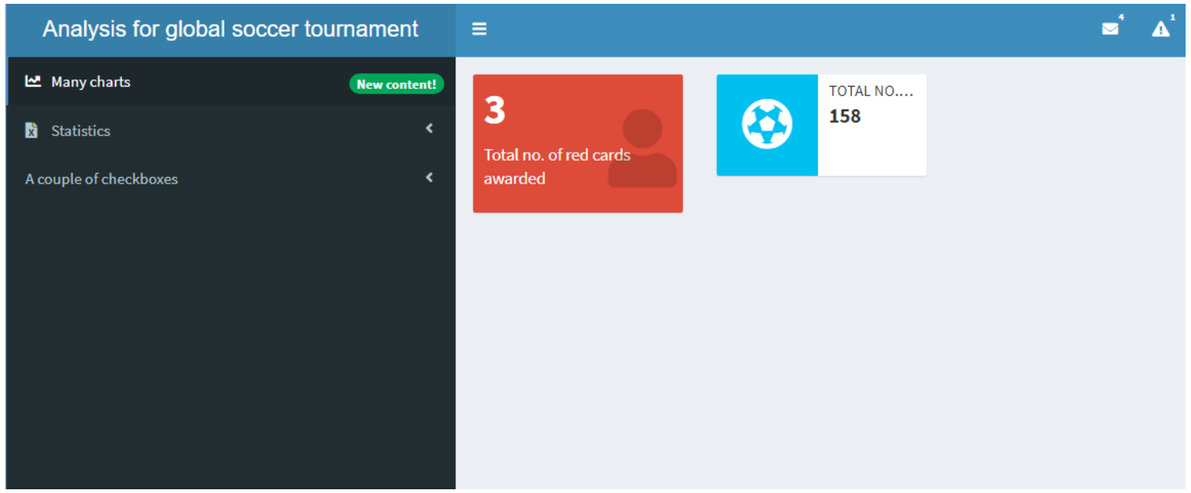
- There are also boxes that display small amounts of information
  - `valueBox()`
  - `infoBox()`




```
body <- dashboardBody(
  fluidRow(valueBox(value = 3,
                    subtitle = "Total no. of red cards awarded",
                    icon = icon("user"),
                    color = "red"),
           infoBox(value = 158,
                   title = "Total no. of goals scored",
                   icon = icon("futbol"))
           ))
```

valueBoxOutput and infoBoxOutput

- Variations of `valueBox()` and `infoBox()`: `valueBoxOutput()` and `infoBoxOutput()`
- These are output functions
- There are also `renderValueBox()` and `renderInfoBox()`
- `valueBox()` and `infoBox()` are needed




```
body <- dashboardBody(
  fluidRow(valueBoxOutput(outputId = "valuebox1"),
           infoBoxOutput(outputId = "infobox1")
           ))
```

```
server <- function(input, output){
  output$valuebox1 <- renderValueBox(
    valueBox(3, "Total no. of red cards awarded",
                icon = icon("user"), color = "red"))
  output$infobox1 <- renderInfoBox(
    infoBox(158,
                    title = "Total no. of goals scored",
                    icon = icon("futbol")))
}
```

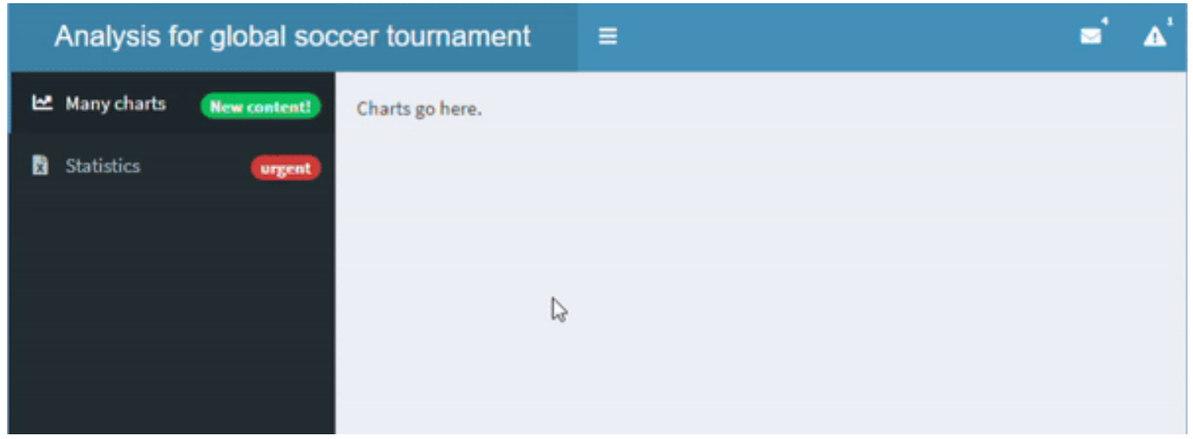
Analogy between valueBoxOutput and plotOutput

- `plotOutput()` :
  - Placed in the UI
  - Determines its **position** in the shinydashboard
- `renderPlot()` :
  - Placed in `server()`
  - Contains a `ggplot()` object

- `valueBoxOutput()` :
  - Placed in the UI
  - Determines its **position** in the shinydashboard
- `renderValueBox()` :
  - Placed in `server()`
  - Contains a `valueBox()` object

Linking the sidebar and body

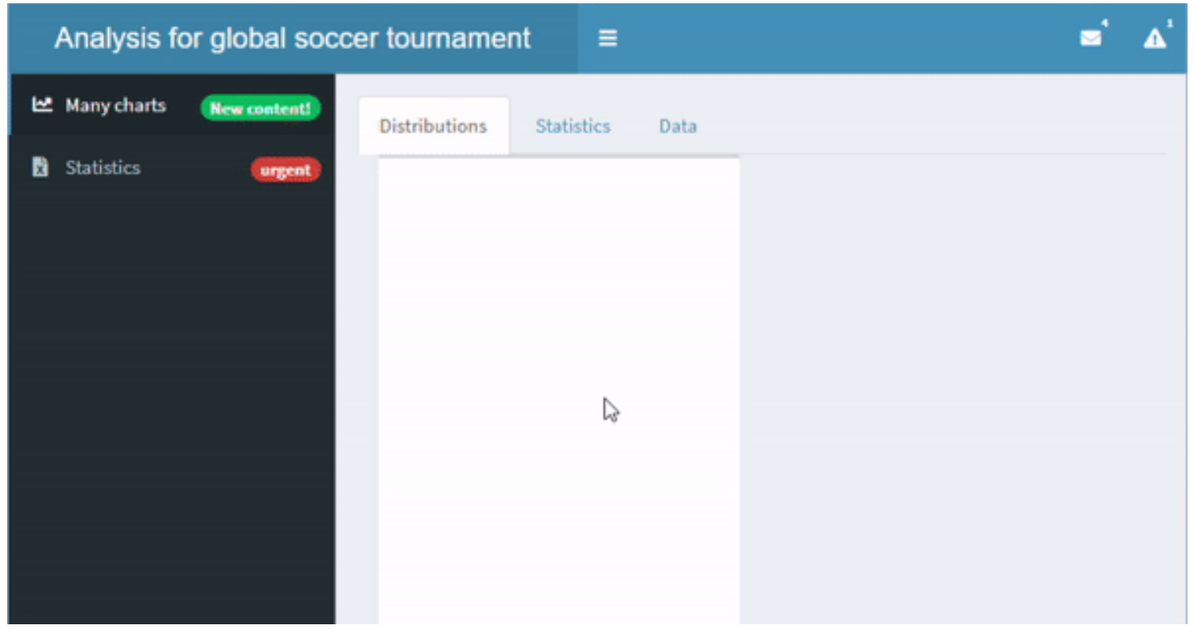
- Recall earlier when we had a sidebar with two buttons called "charts" and "statistics"
- To link these to the body, we will need to add `tabItems()`
  - Each page is defined by a `tabItem()`
  - The labels on each `menuItem()` - `tabItem()` pair must match




```
sidebar <- dashboardSidebar(
  sidebarMenu(
    id = "pages",
    menuItem("Many charts", tabName = "charts",
             icon = icon("chart-line"),
             badgeLabel = "New content!", badgeColor = "green"),
    menuItem("Statistics", icon = icon("file-excel"),
             tabName = "statistics",
             badgeLabel = "urgent", badgeColor = "red")
 ))
body <- dashboardBody(
  tabItems(
    tabItem("charts",
      "Charts go here."
    ),
    tabItem("statistics",
      "Statistics go here."
)))
```

Tabs in the body

- In a shinyApp, a set of tabs can also be defined using `tabsetPanel()`
  - Each tab is defined by `tabPanel()`
- Do not confuse with `tabItems()` and `tabItem()`




```
body <- dashboardBody(tabsetPanel(
    tabPanel("Distributions",
             box(plotOutput("dist"))),
    tabPanel("Statistics",
             dateInput("matchdate",
                       "Enter the match date:",
                       value = "2022-11-20"),
             valueBoxOutput("red"),
             valueBoxOutput("yellow")),
    tabPanel("Data", "Data goes here.",
             tableOutput("data"))
))
server <- function(input, output){
  output$red <- renderValueBox(valueBox(0, "Red cards"))
  output$yellow <- renderValueBox(valueBox(6, "Yellow cards"))
}
```

Let's practice!

BUILDING DASHBOARDS WITH SHINYDASHBOARD

Your first shinydashboard

BUILDING DASHBOARDS WITH SHINYDASHBOARD

Png Kee Seng
Researcher

datacamp

The road so far

- Wireframing in the user interface (UI)
  - Header, side bar, body
- Define outputs and their interactions with user inputs
- The **server** will play this role

```
header <- dashboardHeader(...)
sidebar <- dashboardSidebar(...)
body <- dashboardBody(...)
ui <- dashboardPage(header,
                    sidebar, body)
server <- function(input, output) {
  ...
}
shinyApp(ui, server)
```

The server

- Need to define outputs and inputs
- Specify outputs, and how they interact with user inputs
- Useful to define our own helper functions to reduce clutter and improve readability
  - Place these helper functions at the top of our code

Adding a plot

- Back to the global soccer tournament dashboard
- Let's add a hexbin plot for *team 1*
- We will then need to place this plot in the server
- What if we want the same plot for *team 2*?

```
soccer %>%
  ggplot(aes(x=`1_passes_compeletd`,
             y=`1_goal_prevented`)) +
  geom_hex(bins=10) + theme_classic()
```

```
server <- function(input, output){
  output$plot1 <- renderPlot({
  fifa %>%
    ggplot(aes(x=`1_passes_compeletd`,
               y=`1_goal_prevented`)) +
    geom_hex(bins=10) + theme_classic()
  })
}
```

Defining ggplot objects within server

```
server <- function(input, output) {
  output$plot1 <- renderPlot({
    if (input$team == 1){
      fifa %>% ggplot(aes(x=`1_passes_compeletd`, y=`1_goal_prevented`)) +
        geom_hex(bins=10) + theme_classic()
      }
    else if (input$team == 2){
      fifa %>% ggplot(aes(x=`2_passes_compeletd`, y=`2_goal_prevented`)) +
        geom_hex(bins=10) + theme_classic()
      }
    })
}
```

Using helper functions

```
plot_hex <- function(team){
  if (team == 1){
    soccer %>% ggplot(aes(x=`1_passes_compeletd`,
                          y=`1_goal_prevented`)) +
      geom_hex(bins=10) + theme_classic()
  }
  else if (team == 2){
    soccer %>% ggplot(aes(x=`2_passes_compeletd`,
                          y=`2_goal_prevented`)) +
      geom_hex(bins=10) + theme_classic()
  }
}
```

Using helper functions

With helper function

```
server <- function(input, output) {
  output$plot1 <- renderPlot(plot_hex(input$team))
}
```

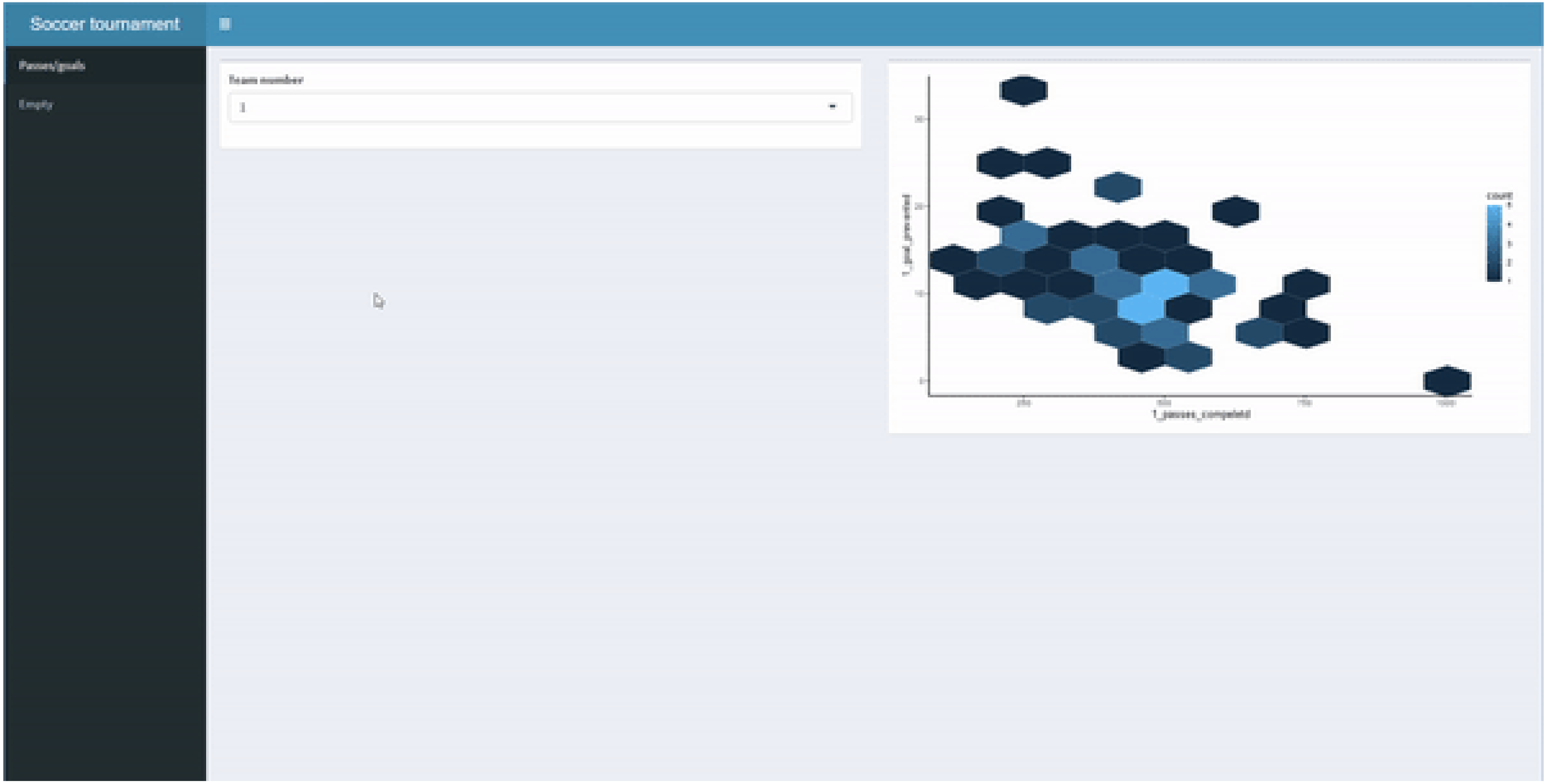
- Isn't this more readable?

Without helper function

```
server <- function(input, output) {
  output$plot1 <- renderPlot({
    if (input$team == 1){
      soccer %>% ggplot(aes(x=`1_passes_compeletd`,
                 y=`1_goal_prevented`)) +
        geom_hex(bins=10) + theme_classic()
      }
    else if (input$team == 2){
      soccer %>% ggplot(aes(x=`2_passes_compeletd`,
                 y=`2_goal_prevented`)) +
        geom_hex(bins=10) + theme_classic()
      }
    })
}
```

Putting it all together

```
# A complete shinydashboard
header <- dashboardHeader(title = "Soccer tournament")
sidebar <- dashboardSidebar(sidebarMenu(menuItem("Passes/goals",
                                                 tabName = "passtab"),
                                        menuItem("Empty",
                                                 tabName = "emptytab")))
body <- dashboardBody(tabItems(
  tabItem("passtab", fluidRow(box(selectInput("team", "Team number",
          choices = c(1,2))), box(plotOutput("plot1"))) ), tabItem("emptytab")) )
ui <- dashboardPage(header, sidebar, body)
server <- function(input, output) {
output$plot1 <- renderPlot(plot_hex(input$team)) }
shinyApp(ui, server)
```


Soccer tournament
Passes/goals
Empty
Team number
1
count
T_goal_prevented
T_passes_completed

Let's practice!

BUILDING DASHBOARDS WITH SHINYDASHBOARD