

PERTEMUAN 7

Visualisasi Menggunakan R Shiny

SUB-CAPAIAN PEMBELAJARAN MATA KULIAH:

Mahasiswa mampu membuat grafik interaktif dengan R Shiny.

POKOK BAHASAN:

1. Menginstall dan memuat R shiny
2. Bagian dari skrip aplikasi Shiny serta kegunaannya masing-masing
3. Membuat aplikasi shiny dari contoh bawaan R
4. Membuat visualisasi interaktif dengan R Shiny

TUJUAN:

Setelah melakukan praktikum, mahasiswa diharapkan untuk dapat memahami komponen dari R Shiny dan kegunaannya, dapat membuat aplikasi web untuk visualisasi interaktif dengan paket shiny, dapat membuat dashboard berupa grafik interaktif dengan R Shiny, serta dapat menginterpretasikan grafik yang ditampilkan.

DASAR TEORI:

1. Visualisasi Interaktif dengan R Shiny

Visualisasi yang telah dibahas pada bab-bab sebelumnya sebagian besar bersifat statis, yaitu berupa gambar yang dapat ditampilkan di layar atau disimpan dalam bentuk file gambar. Untuk membuat visualisasi interaktif, diperlukan pembuatan antarmuka pengguna (*user interface* atau UI) yang menangkap input pengguna. Paket Shiny adalah paket R yang dapat membuat UI interaktif dalam bentuk aplikasi Web interaktif yang dapat berubah berdasarkan input pengguna. Setiap aplikasi Shiny memiliki halaman web yang dikunjungi pengguna dan di belakang halaman web terdapat komputer yang melayani halaman tersebut dengan menjalankan R.

Shiny memiliki kumpulan fitur yang cukup banyak yang dapat digunakan untuk membuat aplikasi web yang kompleks. Keuntungan dari membuat aplikasi web adalah sebagian besar pengguna dapat mengakses visualisasi interaktif melalui browser web mereka tanpa perlu meng-*install* paket perangkat lunak apa pun termasuk R dan mengunduh data. Pembuatan aplikasi web dengan paket Shiny seluruhnya dapat dilakukan dalam R, tanpa

menulis kode HTML atau Javascript. Meskipun demikian, dimungkinkan untuk menentukan UI menggunakan halaman HTML jika diperlukan. Namun, dalam modul ini kita akan melihat pembuatan aplikasi web di R.

2. Aplikasi Shiny dari Contoh Bawaan Paket shiny

Shiny adalah paket R yang memudahkan untuk membangun aplikasi web interaktif langsung dari R. Berikut tutorial membangun aplikasi Shiny.

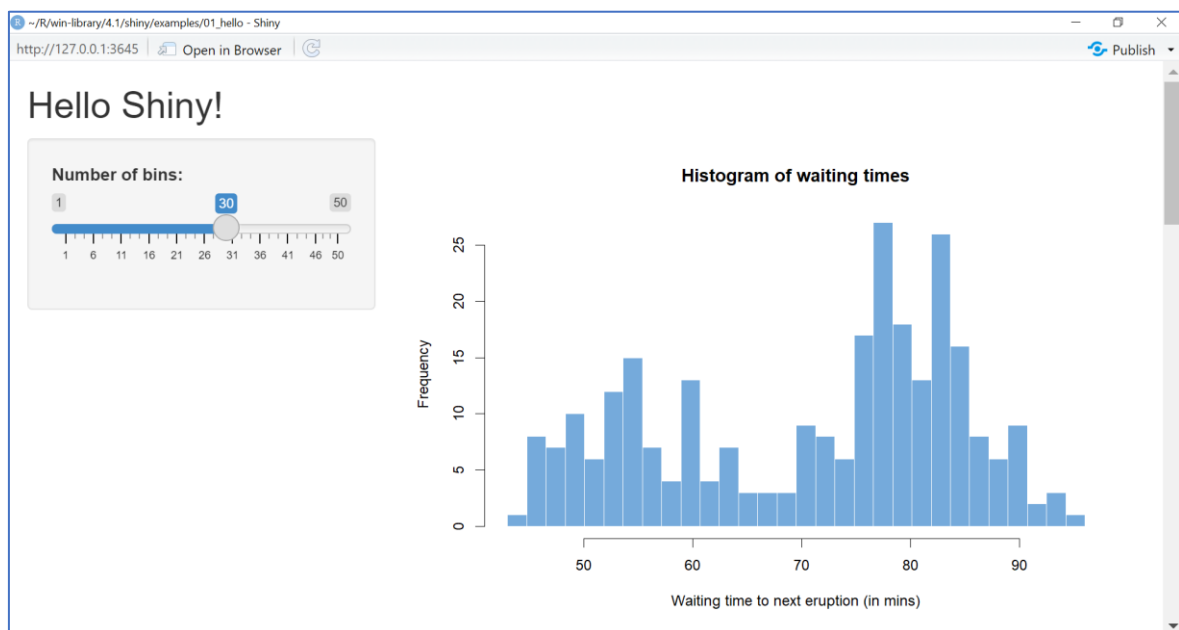
1. Jika Anda masih belum meng-*install* paket Shiny, buka sesi R, sambungkan ke internet, dan jalankan

```
install.packages("shiny")
```

2. Paket shiny memiliki sebelas contoh bawaan yang masing-masing menunjukkan cara kerja Shiny. Untuk kali ini, digunakan contoh **Hello Shiny** untuk membuat plot histogram dari dataset R dengan jumlah bin yang dapat dikonfigurasi. Pengguna dapat mengubah jumlah bin dengan cara menggeser bilah penggeser, lalu aplikasi akan merespon masukan dari pengguna. Untuk menjalankan Hello Shiny, ketik syntax berikut di RGui atau RStudio.

```
library(shiny)
runExample("01_hello")
```

Jika menggunakan RGui, akan ditampilkan langsung ke web browser yang sedang dibuka. Jika menggunakan RStudio, plot akan muncul di jendela baru dari RStudio seperti dalam Gambar 1. Dengan menekan tab *Open in Browser*, jendela Hello Shiny akan muncul di web browser.



Gambar 1 Aplikasi Hello Shiny

Aplikasi Shiny terdapat dalam satu skrip yang disebut **app.R**. Skrip app.R berada di suatu direktori, misalnya direktori `newdir/` dan aplikasi dapat dijalankan dengan `runApp("newdir")`. Skrip app.R terdiri dari tiga komponen, yaitu:

- objek antarmuka pengguna (UI)**, berfungsi untuk mengontrol tata letak dan tampilan aplikasi yang akan dilihat pengguna di aplikasi web serta menjabarkan input atau masukan dari pengguna.
- fungsi server**, berisi instruksi yang dibutuhkan komputer untuk membangun aplikasi, misalnya logika aplikasi yang menghasilkan plot.
- panggilan ke fungsi shinyApp**, fungsi `shinyApp` membuat objek aplikasi Shiny dari pasangan UI/server eksplisit.

Sebelum versi 0.10.2, Shiny tidak mendukung aplikasi file tunggal dan objek UI serta fungsi server harus dimuat dalam skrip terpisah yang disebut **ui.R** dan **server.R**. Fungsionalitas dengan skrip terpisah tersebut masih didukung di Shiny, tetapi tutorial dan sebagian besar dokumentasi pendukung berfokus pada aplikasi file tunggal.

Berikut skrip app.R dari Hello Shiny diawali dengan memuat paket shiny.

```
library(shiny)

# Define UI for app that draws a histogram ----
ui <- fluidPage(

  # App title ----
  titlePanel("Hello Shiny!"),

  # Sidebar layout with input and output definitions ----
  sidebarLayout(

    # Sidebar panel for inputs ----
    sidebarPanel(

      # Input: Slider for the number of bins ----
      sliderInput(inputId = "bins",
                  label = "Number of bins:",
                  min = 1,
                  max = 50,
                  value = 30)

    ),

    # Main panel for displaying outputs ----
    mainPanel(

      # Output: Histogram ----
      plotOutput(outputId = "distPlot")

    )
  )
)
```

UI

```
# Define server logic required to draw a histogram ----
server <- function(input, output) {
```

```
  # Histogram of the Old Faithful Geyser Data ----
  # with requested number of bins
  # This expression that generates a histogram is wrapped in a call
  # to renderPlot to indicate that:
  #
  # 1. It is "reactive" and therefore should be automatically
  #    re-executed when inputs (input$bins) change
  # 2. Its output type is a plot
  output$distPlot <- renderPlot({
```

```
    x <- faithful$waiting
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

    hist(x, breaks = bins, col = "#75AADB", border = "white",
         xlab = "Waiting time to next eruption (in mins)",
         main = "Histogram of waiting times")
```

```
  })
```

```
}
```

```
# Create Shiny app ----
```

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

**Fungsi
Server**

**Fungsi
shinyApp**

Kode sumber di atas terdiri dari 3 komponen, yaitu UI, fungsi server, dan panggilan ke fungsi shinyApp. Bagian UI memuat `fluidPage()` untuk mengontrol tata letak dan tampilan dari aplikasi. Fungsi `slidebar()` untuk menampilkan batang penggeser sebagai input bagi pengguna untuk mengatur sendiri banyaknya bin dari histogram sehingga visualisasi bersifat interaktif. Pada fungsi server Hello Shiny, script melakukan beberapa perhitungan dan kemudian memplot histogram sesuai dengan jumlah bin yang diminta pengguna (diatur di bagian UI). Sebagian besar skrip fungsi server dibungkus dengan panggilan ke `renderPlot`. Komentar di atas fungsi menjelaskan sedikit tentang ini. Baris terakhir `app.R` memuat fungsi `shinyApp()` untuk memanggil pasangan UI dan server agar dihasilkan aplikasi Shiny. Setelah semua instruksi dijalankan di RGui/ RStudio, aplikasi web muncul seperti dalam Gambar 10.1.

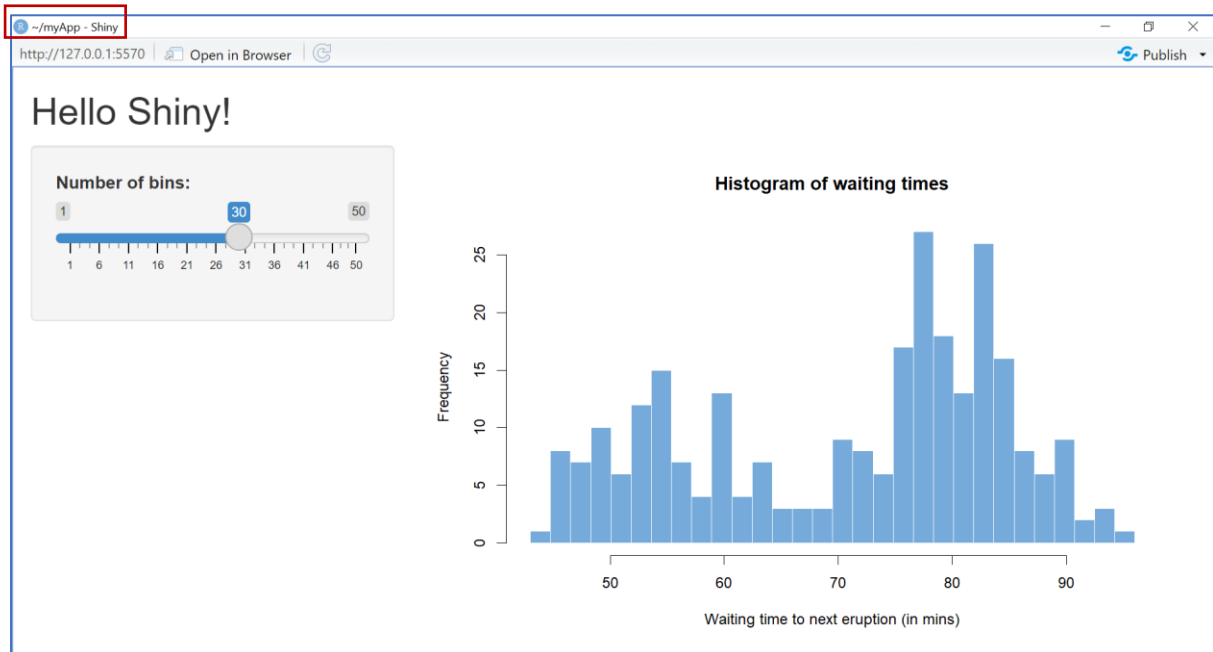
Saat aplikasi Hello Shiny aktif, sesi R akan sibuk sehingga Anda tidak akan dapat menjalankan perintah R apapun. Pada saat itu, R memantau aplikasi dan menjalankan reaksi aplikasi, misalnya pengguna mengubah banyaknya bin sehingga R akan menjalankan perintah untuk mengubah tampilan histogram. Untuk mengaktifkan kembali sesi R, tekan escape atau klik ikon STOP atau tanda berhenti yang berada di sudut kanan atas panel konsol RStudio.

Aplikasi Shiny juga dapat dibangun dengan membuat direktori baru dan menyimpan file `app.R` di dalamnya. Setiap aplikasi disarankan agar hidup di direktori uniknya sendiri. Aplikasi

Shiny dapat dijalankan dengan memberikan nama direktorinya ke fungsi `runApp()`. Misalkan kode dari aplikasi Hello Shiny disimpan di direktori `~/myApp` atau `myApp/app.R`, maka jalankan kode berikut:

```
library(shiny)
runApp("~/myApp") #atau runApp("myApp/app.R")
```

Setelah kode di atas berhasil dijalankan, muncul seperti Gambar 2. Pada sisi kiri atas, terlihat nama direktori aplikasi Shiny berada.

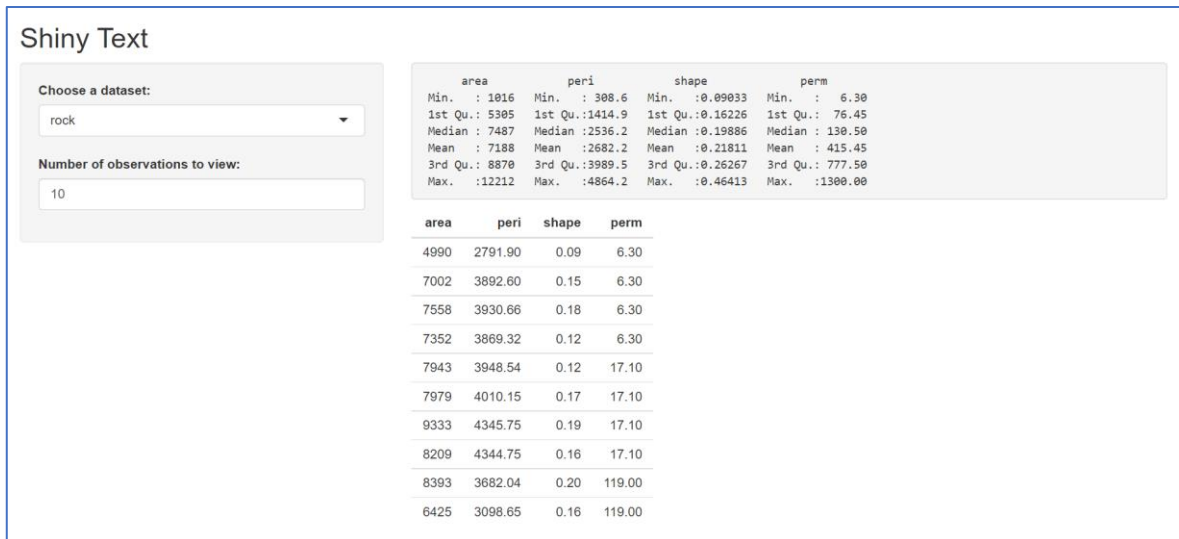


Gambar 2 Aplikasi Hello Shiny di Direktori `~/myApp`

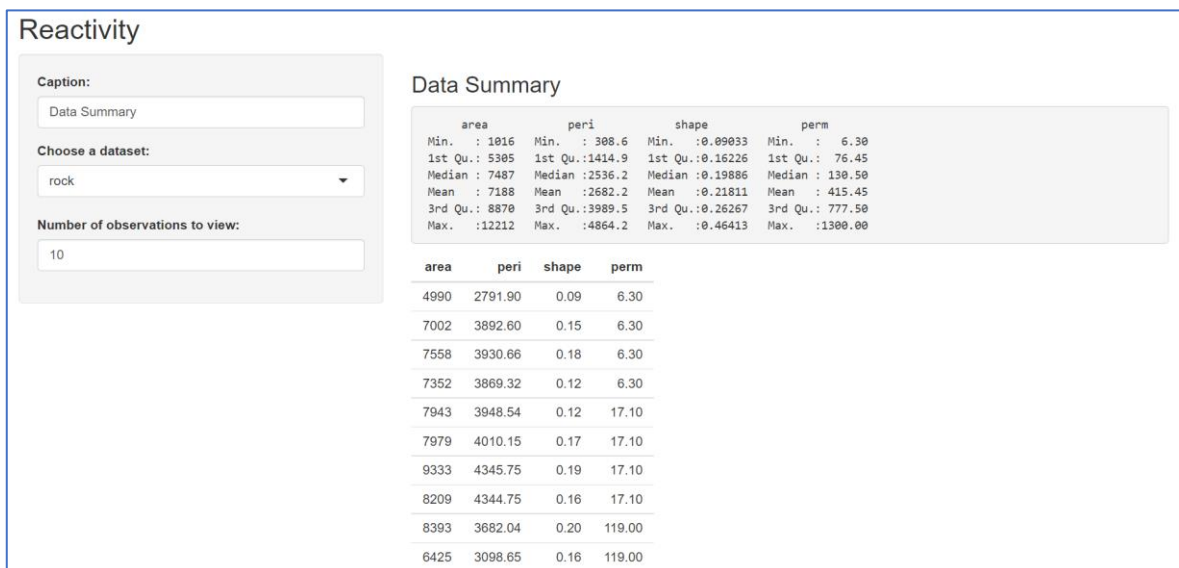
Aplikasi yang sudah ada di paket shiny selain Hello Shiny, diantaranya:

```
runExample("02_text")           # tables and data frames
runExample("03_reactivity")      # a reactive expression
runExample("04_mpg")             # global variables
runExample("05_sliders")         # slider bars
runExample("06_tabsets")         # tabbed panels
runExample("07_widgets")         # help text and submit buttons
runExample("08_html")            # Shiny app built from HTML
runExample("09_upload")          # file upload wizard
runExample("10_download")        # file download wizard
runExample("11_timer")           # an automated timer
```

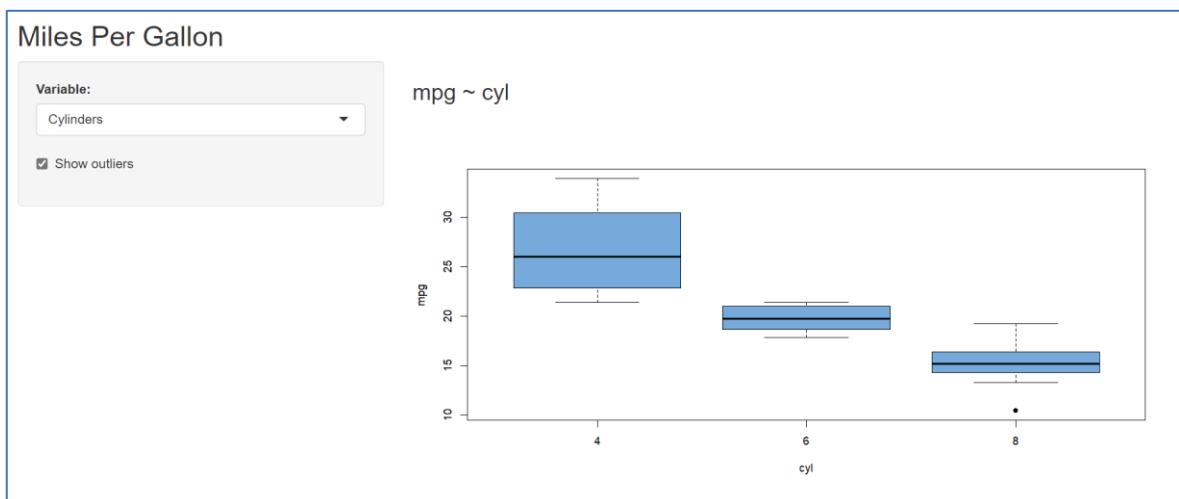
Jika semua kode diatas dijalankan secara langsung, aplikasi contoh Shiny terbuka satu per satu sekaligus dengan `app.R` di dalam layar. Aplikasi yang pertama kali muncul adalah Shiny Text, lalu setelah ditutup, muncul aplikasi Reactivity, begitu seterusnya hingga contoh aplikasi ke-11. Anda dapat memodifikasi dari aplikasi yang ada sebelumnya.



Gambar 3 Tampilan Aplikasi dari runExample("02_text")

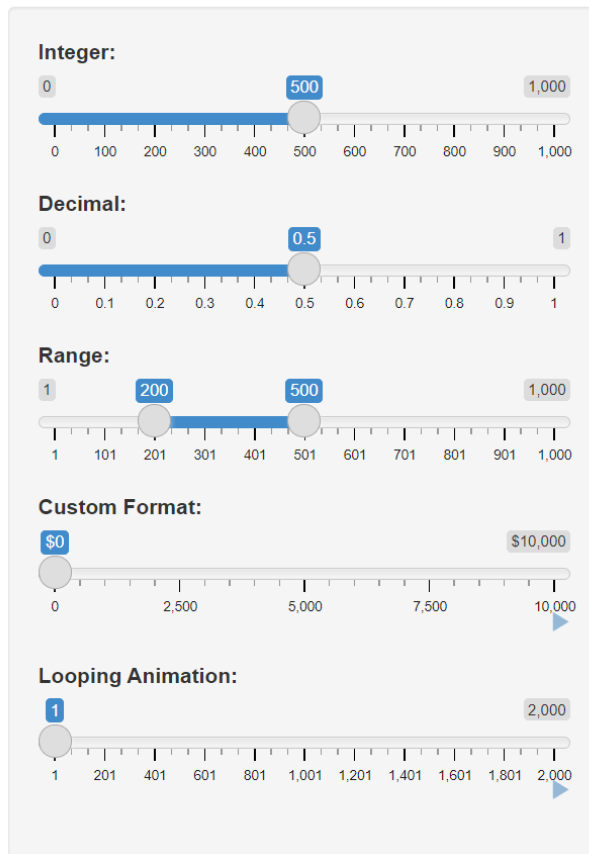


Gambar 4 Tampilan Aplikasi dari runExample("03_reactivity")



Gambar 5 Tampilan Aplikasi dari runExample("04_mpg")

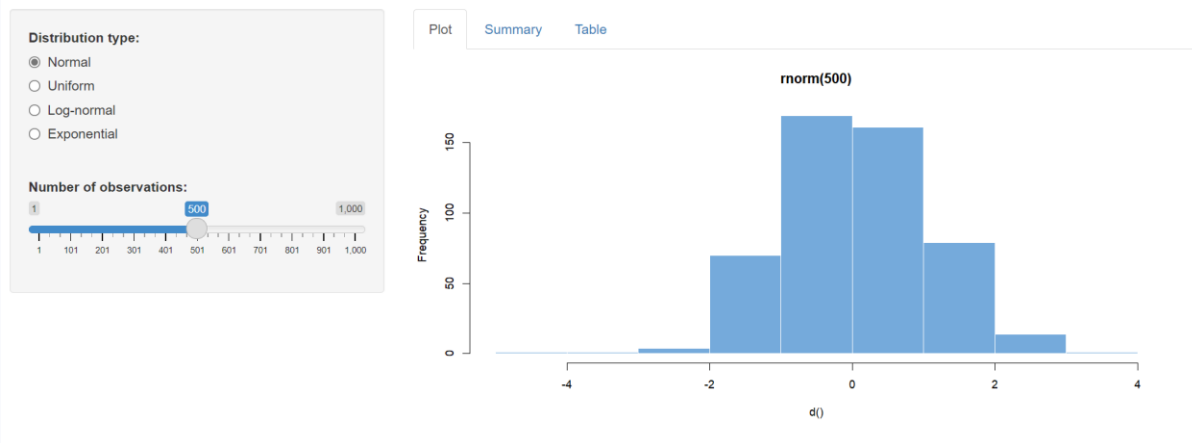
Sliders



Name	Value
Integer	500
Decimal	0.5
Range	200 500
Custom Format	0
Animation	1

Gambar 6 Tampilan Aplikasi dari `runExample("05_sliders")`

Tabsets



Gambar 7 Tampilan Aplikasi dari `runExample("06_tabsets")`

More Widgets

Choose a dataset:

rock

Number of observations to view:

10

Note: while the data view will show only the specified number of observations, the summary will still be based on the full dataset.

Update View

Summary

area	peri	shape	perm
Min. : 1016	Min. : 308.6	Min. : 0.09033	Min. : 6.30
1st Qu.: 5305	1st Qu.: 1414.9	1st Qu.: 0.16226	1st Qu.: 76.45
Median : 7487	Median : 2536.2	Median : 0.19886	Median : 130.50
Mean : 7188	Mean : 2682.2	Mean : 0.21811	Mean : 415.45
3rd Qu.: 8870	3rd Qu.: 3989.5	3rd Qu.: 0.26267	3rd Qu.: 777.50
Max. : 12212	Max. : 4864.2	Max. : 0.46413	Max. : 1300.00

Observations

area	peri	shape	perm
4990	2791.90	0.09	6.30
7002	3892.60	0.15	6.30
7558	3930.66	0.18	6.30
7352	3869.32	0.12	6.30
7943	3948.54	0.12	17.10
7979	4010.15	0.17	17.10
9333	4345.75	0.19	17.10
8209	4344.75	0.16	17.10
8393	3682.04	0.20	119.00
6425	3098.65	0.16	119.00

Gambar 8 Tampilan Aplikasi dari runExample("07_widgets")

HTML UI

Distribution type:

Normal

Number of observations:

500

Summary of data:

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
-2.903607	-0.638970	0.006116	0.023787	0.706622	3.018497

Plot of data:

Head of data:

x
-0.16
-0.69
0.45
-0.52
-0.10
-2.08

Gambar 9 Tampilan Aplikasi dari runExample("08_html")

Uploading Files

Choose CSV File

Browse...

No file selected

☒ Header

Separator

☒ Comma

☐ Semicolon

☐ Tab

Quote

☐ None

☒ Double Quote

☐ Single Quote

Display

☒ Head

☐ All


Gambar 10 Tampilan Aplikasi dari `runExample("09_upload")`

Downloading Data

Choose a dataset:

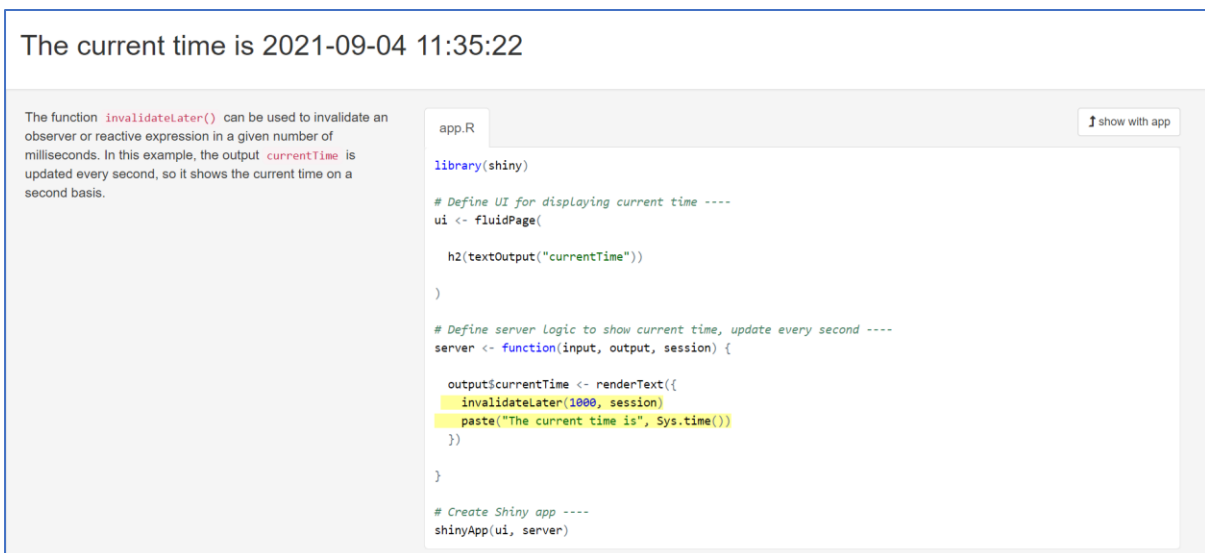
rock

▼

 Download

area	peri	shape	perm
4990	2791.90	0.09	6.30
7002	3892.60	0.15	6.30
7558	3930.66	0.18	6.30
7352	3869.32	0.12	6.30
7943	3948.54	0.12	17.10
7979	4010.15	0.17	17.10
9333	4345.75	0.19	17.10
8209	4344.75	0.16	17.10
8393	3682.04	0.20	119.00
6425	3098.65	0.16	119.00
9364	4480.05	0.15	119.00
8624	3986.24	0.15	119.00
10651	4036.54	0.23	82.40
8868	3518.04	0.23	82.40
9417	3999.37	0.17	82.40
8874	3629.07	0.15	82.40
10962	4608.66	0.20	58.60
10743	4787.62	0.26	58.60

Gambar 11 Tampilan Aplikasi dari `runExample("10_download")`



Gambar 12 Tampilan Aplikasi dari `runExample("11_timer")`

3. Membuat Aplikasi Shiny

Subbab ini menjelaskan cara membuat aplikasi Shiny Anda sendiri dari awal. Pada bagian awal, Anda mempelajari bagian-bagian dari aplikasi Shiny dan menyelesaikannya dengan menerapkan aplikasi Shiny Anda sendiri secara online. Selanjutnya, Anda mempelajari cara membangun tata letak dan tampilan aplikasi Shiny.

Berikut ini adalah kode aplikasi Shiny yang terdiri dari UI, fungsi server, dan pemanggilan ke fungsi `shinyApp()` seperti halnya contoh pada subbab 2.

```
library(shiny)

# Define UI ----
ui <- fluidPage(

)

# Define server logic ----
server <- function(input, output) {

}

# Run the app ----
shinyApp(ui = ui, server = server)
```

Setelah menjalankan kode di atas, didapatkan halaman kosong. Hal ini karena di dalam UI, belum diatur tata letak dan tampilan aplikasinya. Anda dapat menambahkan argumen atau fungsi di dalam `fluidPage()` untuk mengatur tata letak (*layout*) dan tampilan di halaman web, misalnya dengan kode sebagai berikut.

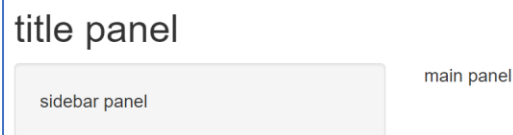
```

ui <- fluidPage(
  titlePanel("title panel"),

  sidebarLayout(
    sidebarPanel("sidebar panel"),
    mainPanel("main panel")
  )
)

```

Setelah menjalankan kode aplikasi yang sudah ditambahkan argumen untuk mengatur tata letak dan tampilan aplikasi web, didapatkan hasil seperti dalam Gambar 13.



Gambar 13 Tampilan Aplikasi Shiny Sendiri Setelah Pengaturan Tata Letak

Aplikasi Shiny juga dapat menampilkan widget untuk menambah tampilan yang ada di halaman web atau antarmuka pengguna. Tabel 1 menunjukkan fungsi yang digunakan untuk menyajikan widget.

Tabel 1 Fungsi untuk menampilkan widget

No	Fungsi	Widget
1	actionBotton	Tombol aksi
2	checkboxGroupInput	Sekelompok kotak centang
3	checkboxInput	Satu kotak centang
4	dateInput	Kalender untuk membantu pemilihan tanggal
5	dateRangeInput	Sepasang kalender untuk memilih rentang tanggal
6	fileInput	Control wizard unggah file
7	helpText	Teks bantuan yang dapat ditambahkan ke formulir input
8	numericInput	Bidang untuk memasukkan angka
9	radioButtons	Satu set tombol radio
10	selectInput	Sebuah kotak dengan pilihan untuk dipilih
11	sliderInput	Bilah penggeser
12	submitButton	Tombol kirim
13	textInput	Bidang untuk memasukkan teks

Kode sebelumnya dapat ditambahkan dengan fungsi dalam Tabel 1 untuk menampilkan widget pada aplikasi Shiny yang Anda buat. Jalankan kode berikut ini.

```
library(shiny)

# Define UI ----
ui <- fluidPage(
  titlePanel("Basic widgets"),

  fluidRow(

    column(3,
      h3("Buttons"),
      actionButton("action", "Action"),
      br(),
      br(),
      submitButton("Submit")),

    column(3,
      h3("Single checkbox"),
      checkboxInput("checkbox", "Choice A", value = TRUE)),

    column(3,
      checkboxGroupInput("checkGroup",
        h3("Checkbox group"),
        choices = list("Choice 1" = 1,
                       "Choice 2" = 2,
                       "Choice 3" = 3),
        selected = 1)),

    column(3,
      dateInput("date",
        h3("Date input"),
        value = "2014-01-01"))
  ),

  fluidRow(

    column(3,
      dateRangeInput("dates", h3("Date range"))),

    column(3,
      fileInput("file", h3("File input"))),

    column(3,
      h3("Help text"),
      helpText("Note: help text isn't a true widget,",
               "but it provides an easy way to add text to",
               "accompany other widgets.")),

    column(3,
      numericInput("num",
        h3("Numeric input"),
        value = 1))
  ),
)
```

```

fluidRow(
  column(3,
    radioButtons("radio", h3("Radio buttons"),
      choices = list("Choice 1" = 1, "Choice 2" = 2,
        "Choice 3" = 3), selected = 1)),

    column(3,
      selectInput("select", h3("Select box"),
        choices = list("Choice 1" = 1, "Choice 2" = 2,
          "Choice 3" = 3), selected = 1)),

    column(3,
      sliderInput("slider1", h3("Sliders"),
        min = 0, max = 100, value = 50),
      sliderInput("slider2", "",
        min = 0, max = 100, value = c(25, 75))
    ),

    column(3,
      textInput("text", h3("Text input"),
        value = "Enter text..."))
  )
)

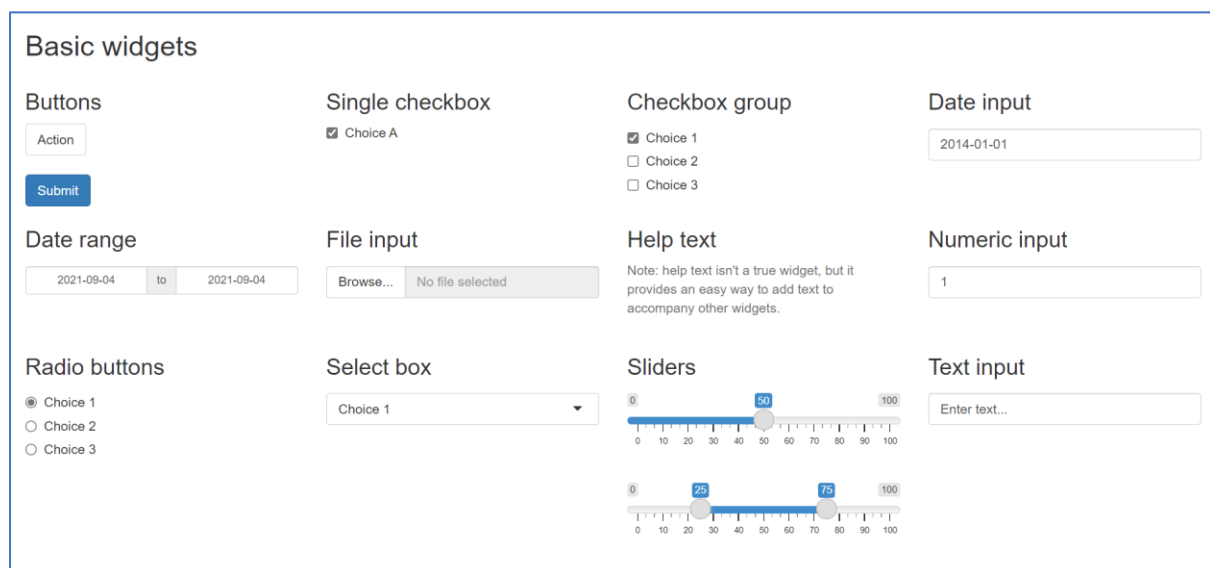
# Define server logic ----
server <- function(input, output) {

}

# Run the app ----
shinyApp(ui = ui, server = server)

```

Aplikasi Shiny setelah ditambahkan semua fungsi untuk menampilkan widget yang ada dalam Tabel 1 ditunjukkan oleh Gambar 14.



Gambar 14 Tampilan Aplikasi Shiny Sendiri Setelah Penambahan Widget

Aplikasi Shiny juga dapat dibangun agar memiliki tampilan output yang reaktif dari visualisasi hasil sensus. Langkah pertama adalah menambahkan objek R ke UI. Di bagian ui, ditambahkan fungsi `textOutput()` untuk membuat teks.

```
library(shiny)

# Define UI ----
ui <- fluidPage(
  titlePanel("censusVis"),

  sidebarLayout(
    sidebarPanel(
      helpText("Create demographic maps with
        information from the 2010 US Census."),

      selectInput("var",
        label = "Choose a variable to display",
        choices = c("Percent White",
                    "Percent Black",
                    "Percent Hispanic",
                    "Percent Asian"),
        selected = "Percent White"),

      sliderInput("range",
        label = "Range of interest:",
        min = 0, max = 100, value = c(0, 100))
    ),

    mainPanel(
      textOutput("selected_var")
    )
  )
)

# Define server logic ----
server <- function(input, output) {

}

# Run the app ----
shinyApp(ui = ui, server = server)
```

Fungsi output lainnya terdapat pada Tabel 2.

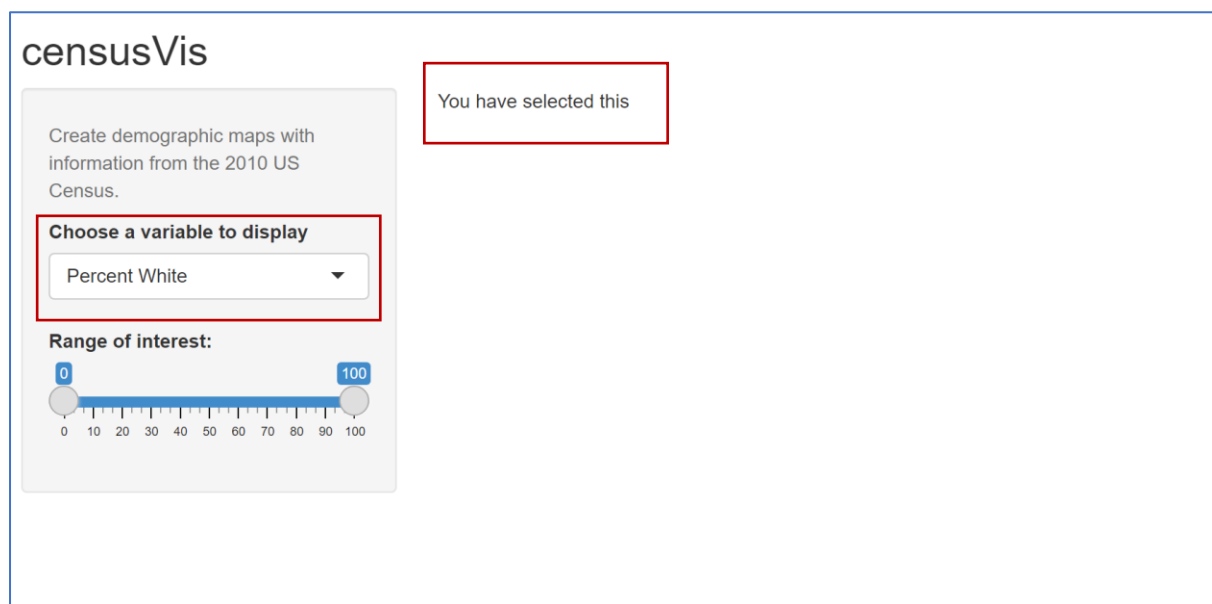
Tabel 2 Fungsi Output

Fungsi Output	Membuat
<code>dataTableOutput</code>	Tabel Data
<code>htmlOutput</code>	HTML mentah
<code>imageOutput</code>	Gambar
<code>plotOutput</code>	Plot
<code>tableOutput</code>	Table
<code>textOutput</code>	Text
<code>uiOutput</code>	HTML mentah

Langkah kedua untuk menampilkan output reaktif adalah menyediakan kode R untuk membangun objek. Dari kode sebelumnya, ditambahkan fungsi render di dalam komponen fungsi server sebagai berikut.

```
server <- function(input, output) {  
  output$selected_var <- renderText({  
    "You have selected this"  
  })  
}
```

Tampilan Aplikasi Shiny setelah ditambahkan fungsi render seperti dalam Gambar 15. Saat tanda panah ke bawah (tombol scroll) diklik dan dipilih variabel lainnya, pada bagian output tetap muncul teks "You have selected this".



Gambar 15 Tampilan Output Reaktif dengan Fungsi render

Fungsi render yang lainnya ada dalam Tabel 3.

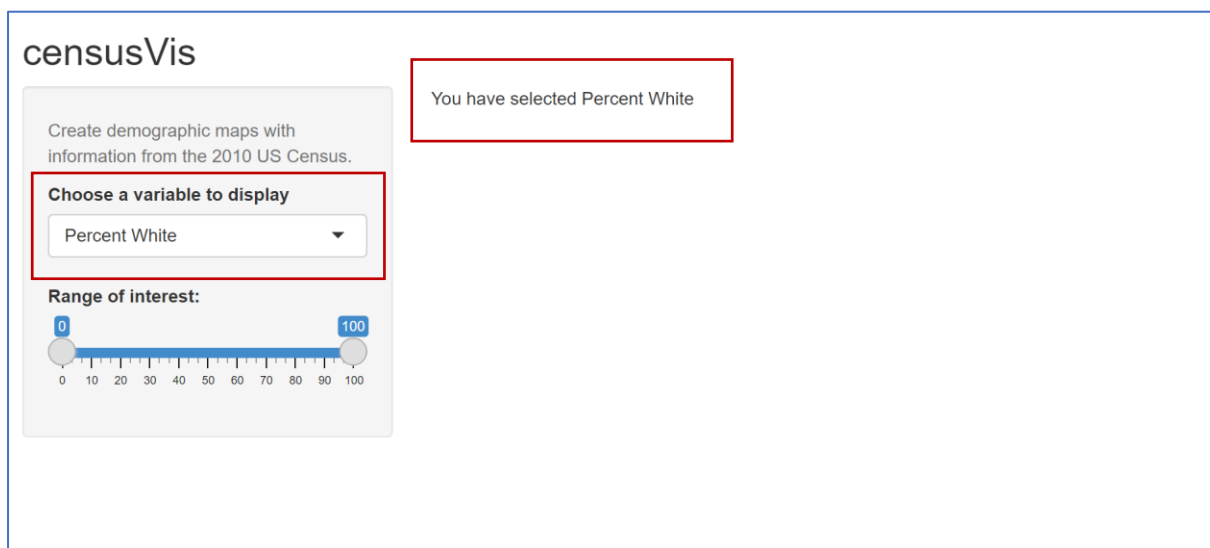
Tabel 3 Fungsi render

Fungsi Output	Membuat
renderDataTable	Tabel data
renderImage	gambar (disimpan sebagai tautan ke file sumber)
renderPlot	Plot
renderPrint	hasil cetak apa pun
renderTable	bingkai data, matriks, tabel lain seperti struktur
renderText	string karakter
renderUI	objek tag Shiny atau HTML

Kode pada komponen fungsi server dapat dimodifikasi dengan menggunakan nilai widget sehingga output akan menampilkan sesuai dengan variabel yang dipilih pada tombol scroll.

```
server <- function(input, output) {  
  output$selected_var <- renderText({  
    paste("You have selected this", input$var)  
  })  
}
```

Berdasarkan Gambar 16, output bersifat reaktif sesuai dengan variabel yang dipilih melalui tombol scroll.



Gambar 16 Tampilan Output Reaktif dengan Fungsi render dengan paste

Kode yang telah dibuat untuk membangun aplikasi Shiny, disimpan dalam direktori 'censusVis/census.R' lalu ditampilkan dengan mode showcase sebagai berikut:

```
runApp('censusVis/census.R', display.mode = "showcase")
```

Tampilan Aplikasi Shiny sama dengan Gambar 15, tetapi pada halaman web juga muncul skrip berisi kode dari aplikasi shiny.

Aplikasi Shiny yang telah Anda buat, dapat dibagikan kepada pengguna, secara umum terdapat 2 cara berbagi. Pertama, membagikan aplikasi Shiny sebagai skrip R. Cara ini adalah cara paling sederhana untuk berbagi aplikasi, tetapi hanya berfungsi jika pengguna memiliki R di komputer mereka sendiri serta mengerti cara menggunakannya. Kedua, membagikan aplikasi Shiny Anda sebagai halaman web. Pengguna Anda dapat menavigasi ke aplikasi Anda melalui internet dengan browser web. Mereka akan menemukan aplikasi Anda dirender sepenuhnya, terbaru, dan siap digunakan. Semua metode di atas memiliki batasan yang sama.

Mereka mengharuskan pengguna Anda untuk meng-*install* R dan Shiny di komputer mereka. Namun, hal tersebut dapat diatasi untuk pengguna yang tidak memiliki R dan tidak berniat menggunakannya, yaitu dengan cara meng-*host* aplikasi Shiny Anda sebagai halaman web diantaranya melalui shinyapps.io, Shiny Server, dan RStudio Connect.

4. Membuat Dashboard dengan R

Dashboard juga merupakan cara visualisasi interaktif. Dengan menggunakan R, dashboard dibuat dengan paket `shinydashboard` sehingga perlu menginstall paket di R.

```
install.packages("shinydashboard")
```

Setelah terinstall, langkah selanjutnya adalah memanggil paket dan menuliskan kode aplikasi Shiny sebagai berikut.

```
library(shiny)
library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = "Basic dashboard"),
  dashboardSidebar(),
  dashboardBody(
    # Boxes need to be put in a row (or column)
    fluidRow(
      box(plotOutput("plot1", height = 250)),

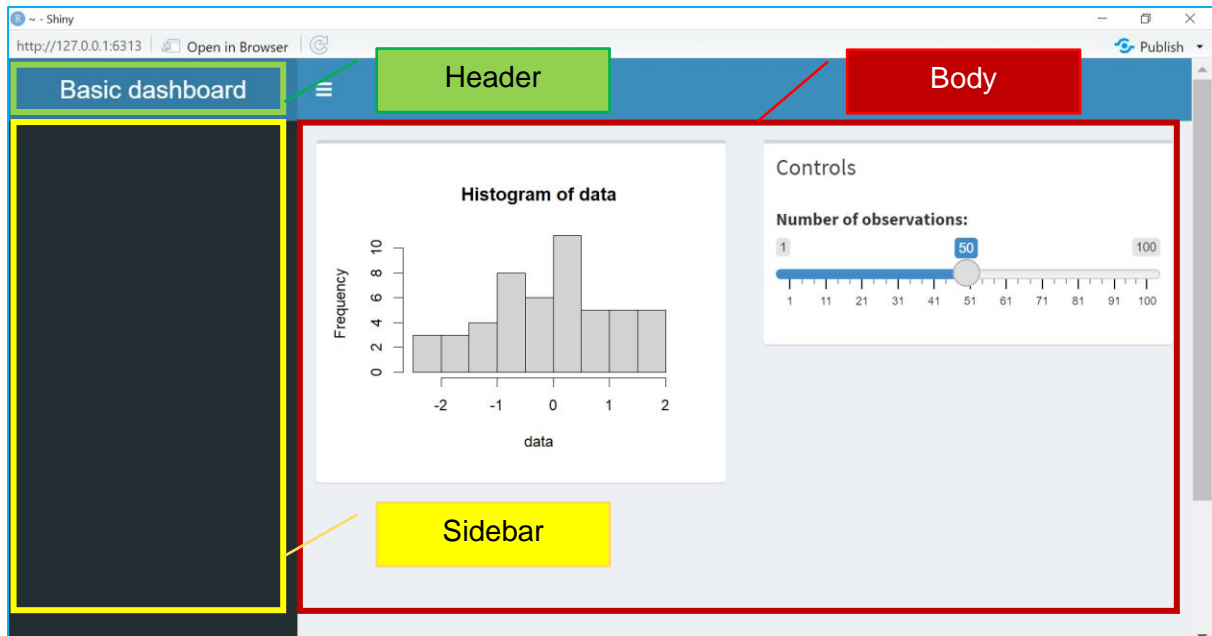
      box(
        title = "Controls",
        sliderInput("slider", "Number of observations:", 1, 100, 50)
      )
    )
  )
)

server <- function(input, output) {
  set.seed(122)
  histdata <- rnorm(500)

  output$plot1 <- renderPlot({
    data <- histdata[seq_len(input$slider)]
    hist(data)
  })
}

shinyApp(ui, server)
```

Dashboard terdiri dari 3 bagian, yaitu *header*, *sidebar*, dan *body* yang dituliskan di dalam fungsi `dashboardPage()` yang ada di komponen UI. Tampilan dashboard dari kode di atas terdapat dalam Gambar 17.



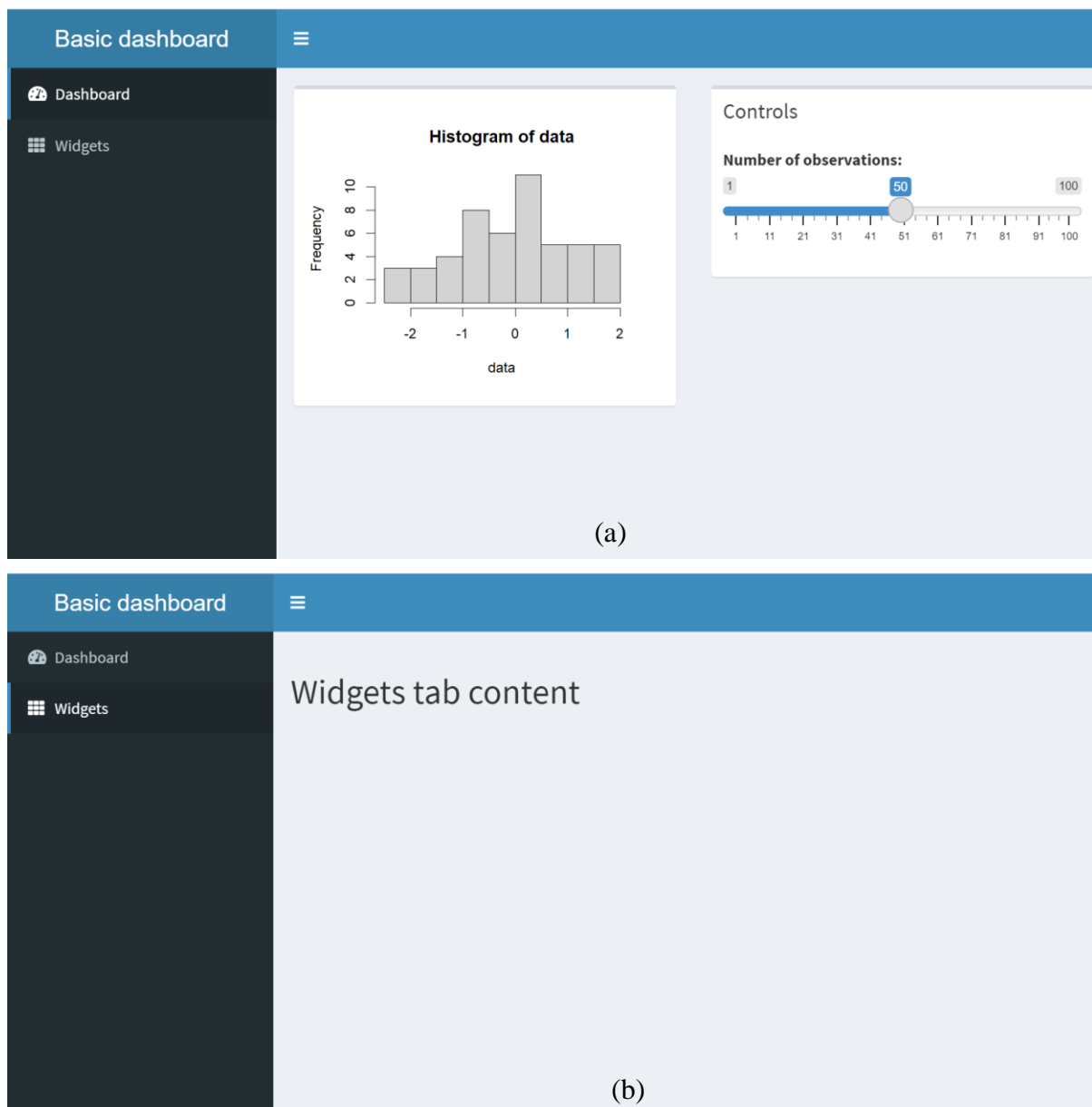
Gambar 17 Basic Dashboard

Selanjutnya, menambahkan konten ke *sidebar* dengan cara menambahkan item menu yang berperilaku seperti tab. Anda perlu menambahkan `menuItems` ke sidebar dengan `tabNames` yang sesuai. Di bagian *body*, tambahkan `tabItems` dengan nilai yang sesuai untuk `tabNames`.

```
ui <- dashboardPage(
  dashboardHeader(title = "Basic dashboard"),
  ## Sidebar content
  dashboardSidebar(
    sidebarMenu(
      menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),
      menuItem("Widgets", tabName = "widgets", icon = icon("th"))
    )
  ),
  ## Body content
  dashboardBody(
    tabItems(
      # First tab content
      tabItem(tabName = "dashboard",
        fluidRow(
          box(plotOutput("plot1", height = 250)),

          box(
            title = "Controls",
            sliderInput("slider", "Number of observations:", 1, 100, 50)
          )
        ),
      ),
      # Second tab content
      tabItem(tabName = "widgets",
        h2("Widgets tab content")
      )
    )
  )
)
```

Tampilan dashboard setelah menambahkan konten sidebar dan item di body, didapatkan hasil seperti Gambar 18.



Gambar 18 Basic Dashboard dengan konten sidebar (a) tab Dashboard (b) tab Widgets

Jika Anda tidak ingin menampilkan sidebar, Anda menuliskan fungsi `dashboardSidebar()` sebagai berikut:

```
dashboardSidebar(disable = TRUE)
```

PERCOBAAN PRAKTIKUM:

1. Menjalankan kode berikut di RGui atau RStudio untuk menampilkan visualisasi interaktif / dashboard dari kode sebagai berikut.

```

library(shiny)

library(shinydashboard)

ui <- dashboardPage(
  dashboardHeader(title = "Info boxes"),
  dashboardSidebar(disable=TRUE),
  dashboardBody(
    # infoBoxes with fill=FALSE
    fluidRow(
      # A static infoBox
      infoBox("New Orders", 10 * 2, icon = icon("credit-card")),
      # Dynamic infoBoxes
      infoBoxOutput("progressBox"),
      infoBoxOutput("approvalBox")
    ),

    # infoBoxes with fill=TRUE
    fluidRow(
      infoBox("New Orders", 10 * 2, icon = icon("credit-card"), fill = TRUE),
      infoBoxOutput("progressBox2"),
      infoBoxOutput("approvalBox2")
    ),

    fluidRow(
      # Clicking this will increment the progress amount
      box(width = 4, actionButton("count", "Increment progress"))
    )
  )
)

server <- function(input, output) {
  output$progressBox <- renderInfoBox({
    infoBox(
      "Progress", paste0(25 + input$count, "%"), icon = icon("list"),
      color = "purple"
    )
  })
  output$approvalBox <- renderInfoBox({
    infoBox(
      "Approval", "80%", icon = icon("thumbs-up", lib = "glyphicon"),
      color = "yellow"
    )
  })

  # Same as above, but with fill=TRUE
  output$progressBox2 <- renderInfoBox({
    infoBox(
      "Progress", paste0(25 + input$count, "%"), icon = icon("list"),
      color = "purple", fill = TRUE
    )
  })
  output$approvalBox2 <- renderInfoBox({
    infoBox(
      "Approval", "80%", icon = icon("thumbs-up", lib = "glyphicon"),
      color = "yellow", fill = TRUE
    )
  })
}

shinyApp(ui, server)

```

```

# Module definition, new method
myModuleUI <- function(id, label = "Input text: ") {
  ns <- NS(id)
  tagList(
    textInput(ns("txt"), label),
    textOutput(ns("result"))
  )
}

myModuleServer <- function(id, prefix = "") {
  moduleServer(
    id,
    function(input, output, session) {
      output$result <- renderText({
        paste0(prefix, toupper(input$txt))
      })
    }
  )
}

# Use the module in an application
ui <- fluidPage(
  myModuleUI("myModule1")
)
server <- function(input, output, session) {
  myModuleServer("myModule1", prefix = "Converted to uppercase: ")
}
shinyApp(ui, server)

```

```

library(shiny)

ui <- basicPage(
  plotOutput("plot1", click = "plot_click"),
  verbatimTextOutput("info")
)

server <- function(input, output) {
  output$plot1 <- renderPlot({
    plot(mtcars$wt, mtcars$mpg)
  })

  output$info <- renderText({
    paste0("x=", input$plot_click$x, "\ny=", input$plot_click$y)
  })
}

shinyApp(ui, server)

```

2. Mengupload file data yang telah diunduh pada tugas sebelumnya dengan syntax: `runExample("09_upload")`, kemudian membuat aplikasi shiny untuk menampilkan grafik interaktif boxplot.

TUGAS PRAKTIKUM:

TUGAS KELOMPOK:

Melakukan visualisasi interaktif dari dataset yang sudah diunduh pada tugas kelompok sebelumnya:

1. Buat **dashboard** dengan nama header **Dashboard Kelompok X** (sesuai nama kelompok), sidebar yang memuat 3 tab (Data, Visualisasi, dan Interpretasi), serta body menampilkan output sesuai dengan masing-masing tab.
2. Tab **Data** berisi input atau upload data. Tab **Visualisasi** berisi statistika deskriptif dan minimal 2 grafik interaktif. Tab **Interpretasi** berisi analisis dan interpretasi dari hasil visualisasi.

```
# Hint: Fungsi untuk Statistika Deskriptif
summary(data)
cor(x,y) # correlation
cov(x,y) # covariance
IQR(x) # interquartile range
mean(x) # mean
median(x) # median
range(x) # min-max
sd(x) # std. dev.
var(x) # variance
```

Cara Pengumpulan:

- File yang dikumpulkan adalah skrip aplikasi shiny (.R) dan dataset (.csv)
- Pengumpulan tugas maksimal 18 Oktober 2023 pukul 23.59 WIB
- Beri nama file : Tugas M7_NIM masing-masing

REFERENSI:

Pathak, M. A., (2014), *Beginning Data Science with R*, Springer International Publishing, Switzerland.

Dietrich, D., Barry H., & Beibei Y., (2015), *Data Science & Big Data Analytics: Discovering, Analyzing, Visualizing and Presenting Data*, John Wiley & Sons, Inc., Indianapolis, Indiana

<https://shiny.rstudio.com/tutorial/>

http://homepage.stat.uiowa.edu/~rdecook/stat6220/Class_notes/R-shiny_intro.pdf#:~:text=To%20make%20an%20R%20Shiny%20app%2C%20start%20with,into%20a%20single%20folder%20named%20for%20your%20app.

<https://shiny.rstudio.com/tutorial/written-tutorial/lesson1/>

http://rstudio.github.io/shinydashboard/get_started.html

<https://shiny.rstudio.com/articles/plot-interaction.html>