A simple shiny app to explore Palmer Penguin data using ChatGPT to prototype.

Ronald (Ryy) Glenn Thomas

2025-05-13

Table of contents

1 Introduction

Shiny apps are a great way to explore data interactively. In this example, we will use ChatGPT to prototype a simple Shiny app to explore the Palmer Penguin data set from the `palmerpenguins` package.

1.1

chatGPT Prompts:

"I want to use the Palmer Penguin dataset to create a Shiny app for data exploration."

"Update shiny app. Add a dropdown menu to select categorical variables, sex, species or island. Also add a dropdown menu to select continuouse variables Use selected categorical variable as a grouping variable in side-by-side boxplots of selected continuouse variables."

"add a second interactive plot to app.R code to provide scatterplots of 2 selected continuous variables. The two cont. vars are selected from drop down menus."

1.2 R code to launch.

From inside R.

```
    shiny::runApp("app.R", launch.browser = TRUE)
```

From shell.

```
    R -e "shiny::runApp('app.R', launch.browser=T)"
```

```
library(shiny)
library(bslib)
library(bsicons)
library(palmerpenguins)
library(ggplot2)
library(corrgram)
library(dplyr)

# Load the dataset
data <- na.omit(penguins) # Remove rows with NA for simplicity

# Helper function for tooltip
add_tooltip <- function(input_ui, tooltip_text) {
  input_ui$children[[1]] <- div(
    input_ui$children[[1]], # Original label
    span(
      bs_icon("info-circle-fill"),
      class = "tooltip-icon ms-2",
      `data-bs-toggle` = "tooltip",
      `data-bs-placement` = "right",
      title = tooltip_text
    ),
    style = "display: flex; align-items: center;"
  )
  input_ui
}

# Module for inputs
inputsUI <- function(id) {
  ns <- NS(id)
  tagList(
    h4("Boxplot Controls"),
    add_tooltip(
      selectInput(
        ns("xvar"),
        "Continuous Variable for Boxplot:",
        choices = names(data)[3:6]
      ),
      "Select a continuous variable to display on the Y-axis of the boxplot."
    ),
    add_tooltip(
      selectInput(
        ns("groupvar"),
        "Group by (Categorical Variable):",
        choices = c("species", "sex", "island"),
        selected = "species"
      ),
      "Select a categorical variable to group data in the boxplot."
    ),
    hr(),
    h4("Scatterplot Controls"),
    add_tooltip(
      selectInput(
        ns("scatter_x"),
        "X-axis for Scatterplot:",
        choices = names(data)[3:6]
      ),
      "Select a variable for the X-axis of the scatterplot."
    ),
    add_tooltip(
      selectInput(
        ns("scatter_y"),
        "Y-axis for Scatterplot:",
        choices = names(data)[3:6]
      ),
      "Select a variable for the Y-axis of the scatterplot."
    ),
    add_tooltip(
      selectInput(
        ns("groupvar_scatter"),
        "Group by (Scatterplot):",
        choices = c("species", "sex", "island"),
        selected = "species"
      ),
      "Select a variable to group points in the scatterplot by color."
    )
  )
}

inputsServer <- function(id) {
  moduleServer(id, function(input, output, session) {
    reactive(input)
  })
}
```

```r
# Module for boxplot
boxplotUI <- function(id) {
  ns <- NS(id)
  plotOutput(ns("boxPlot"), height = "400px")
}

boxplotServer <- function(id, data, inputs) {
  moduleServer(id, function(input, output, session) {
    output$boxPlot <- renderPlot({
      req(data(), inputs())
      ggplot(
        data(),
        aes(
          x = .data[[inputs()$groupvar]],
          y = .data[[inputs()$xvar]],
          fill = .data[[inputs()$groupvar]]
        )
      ) +
        geom_boxplot(alpha = 0.7) +
        theme_minimal() +
        labs(
          x = inputs()$groupvar,
          y = inputs()$xvar,
          fill = inputs()$groupvar
        ) +
        theme(legend.position = "bottom")
    })
  })
}

# Module for scatterplot
scatterplotUI <- function(id) {
  ns <- NS(id)
  plotOutput(ns("scatterPlot"), height = "400px")
}

scatterplotServer <- function(id, data, inputs) {
  moduleServer(id, function(input, output, session) {
    output$scatterPlot <- renderPlot({
      req(data(), inputs())
      ggplot(
        data(),
        aes(
          x = .data[[inputs()$scatter_x]],
          y = .data[[inputs()$scatter_y]],
          color = .data[[inputs()$groupvar_scatter]]
        )
      ) +
        geom_point(alpha = 0.7, size = 2) +
        geom_smooth(
          method = "lm",
          se = FALSE,
          aes(group = 1),
          color = "black",
          linetype = "dashed"
        ) +
        geom_smooth(method = "lm", se = FALSE) +
        theme_minimal() +
        labs(
          x = inputs()$scatter_x,
          y = inputs()$scatter_y,
          color = inputs()$groupvar_scatter
        ) +
        theme(legend.position = "bottom")
    })
  })
}

# Module for correlation matrix
correlationUI <- function(id) {
  ns <- NS(id)
  plotOutput(ns("correlationMatrix"), height = "400px")
}

correlationServer <- function(id, data) {
  moduleServer(id, function(input, output, session) {
    output$correlationMatrix <- renderPlot({
      req(data())
      numericData <- data()[, sapply(data(), is.numeric)]
      corrgram(
        numericData,
        order = TRUE,
        lower.panel = panel.shade,
        upper.panel = panel.pie,
        text.panel = panel.txt,
        main = "Correlation Matrix"
      )
    })
  })
}

# Main UI and server
ui <- fluidPage(
  theme = bs_theme(bootswatch = "litera"), # Use a bslib theme
  titlePanel("Palmer Penguins Explorer"),
  fluidRow(
    column(3, inputsUI("inputs"),
       style = "overflow-y: auto; max-height: 600px;"),
    column(
      6,
      fluidRow(
        column(12, h4("Boxplot"), boxplotUI("boxplot"), style = "height: 50%;"),
        column(12, h4("Scatterplot"), scatterplotUI("scatterplot"),
           style = "height: 50%;")
      ),
      style = "overflow-y: auto;"
    ),
    column(3, h4("Correlation Matrix"), correlationUI("correlation"))
  )
)

server <- function(input, output, session) {
  # Reactive dataset
  filteredData <- reactive({
    data
  })

  # Call modules
  inputs <- inputsServer("inputs")
  boxplotServer("boxplot", filteredData, inputs)
  scatterplotServer("scatterplot", filteredData, inputs)
  correlationServer("correlation", filteredData)
}

shinyApp(ui, server)
```

1.3 Prerequisites

In development

1.4 Step-by-Step Implementation

In development

1.5 Key Takeaways

In development

1.6 Further Reading

In development