National University of the Altiplano
Faculty of Statistical and Computer Engineering
Professor: Fred Torres Cruz
Author: Eva Ruth Mamani Josec
GitHub Repository: `https://github.com/evaruth270/estadistica_app002.git`
Shiny App Link: `https://evaruthmj.shinyapps.io/app_est_comp01/`
Facebook Link: `https://www.facebook.com/share/p/16CpLtkfjk//`

STATISTICAL APPLICATION

1. **Central Limit Theorem (CLT)**
This module allows simulating the distribution of sample means from a population with a discrete uniform distribution (values from 1 to 100). The user can choose the sample size and the number of simulations. As a result, a histogram of the sample means is generated. The graph shows how, as the number of simulations increases, the distribution of the means tends to be normal, demonstrating the Central Limit Theorem.

2. **Normality Tests**
In this section, the user can upload a database in `.csv` or `.xlsx` format to perform normality tests on a numerical variable. The tests that can be applied are:

   - Shapiro-Wilk
   - Kolmogorov-Smirnov
   - Lilliefors
   - Jarque-Bera

   In addition to the statistical result (p-value), a histogram of the variable is shown, which allows for a visual interpretation of the distribution's shape.

3. **Chi-Square Test**
This module allows applying the chi-square test to compare observed frequencies with expected frequencies. The user can manually input data or upload a file. The result shows the chi-square statistic and the corresponding p-value. A bar chart comparing observed and expected frequencies is also provided.

4. **Student's t-Test**
This section allows applying the Student's t-test for two types of comparisons:

   - Independent samples
   - Paired samples

   The user must upload a file with two columns representing the groups. The t-statistic and p-value are calculated, and a boxplot is shown to visually compare the groups.

5. **ANOVA (Analysis of Variance)**
   This module allows comparing three or more independent groups using one-way ANOVA. The user must upload a file with one numeric column (dependent variable) and one categorical column (group or treatment). The ANOVA summary is presented along with a boxplot for group comparison.

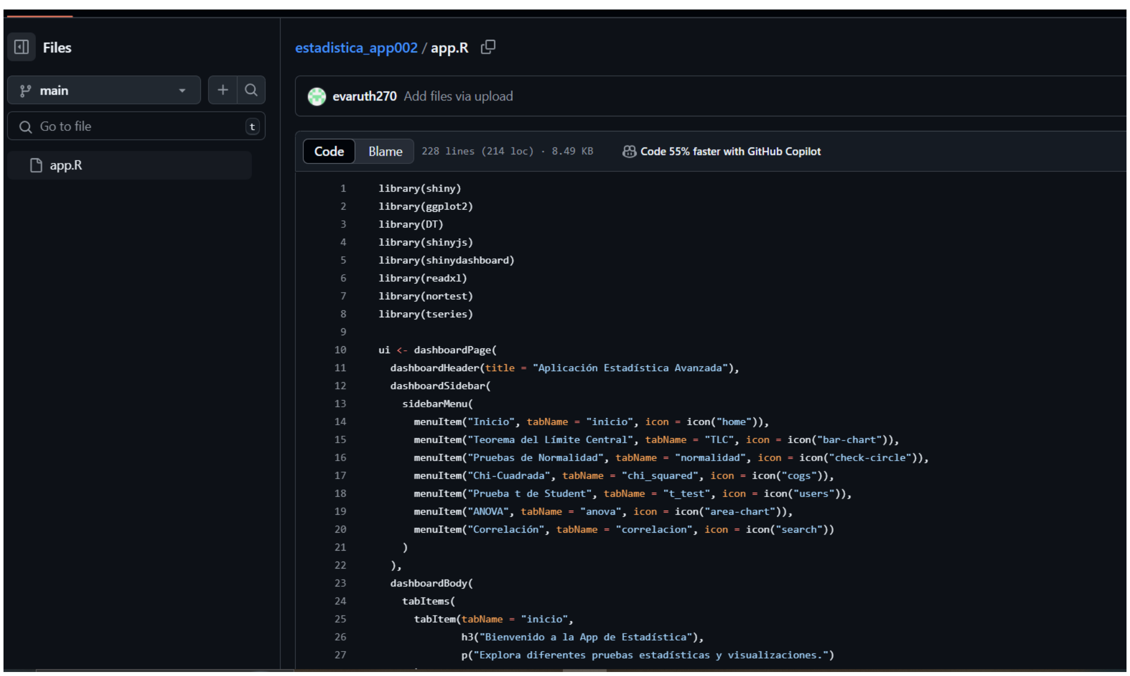
6. **Correlation**
   This section allows calculating the correlation between two numerical variables. Depending on the sample size, one can apply:

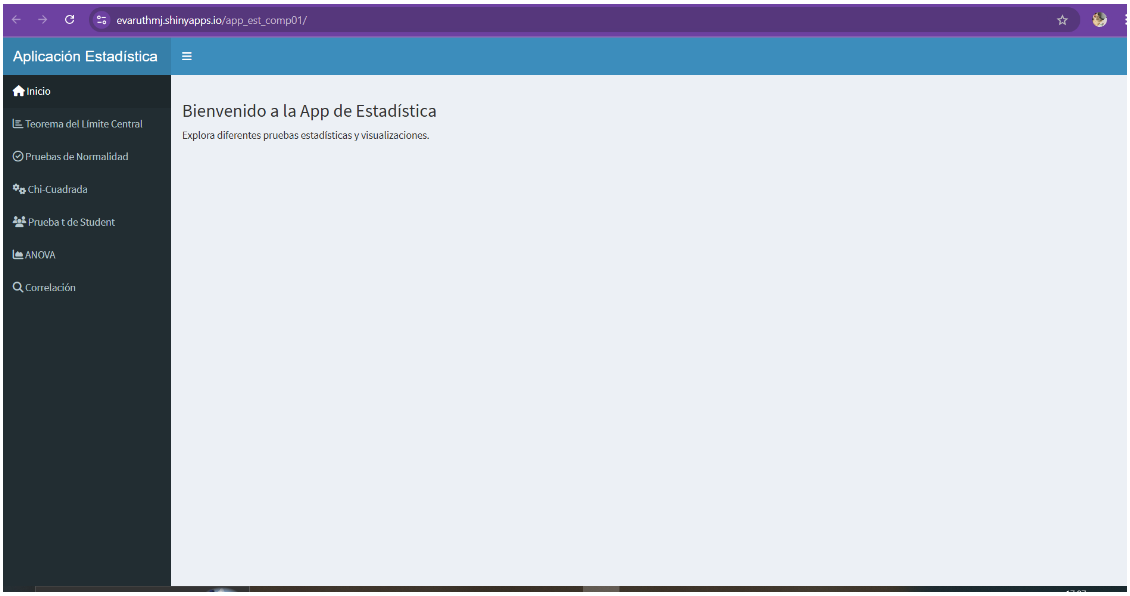
   - Pearson correlation (when $n > 30$)
   - Spearman correlation (when $n < 30$ or for ordinal data)

   The user uploads a file with two numeric columns. The output includes the correlation coefficient, the p-value, and a scatter plot with a trend line.

General Considerations

The application is built using the `shinydashboard` package, which organizes the contents into tabs and collapsible panels. Each analysis displays clear statistical results accompanied by graphs created with `ggplot2`, allowing users to visually interpret the data. The app is intended both for educational purposes and for practical statistical data analysis.





R Code

```
library(shiny)
library(ggplot2)
library(DT)
library(shinyjs)
library(shinydashboard)
library(readxl)
library(nortest)
library(tseries)

ui <- dashboardPage(
  dashboardHeader(title = "Aplicaci n Estad stica Avanzada"),
  dashboardSidebar(
    sidebarMenu(
      menuItem("Inicio", tabName = "inicio", icon = icon("home")),
      menuItem("Teorema del L mite Central", tabName = "TLC", icon = icon("b
      menuItem("Pruebas de Normalidad", tabName = "normalidad", icon = icon("
      menuItem("Chi-Cuadrada", tabName = "chi_squared", icon = icon("cogs")),
      menuItem("Prueba t de Student", tabName = "t_test", icon = icon("users"
      menuItem("ANOVA", tabName = "anova", icon = icon("area-chart")),
      menuItem("Correlaci n", tabName = "correlacion", icon = icon("search")
    )
  ),
  dashboardBody(
    tabItems(
```

```
tabItem(tabName = "inicio",
        h3("Bienvenido a la App de Estad stica"),
        p("Explora diferentes pruebas estad sticas y visualizaciones."
),
tabItem(tabName = "TLC",
        sidebarLayout(
          sidebarPanel(
            numericInput("n_size", "Tama o de muestra:", value = 30, n
            numericInput("n_sims", "N mero de simulaciones:", value =
          ),
          mainPanel(
            plotOutput("TLCPlot")
          )
        )
),
tabItem(tabName = "normalidad",
        sidebarLayout(
          sidebarPanel(
            fileInput("file", "Sube CSV o XLSX", accept = c(".csv", ".x
            selectInput("normal_test", "Prueba de normalidad",
                        choices = c("Shapiro-Wilk", "Kolmogorov-Smirnov
          ),
          mainPanel(
            verbatimTextOutput("normality_result"),
            plotOutput("normality_plot")
          )
        )
),
tabItem(tabName = "chi_squared",
        sidebarLayout(
          sidebarPanel(
            textInput("observed", "Observados (separados por coma):", v
            textInput("expected", "Esperados (separados por coma):", va
          ),
          mainPanel(
            verbatimTextOutput("chi_squared_result"),
            plotOutput("chi_plot")
          )
        )
),
tabItem(tabName = "t_test",
        sidebarLayout(
          sidebarPanel(
            fileInput("t_file", "Sube CSV o XLSX", accept = c(".csv", "
            selectInput("t_test_type", "Tipo de prueba t",
```

```
                    choices = c("Muestras independientes", "Muestra
              ),
              mainPanel(
                verbatimTextOutput("t_test_result"),
                plotOutput("t_plot")
              )
            )
    ),
    tabItem(tabName = "anova",
            sidebarLayout(
              sidebarPanel(
                fileInput("anova_file", "Sube CSV o XLSX", accept = c(".csv
              ),
              mainPanel(
                verbatimTextOutput("anova_result"),
                plotOutput("anova_plot")
              )
            )
    ),
    tabItem(tabName = "correlacion",
            sidebarLayout(
              sidebarPanel(
                fileInput("corr_file", "Sube CSV o XLSX", accept = c(".csv"
                selectInput("corr_type", "Tipo de correlaci n", choices =
              ),
              mainPanel(
                verbatimTextOutput("correlation_result"),
                plotOutput("corr_plot")
              )
            )
    )
   )
  )
)

server <- function(input, output) {
  # TLC
  output$TLCPlot <- renderPlot({
    set.seed(123)
    sample_means <- replicate(input$n_sims, mean(sample(1:100, input$n_size,
    ggplot(data.frame(x = sample_means), aes(x = x)) +
      geom_histogram(bins = 30, fill = "skyblue", color = "black") +
      ggtitle("Distribuci n de medias (TLC)") +
      theme_minimal()
  })
```

```
# Pruebas de Normalidad
output$normality_result <- renderPrint({
  req(input$file)
  ext <- tools::file_ext(input$file$name)
  data <- if (ext == "csv") read.csv(input$file$datapath) else read_xlsx(in
  x <- data[[1]]
  switch(input$normal_test,
         "Shapiro-Wilk" = shapiro.test(x),
         "Kolmogorov-Smirnov" = ks.test(x, "pnorm", mean = mean(x), sd = sd
         "Lilliefors" = lillie.test(x),
         "Jarque-Bera" = jarque.bera.test(x))
})

output$normality_plot <- renderPlot({
  req(input$file)
  ext <- tools::file_ext(input$file$name)
  data <- if (ext == "csv") read.csv(input$file$datapath) else read_xlsx(in
  x <- data[[1]]
  ggplot(data.frame(x), aes(x = x)) +
    geom_histogram(bins = 30, fill = "lightgreen", color = "black") +
    ggtitle("Histograma de la variable") +
    theme_minimal()
})

# Chi-Cuadrada
output$chi_squared_result <- renderPrint({
  obs <- as.numeric(strsplit(input$observed, ",")[[1]])
  exp <- as.numeric(strsplit(input$expected, ",")[[1]])
  chisq.test(obs, p = exp / sum(exp))
})

output$chi_plot <- renderPlot({
  obs <- as.numeric(strsplit(input$observed, ",")[[1]])
  exp <- as.numeric(strsplit(input$expected, ",")[[1]])
  df <- data.frame(Grupo = factor(1:length(obs)), Observado = obs, Esperado
  ggplot(df, aes(x = Grupo)) +
    geom_bar(aes(y = Observado), stat = "identity", fill = "orange") +
    geom_point(aes(y = Esperado), color = "red", size = 3) +
    ggtitle("Comparaci n Observado vs Esperado") +
    theme_minimal()
})

# Prueba t
output$t_test_result <- renderPrint({
```

```
    req(input$t_file)
    ext <- tools::file_ext(input$t_file$name)
    data <- if (ext == "csv") read.csv(input$t_file$datapath) else read_xlsx(
    if (input$t_test_type == "Muestras independientes") {
      t.test(data[[1]], data[[2]])
    } else {
      t.test(data[[1]], data[[2]], paired = TRUE)
    }
  })

  output$t_plot <- renderPlot({
    req(input$t_file)
    ext <- tools::file_ext(input$t_file$name)
    data <- if (ext == "csv") read.csv(input$t_file$datapath) else read_xlsx(
    df <- data.frame(grupo = rep(c("Grupo 1", "Grupo 2"), each = nrow(data)),
                     valor = c(data[[1]], data[[2]]))
    ggplot(df, aes(x = grupo, y = valor, fill = grupo)) +
      geom_boxplot() +
      ggtitle("Boxplot Comparativo") +
      theme_minimal()
  })

  # ANOVA
  output$anova_result <- renderPrint({
    req(input$anova_file)
    ext <- tools::file_ext(input$anova_file$name)
    data <- if (ext == "csv") read.csv(input$anova_file$datapath) else read_x
    colnames(data) <- c("valor", "grupo")
    aov_result <- aov(valor ~ as.factor(grupo), data = data)
    summary(aov_result)
  })

  output$anova_plot <- renderPlot({
    req(input$anova_file)
    ext <- tools::file_ext(input$anova_file$name)
    data <- if (ext == "csv") read.csv(input$anova_file$datapath) else read_x
    colnames(data) <- c("valor", "grupo")
    ggplot(data, aes(x = as.factor(grupo), y = valor, fill = grupo)) +
      geom_boxplot() +
      ggtitle("Boxplot por grupo (ANOVA)") +
      theme_minimal()
  })

  # Correlaci n
  output$correlation_result <- renderPrint({
```

```
    req(input$corr_file)
    ext <- tools::file_ext(input$corr_file$name)
    data <- if (ext == "csv") read.csv(input$corr_file$datapath) else read_xl
    if (input$corr_type == "Pearson") {
      cor.test(data[[1]], data[[2]], method = "pearson")
    } else {
      cor.test(data[[1]], data[[2]], method = "spearman")
    }
  })

  output$corr_plot <- renderPlot({
    req(input$corr_file)
    ext <- tools::file_ext(input$corr_file$name)
    data <- if (ext == "csv") read.csv(input$corr_file$datapath) else read_xl
    ggplot(data, aes(x = data[[1]], y = data[[2]])) +
      geom_point(color = "blue", size = 2) +
      geom_smooth(method = "lm", se = FALSE, color = "red") +
      ggtitle("Gr fico de dispersi n (correlaci n)") +
      theme_minimal()
  })
}

shinyApp(ui, server)
```