



Leveraging Shiny Modules for Modular and Scalable Web Applications For Clinical Trials/Research

Presentation Number: OS04

Presenter Name: Sohan Lal

Email: sohan.l225@geninvo.com

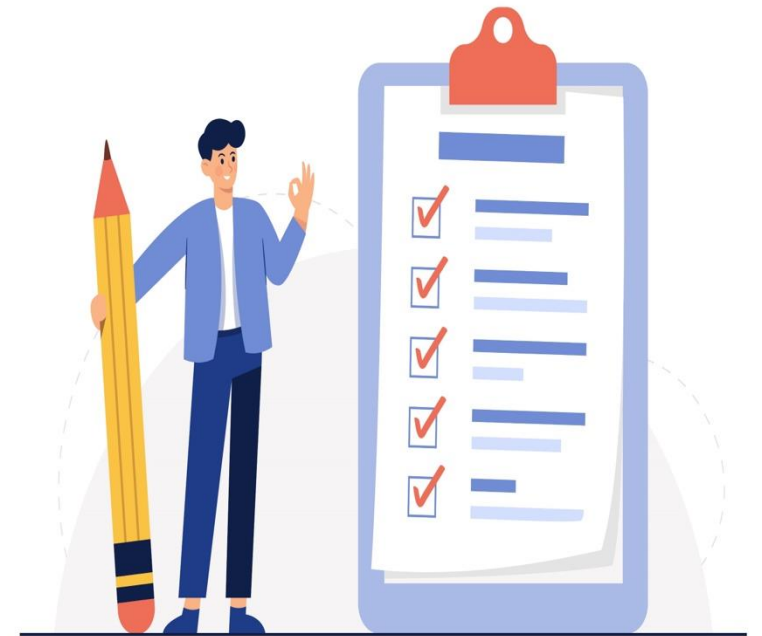
www.geninvo.com

North America | UK | Europe | India



Agenda

- **What is R Shiny Application?**
- **Challenge with Monolith Shiny Application**
- **What is Shiny Modules?**
- **Why should we use Shiny modules in Shiny application?**
- **Traditional Shiny and Shiny Module Architecture**
- **Decompose your application components with some module examples**



What is R Shiny Application ?

Shiny is a framework for building web applications using R and python.

Realtime Clinical Summaries
Data from CDISC SDTM/ADaM Pilot Project

SubjectsPopulationResponseAdverse EventsLabs+ProfilesDo-it-yourself ListingsToolsAbout

Navigate To Tab With:
Subject Listing

Subsets

Extract
☒ Current

Population
☐ All Subjects
☐ First 12 Patients
☒ First 30 Patients
☐ Week 24 Completers
☐ Safety Analysis Set

Phase
All/None
☒ H2Q-MC-LZZT(c)

Cohort
All/None
☒ Placebo
☒ Xanomeline High Dose
☒ Xanomeline Low Dose

Ad Hoc Subset
☐ Only summarize subjects on Subjects tab

Columns

All/None/Reset

☐ Phase
☒ Cohort
☒ Subject
☐ First 12 Patients
☐ First 30 Patients
☐ Week 24 Completer Population
☒ Safety Population
☐ Site ID
☒ Age
☐ Age Group
☒ Sex
☒ Race
☐ Baseline MMSE Category
☐ Baseline MMSE Score
☐ Duration of Disease (months)
☐ Ethnicity
☐ Baseline Height (cm)

Listing of Subjects - First 30 Patients

Download Selected Columns as CSV

Download Selected Columns as RTF

Note: RTF downloads may fail if output has too many columns.

Search:

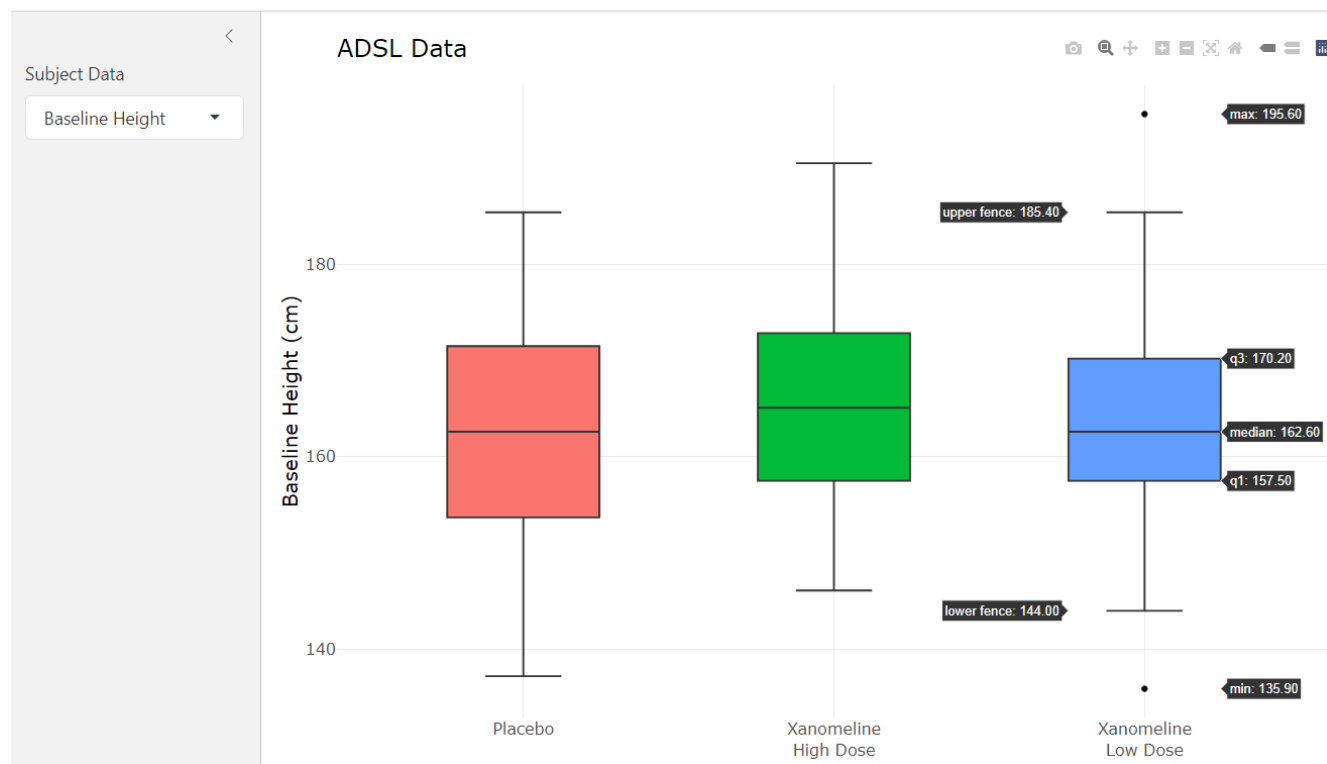
Cohort	Subject	Safety Population	Age	Sex	Race	Last Recorded Dose Day	Reason for Treatment Discontinuation
All	All	All	All	All	All	All	All
Placebo	01-701-1023	Y	64	Male	WHITE	28	ADVERSE EVENT
Placebo	01-701-1392	Y	78	Male	WHITE	183	COMPLETED
Placebo	01-703-1299	Y	81	Female	WHITE	183	COMPLETED
Placebo	01-704-1164	Y	67	Female	WHITE	198	COMPLETED
Placebo	01-704-1260	Y	71	Female	WHITE	67	WITHDRAWAL BY SUBJECT
Placebo	01-708-1087	Y	74	Female	WHITE	189	COMPLETED
Placebo	01-710-1271	Y	86	Female	WHITE	56	ADVERSE EVENT

RI ACK OR

www.geninvo.com
North America | UK | Europe | India

Shiny example app. R file

ADaM Subject-Level Analysis



www.geninvo.com
North America | UK | Europe | India

ui.R

```
# Read in Data -----
adsl <- read_xpt("adsl.xpt")

# User Interface -----
ui <- page_fillable(
  theme = bs_theme(version = 5, bootswatch = "litera"),
  titlePanel("ADaM Subject-Level Analysis"),
  card(
    layout_sidebar(
      fill = TRUE,
      fillable = TRUE,
      sidebar = sidebar(
        selectInput(
          inputId = "subject_data",
          label = "Subject Data",
          choices = c(
            "Age" = "AGE",
            "Baseline BMI" = "BMIBL",
            "Baseline Height" = "HEIGHTBL",
            "Baseline Weight" = "WEIGHTBL",
            "Years of Education" = "EDUCLVL"
          )
        ),
        plotlyOutput("boxplot")
      )
    )
  )
)
```

server.R

```
server <- function(input, output, session) {
  output$boxplot <- renderPlotly({
    the_plot <- ggplot(data = adsl, aes(x = TRT01A,
                                          y = .data[[input$subject_data]],
                                          fill = TRT01A)) +
      geom_boxplot() +
      theme_minimal() +
      theme(legend.position = "none",
            text = element_text(size = 15)) +
      labs(
        title = "ADSL Data",
        subtitle = "Comparing Treatment Groups",
        x = "",
        y = attributes(adsl[[input$subject_data]])$label
      ) +
      scale_x_discrete(labels = label_wrap(10))
    ggplotly(the_plot)
  })
}
```

Challenge with Monolith Application



- Harder for someone new to learn and contribute
- Difficult to keep the track of components
- Tedious to reuse its element within itself and other applications
- Testing and debugging individual components become challenging
- Team collaboration



What is Shiny Modules and Why to use Shiny modules?

Modules are self-contained components in R shiny that encapsulate UI and Server logic. They promote **reusability** and **modularity** allowing developers to build complex applications more efficiently.

Encapsulation

- Data Upload Module (UI + Server)
- Visualization Module (UI + Server)
- Analysis Module (UI + Server)

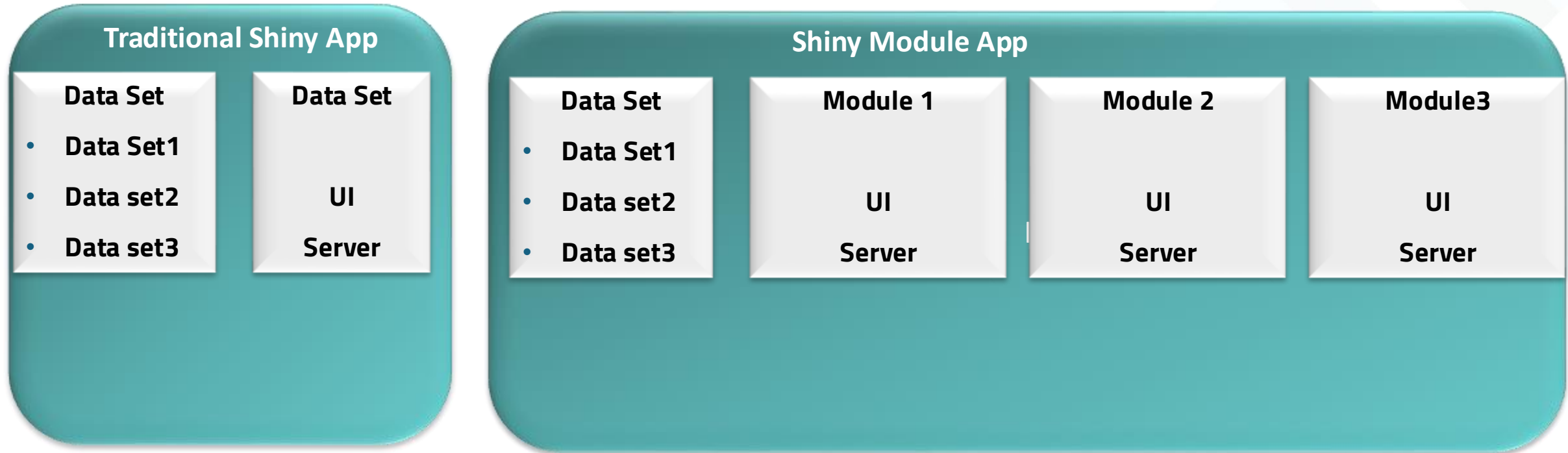
Reusability

- Step1 Module (UI + Server)
- Step 2 Module (UI + Server)
- Step 3 Module (UI + Server)

Scalability

- Baseline Module (UI + Server)
- AE Module (UI + Server)
- LB Module (UI + Server)

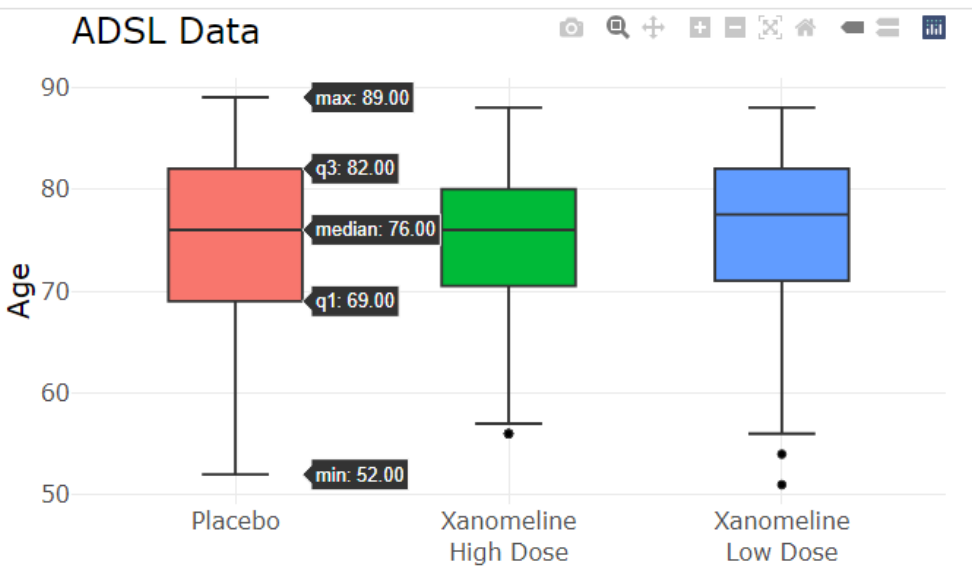
Shiny Module Architecture



Step1: Decompose your application components

Select Subject Data:

Age



Demographic	Variable	Placebo	Xanomeline High Dose	Xanomeline Low Dose
Sex	F	53	40	50
Sex	M	33	44	34
Race	BLACK OR AFRICAN AMERICAN	8	9	6
Race	WHITE	78	74	78
Race	AMERICAN INDIAN OR ALASKA NATIVE	0	1	0
Ethnicity	HISPANIC OR LATINO	3	3	6
Ethnicity	NOT HISPANIC OR LATINO	83	81	78

Input Module

Plot Module

Table Module



www.geninvo.com
North America | UK | Europe | India

Step2: Convert into Modules

Module2: plot_module.R

```
# Module for Boxplot
mod_boxplot_ui <- function(id) {
  ns <- NS(id)
  plotlyOutput(ns("boxplot"))
}

mod_boxplot_server <- function(id, selected_data, adsl) {
  moduleServer(id, function(input, output, session) {
    output$boxplot <- renderPlotly({
      the_plot <- ggplot(data = adsl, aes(x = TRT01A,
                                           y = .data[[selected_data()]],
                                           fill = TRT01A)) +

      geom_boxplot() +
      theme_minimal() +
      theme(legend.position = "none",
            text = element_text(size = 15)) +
      labs(
        title = "ADSL Data",
        subtitle = "Comparing Treatment Groups",
        x = "",
        y = attributes(adsl[[selected_data()]])$label
      ) +
      scale_x_discrete(labels = label_wrap(10))

      ggplotly(the_plot)
    })
  })
}
```

Traditional Shiny app.R

```
adsl <- read_xpt("~/Phuse Test/adsl.xpt")

# User Interface -----
ui <- fluidPage(
  titlePanel("ADaM Subject-Level Analysis"),
  tabsetPanel(
    tabPanel("Analysis 1",
      column(3,
        selectInput(inputId = "subject_data",
          label = "Select Subject Data:",
          choices = c("Age" = "AGE", "Baseline BMI" = "BMIBL",
            "Baseline Height" = "HEIGHTBL",
            "Baseline Weight" = "WEIGHTBL",
            "Years of Education" = "EDUCLVL"
          )),
        column(9, plotlyOutput("boxplot"),
          tableOutput("summary_stats")#
        ))
      )
  )

server <- function(input, output, session) {

  output$boxplot <- renderPlotly({
    the_plot <- ggplot(data = adsl, aes(x = TRT01A,
                                         y = .data[[input$subject_data]],
                                         fill = TRT01A)) +

    geom_boxplot() +
    theme_minimal() +
    theme(legend.position = "none",
          text = element_text(size = 15)) +
    labs(title = "ADSL Data",
         subtitle = "Comparing Treatment Groups",
         x = "", y = attributes(adsl[[input$subject_data]])$label
    ) + scale_x_discrete(labels = label_wrap(10))
    ggplotly(the_plot)
  })

  # Summary Statistics
  summary_data <- reactive({adsl %>%
    group_by(TRT01A) %>% summarize(
      Mean = mean(.data[[input$subject_data]], na.rm = TRUE),
      Median = median(.data[[input$subject_data]], na.rm = TRUE),
      SD = sd(.data[[input$subject_data]], na.rm = TRUE)
    )
  })
  output$summary_stats <- renderTable({
    summary_data()
  })
}
```

Modular and non- modular aap. R file

Modular app.R

```
# Load the modules
source("~/modules_final/mod_subject_input.R")
source("~/modules_final/mod_boxplot.R")
source("~/modules_final/mod_summary_stats.R")
source("~/modules_final/mod_data_table.R")

adsl <- read_xpt("~/adsl.xpt")

ui <- fluidPage(
  titlePanel("ADaM Subject-Level Analysis"),

  fluidRow(
    column(3, # Sidebar for subject data selection
      mod_subject_input_ui("subject_data")
    ),

    column(9, # Main area for the plot and tables
      mod_boxplot_ui("boxplot"),
      mod_summary_stats_ui("summary_stats"),
    )
  )
)

server <- function(input, output, session) {

  # Call the input module
  selected_data <- mod_subject_input_server("subject_data")
  mod_boxplot_server("boxplot", selected_data, adsl)
  mod_summary_stats_server("summary_stats", selected_data, adsl)

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

Non- Modular app.R

```
adsl <- read_xpt("~/Phuse Test/adsl.xpt")

# User Interface -----
ui <- fluidPage(
  titlePanel("ADaM Subject-Level Analysis"),
  tabsetPanel(
    tabPanel("Analysis 1",
      column(3,
        selectInput(inputId = "subject_data",
          label = "Select Subject Data:",
          choices = c("Age" = "AGE", "Baseline BMI" = "BMIBL",
            "Baseline Height" = "HEIGHTBL",
            "Baseline Weight" = "WEIGHTBL",
            "Years of Education" = "EDUCLVL"
          )),
        column(9, plotlyOutput("boxplot"),
          tableOutput("summary_stats")#
        )
      )
    )
  )

server <- function(input, output, session) {

  output$boxplot <- renderPlotly({
    the_plot <- ggplot(data = adsl, aes(x = TRT01A,
      y = .data[[input$subject_data]],
      fill = TRT01A)) +

    geom_boxplot() +
    theme_minimal() +
    theme(legend.position = "none",
      text = element_text(size = 15)) +
    labs(title = "ADSL Data",
      subtitle = "Comparing Treatment Groups",
      x = "", y = attributes(adsl[[input$subject_data]])$label
    ) + scale_x_discrete(labels = label_wrap(10))
    ggplotly(the_plot)
  })

  # Summary Statistics
  summary_data <- reactive({adsl %>%
    group_by(TRT01A) %>% summarize(
      Mean = mean(.data[[input$subject_data]], na.rm = TRUE),
      Median = median(.data[[input$subject_data]], na.rm = TRUE),
      SD = sd(.data[[input$subject_data]], na.rm = TRUE)
    )
  })

  output$summary_stats <- renderTable({
    summary_data()
  })
}
```



www.geninvo.com
North America | UK | Europe | India

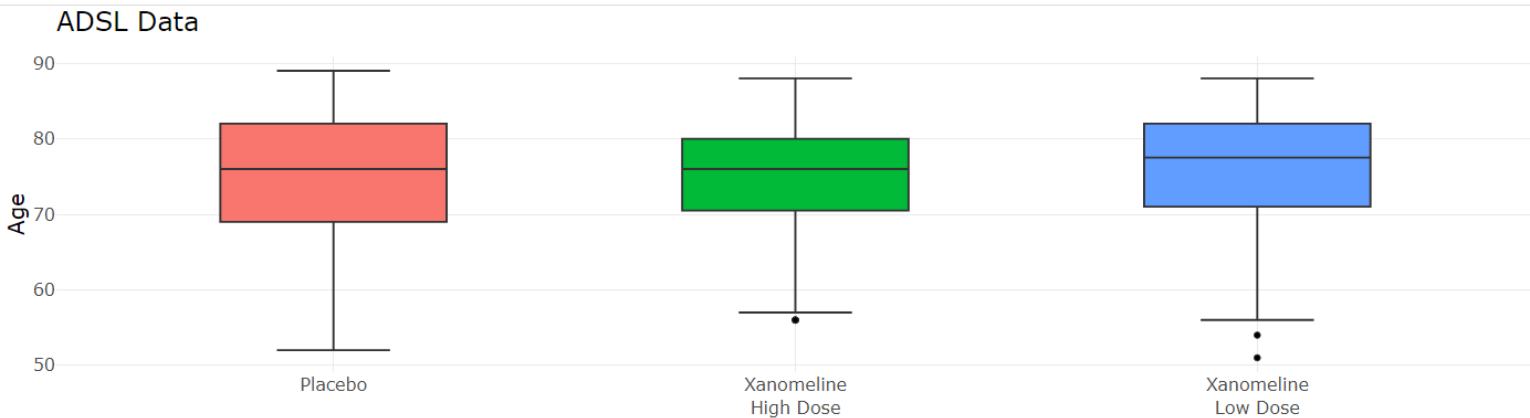
Same Module Calling app. R file

ADaM Subject-Level Analysis

Analysis 1 Analysis 2

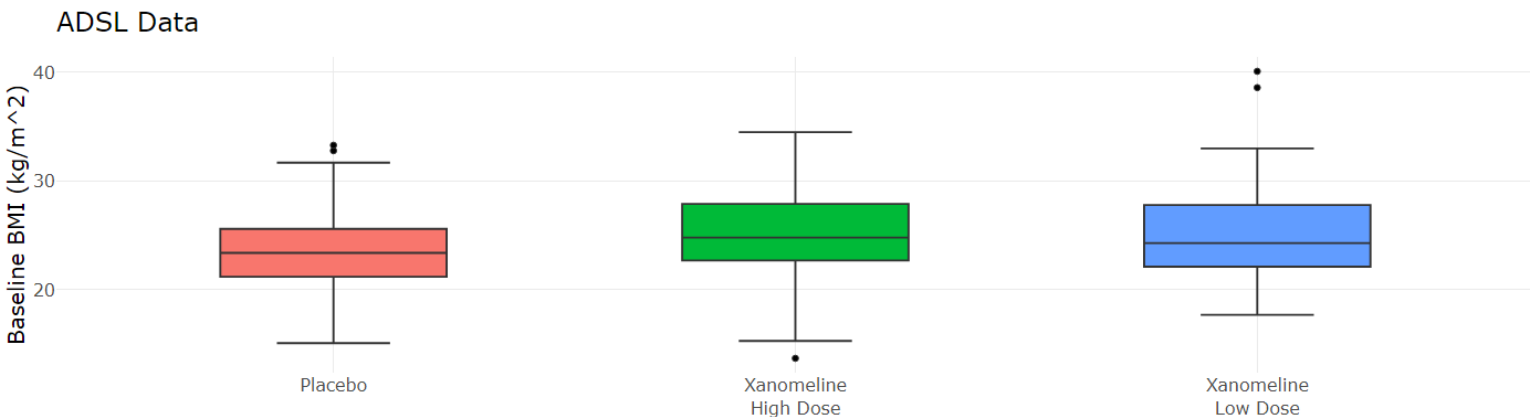
Select Subject Data:

Age



Select Subject Data:

Baseline BMI



www.geninvo.com
North America | UK | Europe | India

Same Tab Module Calling

```
ui <- fluidPage(  
  titlePanel("ADaM Subject-Level Analysis"),  
  tabsetPanel(  
    tabPanel("Analysis 1",  
      fluidRow(column(3,  
                    mod_subject_input_ui("subject_data1")  
                ),  
                column(9,  
                    mod_boxplot_ui("boxplot1")  
                )),  
      fluidRow(column(3,  
                    mod_subject_input_ui("subject_data4")  
                ),  
                column(9,  
                    mod_boxplot_ui("boxplot4")  
                ))  
    ),  
  ))
```

```
> # Server Function -----  
> server <- function(input, output, session) {
```

```
  selected_data1 <- mod_subject_input_server("subject_data1")  
  
  mod_boxplot_server("boxplot1", selected_data1, adsl)  
  #mod_summary_stats_server("summary_stats1", selected_data1, adsl)  
  mod_data_table_server("data_table1", selected_data1, adsl)
```

```
  selected_data4 <- mod_subject_input_server("subject_data4")  
  mod_boxplot_server("boxplot4", selected_data4, adsl)  
  #mod_summary_stats_server("summary_stats1", selected_data1, adsl)  
  mod_data_table_server("data_table4", selected_data4, adsl)
```

Multiple Tab Module Calling

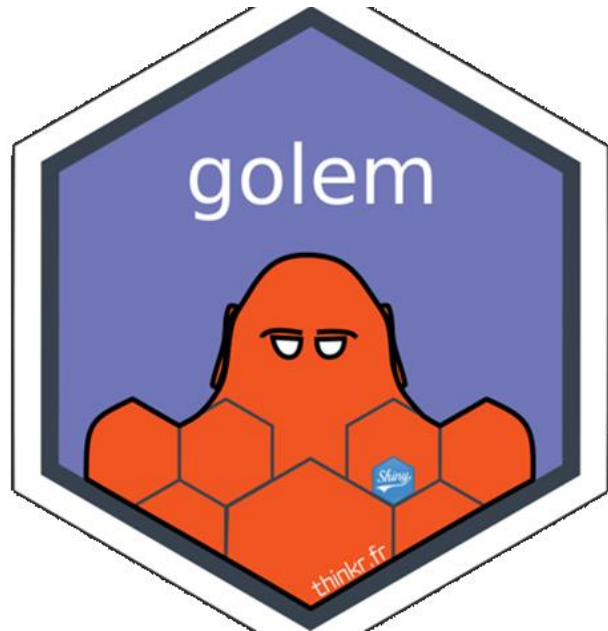
```
ui <- fluidPage(  
  titlePanel("ADaM Subject-Level Analysis"),  
  tabsetPanel(  
    | tabPanel("Analysis 1",  
      fluidRow(column(3,  
        mod_subject_input_ui("subject_data1")  
      ),  
      column(9,  
        mod_boxplot_ui("boxplot1"),  
        mod_summary_stats_ui("summary_stats1"),  
        mod_data_table_ui("data_table1")  
      ))  
    ),  
    tabPanel("Analysis 2", fluidRow(  
      column(3, mod_subject_input_ui("subject_data2")  
    ),  
      column(9,  
        mod_boxplot_ui("boxplot2"),  
        mod_summary_stats_ui("summary_stats2"),  
        mod_data_table_ui("data_table2")  
      ))))  
  )  
  
# Server Function -----  
server <- function(input, output, session) {  
  
  selected_data1 <- mod_subject_input_server("subject_data1")  
  
  mod_boxplot_server("boxplot1", selected_data1, adsl)  
  mod_summary_stats_server("summary_stats1", selected_data1, adsl)  
  mod_data_table_server("data_table1", selected_data1, adsl)  
  
  selected_data2 <- mod_subject_input_server("subject_data2")  
  
  mod_boxplot_server("boxplot2", selected_data2, adsl)  
  mod_summary_stats_server("summary_stats2", selected_data2, adsl)  
  mod_data_table_server("data_table2", selected_data2, adsl)  
  
}  
  
# Run the Shiny App -----  
shinyApp(ui = ui, server = server)
```

Benefits of Shiny Module



- Helps you break your big, complex app into smaller parts.
- Easy to keep the track of components
- Single location to debug if something fails
- Structured cleanly for someone new to learn and contribute
- Element usability within itself and other applications
- No need to worry about the same space conflicts.

R Shiny Module Based Frameworks



Next Step



New to Shiny? Start with modular approach



Learned the monolith way? Start with converting an existing app modular



Check out the [Chapter 19 Shiny modules | Mastering Shiny \(mastering-shiny.org\)](#)

Questions & Answer



www.geninvo.com
North America | UK | Europe | India



References

- [Reproducible and Reliable Shiny Apps for Regulatory Submissions | R-bloggers](#)



www.geninvo.com
North America | UK | Europe | India



Ideas
Delivered
Intuitively

Thank You!

+1-833-GEN-INVO (436-4686) | info@geninvo.com | www.geninvo.com

