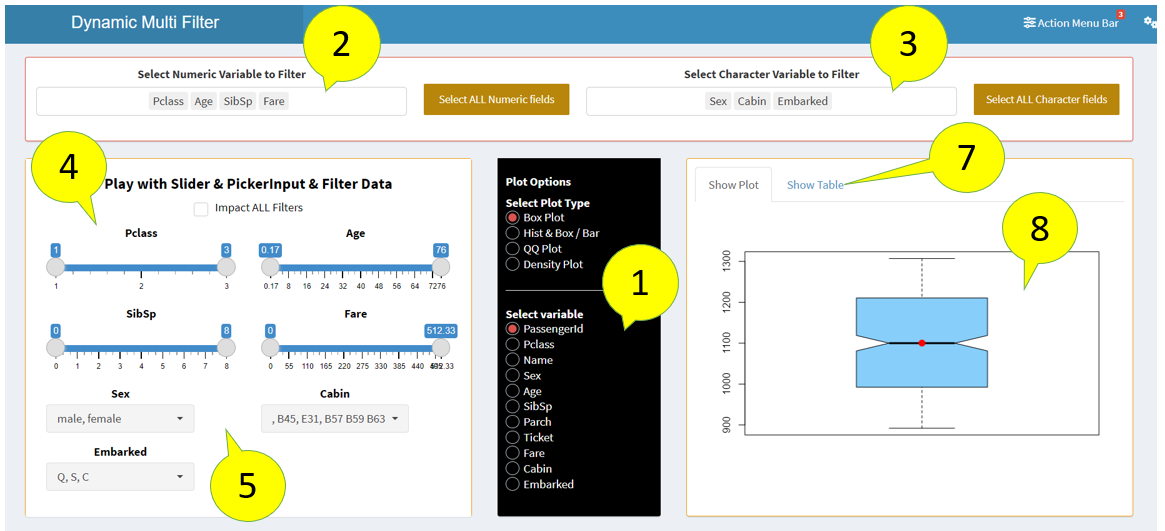
**Eight Key Sections**

These are the eight key functionalities in this Shiny App, for which we need to review the rscript.



## Get Dataset & Update Check Box and Select Input

I have uploaded Titanic dataset, once you upload, all column names get populated in section 1 as awesome checkbox and select input for numeric and character fields get updated. Here is the rscript.

# data upload section has this function

fnupdatevarchoice()

###############

fnupdatevarchoice <- function(){

updateSelectInput(session,inputId = "mNumVarFilter",choices = names(dplyr::select\_if(vmy$mydata,is.numeric)) )

updateSelectInput(session,inputId = "mChrVarFilter",choices = names(dplyr::select\_if(vmy$mydata,is.character)) )

}

## Dynamic Slider and Select Input

This section of the code in UI captures the ui output element of dynamic slider input and select input

box (

id = "slidebarbox206",

width = 5,

height = '425px',

align ='center',

HTML(paste('<h4><b>','Play with Slider & PickerInput & Filter Data','</b><h5>')),

title = NULL,

status = "warning",

solidHeader = TRUE,

collapsible = FALSE,

awesomeCheckbox(inputId = "mRedesignSliders",label = "Impact ALL Filters",value = FALSE),

uiOutput(outputId = "muimultisliderplay"),

uiOutput(outputId = "multislidertext")

)#box closure slider input

Here is the code on the server side. A variable called slidercolrange is declared and initiated on the server side.

slidercolrange <- -2

slider\_options variable gets initiated with numeric fields which you select one by one <- input$mNumVarFilter

As you select one by one the code create a list of sliders each with selected column names.

What this code does slidercolrange <- slidercolrange ++ 4 ?

A column as 12 divisions, for each slider we allocate 4 columns which gets incremented for each selection, once slidercolrange value reaches 12, slidercolrange gets initiated to 1, so that the slider takes next row

######################

output$muimultisliderplay <- renderUI({

tryCatch({

if (length(input$mNumVarFilter) != 0){

slider\_options <- input$mNumVarFilter

}

else{

return()

}

# First, create a list of sliders each with a different name

sliders <- lapply(1:length(slider\_options), function(i) {

if (slidercolrange==12){

Each selection is assigned to inputName1A variable one after another.

Than the slider input is dynamically built using the code next to that

slidercolrange <- 1

}

else{

slidercolrange <- slidercolrange ++ 4

}

inputName1A <- slider\_options[i]

if (input$mRedesignSliders == TRUE){

column(slidercolrange+4,sliderInput(inputId = inputName1A, label = inputName1A,

What are these two options?

There is a check box on the top of the slider section, whether you want all slider and select input to get affected by each selection or each slider values are independent

min=min(vmy$data\_1()[,inputName1A],na.rm = TRUE),

max=max(vmy$data\_1()[,inputName1A],na.rm = TRUE),

value=c(min(vmy$data\_1()[[inputName1A]],na.rm = TRUE),max(vmy$data\_1()[[inputName1A]],na.rm = TRUE)),

width = "500px"))

}

else{

column(slidercolrange+4,sliderInput(inputId = inputName1A, label = inputName1A,

min=min(vmy$mydata[,inputName1A],na.rm = TRUE),

max=max(vmy$mydata[,inputName1A],na.rm = TRUE),

value=c(min(vmy$mydata[[inputName1A]],na.rm = TRUE),max(vmy$mydata[[inputName1A]],na.rm = TRUE)),

What is this reactive value vmy$data\_1()?

vmy$data\_1 is a temporary dataset, which gets created as you apply slider or select any input. vmy is a reactive element declared at the starting of the server section

vmy <- reactiveValues(mydata=NULL,data\_1=NULL)

next let us understand the role of vmy$data\_1()

width = "500px"))

}

})

# Create a tagList of sliders (this is important)

do.call(tagList, sliders)

}, error=function(e){cat("ERROR :",conditionMessage(e), "\n")})

})

## vmy$data\_1 reactive variable

For our plot to get dynamically filtered, first the dataset (table) should get filtered dynamically for each adjustments in the range and also for select input. This reactive function does that job

Here mydata get temporarily assigned to data\_

vmy$data\_1 <-reactive({

tryCatch({

data\_ <- vmy$mydata

slider\_options <- input$mNumVarFilter

# this is how you fetch the input variables from ui

With this for loop function, date\_ gets filtered each slider adjustments for range value greater than or equal to as well as for less than or equal to. Here [2] is the upper value of the range and [1] is on the lower side

for(i in slider\_options) {

xxtt<-as.double(eval(parse(text=paste0("input$",i))))

data\_ <- data\_[data\_[[i]] <= xxtt[2] &

data\_[[i]] >= xxtt[1],]

}

slider\_optionsTXT <- input$mChrVarFilter

# this is how you fetch the input variables from ui component component character fields

for(i in slider\_optionsTXT) {

Similarly, the temporary table data\_ is getting filtered for each selection of character columns. Here the eval function takes the value of “i” and addes it to input$. for example if the value of “i” is column Sex. The temporary variable of xxttTXT gets the value of input$Sex

xxttTXT<-eval(parse(text=paste0("input$",i)))

data\_ <- data\_[data\_[[i]] %in% xxttTXT,]

}

data\_

}, error=function(e){cat("ERROR :",conditionMessage(e), "\n")})

return(data\_)

})