

#Dashboard header carrying the title of the dashboard

library(shiny)

library(shinydashboard)

library(dplyr)

library(ggplot2)

library(plotly)

library(shinyjs)

library(shinyBS)

library(shinydashboardPlus)

header <- dashboardHeaderPlus(title = "Accounts", enable\_rightsidebar = TRUE,

rightSidebarIcon = "info-circle")

dashboardHeader(dropdownMenuOutput("messageMenu"))

dropdownMenu(type = "messages",

messageItem(

from = "Sales Dept",

message = "Sales are steady this month."

),

messageItem(

from = "New User",

message = "How do I register?",

icon = icon("question"),

time = "13:00"

),

messageItem(

from = "Support",

message = "The new server is ready.",

icon = icon("life-ring"),

time = "2020-12-01"

)

)

#Sidebar content of the dashboard

sidebar <- dashboardSidebar(

fluidPage(

sidebarLayout(

sidebarPanel(

),

mainPanel(

textOutput("txtOutput"),

selectInput("NamesOutput","Names of the Individual",choices = c("Rahul"))

)

)

),

sidebarMenu(

menuItem("Dashboard", tabName = "dashboard", icon = icon("dashboard")),

menuItem("Dashboard", tabName = "dynamic", icon = icon("dashboard")),

menuItem("Widgets", icon = icon("th"), tabName = "widgets", badgeLabel = "new",

badgeColor = "green"),

menuItem("Visit-us", icon = icon("send",lib='glyphicon'),

href = "http://sparshv2/portals/PDP/Pages/Home.aspx"),

menuItem("DPO", icon = icon("send",lib='glyphicon'),

href = "https://infosystechnologies.sharepoint.com/slides/DPOBL"),

sidebarMenu(

menuItem("Total Accounts", tabName = "dashboard", icon = icon("tree")),

menuItem("Compliance",tabName = "Month Cost", icon = icon("tree"))

)

)

)

frow1 <- fluidRow(

tabItems(

# First tab content

tabItem(tabName = "dashboard",

tags$div(wellPanel(fluidRow(

tags$div(fileInput('file1', 'Load Master Data',

accept = c('text/csv','text/comma-separated-values',

'text/tab-separated-values','text/plain','.csv','.tsv'))),

tags$div(fileInput('file2', 'Load Calendar Hierarchy',

accept = c('text/csv','text/comma-separated-values',

'text/tab-separated-values','text/plain','.csv','.tsv'))),

tags$div(fileInput('file3', 'Load Location Hierarchy',

accept = c('text/csv','text/comma-separated-values',

'text/tab-separated-values','text/plain','.csv','.tsv')))

))),style="background-color:#FAEBD7;"),

valueBoxOutput("value1")

,valueBoxOutput("value2")

,valueBoxOutput("value3")

))

frow2 <- fluidRow(

tabItems(

tabItem(tabName = "dynamic",

fluidRow(

tabBox(width=13,

# Title can include an icon

title = tagList(shiny::icon("gear"), "Forecasts"),

tabPanel("Sales",

wellPanel(dataTableOutput("table"))),

tabPanel("Selected Forecast",

wellPanel(fluidRow(

tags$em(tags$u( h4("Forecast Accuracy "))),

verbatimTextOutput("facc"),

hr(),

# tags$em(tags$u(h4("Forecasted Sales"))),

# dataTableOutput("fcast"),

# hr(),

plotlyOutput("sgraph")))),

tabPanel("Forecast Comparison", plotlyOutput("fplot",height = "540px")),

tabPanel("Best Forecast",

wellPanel(fluidRow(

valueBoxOutput("mape",5),

valueBoxOutput("mod",7),

dataTableOutput("bfcast"),

plotlyOutput("graph")

)))

)

)

)))

frow3 <- fluidRow(

tabItems(

tabItem(tabName = "widgets",

h2("Widgets tab content"),

fluidRow(

box(plotOutput("plot1", height = 250)),

box(

title = "Controls",

sliderInput("slider", "Number of observations:", 1, 100, 50)),

box(

title = "BLR Accounts"

,status = "primary"

,solidHeader = TRUE

,collapsible = TRUE

,plotOutput("BLRAccountsbyNames", height = "300px")

)),

box(

title = "Total Accounts"

,status = "primary"

,solidHeader = TRUE

,collapsible = TRUE

,plotOutput("TotalAccountsbyNames", height = "300px")

))))

box(plotOutput("correlation\_plot"), width = 10)

box(

selectInput("features", "Features:",

c("Month Cost", "Revenue",

"Total Accounts")), width = 5

)

# combine the two fluid rows to make the body

body <- dashboardBody(frow1, frow2, frow3)

#completing the ui part with dashboardPage

ui <- dashboardPage(title = 'Dashboard', header, sidebar, body, skin='red')

observe({

if (req(input$nav) == "dashboard"){

message("Section A has been selected")

shinyjs::removeClass(id = "control-sidebar-T\_A-tab", class = "tab-pane")

shinyjs::removeClass(id = "control-sidebar-T\_B-tab", class = "tab-pane active")

shinyjs::addClass(id = "control-sidebar-T\_A-tab", class = "tab-pane active")

shinyjs::addClass(id = "control-sidebar-T\_B-tab", class = "tab-pane")

}

if (req(input$nav) == "widgets"){

message("Section B has been selected")

shinyjs::removeClass(id = "control-sidebar-T\_B-tab", class = "tab-pane")

shinyjs::removeClass(id = "control-sidebar-T\_A-tab", class = "tab-pane active")

shinyjs::addClass(id = "control-sidebar-T\_B-tab", class = "tab-pane active")

shinyjs::addClass(id = "control-sidebar-T\_A-tab", class = "tab-pane")

}

if (req(input$nav) == "charts"){

message("Section C has been selected")

shinyjs::removeClass(selector = "aside.control-sidebar-open aside.control-sidebar-dark", class = "control-sidebar-open aside.control-sidebar-dark-open")

shinyjs::addClass(selector = "aside.control-sidebar", class = "control-sidebar")

}

})

# create the server functions for the dashboard

server <- function(input, output){

#some data manipulation to derive the values of KPI boxes

total.Accounts <- sum(BLR\_Account\_Data$`Total Accounts`)

total.BLRAccounts <- sum(BLR\_Account\_Data$`BLR Accounts`)

prof.Com <- sum(BLR\_Account\_Data$Comp)

#creating the valueBoxOutput content

output$value1 <- renderValueBox({

valueBox(

formatC(total.BLRAccounts, format="d", big.mark=',')

,'Total BLRAccounts'

,icon = icon("stats",lib='glyphicon')

,color = "purple")

})

output$value2 <- renderValueBox({

valueBox(

formatC(total.Accounts, format="d", big.mark=',')

,'Total Accounts'

,icon = icon("gbp",lib='glyphicon')

,color = "green")

})

output$value3 <- renderValueBox({

valueBox(

formatC(prof.Comp, format="d", big.mark=',')

,paste('Comp:',prof.Comp)

,icon = icon("menu-hamburger",lib='glyphicon')

,color = "yellow")

})

#creating the plotOutput content

output$BLRAccountsbyNames <- renderPlot({

ggplot(data = BLR\_Account\_Data,

aes(x=Names, y=total.Accounts, fill=factor(Names))) +

geom\_bar(position = "stack", stat = "identity") + ylab("Total Count") +

xlab("Names") + theme(legend.position="bottom"

,plot.title = element\_text(size=15, face="bold")) +

ggtitle("BLR Accounts") + labs(fill = "Names")

})

output$TotalAccountsbyNames<- renderPlot({

ggplot(data = BLR\_Account\_Data,

aes(x=Names, y=total.BLRAccounts, fill=factor(Names))) +

geom\_bar(position = "identity", stat = "identity") + ylab("Counts") +

xlab("Names") + theme(legend.position="bottom"

,plot.title = element\_text(size=15, face="bold")) +

ggtitle("Total Accounts") + labs(fill = "Names")

})

output$txtOutput <- renderText({

paste(input$txtInput)

})

output$CountrOutput<- renderText({

paste(input$slctInput)

})

output$correlation\_plot <- renderPlot({

plot(BLR\_Account\_Data$Names, BLR\_Account\_Dat[[input$features]],

xlab = "Compliant", ylab = "Features")

})

output$BLR\_Account\_Data<- renderTable({

cityfilter <- subset(Names,BLR\_Account\_Data$City == input$Revenue)

})

}

#run/call the shiny app

shinyApp(ui, server)