# ui.R

library(shiny) # shiny makes it easy to build interactive web applications with R  
library(shinydashboard) # Create dashboard in the web applications

##   
## Attaching package: 'shinydashboard'

## The following object is masked from 'package:graphics':  
##   
## box

library(DT) # Provides a R interface to manipulate data tables

##   
## Attaching package: 'DT'

## The following objects are masked from 'package:shiny':  
##   
## dataTableOutput, renderDataTable

library(ggplot2) # Makes it easy to plot  
library(plotly) # Plotly creates interactive plots

##   
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':  
##   
## last\_plot

## The following object is masked from 'package:stats':  
##   
## filter

## The following object is masked from 'package:graphics':  
##   
## layout

library(leaflet) # leaflet enables us to display map with geographical data  
  
  
  
# Data preparation  
data = readRDS("D:\\Warwick\\Courses\\Term 1\\Data Management\\Group Assignment\\PartD\\Demo\\final\_result.rds")  
data$Longitude = as.numeric(data$Longitude)  
data$Latitude = as.numeric(data$Latitude)  
# Change encoding to utf-8  
Encoding(data$LocalAuthorityName) = "UTF-8"  
Encoding(data$AddressLine) = "UTF-8"  
  
  
  
dbHeader = dashboardHeader(title="UK Food Hygiene",  
 tags$li(a(onclick = "openTab('HygieneMap')",  
 icon("home"),  
 title = "Back to Home",  
 style = "cursor: pointer;"),  
 class = "dropdown",  
 tags$script(HTML("  
 var openTab = function(tabName){  
 $('a', $('.sidebar')).each(function() {  
 if(this.getAttribute('data-value') == tabName) {  
 this.click()};});}"))),  
 tags$li(a(href = 'https://www.food.gov.uk/',  
 icon("utensils"),  
 title = "Food Hygiene Home"),  
 class = "dropdown"))  
  
ui = dashboardPage(  
 dbHeader,   
 dashboardSidebar(   
   
 sidebarMenu(  
 menuItem("Food Map", tabName = "HygieneMap", icon = icon("map-marked-alt")),  
 menuItem("Plots", tabName = "plotdashboard", icon = icon("th")),  
 menuItem("Controls", tabName = "Hygieneplotswithcontrols", icon = icon("filter")),  
 menuItem("Data", tabName = "Hygienedata", icon = icon("dashboard"))  
   
 )),   
 dashboardBody(   
 tabItems(  
 # First tab content  
 tabItem(tabName = "HygieneMap",  
 fluidRow(h1(style = "font-family:Times New Roman;padding-left:16px", "Food Map")),   
 fluidRow(  
 column(  
 4,  
 selectInput('City', 'Select City', unique(data$LocalAuthorityName), selectize = TRUE)  
 ),  
 column(  
 4,  
 selectInput('BusinessType', 'Select Business Type', choices = NULL, selectize = TRUE)  
 ),  
 column(  
 4,  
 selectInput('RatingValue', 'Select Rating Value', choices = NULL, selectize = TRUE)  
 ),  
 ),  
 fluidRow(column(width = 12, leafletOutput("mymap", "100%", height = "500"))),  
 fluidRow(column(width = 12, p(textOutput("desc")))),  
 fluidRow(a(style = "padding-left:25px;font-size:10;font-family:Lato;text-decoration:underline",  
 "Source: Food Standards Agency",  
 href = "https://www.food.gov.uk/uk-food-hygiene-rating-data-api"))  
 ),  
   
 # Second tab  
 tabItem(tabName = "plotdashboard",  
 fluidRow(h1(style = "font-family:Times New Roman;padding-left:16px", "Plots")),  
   
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "The share of Scheme Type")),  
 fluidRow(  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotlyOutput("myplot2"))))),  
 tags$hr(style="border-color:black"),  
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "The share of New Rating Pending")),  
 fluidRow(  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotlyOutput("myplot4"))))),  
 tags$hr(style="border-color:black"),  
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "The Number of Missing values for Scores")),  
 fluidRow(  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotlyOutput("myplot5"))))),  
 fluidRow(a(style = "padding-left:25px;font-size:10;font-family:Lato;text-decoration:underline",  
 "Source: Food Standards Agency",  
 href = "https://www.food.gov.uk/uk-food-hygiene-rating-data-api"))  
 ),  
   
 # Third tab  
 tabItem(tabName = "Hygieneplotswithcontrols",  
 fluidRow(h1(style = "font-family:Times New Roman;padding-left:16px", "Controls")),  
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "The Share of Rating Values in each city")),  
 fluidRow(  
 column(  
 width = 5,  
 selectInput('SchemeForPlot', 'Select Scheme',   
 c("Overall",   
 ifelse(unique(data$SchemeType) == "FHIS", "FHIS (Scotland)",  
 "FHRS (England, Wales and Northern Ireland)")),   
 selectize = TRUE)  
 ),  
 column(  
 width = 5,  
 selectInput('CityForPlot', 'Select City',   
 choices = NULL,   
 selectize = TRUE)  
 ),  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotOutput("myplot1"))))),  
   
 tags$hr(style="border-color:black"),  
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "Business Type Share")),  
 fluidRow(  
 column(  
 width = 4,  
 selectInput('CityForPlot2', 'Select City',   
 c("Overall", unique(data$LocalAuthorityName)),   
 selectize = TRUE)  
 ),  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotOutput("myplot3"))))),  
 tags$hr(style="border-color:black"),  
 fluidRow(h3(style = "font-family:Times New Roman;padding-left:16px",   
 "Scores")),  
 fluidRow(  
 column(  
 width = 4,  
 selectInput('ScorePlot', 'Select Score Type',   
 c(unique(colnames(data[,c(19:21)]))),   
 selectize = TRUE)  
 ),  
 fluidRow(  
 column(width = 12,  
 box(width = 12,  
 plotlyOutput("myplot6"))))),  
 fluidRow(a(style = "padding-left:25px;font-size:10;font-family:Lato;text-decoration:underline",  
 "Source: Food Standards Agency",  
 href = "https://www.food.gov.uk/uk-food-hygiene-rating-data-api"))  
 ),  
   
 # Last tab content  
 tabItem(tabName = "Hygienedata",  
 fluidRow(h1(style = "font-family:Times New Roman;padding-left:16px", "Dataset")),  
 fluidRow(  
 column(width = 12,   
 div(style = 'overflow-x: scroll', DT::dataTableOutput("mydata"),  
 tags$style(HTML(".dataTables\_wrapper .dataTables\_length {  
 float: left;}.dataTables\_wrapper .dataTables\_filter {  
 float: right; text-align: left;}"))  
 )  
 )  
 ),  
 fluidRow(a(style = "padding-left:25px;font-size:10;font-family:Lato;text-decoration:underline",  
 "Source: Food Standards Agency",  
 href = "https://www.food.gov.uk/uk-food-hygiene-rating-data-api"))  
 )  
 )  
 )  
)

# server.R

#  
# This is the server logic of a Shiny web application. You can run the  
# application by clicking 'Run App' above.  
#  
# Find out more about building applications with Shiny here:  
#  
# http://shiny.rstudio.com/  
#  
library(tidyverse) # tidyverse provides data manipulation functions and pipeline

## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.2 √ purrr 0.3.4  
## √ tibble 3.0.4 √ dplyr 1.0.2  
## √ tidyr 1.1.2 √ stringr 1.4.0  
## √ readr 1.4.0 √ forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(shiny) # shiny makes it easy to build interactive web applications with R  
library(ggplot2) # Makes it easy to plot  
library(plotly) # Plotly creates interactive plots

##   
## Attaching package: 'plotly'

## The following object is masked from 'package:ggplot2':  
##   
## last\_plot

## The following object is masked from 'package:stats':  
##   
## filter

## The following object is masked from 'package:graphics':  
##   
## layout

library(ggpubr) # ggpubr provides ggarrange function that can arrange multiple ggplots on the same page  
# install.packages("ggpubr")  
  
  
server = function(input, output, session){  
 data\_no\_na\_geo = data %>%  
 filter(!is.na(Longitude))  
   
 # Tab 1: select a city  
 observeEvent(input$City,{  
 updateSelectInput(session,'BusinessType',  
 choices=c("Select a business type", data\_no\_na\_geo %>%  
 filter(LocalAuthorityName == input$City) %>%  
 distinct(BusinessType) %>%  
 arrange(BusinessType)))  
 })  
   
 # If the size of selection is 1, updateSelectInput will use the column name as the choice ("https://stackoverflow.com/questions/43098849/display-only-one-value-in-selectinput-in-r-shiny-app"), so I create another dataframe and change its column name to the selection to solve this problem  
 temp = reactive({  
 df = data\_no\_na\_geo  
 if (nrow(df %>%   
 filter(LocalAuthorityName == input$City) %>%   
 filter(BusinessType == input$BusinessType) %>%  
 distinct(RatingValue)) == 1){  
 temp\_name = df %>%   
 filter(LocalAuthorityName == input$City) %>%   
 filter(BusinessType == input$BusinessType) %>%  
 select(RatingValue) %>%  
 distinct(RatingValue) %>%  
 as.vector()  
 colnames(df) = c(colnames(df)[1:7], temp\_name, colnames(df)[9:22])   
 return(df)  
 }  
 else{  
 return(df)  
 }  
 })  
   
 # Tab 1: Select a business type  
 observeEvent(input$BusinessType,{  
 updateSelectInput(session,'RatingValue',  
 choices=c("Select a rating value",   
 temp() %>%  
 filter(LocalAuthorityName == input$City &  
 BusinessType == input$BusinessType) %>%  
 select(colnames(temp())[8]) %>%  
 distinct() %>%  
 arrange() %>%  
 as.vector()  
 )  
 )  
 })  
   
 # Get data from a specified city and a particular type of business  
 filteredData = reactive({  
 df = data\_no\_na\_geo %>%   
 filter(LocalAuthorityName == input$City) %>%   
 filter(BusinessType == input$BusinessType) %>%   
 filter(RatingValue == input$RatingValue)  
 return(df)   
 })  
   
 # Plot the whole map  
 output$mymap = renderLeaflet({  
 leaflet(  
 options = leafletOptions(zoomControl = FALSE)) %>%  
 htmlwidgets::onRender("function(el, x) {L.control.zoom({position: 'topright'}).addTo(this)}") %>%   
 addProviderTiles(providers$OpenStreetMap) %>%   
 setView(lng = -4.53067, lat = 54.22864, zoom = 5)  
 })  
   
 # Add markers to the map  
 observe({  
 icons <- awesomeIcons(  
 icon = 'ios-close',  
 iconColor = 'white',  
 library = 'ion',  
 markerColor = "red"  
 )  
 mymap\_proxy <- leafletProxy("mymap", data = filteredData()) %>%   
 clearMarkers() %>%   
 addAwesomeMarkers(filteredData()$Longitude,   
 filteredData()$Latitude,   
 icon = icons,   
 popup = paste("<b>",filteredData()$BusinessName,"</b>", "<br>",  
 "<b>","Postcode:","</b>", filteredData()$PostCode, "<br>",  
 "<b>","Address:","</b>", filteredData()$AddressLine, "<br>",  
 "<b>","Food hygiene ratings:","</b>",filteredData()$RatingValue, "<br>",  
 "<b>","Rating Scheme:","</b>",filteredData()$SchemeType, "<br>",  
 "<b>","Hygiene Score:","</b>",filteredData()$Hygiene, "<br>",  
 "<b>","Structural Score:","</b>",filteredData()$Structural, "<br>",  
 "<b>","Management Score:","</b>",filteredData()$ConfidenceInManagement) %>%   
 lapply(htmltools::HTML)) %>%   
 flyToBounds(lng1 = max(filteredData()$Longitude), lng2 = min(filteredData()$Longitude),  
 lat1 = max(filteredData()$Latitude), lat2 = min(filteredData()$Latitude))  
   
 })  
   
   
 ## Tab 2: Plots  
 # Plot1 - Scheme Type share  
 output$myplot2 = renderPlotly({  
 a = ggplot(data = data, aes(x = SchemeType)) +   
 geom\_bar() +  
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=14),   
 axis.text.y = element\_text(face="bold", color="#993333", size=14),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14))  
 ggplotly(a)  
 })  
   
 # Plot2 - New Rating Pending share  
 output$myplot4 = renderPlotly({  
 a = ggplot(data = data, aes(x = NewRatingPending)) +   
 geom\_bar() +  
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=14),   
 axis.text.y = element\_text(face="bold", color="#993333", size=14),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14))  
 ggplotly(a)  
 })  
   
   
 # Remove data with missing scores  
 data\_no\_missing\_score = data %>%  
 filter(!(is.na(Hygiene) & is.na(Structural) & is.na(ConfidenceInManagement)))  
   
 output$myplot5 = renderPlotly({  
 a = ggplot(data = data.frame(Scores = c("NA",   
 "Non-NA"),  
 Count = c(nrow(data)-nrow(data\_no\_missing\_score),   
 nrow(data\_no\_missing\_score))),  
 aes(x = Scores, y = Count)) +  
 geom\_bar(stat = "identity") +   
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=14),   
 axis.text.y = element\_text(face="bold", color="#993333", size=14),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14))  
 ggplotly(a)  
 })  
   
   
 ## Tab 3: Controlled plot  
 # Plot1 - the bar chart for rating values in each city  
 # Select a scheme type  
 observeEvent(input$SchemeForPlot,{  
 updateSelectInput(session,'CityForPlot',  
 choices=c(data %>%  
 filter(SchemeType == substr(input$SchemeForPlot, start = 1, stop = 4)) %>%  
 arrange(LocalAuthorityName) %>%  
 distinct(LocalAuthorityName)))})  
   
 # Get data from a particular city and a specified type of scheme  
 filteredData2 = reactive({  
 df = data %>%  
 filter(SchemeType == substr(input$SchemeForPlot, start = 1, stop = 4)) %>%  
 filter(LocalAuthorityName == input$CityForPlot)  
 return(df)  
 })  
   
 # Plot the bar chart  
 output$myplot1 = renderPlot({  
 if (input$SchemeForPlot == "Overall"){  
 g1 = ggplot(data = data %>% filter(SchemeType == "FHIS"),   
 aes(x = RatingValue)) +   
 geom\_bar(fill = "cadetblue1", col = "black") +  
 geom\_text(stat='count', aes(label=..count..), vjust=0.5) +  
 ggtitle("FHIS") +  
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=9),   
 axis.text.y = element\_text(face="bold", color="#993333", size=12),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14),  
 plot.title = element\_text(size = 22, face = "bold"))  
 g2 = ggplot(data = data %>% filter(SchemeType == "FHRS"), aes(x = RatingValue)) +  
 geom\_bar(fill = "cadetblue1", col = "black") +  
 geom\_text(stat='count', aes(label=..count..), vjust=0.5) +  
 ggtitle("FHRS") +  
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=9),   
 axis.text.y = element\_text(face="bold", color="#993333", size=12),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14),  
 plot.title = element\_text(size = 22, face = "bold"))  
 ggpubr::ggarrange(g1, g2, nrow = 2)  
 }  
 else{  
 ggplot(data = filteredData2(), aes(x = RatingValue)) +   
 geom\_bar(fill = "cadetblue1", col = "black") +  
 geom\_text(stat='count', aes(label=..count..), vjust=0.5) +  
 ggtitle(paste0("Rating Value distribution in ",   
 input$CityForPlot)) +  
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=12),   
 axis.text.y = element\_text(face="bold", color="#993333", size=12),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14),  
 plot.title = element\_text(size = 18, face = "bold"))  
 }  
 })  
   
 # Plot2 - the pie chart for business type share  
 # Select a city  
 data\_Pie\_each\_city = reactive({  
 df = data %>%  
 filter(LocalAuthorityName == input$CityForPlot2) %>%  
 group\_by(BusinessType) %>%  
 summarise(Total = n()) %>%  
 ungroup()  
 return(df)  
 })  
 # Build a data frame to store the total amount for each business type  
 data\_Pie\_all = data %>%  
 group\_by(BusinessType) %>%  
 summarise(Total = n()) %>%  
 ungroup()  
 # Plot a pie chart  
 output$myplot3 = renderPlot({  
 if (input$CityForPlot2 == "Overall"){  
 ggplot(data = data\_Pie\_all, aes(x = "", y = Total, fill = BusinessType)) +  
 geom\_bar(stat = "identity", col = "black") +  
 coord\_polar("y") +  
 ggtitle(paste0("Overall Business Type distribution")) +  
 # scale\_fill\_brewer("Blues") +  
 theme\_minimal() +  
 theme(  
 axis.title.x = element\_blank(),  
 axis.title.y = element\_blank(),  
 panel.border = element\_blank(),  
 panel.grid = element\_blank(),  
 axis.ticks = element\_blank(),  
 plot.title = element\_text(size=22, face="bold"),  
 legend.title = element\_text(size = 10),  
 legend.text = element\_text(size = 10)  
 ) +  
 theme(axis.text.x=element\_blank()) +  
 geom\_text(aes(label = paste0(round((Total/sum(Total))\*100, 2), "%")),  
 size = 3, position = position\_stack(vjust = 0.5))  
 }  
 else{  
 ggplot(data = data\_Pie\_each\_city(), aes(x = "", y = Total, fill = BusinessType)) +  
 geom\_bar(stat = "identity", col = "black") +  
 coord\_polar("y") +  
 ggtitle(paste0("Business Type distribution in ", input$CityForPlot2)) +  
 # scale\_fill\_brewer("Blues") +  
 theme\_minimal() +  
 theme(  
 axis.title.x = element\_blank(),  
 axis.title.y = element\_blank(),  
 panel.border = element\_blank(),  
 panel.grid = element\_blank(),  
 axis.ticks = element\_blank(),  
 plot.title = element\_text(size=22, face="bold"),  
 legend.title = element\_text(size = 10),  
 legend.text = element\_text(size = 10)  
 ) +  
 theme(axis.text.x=element\_blank()) +  
 geom\_text(aes(label = paste0(round((Total/sum(Total))\*100, 2), "%")),  
 size = 3, position = position\_stack(vjust = 0.5))  
 }  
 })  
   
 # Plot3 -  
   
 filteredData3 = reactive({  
 df = data.frame(Score = as.integer(data\_no\_missing\_score[, input$ScorePlot])) %>%  
 arrange(Score)  
 return(df)  
 })  
 output$myplot6 = renderPlotly({  
 score.plot = ggplot(data = filteredData3(), aes(x = Score)) +  
 geom\_bar(fill = "cadetblue1", col = "black") +  
 ggtitle(paste0(input$ScorePlot, " Score Distribution")) +   
 theme(axis.text.x = element\_text(face="bold", color="#993333", size=12),   
 axis.text.y = element\_text(face="bold", color="#993333", size=12),  
 axis.title.x = element\_text(face="bold", color="black", size=14),  
 axis.title.y = element\_text(face="bold", color="black", size=14),  
 plot.title = element\_text(size = 20, face = "bold"))  
 ggplotly(score.plot)  
 })  
   
   
   
 # Display our data  
 output$mydata = DT::renderDataTable(  
 DT::datatable(data)  
 )  
}