library(shiny)

library(shinydashboard)

library(DT)

library(dplyr)

library(ggplot2)

library(bslib)

library(shinyWidgets)

library(shinyjs)

# UI Definition

ui <- dashboardPage(

dashboardHeader(title = span("🎬 Movie Dataset Analysis App", style = "color:white")),

dashboardSidebar(

sidebarMenu(

menuItem("Upload & Edit", tabName = "upload", icon = icon("upload")),

menuItem("Summary & Stats", tabName = "summary", icon = icon("chart-line")),

menuItem("Charts", tabName = "charts", icon = icon("chart-pie")),

menuItem("Export", tabName = "export", icon = icon("file-export"))

)

),

dashboardBody(

useShinyjs(),

theme = bs\_theme(bootswatch = "cosmo"),

tabItems(

# Upload & Edit Tab

tabItem(tabName = "upload",

fluidRow(

box(width = 6, status = "primary", solidHeader = TRUE, title = "Upload Dataset",

fileInput("file", "Choose a Movie CSV/Excel File", accept = c(".csv", ".xlsx"))

),

box(width = 12, status = "warning", solidHeader = TRUE, title = "Editable Data Table",

DTOutput("editableTable"))

)

),

# Summary Tab

tabItem(tabName = "summary",

fluidRow(

box(width = 12, status = "info", solidHeader = TRUE, title = "Summary Statistics",

verbatimTextOutput("summaryOutput"))

)

),

# Charts Tab

tabItem(tabName = "charts",

fluidRow(

box(width = 4, status = "primary", solidHeader = TRUE, title = "Chart Settings",

selectInput("chartType", "Select Chart Type",

choices = c("Bar Chart", "Line Chart", "Pie Chart", "Heatmap")),

pickerInput("selectedGenre", "Filter by Genre",

choices = NULL, multiple = TRUE, options = list(`actions-box` = TRUE))

),

box(width = 8, status = "success", solidHeader = TRUE, title = "Visualization",

plotOutput("chartOutput", height = "400px"))

)

),

# Export Tab

tabItem(tabName = "export",

fluidRow(

box(width = 6, status = "success", solidHeader = TRUE, title = "Download Your Data",

downloadButton("downloadCSV", "📥 Download as CSV")

)

)

)

)

)

)

# Server Logic

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

# Reactive Data Store

movie\_data <- reactiveVal(data.frame())

# Load CSV or Excel File

observeEvent(input$file, {

ext <- tools::file\_ext(input$file$name)

df <- if (ext == "csv") {

read.csv(input$file$datapath)

} else if (ext == "xlsx") {

readxl::read\_excel(input$file$datapath)

}

req(all(c("Title", "Genre", "Year", "Revenue") %in% colnames(df)))

movie\_data(df)

updatePickerInput(session, "selectedGenre", choices = unique(df$Genre), selected = unique(df$Genre))

})

# Editable Table

output$editableTable <- renderDT({

datatable(movie\_data(), editable = TRUE, options = list(pageLength = 10))

})

# Summary Statistics

output$summaryOutput <- renderPrint({

summary(movie\_data())

})

# Chart Rendering

output$chartOutput <- renderPlot({

df <- movie\_data()

if (!is.null(input$selectedGenre)) {

df <- df[df$Genre %in% input$selectedGenre, ]

}

if (input$chartType == "Bar Chart") {

ggplot(df, aes(x = Genre, fill = Genre)) + geom\_bar() + theme\_minimal()

} else if (input$chartType == "Line Chart") {

ggplot(df, aes(x = Year, y = Revenue, color = Genre)) + geom\_line() + theme\_minimal()

} else if (input$chartType == "Pie Chart") {

pie(table(df$Genre), col = rainbow(length(unique(df$Genre))))

} else if (input$chartType == "Heatmap") {

ggplot(df, aes(x = Genre, y = Year, fill = Revenue)) +

geom\_tile(color = "white") + theme\_minimal()

}

})

# Download as CSV

output$downloadCSV <- downloadHandler(

filename = function() {"movie\_data\_export.csv"},

content = function(file) {

write.csv(movie\_data(), file, row.names = FALSE)

}

)

}

# Run the App

shinyApp(ui, server)