# UI

This code is using the Shiny library in R to create a web application. The application has two main tabs: "Data" and "Monte Carlo Simulation".

The first line imports the "shinythemes" library, which allows the user to change the theme of the application. In this case, the theme is set to "superhero" using the line: theme=shinytheme("superhero").

The "Data" tab allows the user to upload a data file using the fileInput function, which is displayed in the sidebar panel. The main panel of the "Data" tab contains a tabsetPanel with three tabs: "Graph", "Data", and "Normally test". The first tab, "Graph", displays a plot of the uploaded data using the plotOutput function. The second tab, "Data", displays summary statistics of the data using verbatimTextOutput function and the third tab is for testing the normality of the data using verbatimTextOutput function.

The "Monte Carlo Simulation" tab contains a sidebar panel with text inputs for the initial investment and number of simulations, as well as an action button for running the simulation. The main panel of this tab contains a single tab, "Result of Simulation of Monte Carlo", which displays the results of the simulation using verbatimTextOutput function and also interpretation of the results.

The overall layout of the application is created using the fluidPage function and the navbarPage function, which creates a navigation bar at the top of the page. The layout is also defined by sidebarLayout and mainPanel functions.

# Server

This code defines several functions for calculating Value at Risk (VaR) using Monte Carlo simulation. The VaR is a statistical measure of the potential loss on a portfolio of investments over a given time period with a given confidence level.

The first function VaR.MC takes in four arguments: x which is the input data, Wo which is the initial investment, n which is the number of simulations, and alpha which is the confidence level. It first converts the input data x into a matrix and finds the mean and standard deviation of the data. Then it simulates n returns using the normal distribution with mean and standard deviation of the data. Using the quantile function it finds the VaR at the given confidence levels and multiples it with the initial investment and square root of the number of days in the given time period. Finally it returns the VaR values as a matrix with column names representing the confidence level.

The second function VaR\_MC takes in three arguments: x which is the input data, m which is the number of simulations, Wo which is the initial investment. This function uses the first function 'VaR.MC' and runs the simulation 'm' times and storing the VaR values for each run in a matrix. It then finds the average of the VaR values for each confidence level and returns it as a matrix with column names representing the confidence level.

Finally, the server function is using the Shiny's functionality to render the output for the corresponding inputs. It is reading the data file uploaded by the user and then it is using the 'VaR\_MC' function to calculate VaR and showing the results, graph, data and descriptive statistics of the data.

# VaR

This code defines two R functions for calculating Value at Risk (VaR) using Monte Carlo simulation. The VaR is a statistical measure of the potential loss on a portfolio of investments over a given time period with a given confidence level.

The first function VaR.MC takes in five arguments: x which is the input data, Wo which is the initial investment, n which is the number of simulations, alpha which is the confidence level and k which is the square root of the number of days in the given time period. It first converts the input data x into a matrix and finds the mean and standard deviation of the data. Then it simulates n returns using the normal distribution with mean and standard deviation of the data. Using the quantile function it finds the VaR at the given confidence levels and multiples it with the initial investment and square root of the number of days in the given time period. Finally it returns the VaR values as a matrix with column names representing the confidence level.

The second function VaR\_MC takes in three arguments: x which is the input data, m which is the number of simulations, Wo which is the initial investment. This function uses the first function 'VaR.MC' and runs the simulation 'm' times and storing the VaR values for each run in a matrix. It then finds the average of the VaR values for each confidence level and returns it as a matrix with column names representing the confidence level. It also prints out the VaR values for each confidence level for each simulation and the average VaR values for each confidence leve