```r
# -------------------------------------------------------------------------
# Required Libraries
# -------------------------------------------------------------------------
if (!require(ggplot2)) install.packages("ggplot2")
if (!require(tseries)) install.packages("tseries")
if (!require(forecast)) install.packages("forecast")
library(ggplot2)
library(tseries)
library(forecast)

# -------------------------------------------------------------------------
# 1. Data Retrieval and Preprocessing (15 Marks)
# -------------------------------------------------------------------------

# Load Dataset
walmart_data <- read.csv("Walmart Stock Price History.csv")

# Clean Numeric Columns
walmart_data$Price <- as.numeric(gsub(",", "", walmart_data$Price))
walmart_data$Open <- as.numeric(gsub(",", "", walmart_data$Open))
walmart_data$High <- as.numeric(gsub(",", "", walmart_data$High))
walmart_data$Low <- as.numeric(gsub(",", "", walmart_data$Low))

# Convert 'Date' to Date Format
walmart_data$Date <- as.Date(walmart_data$Date, format = "%d/%m/%Y")

# Remove Missing Values and Sort by Date
walmart_data <- walmart_data[!is.na(walmart_data$Price), ]
walmart_data <- walmart_data[order(walmart_data$Date), ]

# Plot Time Series Data
ggplot(data = walmart_data, aes(x = Date, y = Price)) +
  geom_line(color = "blue") +
  labs(title = "Walmart Stock Prices Over Time", x = "Date", y = "Closing Price")

# -------------------------------------------------------------------------
# 2. Preliminary Analysis and Model Identification (20 Marks)
# -------------------------------------------------------------------------

# Log Transformation for Variance Stabilization
walmart_data$Log_Price <- log(walmart_data$Price)

# First-order Differencing for Stationarity
Diff_Log_Price <- diff(walmart_data$Log_Price)

# Adjust Dataset After Differencing
walmart_data <- walmart_data[-1, ]  # Remove the first row
walmart_data$Diff_Log_Price <- Diff_Log_Price

# ADF Test for Stationarity
adf_test <- adf.test(walmart_data$Diff_Log_Price, alternative = "stationary")
print(adf_test)

# Plot ACF and PACF to Identify ARIMA Parameters
acf(walmart_data$Diff_Log_Price, main = "ACF of Differenced Log Prices")
pacf(walmart_data$Diff_Log_Price, main = "PACF of Differenced Log Prices")

# -------------------------------------------------------------------------
# 3. ARIMA Model Fitting and Comparison (30 Marks)
# -------------------------------------------------------------------------

# Fit ARIMA Models
model1 <- Arima(walmart_data$Log_Price, order = c(1, 1, 1))
model2 <- Arima(walmart_data$Log_Price, order = c(2, 1, 2))
```

```r
# Compare Models Using AIC and BIC
aic_values <- c(AIC(model1), AIC(model2))
bic_values <- c(BIC(model1), BIC(model2))
comparison <- data.frame(Model = c("ARIMA(1,1,1)", "ARIMA(2,1,2)"), AIC = aic_values, BIC
= bic_values)
print(comparison)

# Forecast Using Best Model (ARIMA(2,1,2))
forecast_model <- forecast(model2, h = 20)
autoplot(forecast_model) +
  labs(title = "Forecast Using ARIMA(2,1,2)", x = "Time", y = "Log of Closing Prices")

# --------------------------------------------------------------------------
# 4. Residual Diagnostics and Assumption Validation (20 Marks)
# --------------------------------------------------------------------------

# Residual Diagnostics
checkresiduals(model2)

# Ljung-Box Test for Randomness of Residuals
ljung_box_test <- Box.test(residuals(model2), lag = 10, type = "Ljung-Box")
print(ljung_box_test)

# Residual Plots
hist(residuals(model2), main = "Residuals Histogram", xlab = "Residuals")
qqnorm(residuals(model2))
qqline(residuals(model2))

# --------------------------------------------------------------------------
# 5. Model Evaluation and Performance Comparison (15 Marks)
# --------------------------------------------------------------------------

# Calculate Forecast Accuracy Metrics
accuracy_metrics <- accuracy(forecast_model)
print(accuracy_metrics)

# Plot Actual vs Predicted Values
autoplot(forecast_model) +
  labs(title = "Actual vs Predicted Data", x = "Time", y = "Log of Closing Prices")
```