**Exercise 1: Simple Linear Regression for Sales Forecasting**

**Problem Statement:**

Predict future sales based on past sales data.

**Algorithm:**

1. Load the sales data.

2. Split the data into training and test sets.

3. Build a simple linear regression model where `Sales` is the dependent variable and `Time` is the independent variable.

4. Predict future sales using the model.

5. Plot the actual vs predicted sales.

**R Code:**

# Step 1: Load data

sales\_data <- data.frame(Time = 1:10, Sales = c(100, 120, 130, 140, 150, 160, 180, 190, 210, 220))

# Step 2: Build the linear model

model <- lm(Sales ~ Time, data = sales\_data)

# Step 3: Predict future sales

future\_time <- data.frame(Time = 11:15)

predicted\_sales <- predict(model, newdata = future\_time)

# Step 4: Output the predictions

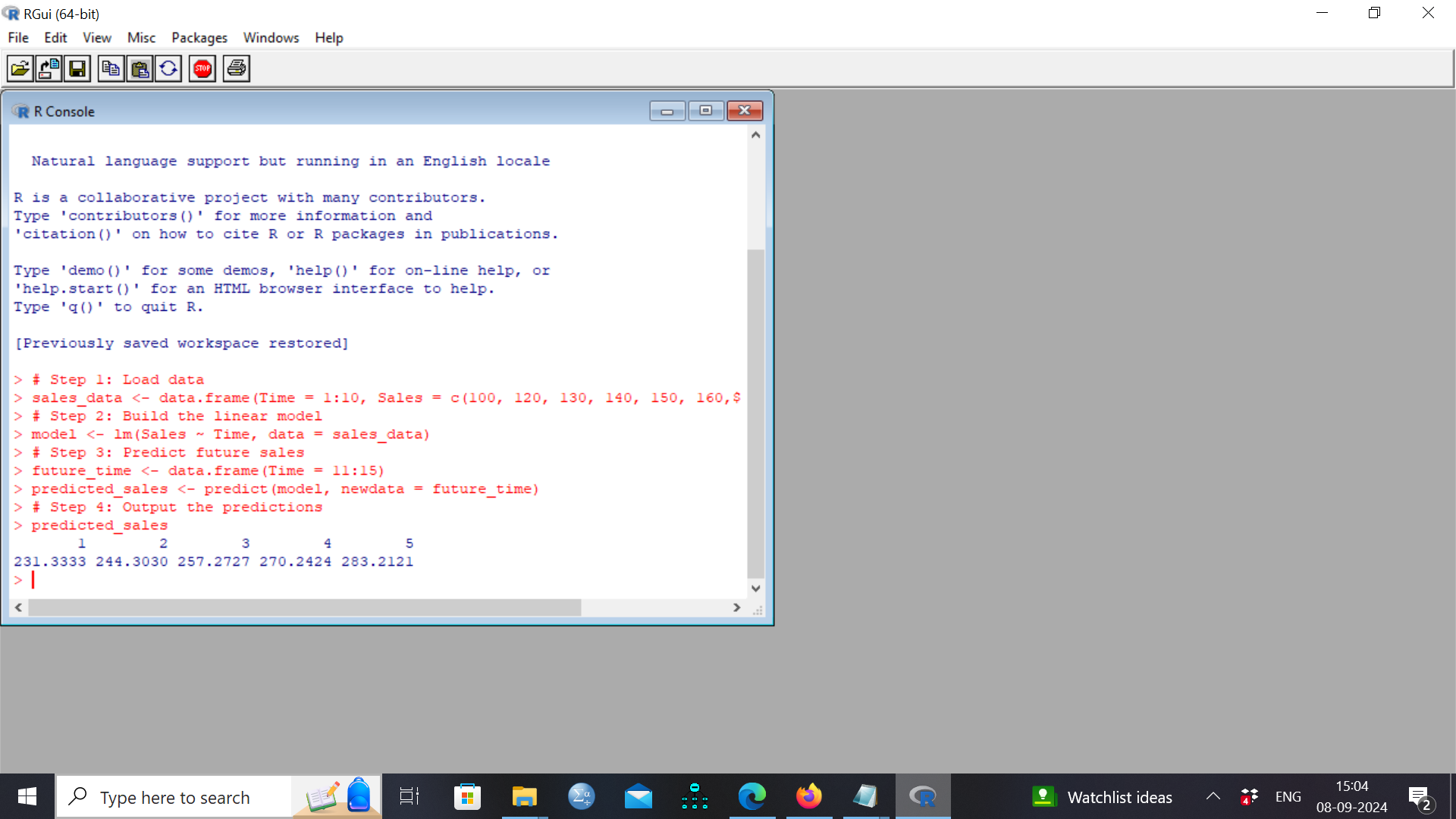
predicted\_sales

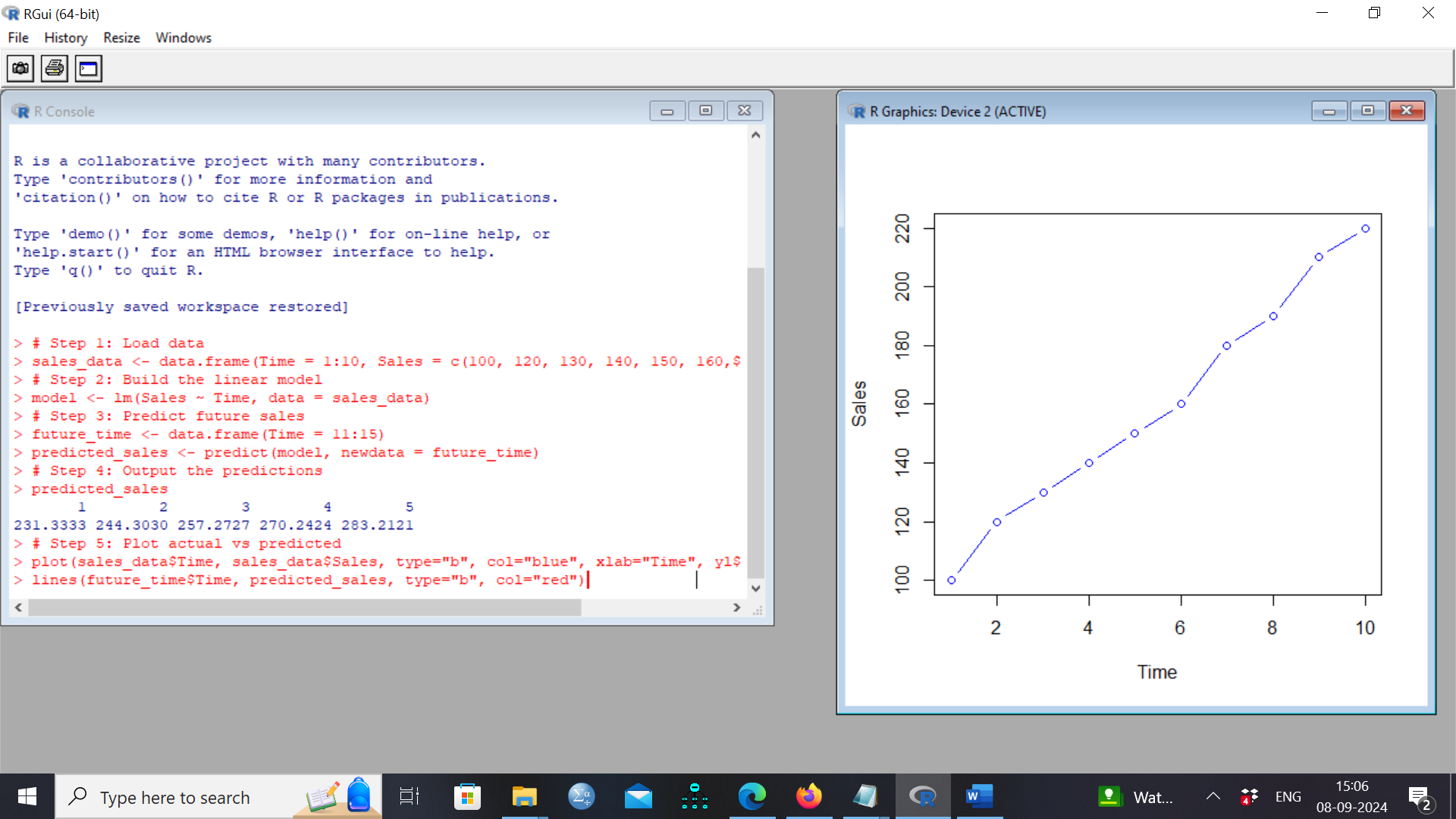
# Step 5: Plot actual vs predicted

plot(sales\_data$Time, sales\_data$Sales, type="b", col="blue", xlab="Time", ylab="Sales")

lines(future\_time$Time, predicted\_sales, type="b", col="red")

**Output:**





**Exercise 2: Moving Average for Sales Smoothing**

**Problem Statement:**

Use a moving average to smooth monthly sales data.

**Algorithm:**

1. Load the monthly sales data.

2. Define a function for the moving average.

3. Apply the moving average function to the sales data.

4. Plot the original and smoothed data.

**R Code:**

# Step 1: Load data

sales\_data <- c(120, 130, 135, 145, 150, 155, 160, 165, 170, 175)

# Step 2: Define moving average function

moving\_avg <- function(x, n=3){

filter(x, rep(1/n, n), sides=2)

}

# Step 3: Apply moving average

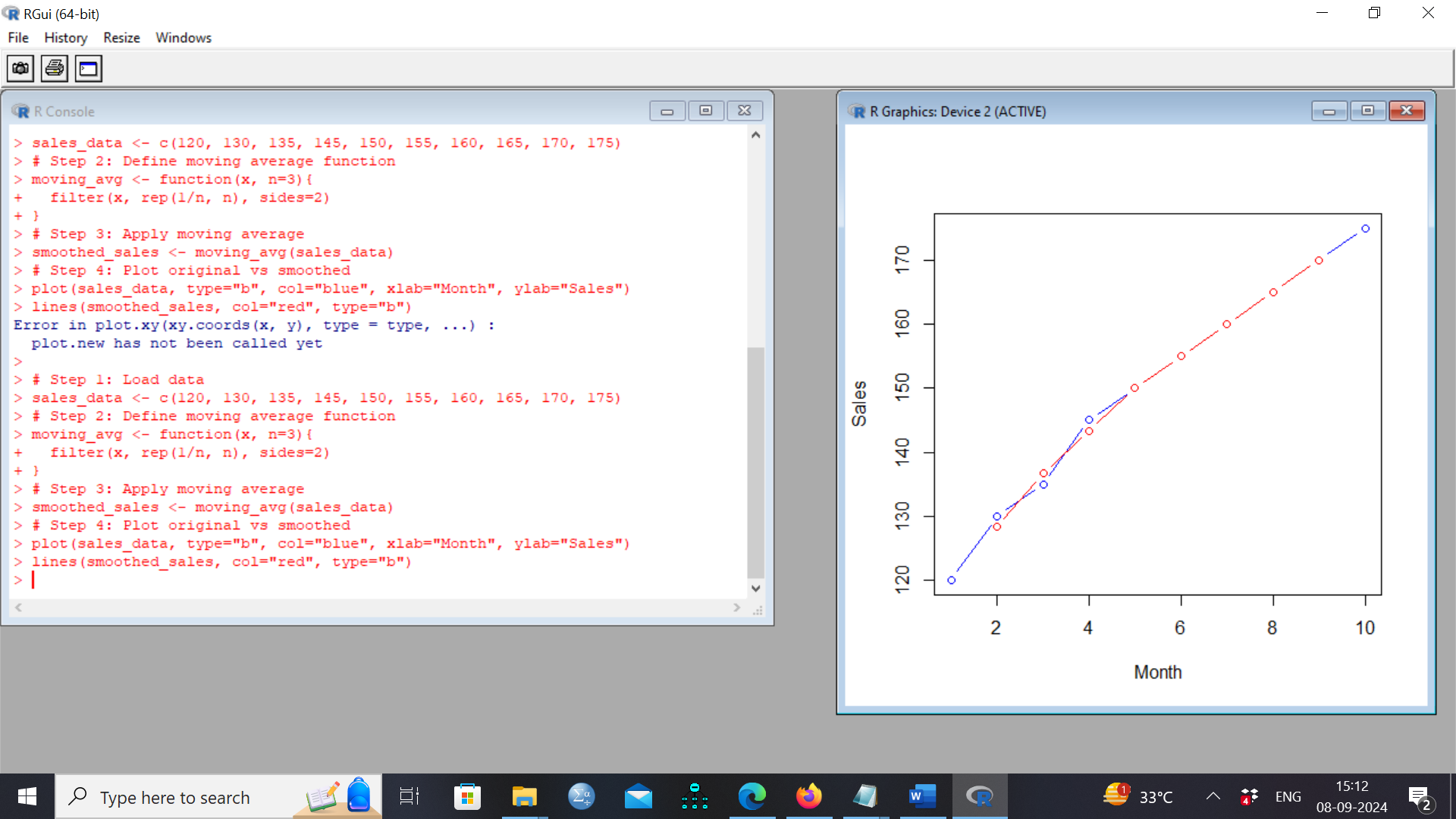
smoothed\_sales <- moving\_avg(sales\_data)

# Step 4: Plot original vs smoothed

plot(sales\_data, type="b", col="blue", xlab="Month", ylab="Sales")

lines(smoothed\_sales, col="red", type="b")

**Output:**



**Exercise 3: Exponential Smoothing for Sales Forecasting**

**Problem Statement:**

Use exponential smoothing to forecast future sales.

**Algorithm:**

1. Load the sales data.

2. Apply exponential smoothing using the `HoltWinters` function.

3. Forecast future sales.

4. Plot the original data and the forecast.

**R Code:**

# Step 1: Load data with yearly frequency

sales\_data <- ts(c(110, 130, 150, 170, 190, 210), frequency = 1)

# Step 2: Apply exponential smoothing without seasonality

sales\_model <- HoltWinters(sales\_data, gamma=FALSE)

# Step 3: Load forecast package

install.packages("forecast")

library(forecast)

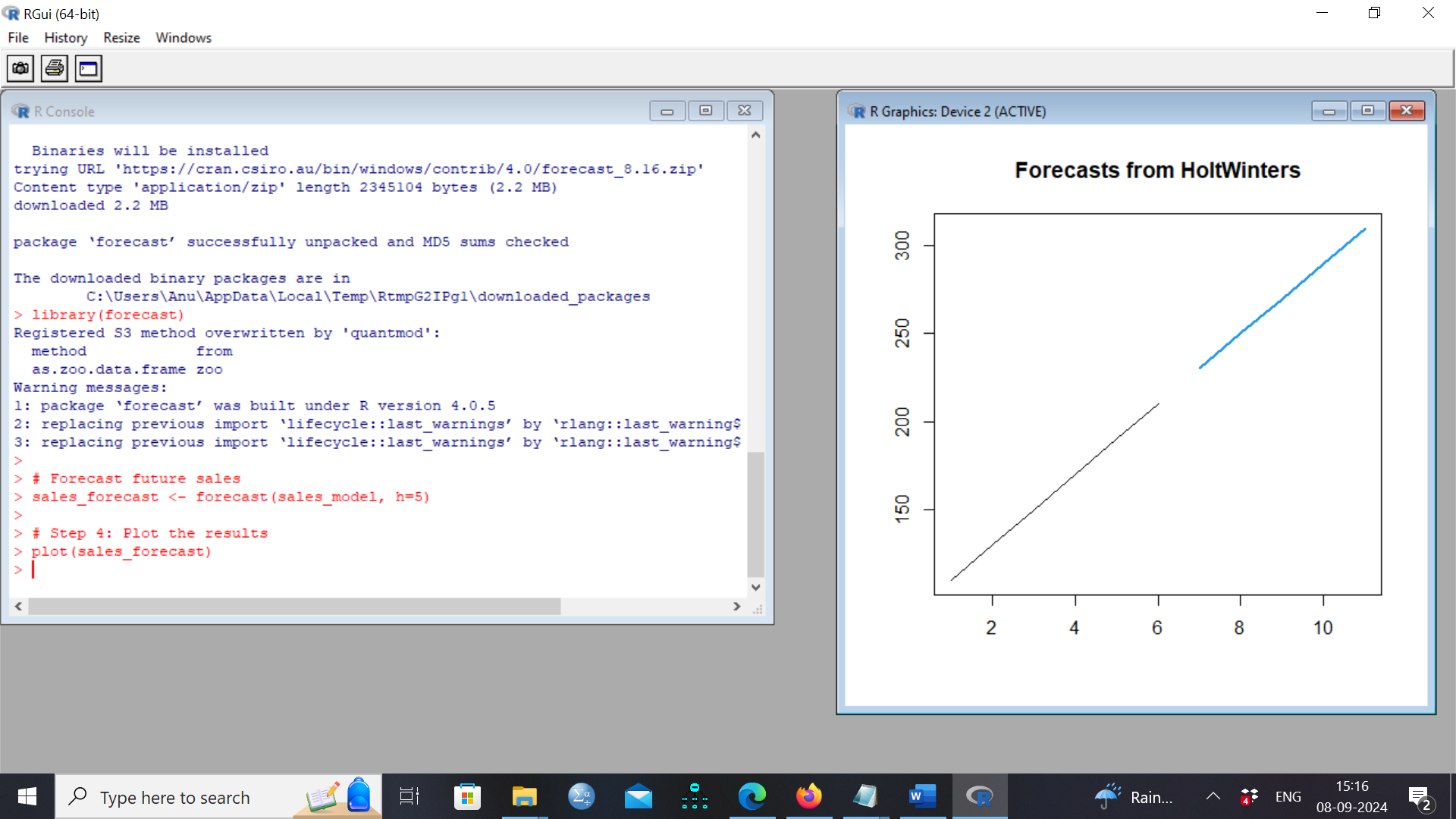
# Forecast future sales

sales\_forecast <- forecast(sales\_model, h=5)

# Step 4: Plot the results

plot(sales\_forecast)

**Output:**



**Exercise 4: Seasonal Decomposition of Time Series (STL)**

**Problem Statement:**

Decompose a sales time series into trend, seasonal, and irregular components.

**Algorithm:**

1. Load the sales data with seasonal patterns.

2. Apply STL decomposition.

3. Visualize the decomposed components (trend, seasonality, and residual).

**R Code:**

# Step 1: Load data with more observations

sales\_data <- ts(c(100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650), frequency=4)

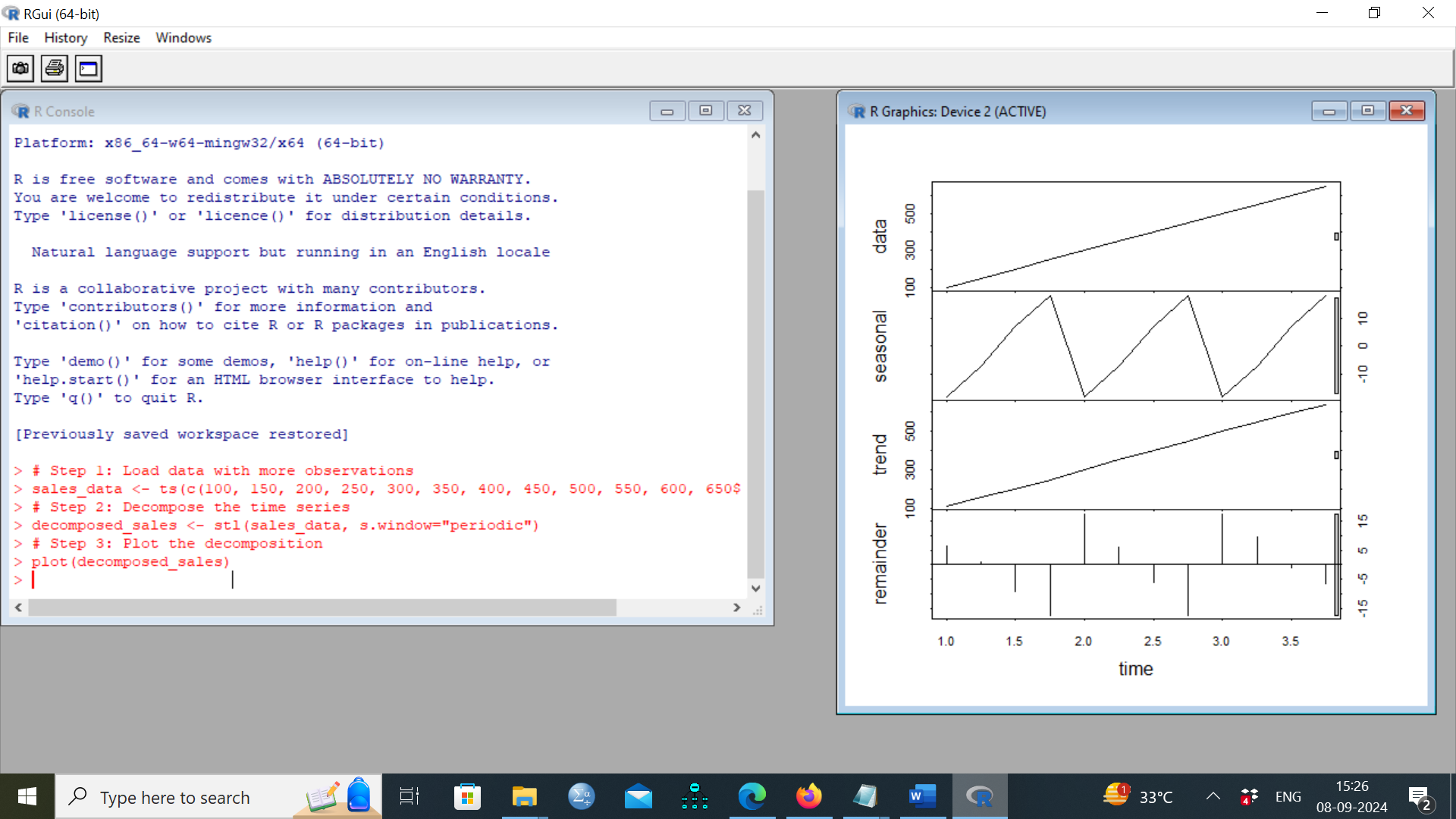
# Step 2: Decompose the time series

decomposed\_sales <- stl(sales\_data, s.window="periodic")

# Step 3: Plot the decomposition

plot(decomposed\_sales)

**Output:**



**Exercise 5: ARIMA Model for Sales Forecasting**

**Problem Statement:**

Forecast sales using the ARIMA model.

**Algorithm:**

1. Load the sales data.

2. Check the stationarity of the data.

3. Fit an ARIMA model to the data.

4. Forecast future sales using the fitted ARIMA model.

5. Plot the forecast results.

**R Code:**

# Install and load the forecast package

install.packages("forecast") # Run this line only if you haven't installed the package

library(forecast)

# Step 1: Load data with more observations and decompose the time series

sales\_data <- ts(c(100, 150, 200, 250, 300, 350, 400, 450, 500, 550, 600, 650), frequency=4)

decomposed\_sales <- stl(sales\_data, s.window="periodic")

plot(decomposed\_sales)

# Step 2: Load new data and fit ARIMA model

sales\_data <- ts(c(150, 160, 170, 180, 190, 200, 210, 220), frequency=12)

sales\_arima <- auto.arima(sales\_data) # Fit the ARIMA model

# Step 3: Forecast future sales

sales\_forecast <- forecast(sales\_arima, h=6) # Forecast for 6 periods ahead

# Step 4: Plot the forecast

plot(sales\_forecast)

**Output:**

