C# AUTOMATION TESTING

Coding:

```
using System;
using System.Collections.Generic;
using System.Net;
using System.Text.RegularExpressions;

namespace WebPageWordCounter
{
    public class WebPageWordCounter
    {
        public static void Main(string[] args)
        {
            string url = "https://en.wikipedia.org/wiki/Test_automation"; // URL of the webpage to
navigate

            string pageContent = GetPageContent(url); // Get the content of the webpage

            if (pageContent != null)
            {
                Dictionary<string, int> wordCount = ExtractWordCount(pageContent);

                foreach (var pair in wordCount)
                {
                    Console.WriteLine($"{pair.Key}: {pair.Value}");
                }
            }
            else
            {
                Console.WriteLine("Failed to retrieve page content.");
            }
        }

        private static string GetPageContent(string url)
        {
            try
            {
                using (WebClient client = new WebClient())
                {
                    return client.DownloadString(url);
```

```
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error: {ex.Message}");
            return null;
        }
    }

    private static Dictionary<string, int> ExtractWordCount(string content)
    {
        // Remove HTML tags and special characters
        string plainText = Regex.Replace(content, @"<\/?\w+.*?>|[:\[\]]", "");

        // Split text into words
        string[] words = plainText.Split(new char[] { ' ', '\n', '\r', '.', ',', '-', ';', '(', ')' },
StringSplitOptions.RemoveEmptyEntries);

        // Count words
        Dictionary<string, int> wordCount = new Dictionary<string, int>();
        foreach (string word in words)
        {
            string cleanedWord = word.Trim().ToLower();
            if (!wordCount.ContainsKey(cleanedWord))
            {
                wordCount[cleanedWord] = 1;
            }
            else
            {
                wordCount[cleanedWord]++;
            }
        }
        return wordCount;
    }
  }
}
```