

SAUCE DEMO – TESTARE AUTOMATĂ

PROIECT LABORATOR TAS

Autor: **Andreea – Maria CIGHI**

Conducător științific: **Dan GOTA**

Cuprins

1	INTRODUCERE.....	3
1.1	DESCRIEREA PROIECTULUI.....	3
1.2	MEDIUL DE DEZVOLTARE	3
1.3	INSTRUMENTELE DE TESTARE FOLOSITE.....	3
1.3.1	<i>MSTest</i>	3
1.3.2	<i>Selenium</i>	4
2	TESTAREA INTERFEȚEI	6
3	SCENARIILE DE TESTARE	6
3.1	TEST CASE 1: AUTENTIFICARE CU CREDENȚIALE VALIDE.....	6
3.2	TEST CASE 2: AUTENTIFICARE CU CREDENȚIALE INVALIDE	8
3.3	TEST CASE 3 : DELOGARE DE PE SAUCEDEMO.....	9
3.4	TEST CASE 4: SORTARE PRODUSE DUPĂ PREȚ CRESCĂTOR	10
3.5	TEST CASE 5: SORTARE PRODUSE DUPĂ PREȚ DESCRESCĂTOR	12
3.6	TEST CASE 6: SORTARE PRODUSE DUPĂ NUME CRESCĂTOR	13
3.7	TEST CASE 7: SORTARE PRODUSE DUPĂ NUME DESCRESCĂTOR	15
3.8	TEST CASE 8: VERIFICARE TITLU PAGINĂ	16
3.9	TEST CASE 9: VERIFICARE PREȚ PRODUS	17
3.10	TEST CASE 10: VERIFICARE IMAGINE PRODUS.....	18
3.11	TEST CASE 11: VERIFICARE DETALII PRODUS	19
3.12	TEST CASE 12: VERIFICARE DESCRIERE PRODUS	20
3.13	TEST CASE 13: ADĂUGARE PRODUS ÎN COȘ	21
3.14	TEST CASE 14: ȘTERGERE PRODUS DIN COȘ	23
3.15	TEST CASE 15: CUMPĂRARE PRODUS FĂRĂ CHECKOUT	24
3.16	TEST CASE 16: CUMPĂRARE PRODUS FĂRĂ INFORMAȚII DE LIVRARE	25
3.17	TEST CASE 17: CUMPĂRARE PRODUS	26
3.18	TEST CASE 18: NAVIGAREA PE PAGINA ABOUT	29
3.19	TEST CASE 19: MĂSURARE TIMP ÎNCĂRCARE PAGINĂ	30
3.20	TEST CASE 20: MĂSURARE TIMP ADĂUGARE PRODUS ÎN COȘ	30

1 Introducere

1.1 Descrierea proiectului

Pentru acest proiect, am ales să testez site-ul **SauceDemo**. Acesta este un site destinat să demonstreze și să ilustreze abilitățile funcționale ale platformei Sauce Labs, care oferă servicii de testare automată în cloud.

Proiectul este conceput pentru a testa funcționalitățile unui site de comerț electronic simplu, oferind o experiență practică pentru testarea automată a interfeței de utilizator și a funcționalităților asociate.

1.2 Mediul de dezvoltare

Mediul de dezvoltare folosit pentru a rula testele este **Visual Studio Community**, iar limbajul de programare este **C#**.

Visual Studio este un mediu de dezvoltare integrat (IDE) dezvoltat de Microsoft, utilizat pentru dezvoltarea de software pentru diferite platforme și limbaje de programare. Acesta este un instrument puternic și cuprinzător pentru dezvoltarea software, adaptabil la diverse nevoi de dezvoltare și oferind un set bogat de funcționalități pentru programatori.

1.3 Instrumentele de testare folosite

1.3.1 MSTest

MSTest este un cadru (framework) de testare integrat în platforma Microsoft .NET, folosit pentru a realiza teste unitare și de integrare în cadrul aplicațiilor dezvoltate în limbajele de programare .NET, precum C# și Visual Basic. Acest cadru face parte din suita de instrumente de testare ale platformei .NET și este integrat în mediul de dezvoltare Visual Studio.

Aspectele cheie ale **MSTest**, care m-au ajutat la testarea mai ușoară a interfeței web, sunt:

1. Atribuții de Testare

- Testele în MSTest sunt definite prin intermediul atributelor specifice, precum **[TestMethod]** pentru identificarea unei metode ca fiind un test.

2. Organizarea Testelor:

- Testele sunt grupate în clase, iar fiecare clasă poate conține mai multe metode de testare.

3. Dispozitiv de testare

- Clasele pot conține metode de inițializare și curățare ([**ClassInitialize**], [**ClassCleanup**], [**TestInitialize**], [**TestCleanup**]) pentru a stabili și elibera resurse înainte și după rularea testelor. Eu am folosit [**TestInitialize**] pentru a pregăti mediul de testare înainte de a rula fiecare test și [**TestCleanup**] pentru a elibera resursele alocate pentru instanța WebDriver după terminarea testelor.
4. **Aserții:**
 - Folosind metode precum **Assert.AreEqual()**, **Assert.IsTrue()**, **Assert.IsFalse()**, programatorii pot valida rezultatele testelor și pot verifica condițiile așteptate.
 5. **Categorizare a Testelor**
 - Posibilitatea de a organiza și executa teste în funcție de categorii, prin intermediul atributelor precum [**TestCategory**].
 6. **Teste de Performanță:**
 - Posibilitatea de a realiza teste de performanță, prin utilizarea atributelor precum [**TestCategory("Performance")**].

1.3.2 Selenium

Selenium este un cadru (framework) de testare automată pentru aplicații web, utilizat pentru a automatiza interacțiunile cu un browser web. Selenium oferă un set de instrumente și biblioteci care permit dezvoltatorilor și testatorilor să scrie scripturi de testare automate într-un mod eficient și fiabil.

Câteva aspecte cheie legate de Selenium :

1. **WebDriver :**

WebDriver este componenta principală a Selenium care furnizează o interfață simplificată pentru a interacționa cu browserele web. Există diferite implementări WebDriver pentru diferite browsere, cum ar fi ChromeDriver, FirefoxDriver, EdgeDriver, etc.

2. **Scripturi de Testare în Limbajul Seleniu:**

Selenium suportă mai multe limbaje de programare, inclusiv Java, Python, C#, Ruby, și JavaScript. Aceasta oferă dezvoltatorilor flexibilitate în alegerea limbajului preferat pentru a scrie teste automate.

3. **Element Locators:**

Selenium oferă diverse strategii pentru a localiza elementele pe o pagină web, inclusiv identificarea după ID, nume, clasă, etichetă, CSS selector sau XPath.

4. **Interacțiuni cu Elementele:**

Selenium oferă metode pentru a efectua acțiuni precum clic, introducerea de texte, trimiterea de formulare, manipularea ferestrelor pop-up și gestionarea cookie-urilor.

5. Wait Strategies:

Pentru a trata sincronizarea între aplicație și browser, Selenium oferă strategii de așteptare (wait) pentru a se asigura că elementele sunt încărcate complet înainte de a efectua acțiuni.

6. Testare Multi-Browser:

Selenium permite testarea pe mai multe browsere, asigurând astfel că aplicația funcționează corect pe o varietate de medii.

7. Headless Browsing:

Posibilitatea de a rula teste în mod headless, fără a deschide efectiv interfața vizuală a browserului.

8. Selenium Grid:

Permite testarea în paralel pe diferite dispozitive și browsere utilizând Selenium Grid. Acesta oferă o soluție scalabilă pentru testarea distribuită.

9. Suport pentru Testing Frameworks:

Selenium se integrează cu diverse cadre de testare, cum ar fi TestNG, JUnit, NUnit, PyTest, etc., pentru a organiza și executa teste într-un mod structurat.

10. Suport pentru Mobile Testing:

Selenium poate fi utilizat în combinație cu Appium pentru a efectua teste automate pe aplicații mobile.

11. Capturi de Ecran și Rapoarte:

Selenium permite realizarea de capturi de ecran în timpul testelor, iar acestea pot fi integrate în rapoarte de testare pentru o analiză detaliată.

12. Integrare cu Sisteme CI/CD:

Selenium poate fi utilizat în cadrul proceselor de integrare continuă (CI) și distribuție continuă (CD), asigurând testarea automată la fiecare actualizare de cod.

2 Testarea interfeței

Pentru a realiza testarea automată a site-ului **SauceDemo** folosind **Selenium** și **MSTest**, am urmat următorii pași esențiali :

1. Instalarea Selenium WebDriver:

Pentru a instala pachetul 'Selenium.WebDriver' am utilizat gestionarul de pachete NuGet din cadrul mediului de dezvoltare VisualStudio.

2. Instalarea Driver-ului Browser-ului:

Am instalat driver-ul corespunzător pentru browser-ul pe care am droit să-l utilizez(ChromeDriver pentru GoogleChrome), având grijă ca versiunile dintre driver și browser să coincidă. Am utilizat versiunea 120.0.6099.225 pe 64 biți.

3. Scrierea testelor automate

Am creat un proiect de testare MSTest în Visual Studio și am scris scripturile de testare automate.

3 Scenariile de testare

Scenariile de testare sunt descrieri detaliate ale comportamentului așteptat al unei aplicații sau sistem în diverse situații sau contexte. Acestea sunt utilizate pentru a ghida procesul de testare și pentru a asigura că aplicația funcționează conform specificațiilor și cerințelor inițiale. Scenariile de testare sunt esențiale pentru planificarea, proiectarea și implementarea testelor, și oferă o bază structurată pentru evaluarea calității software-ului.

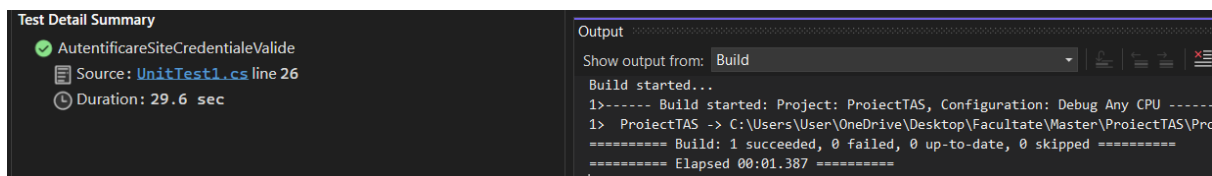
3.1 Test Case 1: Autentificare cu Credențiale Valide

Descriere:	Scopul acestui scenariu de testare automată este să verifice funcționalitatea de autentificare pe site-ul SauceDemo utilizând credențiale valide.		
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că aveți un cont valid cu credențialele: <ul style="list-style-type: none">• Nume Utilizator: standard_user• Parolă: secret_sauce		
Pași de Execuție:		Așteptări	Rezultat Status

Test Case 1: Autentificare cu Credențiale Valide

<ol style="list-style-type: none"> 1. Deschideți pagina principală a SauceDemo. 2. Identificați câmpurile de introducere a credențialelor și butonul de autentificare. 3. Introduceți numele de utilizator "standard_user". 4. Introduceți parola "secret_sauce". 5. Faceți clic pe butonul de autentificare. 	<ul style="list-style-type: none"> • Dacă autentificarea este realizată cu succes, utilizatorul ar trebui să fie redirecționat către pagina de inventar (/inventory.html). • Pagina de inventar ar trebui să conțină elemente relevante pentru autentificarea cu credențiale valide. • Nu ar trebui să apară mesaje de eroare sau avertizări în timpul autentificării. 	OK
Test Case Status :	Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Login")]
public void AutentificareSiteCredentialeValide()
{
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Găsește elementele de autentificare
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Name("password"));

    // Completează câmpurile de autentificare
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);

    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    // Apasă butonul de continua pentru autentificare
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);


    // Așteaptă pentru a se încărca pagina după autentificare
    System.Threading.Thread.Sleep(5000);

    // Verifică dacă autentificarea a reușit
    IWebElement productsTitle = driver.FindElement(By.ClassName("title"));
    Assert.IsTrue(productsTitle.Displayed, "Autentificarea cu credențiale valide a eșuat!");
}
```

3.2 Test Case 2: Autentificare cu Credențiale Invalide

Descriere:	Scopul acestui scenariu de testare automată este să verifice comportamentul sistemului în cazul încercării de autentificare cu credențiale invalide pe site-ul SauceDemo.	
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că aveți un cont valid cu credențialele: <ul style="list-style-type: none"> • Nume Utilizator: standard_user • Parolă: secret_sauce 	
Pași de Execuție:		Rezultat Status
1. Deschideți pagina principală a SauceDemo. 2. Identificați câmpurile de introducere a credențialelor și butonul de autentificare. 3. Introduceți un nume de utilizator invalid (de exemplu, "utilizator_inexistent"). 4. Introduceți o parolă invalidă (de exemplu, "parola_gresita"). 5. Faceți clic pe butonul de autentificare.		OK
Test Case Status :		Testul a trecut cu succes

Rezultat Test :

Test Detail Summary
 AutentificareSiteCredentialeInvalide
 Source: [UnitTest1.cs](#) line 60
 Duration: 13.4 sec

Output
 Show output from: Build
 Build started...
 1>----- Build started: Project: ProiectTAS, Conf
 1> ProiectTAS -> C:\Users\User\OneDrive\Desktop\
 ===== Build: 1 succeeded, 0 failed, 0 up-to-
 ===== Elapsed 00:01.387 =====

Cod :


```

[TestMethod]
[TestCategory("Login")]
public void AutentificareSiteCredentialeInvalide()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Introducere credentiale invalide
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    usernameField.SendKeys("utilizator_inexistent");
    System.Threading.Thread.Sleep(1000);

    IWebElement passwordField = driver.FindElement(By.Id("password"));
    passwordField.SendKeys("parola_gresita");
    System.Threading.Thread.Sleep(1000);

    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

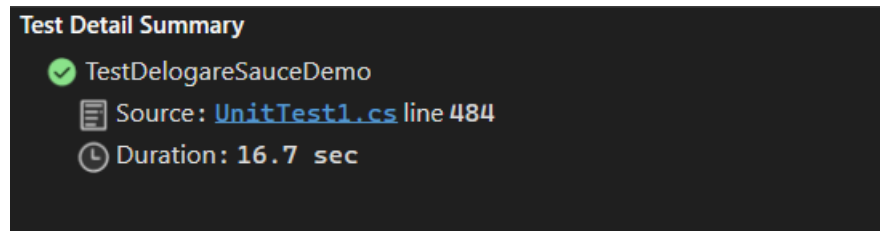
    // Verifică dacă mesajul de eroare este afișat
    IWebElement errorElement = driver.FindElement(By.CssSelector("[data-test='error']"));
    Assert.IsTrue(errorElement.Displayed, "Autentificarea cu credentiale invalide nu a generat eroare!");
}

```

3.3 Test Case 3 : Delogare de pe SauceDemo

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea procesului de delogare de pe site-ul SauceDemo.		
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și vă aflați pe o pagină relevantă pentru delogare.		
Pași de Execuție:		Așteptări	Rezultat Status
1. Identificați elementul sau butonul pentru delogare. 2. Faceți clic pe elementul sau butonul de delogare. 3. Așteptați ca procesul de delogare să fie finalizat și pagina să se încarce complet.		<ul style="list-style-type: none">Utilizatorul ar trebui să fie delogat cu succes.Pagina curentă ar trebui să se schimbe la o pagină de autentificare sau altă pagină relevantă pentru delogare.Nu ar trebui să apară erori sau avertizări în timpul procesului de delogare.	OK
Test Case Status :		Testul a trecut cu succes	

Rezultat Test :



Cod :

```
[TestMethod]
[TestCategory("Logout")]
0 references
public void TestDelogareSauceDemo()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

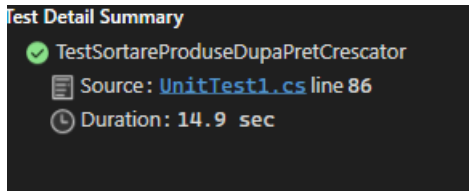
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    // Așteaptă ca pagina să se încarce
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.UrlContains("/inventory.html"));
    // Verifică dacă ești pe pagina de produse
    Assert.IsTrue(driver.Url.Contains("/inventory.html"), "Nu ești pe pagina corectă cu produse.");
    // Delogare
    IWebElement logoutButton = driver.FindElement(By.CssSelector(".bm-burger-button"));
    logoutButton.Click();
    System.Threading.Thread.Sleep(3000);
    IWebElement logoutLink = driver.FindElement(By.Id("logout_sidebar_link"));
    logoutLink.Click();
    // Așteaptă ca pagina să se încarce
    wait.Until(SeleniumExtras.WaitHelpers.ExpectedConditions.UrlContains("https://www.saucedemo.com/"));
    // Verifică dacă ești delogat și revenit la pagina de autentificare
    Assert.IsTrue(driver.Url.Equals("https://www.saucedemo.com/"), "Nu ești delogat și revenit la pagina de auten");
}
```

3.4 Test Case 4: Sortare Produse dupa Pret Crescator

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea funcționalității de sortare a produselor după preț în ordine crescătoare pe pagina de inventar a site-ului SauceDemo.		
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat cu un cont valid. 		
Pași de Execuție:	Așteptări	Rezultat Status	
<ol style="list-style-type: none"> Deschideți pagina de inventar (/inventory.html). Identificați elementul de selecție a sortării și selectați opțiunea de sortare după preț în ordine crescătoare. Așteptați ca pagina să se încarce complet după sortare. 	<ul style="list-style-type: none"> Produsele ar trebui să fie afișate în ordine crescătoare în funcție de preț. Nu ar trebui să existe mesaje de eroare sau avertizări în timpul procesului de sortare. 	OK	

4. Identificați lista de produse și verificați dacă aceasta este afișată în ordine crescătoare în funcție de preț.	• Alte elemente ale paginii (ex. filtre) ar trebui să rămână neschimbate.	
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Sortare")]
public void TestSortareProduseDupaPretCrescator()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptă acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Selectează opțiunea de sortare după preț
    IWebElement sortDropdown = driver.FindElement(By.ClassName("product_sort_container"));
    sortDropdown.Click();
    System.Threading.Thread.Sleep(1000);

    SelectElement selectSort = new SelectElement(sortDropdown);
    selectSort.SelectByText("Price (low to high)");
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca produsele să se sorteze
    System.Threading.Thread.Sleep(3000);

    // Verifică ordinea produselor după preț
    IList<IWebElement> productPrices = driver.FindElements(By.CssSelector(".inventory_item_price"));
    List<decimal> prices = new List<decimal>();

    foreach (var priceElement in productPrices)
    {
        string priceText = priceElement.Text.Replace("$", "");
        if (Decimal.TryParse(priceText, out decimal price))
        {
            prices.Add(price);
        }
    }

    // Verifică dacă lista de prețuri este sortată în ordine crescătoare
    Assert.IsTrue(IsSortedAscending(prices), "Produsele nu sunt sortate corect după preț.");
}
```

```

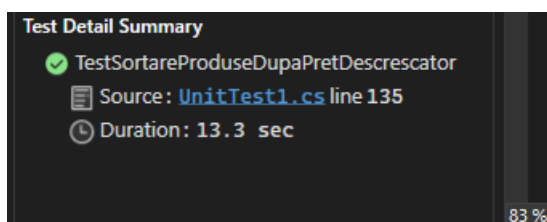
1 reference | 1/1 passing
private bool IsSortedAscending(List<decimal> list)
{
    for (int i = 0; i < list.Count - 1; i++)
    {
        if (list[i] > list[i + 1])
        {
            return false;
        }
    }
    return true;
}

```

3.5 Test Case 5: Sortare Produse după Preț Descrescător

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea sortării produselor după preț descrescător pe pagina de inventar a SauceDemo.		
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat si va aflați pe pagina (/inventory.html).		
Pași de Execuție:		Așteptări	Rezultat Status
1. Identificați elementul de selecție al sortării după preț. 2. Alegeți opțiunea de sortare după preț descrescător. 3. Capturați lista actuală de produse și prețurile lor înainte de sortare. 4. Aplicați sortarea și așteptați ca pagina să se încarce complet. 5. Capturați noua listă de produse și prețurile lor după sortare.		<ul style="list-style-type: none">• Lista de produse după sortare ar trebui să fie ordonată descrescător după preț.• Prețurile produselor în lista sortată ar trebui să fie în ordine descrescătoare.• Nu ar trebui să apară erori sau avertizări în timpul operațiunii de sortare.	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:



Cod:

Test Case 6: Sortare Produse după Nume Crescător

```

public void TestSortareProduseDupaPretDescrescator()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptă acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Selectează opțiunea de sortare după preț descrescător
    IWebElement sortDropdown = driver.FindElement(By.ClassName("product_sort_container"));
    SelectElement selectSort = new SelectElement(sortDropdown);
    selectSort.SelectByText("Price (high to low)");
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca produsele să se sorteze
    System.Threading.Thread.Sleep(2000);

    // Verifică ordinea produselor după preț
    IList<IWebElement> productPrices = driver.FindElements(By.CssSelector(".inventory_item_price"));
    List<decimal> prices = new List<decimal>();

    foreach (var priceElement in productPrices)
    {
        string priceText = priceElement.Text.Replace("$", "");
        if (Decimal.TryParse(priceText, out decimal price))
        {
            prices.Add(price);
        }
    }

    // Verifică dacă lista de prețuri este sortată în ordine descrescătoare
    Assert.IsTrue(IsSortedDescending(prices), "Produsele nu sunt sortate corect după preț descrescător.");
}

private bool IsSortedDescending(List<decimal> list)
{
    for (int i = 0; i < list.Count - 1; i++)
    {
        if (list[i] < list[i + 1])
        {
            return false;
        }
    }
    return true;
}

```

3.6 Test Case 6: Sortare Produse după Nume Crescător

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea sortării produselor după nume crescător pe pagina de inventar a SauceDemo.		
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat și va aflați pe pagina (/inventory.html). 		
Pași de Execuție:		Așteptări	Rezultat Status
<ol style="list-style-type: none"> Identificați elementul de selecție al sortării după nume. Alegeți opțiunea de sortare după nume crescător. Capturați lista actuală de produse și numele lor înainte de sortare. Aplicați sortarea și așteptați ca pagina să se încarce complet. 		<ul style="list-style-type: none"> Lista de produse după sortare ar trebui să fie ordonată crescător după nume. Numele produselor în lista sortată ar trebui să fie în ordine alfabetică crescătoare. 	OK

5. Capturați noua listă de produse și numele lor după sortare.	<ul style="list-style-type: none"> Nu ar trebui să apară erori sau avertizări în timpul operațiunii de sortare. 	
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:

Test Detail Summary

- TestSortareProduseInOrdineAlfabetica
- Source: [UnitTest1.cs](#) line 181
- Duration: 16.4 sec

Output

Show output from: Bu

```

Build started...
1>----- Build sta
1> ProjectTAS ->
===== Build:
===== Elapsed

```

Cod:

```

[TestCategory("Sortare")]
public void TestSortareProduseInOrdineAlfabetica()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități!)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Selectează opțiunea de sortare alfabetică
    IWebElement sortDropdown = driver.FindElement(By.ClassName("product_sort_container"));
    SelectElement selectSort = new SelectElement(sortDropdown);
    selectSort.SelectByText("Name (A to Z)");
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca produsele să se sorteze
    System.Threading.Thread.Sleep(2000);

    // Verifică ordinea produselor după nume
    IList<IWebElement> productNames = driver.FindElements(By.CssSelector(".inventory_item_name"));
    List<string> names = new List<string>();

    foreach (var nameElement in productNames)
    {
        names.Add(nameElement.Text);
    }

    // Verifică dacă lista de nume este sortată în ordine alfabetică
    Assert.IsTrue(IsSortedAlphabetically(names), "Produsele nu sunt sortate corect în ordine alfabetică.");
}

```

```

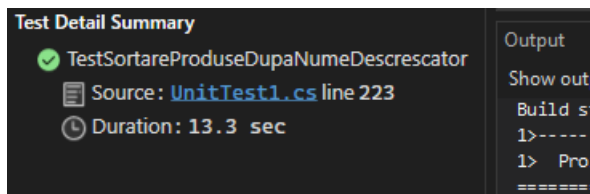
reference [✓] 1/1 passing
private bool IsSortedAlphabetically(List<string> list)
{
    for (int i = 0; i < list.Count - 1; i++)
    {
        if (string.Compare(list[i], list[i + 1], StringComparison.Ordinal) > 0)
        {
            return false;
        }
    }
    return true;
}

```

3.7 Test Case 7: Sortare Produse după Nume Descrescător

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea sortării produselor după nume descrescător pe pagina de inventar a SauceDemo.	
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și vă aflați pe pagina (/inventory.html).	
Pași de Execuție:		Rezultat Status
1. Identificați elementul de selecție al sortării după nume. 2. Alegeți opțiunea de sortare după nume descrescător. 3. Capturați lista actuală de produse și numele lor înainte de sortare. 4. Aplicați sortarea și așteptați ca pagina să se încarce complet. 5. Capturați noua listă de produse și numele lor după sortare.		OK
Test Case Status:		Testul a trecut cu succes

Rezultat Test:



Cod:


```
[TestMethod]
[TestCategory("Sortare")]
public void TestSortareProduseDupaNumeDescrescator()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Selectează opțiunea de sortare descrescătoare (de la Z la A)
    IWebElement sortDropdown = driver.FindElement(By.ClassName("product_sort_container"));
    SelectElement selectSort = new SelectElement(sortDropdown);
    selectSort.SelectByText("Name (Z to A)");
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca produsele să se sorteze
    System.Threading.Thread.Sleep(2000);
    // Verifică ordinea produselor după nume
    IList<IWebElement> productNames = driver.FindElements(By.CssSelector(".inventory_item_name"));
    List<string> names = new List<string>();

    foreach (var nameElement in productNames)
    {
        names.Add(nameElement.Text);
    }

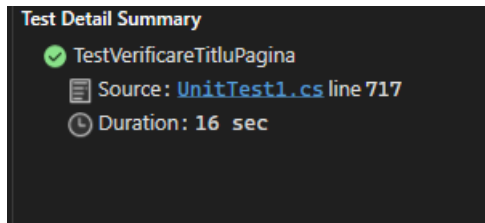
    // Verifică dacă lista de nume este sortată în ordine descrescătoare
    Assert.IsTrue(IsSortedAlphabeticallyDescending(names), "Produsele nu sunt sortate corect în ordine descrescătoare.");
}
```

✓ No issues found

```
1 reference | 1/1 passing
private bool IsSortedAlphabeticallyDescending(List<string> list)
{
    for (int i = 0; i < list.Count - 1; i++)
    {
        if (string.Compare(list[i], list[i + 1], StringComparison.Ordinal) < 0)
        {
            return false;
        }
    }
    return true;
}
```

3.8 Test Case 8: Verificare Titlu Pagină

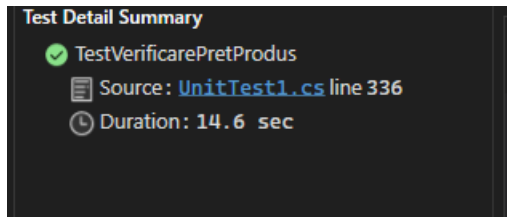
Descriere:	Scopul acestui scenariu de testare automată este să verifice dacă titlul paginii pe SauceDemo corespunde așteptărilor.		
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat cu un cont valid. 		
Pași de Execuție:		Așteptări	Rezultat Status
<ol style="list-style-type: none"> Extrageți titlul paginii Comparați titlul actual cu un titlu așteptat 		<ul style="list-style-type: none"> Titlul paginii ar trebui să corespundă titlului așteptat 	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:**Cod:**

```
[TestMethod]
[TestCategory("DetaliiProdus")]
public void TestVerificareTitluPagina()
{
    // Deschideți pagina principală a SauceDemo
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    // Obțineți titlul paginii
    string pageTitle = driver.Title;
    // Verifică dacă titlul este cel așteptat
    Assert.AreEqual("Swag Labs", pageTitle, "Titlul paginii nu este cel așteptat.");
}
```

3.9 Test Case 9: Verificare Preț Produs

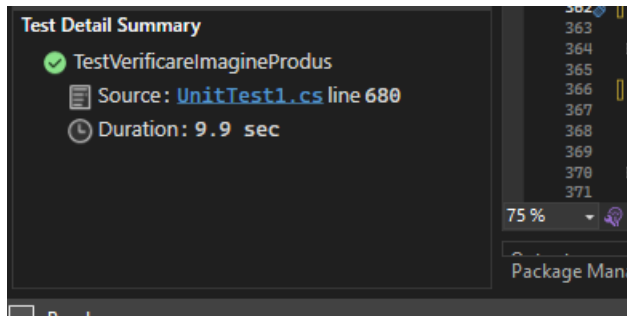
Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea afișării prețului unui produs pe pagina de detalii a produsului pe SauceDemo.	
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat și vă aflați pe o pagină cu detalii despre un produs. 	
Pași de Execuție:	Așteptări	Rezultat Status
<ol style="list-style-type: none"> Alegeți un produs din listă Obțineți prețul produsului Converteți prețul din format text în format decimal Verificați dacă prețul afișat corespunde cu cel așteptat 	<ul style="list-style-type: none"> Prețul afișat al produsului ar trebui să corespundă prețului așteptat. 	OK
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:**Cod:**

```
[TestCategory("DetaliiProdus")]
public void TestVerificarePretProdus()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credentiale valide (adaptati acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    // Accesează pagina cu produse
    IWebElement productsLink = driver.FindElement(By.CssSelector(".inventory_list"));
    productsLink.Click();
    System.Threading.Thread.Sleep(3000);
    // Alege un produs din listă (aici se alege primul produs)
    IWebElement productLink = driver.FindElement(By.CssSelector(".inventory_item a"));
    string productName = productLink.Text;
    productLink.Click();
    System.Threading.Thread.Sleep(3000);
    // Obține prețul produsului
    IWebElement productPriceElement = driver.FindElement(By.CssSelector(".inventory_details_price"));
    string productPriceText = productPriceElement.Text;
    // Convertește prețul din format text la tipul de date corespunzător (de exemplu, decimal)
    if (decimal.TryParse(productPriceText.Replace("$", ""), out decimal productPrice))
    {
        // Verifică dacă prețul produsului este corect
        decimal expectedPrice = 29.99M; // Actualizați cu prețul așteptat
        Assert.AreEqual(expectedPrice, productPrice, $"Prețul produsului {productName} nu este corect.");
    }
    else
    {
        Assert.Fail($"Nu s-a putut converti prețul produsului {productName} într-un număr valid.");
    }
}
```

3.10 Test Case 10: Verificare Imagine Produs

Descriere:	Scopul acestui scenariu de testare automată este să verifice dacă există o imagine afișată pentru un produs pe pagina de detalii a produsului pe SauceDemo.		
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și vă aflați pe o pagină cu detalii despre un produs.		
Pași de Execuție:		Așteptări	Rezultat Status
1. Identificați elementul care ar trebui să afișeze imaginea produsului. 2. Verificați dacă elementul este prezent pe pagină.		<ul style="list-style-type: none">Elementul care ar trebui să afișeze imaginea produsului ar trebui să fie prezent pe pagină	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:**Cod:**

```
[TestMethod]
[TestCategory("DetaliiProdus")]
public void TestVerificareImagineProdus()
{
    // Deschideți pagina principală a SauceDemo
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptati acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));

    // Identificați elementul care conține imaginea produsului (exemplu: primul produs în listă)
    IWebElement productImage = driver.FindElement(By.CssSelector(".inventory_item_img"));

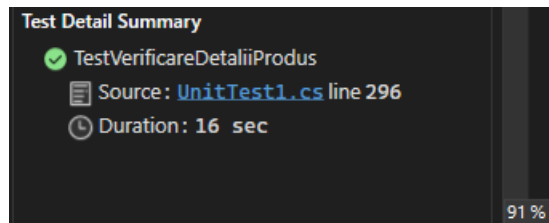
    // Verifică dacă imaginea este prezentă
    Assert.IsTrue(productImage.Displayed, "Imaginea produsului nu este afișată.");
}
```

3.11 Test Case 11: Verificare Detalii Produs

Descriere:	Scopul acestui scenariu de testare automată este să verifice dacă există o descriere afișată pentru un produs pe pagina de detalii a produsului pe SauceDemo.	
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat și vă aflați pe o pagină cu detalii despre un produs. 	
Pași de Execuție:	Așteptări	Rezultat Status
	<ul style="list-style-type: none"> Elementul care ar trebui să afișeze descrierea 	OK

<ol style="list-style-type: none"> 1. Identificați elementul care ar trebui să afișeze descrierea produsului. 2. Verificați dacă elementul este prezent pe pagină. 	<p>produsului ar trebui să fie prezent pe pagină.</p>	
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("DetaliiProdus")]
public void TestVerificareDetaliiProdus()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Accesează pagina cu produse
    IWebElement productsLink = driver.FindElement(By.CssSelector(".inventory_list"));
    productsLink.Click();
    System.Threading.Thread.Sleep(3000);

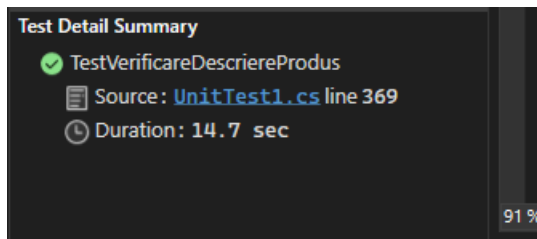
    // Accesează detaliile primului produs
    IWebElement firstProductLink = driver.FindElement(By.CssSelector(".inventory_item a"));
    firstProductLink.Click();
    System.Threading.Thread.Sleep(3000);

    // Verifică dacă sunteți pe pagina de detalii a produsului
    string productDetailsTitle = driver.FindElement(By.CssSelector(".inventory_details_name")).Text;
    Assert.IsTrue(productDetailsTitle.Length > 0, "Nu a fost accesată pagina de detalii a produsului.");
    // Reveniți la pagina cu lista de produse (opțional)
    driver.Navigate().Back();
}
```

3.12 Test Case 12: Verificare Descriere Produs

Descriere:	Scopul acestui scenariu de testare automată este să verifice dacă descrierea afișată pentru un produs pe pagina de detalii a produsului pe SauceDemo este corectă și corespunde așteptărilor.
Precondiții:	<ol style="list-style-type: none"> 1. Accesați site-ul SauceDemo: https://www.saucedemo.com/. 2. Asigurați-vă că sunteți autentificat și vă aflați pe o pagină cu detalii despre un produs.

Pași de Execuție:	Așteptări	Rezultat Status
<ol style="list-style-type: none"> 1. Identificați elementul care afișează descrierea produsului. 2. Obțineți descrierea actuală a produsului. 3. Comparati descrierea actuală cu o descriere așteptată. 	<ul style="list-style-type: none"> • Descrierea afișată a produsului ar trebui să corespundă descrierii așteptate. 	OK
Test Case Status:	Testul a trecut cu succes	

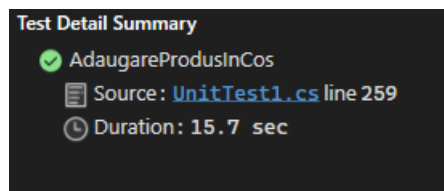
Rezultat Test:**Cod:**

```
[TestMethod]
[TestCategory("DetaliiProdus")]
public void TestVerificareDescriereProdus()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    // Accesează pagina cu produse
    IWebElement productsLink = driver.FindElement(By.CssSelector(".inventory_list"));
    productsLink.Click();
    System.Threading.Thread.Sleep(3000);
    // Alege un produs din listă (aici se alege primul produs)
    IWebElement productLink = driver.FindElement(By.CssSelector(".inventory_item a"));
    string productName = productLink.Text;
    productLink.Click();
    // Obține descrierea produsului
    IWebElement productDescriptionElement = driver.FindElement(By.CssSelector(".inventory_details_desc"));
    string productDescription = productDescriptionElement.Text;
    // Verifică dacă descrierea produsului este neagră și conține informații relevante
    Assert.IsTrue(!string.IsNullOrEmpty(productDescription), $"Descrierea produsului {productName} este goală");
}
```

3.13 Test Case 13: Adăugare produs în coș

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea procesului de adăugare a unui produs în coșul de cumpărături pe SauceDemo.
-------------------	--

Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat cu un cont valid.	
Pași de Execuție:	Așteptări	Rezultat Status
1. Identificați elementul sau butonul care permite adăugarea produsului în coș. 2. Faceți clic pe elementul sau butonul de adăugare în coș. 3. Așteptați ca produsul să fie adăugat în coș 4. Accesați coșul pentru a putea verifica existența produsului 5. Verifică dacă produsul adăugat este prezent în coș	<ul style="list-style-type: none"> • Produsul ar trebui să fie adăugat cu succes în coș. • Mesajele sau indicatorii de succes ar trebui să confirme adăugarea produsului în coș. 	OK
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:**Cod:**

```

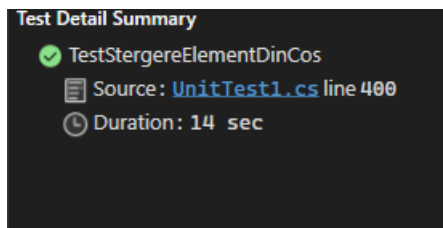
[TestMethod]
[TestCategory("Cart")]
public void AdaugareProdusInCos()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    System.Threading.Thread.Sleep(2000);
    // Adaugă primul produs în coș
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();
    System.Threading.Thread.Sleep(2000);
    // Accesează coșul pentru a verifica adăugarea produsului
    IWebElement cartIcon = driver.FindElement(By.CssSelector(".shopping_cart_container"));
    cartIcon.Click();
    System.Threading.Thread.Sleep(1000);
    // Verifică dacă produsul adăugat este prezent în coș
    IWebElement cartItem = driver.FindElement(By.CssSelector(".cart_item"));
    Assert.IsTrue(cartItem.Displayed, "Produsul nu a fost adăugat în coș.");
}

```


3.14 Test Case 14: Ștergere Produs din Coș

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea procesului de ștergere a unui produs din coșul de cumpărături pe SauceDemo.		
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și aveți cel puțin un produs în coș.		
Pași de Execuție:		Așteptări	Rezultat Status
1. Accesați coșul de cumpărături. 2. Identificați elementul sau butonul care permite ștergerea produsului din coș. 3. Faceți clic pe elementul sau butonul de ștergere. 4. Așteptați confirmarea ștergerii și verificați mesajele sau indicatorii de succes.		<ul style="list-style-type: none"> • Produsul ar trebui să fie șters cu succes din coș. • Mesajele sau indicatorii de succes ar trebui să confirme ștergerea produsului din coș. 	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Cart")]
public void TestStergereElementDinCos()
{
    // Deschide pagina de autentificare
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Accesează pagina cu produse
    IWebElement productsLink = driver.FindElement(By.CssSelector(".inventory_list"));
    productsLink.Click();

    // Adaugă un produs în coș (aici se adaugă primul produs)
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Accesează coșul pentru a verifica adăugarea produsului
    IWebElement cartIcon = driver.FindElement(By.CssSelector(".shopping_cart_container"));
    cartIcon.Click();
    System.Threading.Thread.Sleep(1000);

    // Verifică dacă produsul adăugat este prezent în coș
    IWebElement cartItem = driver.FindElement(By.CssSelector(".cart_item"));
}
```

```

Assert.IsTrue(cartItem.Displayed, "Produsul nu a fost adăugat în coș.");
// Șterge produsul din coș
IWebElement removeButton = driver.FindElement(By.CssSelector("button.btn_secondary.cart_button"));
removeButton.Click();
System.Threading.Thread.Sleep(1000);
// Așteaptă puțin pentru a permite coșului să se actualizeze
System.Threading.Thread.Sleep(2000);
// Verifică dacă produsul a fost șters din coș
try
{
    // Încercă să găsești elementul șters
    IWebElement removedItem = driver.FindElement(By.CssSelector(".cart_item"));
    Assert.Fail("Produsul nu a fost șters din coș.");
}
catch (NoSuchElementException)
{
    // Elementul nu a fost găsit, ceea ce înseamnă că a fost șters corect
}
}
[TestMethod]

```

3.15 Test Case 15: Cumpărare Produs fără CheckOut

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea procesului de cumpărare a unui produs până la pagina de completare a informațiilor pe SauceDemo.	
Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și aveți cel puțin un produs în coș.	
Pași de Execuție:	Așteptări	Rezultat Status
1. Accesați coșul de cumpărături. 2. Identificați elementul sau butonul care inițiază procesul de cumpărare. 3. Faceți clic pe elementul sau butonul de cumpărare. 4. Verificați dacă sunteți pe pagina de completare a informațiilor de livrare	<ul style="list-style-type: none"> Produsul ar trebui să fie în coș Pagina de completare a informațiilor ar trebui să corespundă cu pagina așteptată. 	OK
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:

Test Detail Summary

✓ TestCumparareProdus

Source: [UnitTest1.cs](#) line 515

Duration: 13.4 sec

Cod:

```

[TestMethod]
[TestCategory("Cart")]
public void TestCumparareProdus()
{
    // Deschideți pagina principală a SauceDemo
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    // Adăugați un produs în coș (puteți adapta în funcție de specificul site-ului)
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();
    // Așteaptă ca produsul să fie adăugat în coș
    wait.Until(ExpectedConditions.ElementIsVisible(By.CssSelector(".shopping_cart_badge")));
    // Navigați la coș
    IWebElement cartIcon = driver.FindElement(By.CssSelector(".shopping_cart_container"));
    cartIcon.Click();
    // Așteaptă ca pagina coșului să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/cart.html"));
    // Faceți clic pe butonul "Checkout"
    IWebElement checkoutButton = driver.FindElement(By.CssSelector(".checkout_button"));
    checkoutButton.Click();
    // Așteaptă ca pagina de informații de livrare să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/checkout-step-one.html"));

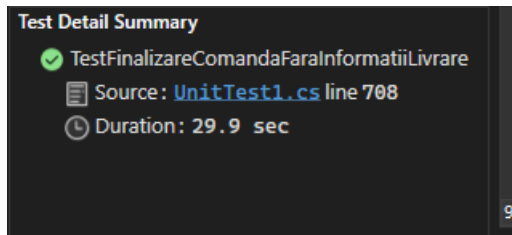
    // Verifică dacă ești pe pagina corectă (de exemplu, pagina de informații de livrare)
    Assert.IsTrue(driver.Url.Contains("/checkout-step-one.html"), "Nu ești pe pagina corectă pentru informații de livrare.");
}

```

3.16 Test Case 16: Cumpărare Produs fără Informații de Livrare

Descriere:	Scopul acestui scenariu de testare automată este să verifice cum gestionează SauceDemo procesul de cumpărare a unui produs atunci când utilizatorul nu completează informațiile de livrare.		
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat și aveți cel puțin un produs în coș. 		
Pași de Execuție:		Așteptări	Rezultat Status
<ol style="list-style-type: none"> Accesați coșul de cumpărături. Identificați elementul sau butonul care inițiază procesul de cumpărare. Faceți clic pe elementul sau butonul de cumpărare. Nu completați informațiile de livrare și încercați să finalizați comanda. 		<ul style="list-style-type: none"> Sistemul ar trebui să afișeze un mesaj de eroare sau să prevină finalizarea comenzii, solicitând completarea informațiilor de livrare. 	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Cart")]
public void TestFinalizareComandaFaraInformatiiLivrare()
{
    // Deschideți pagina principală a SauceDemo și adăugați produse în coș
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    System.Threading.Thread.Sleep(3000);

    // Adăugați un produs în coș (puteți adapta în funcție de specificul site-ului)
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();

    System.Threading.Thread.Sleep(3000);

    // Așteaptă ca produsul să fie adăugat în coș
    wait.Until(ExpectedConditions.ElementIsVisible(By.CssSelector(".shopping_cart_badge")));
    System.Threading.Thread.Sleep(3000);

    // Navigați la coș
    IWebElement cartIcon = driver.FindElement(By.CssSelector(".shopping_cart_container"));
    cartIcon.Click();

    // Așteaptă ca pagina cosului să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/cart.html"));
    System.Threading.Thread.Sleep(3000);
```

```
// Continuați cu operațiunile de plată fără a completa informațiile de livrare
IWebElement checkoutButton = driver.FindElement(By.CssSelector(".checkout_button"));
checkoutButton.Click();
System.Threading.Thread.Sleep(3000);

// Așteaptă ca pagina de informații de livrare să se încarce complet
wait.Until(ExpectedConditions.UrlContains("/checkout-step-one.html"));
System.Threading.Thread.Sleep(3000);

// Faceți clic pe butonul "Finish" pentru a încerca să finalizați comanda
IWebElement finishButton = driver.FindElement(By.CssSelector(".btn_primary"));
finishButton.Click();

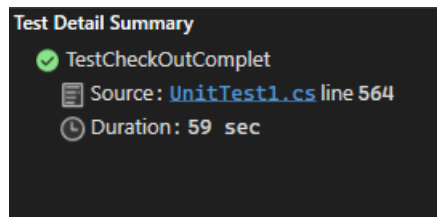
// Verificați dacă aplicația afișează mesaje de eroare pentru câmpurile necompletate
IWebElement errorMessage = driver.FindElement(By.CssSelector("button.error-button"));
Assert.IsTrue(errorMessage.Displayed, "Mesajul de eroare pentru informații de livrare incomplete nu este afișat corespunzător.");

// Puteți adăuga și alți pași de verificare sau de manipulare în funcție de necesități
```

3.17 Test Case 17: Cumpărare Produs

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea procesului de cumpărare a unui produs pe SauceDemo.
------------	---

Precondiții:	1. Accesați site-ul SauceDemo: https://www.saucedemo.com/ . 2. Asigurați-vă că sunteți autentificat și aveți cel puțin un produs în coș.		
Pași de Execuție:	Așteptări	Rezultat Status	
1. Accesați coșul de cumpărături. 2. Identificați elementul sau butonul care inițiază procesul de cumpărare. 3. Faceți clic pe elementul sau butonul de cumpărare. 4. Completați informațiile necesare pentru finalizarea comenzii (ex. adresa de livrare, detalii de plată). 5. Așteptați confirmarea finalizării comenzii și verificați mesajele sau indicatorii de succes.	<ul style="list-style-type: none">Comanda ar trebui să fie plasată cu succes.Mesajele sau indicatorii de succes ar trebui să confirme finalizarea comenzii.	OK	
Test Case Status:	Testul a trecut cu succes		

Rezultat Test:

Cod:

```

[TestMethod]
[TestCategory("Cart")]
public void TestCheckoutComplet()
{
    // Deschideți pagina principală a SauceDemo
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));

    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    System.Threading.Thread.Sleep(3000);
    // Adăugați un produs în coș
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca produsul să fie adăugat în coș
    wait.Until(ExpectedConditions.ElementIsVisible(By.CssSelector(".shopping_cart_badge")));
    System.Threading.Thread.Sleep(3000);

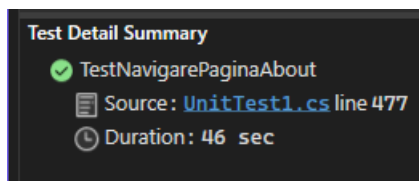
    // Navigați la coș
    IWebElement cartIcon = driver.FindElement(By.CssSelector(".shopping_cart_container"));
    cartIcon.Click();
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca pagina coșului să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/cart.html"));
    System.Threading.Thread.Sleep(3000);
    // Faceți clic pe butonul "Checkout"
    IWebElement checkoutButton = driver.FindElement(By.CssSelector(".checkout_button"));
    checkoutButton.Click();
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca pagina de informații de livrare să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/checkout-step-one.html"));
    System.Threading.Thread.Sleep(3000);
    // Completați informațiile de livrare
    IWebElement firstNameInput = driver.FindElement(By.Id("first-name"));
    firstNameInput.SendKeys("Laura");
    System.Threading.Thread.Sleep(3000);
    IWebElement lastNameInput = driver.FindElement(By.Id("last-name"));
    lastNameInput.SendKeys("Pop");
    System.Threading.Thread.Sleep(3000);
    IWebElement zipCodeInput = driver.FindElement(By.Id("postal-code"));
    zipCodeInput.SendKeys("1234");
    System.Threading.Thread.Sleep(3000);
    // Faceți clic pe butonul "Continue"
    IWebElement continueButton = driver.FindElement(By.CssSelector(".btn_primary"));
    continueButton.Click();
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca pagina de informații de plată să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/checkout-step-two.html"));
    System.Threading.Thread.Sleep(3000);
    // Faceți clic pe butonul "Finish"
    IWebElement finishButton = driver.FindElement(By.CssSelector(".btn_action"));
    finishButton.Click();
    System.Threading.Thread.Sleep(3000);
    // Așteaptă ca pagina de confirmare a comenzii să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("/checkout-complete.html"));
    // Verifică dacă ești pe pagina corectă (de exemplu, pagina de confirmare a comenzii)
    Assert.IsTrue(driver.Url.Contains("/checkout-complete.html"), "Nu ești pe pagina corectă pentru confirmarea comenzii.");
}
[TestMethod]

```

3.18 Test Case 18: Navigarea pe Pagina About

Descriere:	Scopul acestui scenariu de testare automată este să verifice corectitudinea și funcționalitatea navigării pe pagina "About" pe SauceDemo.	
Precondiții:	Accesați site-ul SauceDemo: https://www.saucedemo.com/ .	
Pași de Execuție:	Așteptări	Rezultat Status
<ol style="list-style-type: none"> 1. Identificați elementul sau butonul care duce la pagina "About". 2. Faceți clic pe elementul sau butonul de navigare. 3. Așteptați ca pagina "About" să se încarce complet. 	<ul style="list-style-type: none"> • Utilizatorul ar trebui să fie redirecționat cu succes pe pagina "About". • Elementele specifice ale paginii "About" ar trebui să fie vizibile și accesibile. 	OK
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Navigare")]
public void TestNavigatePaginaAbout()
{
    // Deschideți pagina principală a SauceDemo
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();

    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    System.Threading.Thread.Sleep(1000);
    passwordField.SendKeys("secret_sauce");
    System.Threading.Thread.Sleep(1000);
    loginButton.Click();
    System.Threading.Thread.Sleep(1000);

    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    IWebElement moreButton = driver.FindElement(By.CssSelector(".bm-burger-button"));
    moreButton.Click();
    System.Threading.Thread.Sleep(3000);

    // Identificați și faceți clic pe link-ul către pagina "About"
    IWebElement aboutLink = driver.FindElement(By.Id("about_sidebar_link"));
    aboutLink.Click();

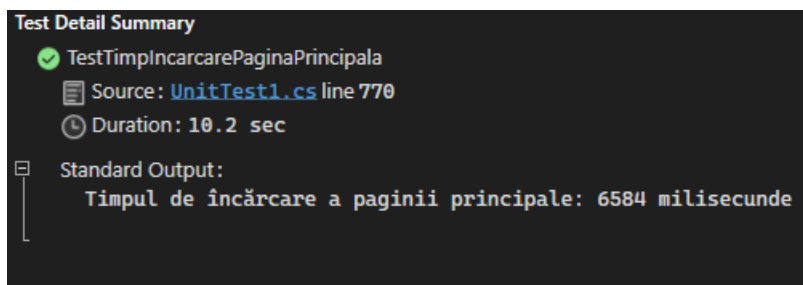
    // Așteaptă ca pagina "About" să se încarce complet
    wait.Until(ExpectedConditions.UrlContains("https://saucelabs.com/"));

    // Verifică dacă ești pe pagina "About"
    Assert.IsTrue(driver.Url.Contains("https://saucelabs.com/"), "Nu ești pe pagina corectă 'About'.");
}
```

3.19 Test Case 19: Măsurare Timp Încărcare Pagină

Descriere:	Scopul acestui scenariu de testare automată este să măsoare timpul de încărcare a paginii principale a SauceDemo și să afișeze acest timp în consolă.		
Precondiții:	Accesați site-ul SauceDemo: https://www.saucedemo.com/ .		
Pași de Execuție:		Așteptări	Rezultat Status
<ol style="list-style-type: none"> Înainte de a încărca pagina, porniți timpul. Accesați pagina principală Opriți timpul Afișați în consolă timpul 		<ul style="list-style-type: none"> Timpul de încărcare al paginii ar trebui să fie afișat în console fără erori. 	OK
Test Case Status:		Testul a trecut cu succes	

Rezultat Test:



Cod:

```
[TestMethod]
[TestCategory("Performance")]
public void TestTimpIncarcarePaginaPrincipala()
{
    Stopwatch stopwatch = new Stopwatch();

    // Măsoară timpul de încărcare a paginii principale
    stopwatch.Start();
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    stopwatch.Stop();

    // Afișează timpul de încărcare
    Console.WriteLine($"Timpul de încărcare a paginii principale: {stopwatch.ElapsedMilliseconds} milisecunde");
}
```

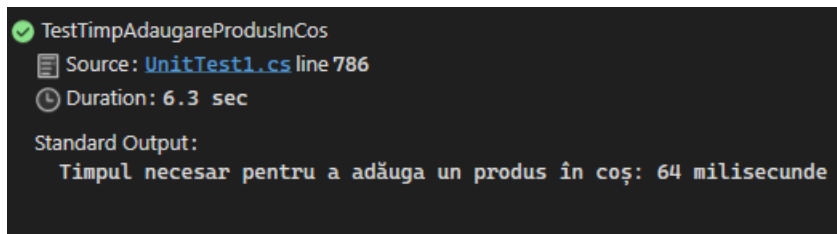
3.20 Test Case 20: Măsurare Timp Adăugare Produs în Coș

Descriere:	Scopul acestui scenariu de testare automată este să măsoare timpul necesar pentru adăugarea unui produs în coș pe SauceDemo și să afișeze acest timp în consolă.		
Precondiții:	<ol style="list-style-type: none"> Accesați site-ul SauceDemo: https://www.saucedemo.com/. Asigurați-vă că sunteți autentificat și aveți cel puțin un produs disponibil pentru adăugare în coș. 		

Test Case 20: Măsurare Timp Adăugare Produs în Coș

Pași de Execuție:	Așteptări	Rezultat Status
<ol style="list-style-type: none"> Înainte de a începe adăugarea produsului în coș, porniți timpul. Identificați produsul pe care doriți să-l adăugați în coș. Faceți clic pe butonul sau elementul de adăugare în coș. După ce produsul a fost adăugat în coș, opriți timpul. Afișați timpul în consolă. 	<ul style="list-style-type: none"> Timpul de adăugare a produsului în coș ar trebui să fie afișat în consolă fără erori. 	OK
Test Case Status:	Testul a trecut cu succes	

Rezultat Test:



Cod:

```

[TestMethod]
[TestCategory("Performance")]
public void TestTimpAdaugareProdusInCos()
{
    // Deschideți pagina principală a SauceDemo și adăugați produse în coș
    driver.Navigate().GoToUrl("https://www.saucedemo.com/");
    driver.Manage().Window.Maximize();
    // Autentifică-te cu credențiale valide (adaptați acești pași în funcție de necesități)
    IWebElement usernameField = driver.FindElement(By.Id("user-name"));
    IWebElement passwordField = driver.FindElement(By.Id("password"));
    IWebElement loginButton = driver.FindElement(By.Id("login-button"));
    usernameField.SendKeys("standard_user");
    passwordField.SendKeys("secret_sauce");
    loginButton.Click();
    // Așteaptă ca pagina să se încarce complet
    WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));
    wait.Until(ExpectedConditions.UrlContains("/inventory.html"));
    // Măsoară timpul necesar pentru a adăuga un produs în coș
    Stopwatch stopwatch = new Stopwatch();
    stopwatch.Start();
    // Adăugați un produs în coș (puteți adapta în funcție de specificul site-ului)
    IWebElement addToCartButton = driver.FindElement(By.CssSelector(".btn_inventory"));
    addToCartButton.Click();
    // Așteaptă ca produsul să fie adăugat în coș
    wait.Until(ExpectedConditions.ElementIsVisible(By.CssSelector(".shopping_cart_badge")));
    // Oprirea ceasului după adăugarea produsului în coș
    stopwatch.Stop();
    // Afișează timpul de adăugare în coș
    Console.WriteLine($"Timpul necesar pentru a adăuga un produs în coș: {stopwatch.ElapsedMilliseconds} milisecunde");
}

```