```java
package páginas;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;

public class LoginPage { // Estes são os primeiros principios do page object.
    private WebDriver navegador; // Primeiro: Tenha um atributo da classe que seja do WebDrive.
    // Segundo: Tenha um construtor da classe que pegue um atributo / navegador de fora e jogue neste navegador.
    public LoginPage(WebDriver navegador) {
        this.navegador = navegador;
    }

    public LoginPage informarOUsername(String username) {
        navegador.findElement(By.cssSelector("label[for='username']")).click();
        navegador.findElement(By.id("username")).sendKeys(username);
        // sendKeys(username), a informação de quem é o username passa a ser dinâmica e não mais fixa.

        return this;
    }

    public LoginPage informarAPassword(String password) {
        navegador.findElement(By.cssSelector("label[for='password']")).click();
        navegador.findElement(By.id("password")).sendKeys(password);

        return this;
    }


   public ListaDeProdutosPage submeterFormularioDeLogin() {
       navegador.findElement(By.cssSelector("button[
```

```
29 type='submit']")).click();
30
31         return new ListaDeProdutosPage(navegador);
32    }
33
34
35
36 }
37
```

```java
package páginas;


import com.sun.deploy.cache.DeployCacheHandler;
import com.sun.javafx.geom.Curve;
import org.junit.jupiter.api.*;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;


import java.time.Duration;

@DisplayName("Testes Web do Modulo de Produtos")

public class ProdutosTest {

    private static int produtos;
    private final Curve input;
    private WebDriver navegador;

    @BeforeEach
    public void beforeEach() {
        // Abrir o navegador
        System.setProperty("webdriver.chrome.driver", "C:\\drivers\\chromedriver.exe");
        this.navegador = new ChromeDriver();

        // Vou maximizar a tela
        this.navegador.manage().window().maximize();

        // Vou definir um tempo de espera padrao de 5 segundos
        this.navegador.manage().timeouts().implicitlyWait(Duration.ofSeconds(10));

        // Navegar para a página da Saucedemo
        this.navegador.get("https://www.saucedemo.com/");
    }
    @Test
    @DisplayName("Posso adicionar produto camisa
```

```
37 vermelha")
38         public void
   testPossoAdicionarProdutoCamisaVermelha() {
39
40             // Fazer login
41         String mensagemApresentada = new LoginPage(
   navegador)
42                 .informarOUsername("standard_user")
43                 .informarAPassword("secret_sauce")
44                 .submeterFormularioDeLogin()
45                 .capturarMensagemApresentada();
46
47
48         Assertions.assertEquals("A compra da camisa
   vermelha foi efetuada.", mensagemApresentada);
49     }
50
51
52 @Test
53 @DisplayName("Posso adicionar produto bolsa")
54         public void testPossoAdicionarProdutoBolsa
   ()  {
55         String mensagemApresentada = new LoginPage(
   navegador)
56         .informarOUsername("standard_user")
57         .informarAPassword("secret_sauce")
58         .submeterFormularioDeLogin()
59         .capturarMensagemApresentada();
60
61         // Produto adiconado com sucesso
62         Assertions.assertEquals("A compra da bolsa
   foi efetuada.", mensagemApresentada);
63     }
64 @Test
65 @DisplayName("Posso adicionar dois produtos com
   valores menores")
66 public void
   testPossoAdicionarDoisProdutosComValoresMenores()  {
67         String mensagemApresentada = new LoginPage(
   navegador)
68         .informarOUsername("standard_user")
```

```
 69          .informarAPassword("secret_sauce")
 70          .submeterFormularioDeLogin()
 71          .capturarMensagemApresentada();
 72
 73          // Produto adicionado com sucesso
 74          Assertions.assertEquals("Produtos com
    valores menores foi efetuada.", mensagemApresentada
    );
 75          }
 76
 77      // Teste:
 78      public String ProdutosTest(Curve input) {
 79          this.input = input;
 80          boolean comprarProduto;
 81          if (comprarProduto > 0) {
 82              System.out.println("Clique no botão do
    produto: \n");
 83
 84              System.out.println("--------Produtos
    disponíveis--------");
 85              for (ProdutosTest p : comprarProduto
    ())  {
 86                  System.out.println(p + "\n");
 87              }
 88
 89              int id = Integer.parseInt(this.input.
    next());
 90              boolean isPresent = false;
 91
 92              ProdutosTest[] comprarProduto;
 93              for (ProdutosTest p : comprarProduto) {
 94                  if (p.getId() == id) {
 95                      int quantidade = 0;
 96                      DeployCacheHandler carrinho;
 97                      try {
 98                          quantidade = carrinho.get(p
    );
 99                          carrinho.put(produtos,
    quantidade + 1);
100                      } catch (NullPointerException e
    ) {
```

```
101                         //se o produto for o
    primeiro no carrinho
102                         carrinho.put(produtos,1);
103                     }
104                     System.out.println(p.getNome
    () + "adicionado ao carrinho.");
105                     isPresent = true;
106
107                     if (isPresent) {
108                         System.out.println("Deseja
    adicionar outro produto ao carrinho? ");
109                         System.out.println("Clique
    no produto para finalizar a compra. \n");
110                         int option = Integer.
    parseInt(this.input.next());
111
112                         if (option == 1) {
113                             comprarProduto();
114
115                         } else {
116                             return("Compra
    finalizada.");
117                         }
118                     }
119                 } else {
120                     System.out.println("Produto não
    encontrado.");
121                     return("produto");
122                 }
123             }
124         }
125     }
126
127     public int comprarProduto() {
128         return comprarProduto();
129     }
130
131     public int getId() {
132         return getId();
133     }
134
```

```
135
136     public static void verCarrinho() {
137         System.out.println("Produtos no seu carrinho
    ");
138         if(carrinho.size() > 0) {
139
140             for(ListaDeProdutosPage p: carrinho.set
    ());
141                 System.out.println("Produto: " + p
     + "\nQuangetidade: " + carrinho.get(produtos));
142         }
143     }
144
145 @AfterEach
146     public void afterEach() {
147         // Vou fechar o navegador
148         navegador.quit();
149     }
150
151     // Não existe uma sequência de execução no JUnit
    , independencia dos testes.
152
153     public String getNome() {
154         return getNome();
155     }
156 }
157
```

```java
package páginas;

import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;

public class ListaDeProdutosPage {
    private WebDriver navegador;

    public ListaDeProdutosPage(WebDriver navegador) {
        this.navegador = navegador;
    }


    public String capturarMensagemApresentada()  {
        // Vou validar que a mensagem de erro foi apresentada
        return navegador.findElement(By.cssSelector(".toast.rounded")).getText();
    }
}

```