1. Tipos de Testes:

**a) Teste Unitário:** são testes que os desenvolvedores fazem para verificar se partes pequenas do código, como funções ou classes, estão funcionando direito? Isso é o teste unitário. Eles são importantes para garantir que as partes individuais do software estejam corretas.

**b) Teste de Integração:** Quando queremos ter certeza de que diferentes partes do software funcionam bem juntas, usamos o teste de integração. Ele olha para a forma como essas partes se comunicam e se encaixam. É importante para ter um sistema estável, como exemplo uma tela que de perfil, geralmente há dados do usuário nesta tela, e precisa estar funcionando bem com o banco de dados CRUD, e também os dados dessa tela precisam aparecer nas páginas que exibem esses dados, como a saudação no header da página com o nome e sobrenome por exemplo.

**c) Teste de UI (User Interface):** Esse teste se concentra na aparência e na interação do usuário com o software. Basicamente, verificamos se os botões, janelas e formulários estão funcionando corretamente. É crucial para garantir uma boa experiência do usuário.

2. Importância da Qualidade de Software em um E-commerce:

A qualidade do software é uma questão gigante em praticamente tudo, mas pensando no e-commerce, se o software não estiver em boa forma, corremos sérios riscos:

Podemos perder clientes, porque ninguém gosta de lidar com um site que são lentos, trava ou tem erros.

* A segurança é crucial, pois qualquer falha pode expor informações sensíveis dos clientes.
* A concorrência está sempre à espreita, e um site com problemas pode fazer com que os clientes vão para o concorrente.
* Corrigir problemas depois do lançamento custa caro e dá muito trabalho.

3. Método "Passo a Passo" para Gherkin:

**GHERKIN**

Funcionalidade: Busca por produto

Cenário: Buscar um produto no site

Dado que eu estou na página inicial do site

Quando eu digito "Processador i9" na barra de busca e clico em pesquisar

Então eu devo ver uma listagem de produtos

4. TDD (Test Driven Development) vs. BDD (Behavior Driven Development):

**TDD** é quando os desenvolvedores escrevem testes antes de escrever o código real. Eles se concentram em aspectos técnicos.

**BDD** se concentra no comportamento do software do ponto de vista do usuário. Os testes são escritos em linguagem natural para descrever o que o software deveria fazer.

5. Page Objects:

Page Objects são uma maneira de organizar o código de automação de testes. Em vez de misturar código de automação com a interface do usuário, você cria classes separadas para representar cada página ou componente. Isso torna os testes mais fáceis de entender e manter, especialmente quando a interface do usuário muda.

6. Automação de Testes com Robot Framework e Selenium:

Após estudar um pouco sobre o Robot Framework, pois nunca cheguei a utilizar, instalei em meu repositório seguindo a documentação fornecida em <https://robotframework.org/?tab=1#getting-started> executando com pip install robotframework

Versão chromewebdriver: https://edgedl.me.gvt1.com/edgedl/chrome/chrome-for-testing/117.0.5938.132/win64/chromedriver-win64.zip

fiz esse teste:  
```robotframework

\*\*\* Settings \*\*\*

Library SeleniumLibrary

\*\*\* Variables \*\*\*

${URL} https://kabum.com.br

\*\*\* Test Cases \*\*\*

Validar Usuário Inválido

Open Browser ${URL} chrome

Input Text id=username usuario\_invalido

Input Text id=password senha

Click Button id=submit

Element Should Be Visible xpath=//div[contains(text(),'Usuário ou senha inválidos')]

Close Browser

Buscar iPhone

Open Browser ${URL} chrome

Input Text id=search-term iphone

Click Button id=search-button

Wait Until Page Contains Element xpath=//div[@class='listagem-box']

Close Browser

Selecionar Categoria e Adicionar ao Carrinho

Open Browser ${URL} chrome

Mouse Over xpath=//a[contains(text(),'Departamento')]

Click Link xpath=//a[contains(text(),'Informática')]

Click Link xpath=//a[contains(text(),'Processadores')]

Click Button xpath=//button[contains(text(),'Adicionar ao carrinho')]

Element Should Be Visible xpath=//div[contains(text(),'Produto adicionado ao carrinho')]

Close Browser

```

mas fiz em Cypress também, por estar mais acostumado em automação de testes.

describe("Validar Usuário Inválido", () => {

it("Deve exibir mensagem de erro para usuário inválido", () => {

cy.visit("https://kabum.com.br");

cy.get("#username").type("usuario\_invalido");

cy.get("#password").type("senha");

cy.get("#submit").click();

cy.get("div:contains('Usuário ou senha inválidos')").should("be.visible");

});

});

describe("Buscar iPhone", () => {

it("Deve realizar uma busca por iPhone e exibir resultados", () => {

cy.visit("https://kabum.com.br");

cy.get("#search-term").type("iphone");

cy.get("#search-button").click();

cy.get("div.listagem-box").should("exist");

});

});

describe("Selecionar Categoria e Adicionar ao Carrinho", () => {

it("Deve selecionar uma categoria e adicionar um produto ao carrinho", () => {

cy.visit("https://kabum.com.br");

cy.contains("Departamento").trigger("mouseover");

cy.contains("Informática").click();

cy.contains("Processadores").click();

cy.contains("Adicionar ao carrinho").click();

cy.get("div:contains('Produto adicionado ao carrinho')").should("be.visible");

});

});