Documentação do arquivo MyProfilePageControllerTest.cls

Introdução

Este arquivo contém uma classe de teste Apex chamada `MyProfilePageControllerTest`. O objetivo desta classe é verificar a funcionalidade de um controlador que atualiza os detalhes de um usuário do portal. A classe garante que os usuários convidados nunca possam acessar a página e valida o comportamento esperado do controlador.

Descrição

A classe `MyProfilePageControllerTest` é uma classe de teste escrita em Apex para validar a funcionalidade de um controlador chamado `MyProfilePageController`. O controlador é responsável por gerenciar os detalhes do perfil de um usuário do portal. A classe de teste verifica cenários como:

- Carregar o usuário atual.
- Alternar entre os modos de edição e visualização.
- Atualizar informações do usuário, como o número de fax.
- Garantir que as alterações sejam persistidas corretamente no banco de dados.

Estrutura

A classe é composta por um único método de teste estático chamado `testSave`. Este método executa diferentes cenários de teste para validar o comportamento do controlador.

Dependências

A classe depende das seguintes entidades:

- **Classe `MyProfilePageController`**: O controlador que está sendo testado.
- **Objeto `User`**: Representa os usuários do Salesforce.
- **Página Visualforce `ChangePassword`**: Utilizada para redirecionar o usuário para a página de alteração de senha.

Imports

Não há importações explícitas, pois o Apex não utiliza declarações de importação como outras linguagens.

Variáveis

A classe utiliza as seguintes variáveis:

- `existingPortalUsers`: Lista de usuários do portal existentes na organização.
- `currentUser`: Representa o usuário atual.
- `existingPortalUser`: Representa um usuário do portal existente.
- `randFax`: Número de fax gerado aleatoriamente para teste.

Métodos

`testSave()`

Este é o único método da classe e realiza os seguintes testes:

1. **Carregar o usuário atual**:
    - Verifica se o controlador carrega corretamente os detalhes do usuário atual.

- Valida que o modo de edição (`isEdit`) está desativado por padrão.

2. **Alternar entre modos de edição e visualização**:
   - Testa os métodos `edit()` e `cancel()` para alternar entre os modos de edição e visualização.
3. **Alterar informações do usuário**:
   - Atualiza o número de fax do usuário e salva as alterações.
   - Valida que as alterações foram persistidas corretamente no banco de dados.
4. **Testar com usuários do portal existentes**:
   - Executa os mesmos testes acima para um usuário do portal existente usando o método `System.runAs()`.

Exemplo

A classe de teste não é utilizada diretamente, mas é executada como parte dos testes unitários no Salesforce. Para executar o teste, use o seguinte comando no Developer Console:

```
@IsTest
public class MyProfilePageControllerTest {
    static void testSave() {
        Test.startTest();
        MyProfilePageControllerTest.testSave();
        Test.stopTest();
    }
}
```

Diagrama de Dependência

O diagrama abaixo ilustra as dependências entre as classes e objetos utilizados:

```
classDiagram
    class MyProfilePageControllerTest {
        +testSave()
    }
    class MyProfilePageController {
        +getUser()
        +getIsEdit()
        +edit()
        +cancel()
        +save()
        +changePassword()
    }
    class User {
        +id
        +profileId
        +userRoleId
        +title
        +firstname
        +lastname
        +email
        +phone
        +mobilephone
        +fax
        +street
        +city
        +state
        +postalcode
        +country
    }
    class Page {
        +ChangePassword
    }

    MyProfilePageControllerTest --> MyProfilePageController
    MyProfilePageController --> User
    MyProfilePageController --> Page
```

Notas

- A anotação `@IsTest(SeeAllData=true)` permite que o teste acesse dados reais na organização. Isso pode ser útil para testes específicos, mas deve ser usado com cuidado para evitar dependências de dados reais.
- O método `System.runAs()` é utilizado para simular a execução de código como um usuário específico.

Vulnerabilidades

- **Dependência de dados reais**: O uso de `SeeAllData=true` pode causar falhas nos testes se os dados reais forem alterados ou removidos.
- **Falta de validação de entrada**: O código não valida os dados inseridos pelo usuário, como o número de fax. Isso pode levar a problemas de integridade de dados.

Esta documentação detalhada deve ajudar qualquer desenvolvedor a entender o propósito e funcionamento da classe `MyProfilePageControllerTest`.