Testes Unitários de Cadastro

```
it('should submit the form', () => {
    spyOn(component, 'CadastrarUsuario'); // Espiona o método
CadastrarUsuario para verificar se é chamado corretamente
    const form = component.Form;

    form.controls['email'].setValue('gabriel@gmail.com');
    form.controls['nome_completo'].setValue('Gabriel Elias');
    form.controls['senha'].setValue('password');

    const submitButton =
fixture.nativeElement.querySelector('button[type="submit"]');
    submitButton.click();

    expect(component.CadastrarUsuario).toHaveBeenCalled();
```

Este teste verifica se o formulário é enviado corretamente quando o botão de submit é clicado. Ele preenche os campos do formulário e simula o clique no botão de submit, em seguida, verifica se o método Cadastrar usuário é chamado corretamente.

```
it('should call exibirlogin() when "Faça login" link is clicked', () =>
{
    spyOn(component, 'exibirlogin'); // Espiona o método exibirlogin
para verificar se é chamado corretamente

    const loginLink = fixture.debugElement.query(By.css('.card-link'));
    loginLink.nativeElement.click();

    expect(component.exibirlogin).toHaveBeenCalled();
```

Este teste verifica se o método exibir login() é chamado corretamente quando o link "Faça login" é clicado. Ele usa a função spiOn para espionar o método ‘‘exibirlogin’’ e, em seguida, simula o clique no link usando fixture.debugElement.query(By.css('.card-link')) para selecionar o link e ‘’nativeElement.click()’’ para disparar o evento de clique. Por fim, ele verifica se o método ‘‘exibirlogin’’ foi chamado corretamente.

Teste unitário de postagem

```
it('should call CadQuadra.Salvar with the correct parameters', () => {
  component.Armazenar();
  expect(cadQuadraMock.Salvar).toHaveBeenCalledWith({
    endereco: 'Test Endereço',
    numero: 'Test Número',
    imagem: undefined,    nome_usuario: undefined   });
});
```

No teste em si, chamamos o método Armazenar() do componente e, em seguida, verificamos se o método “CadQuadra.Salvar()” foi chamado com os parâmetros corretos, usando o método toHaveBeenCalledWith() do Jasmine.

Teste de componentes

```
 import { ComponentFixture, TestBed } from '@angular/core/testing';


import { AcessoComponent } from './acesso.component';


describe('AcessoComponent', () => {
  let component: AcessoComponent;
  let fixture: ComponentFixture<AcessoComponent>;


  beforeEach(async () => {
    await TestBed.configureTestingModule({
      declarations: [ AcessoComponent ]
    })
    .compileComponents();


    fixture = TestBed.createComponent(AcessoComponent);
    component = fixture.componentInstance;
    fixture.detectChanges();
  });


  it('should create', () => {
    expect(component).toBeTruthy();
  });
```

Dentro do teste, estamos usando a função expect() do Jasmine para verificar se o componente foi criado com sucesso. Neste caso, estamos verificando se componente é verdadeiro (ou seja, não é nulo).

```
 it('should enable submit button when form is valid', () => {
    component.formulario.controls['nome'].setValue('Gabriel Elias');

component.formulario.controls['email'].setValue('gabrielelias@gmail.com
');
    component.formulario.controls['senha'].setValue('password');
    fixture.detectChanges();
```

O teste verifica se o botão de envio é habilitado quando o formulário é preenchido corretamente. Para simular isso, definimos valores nos controles do formulário nome, email e senha, e então verificamos se o atributo disabled do elemento de botão é false.

Teste de sistemas

```
import { browser, element, by } from 'protractor';


describe('Teste de Sistema', () => {
  beforeEach(() => {
    browser.get('/');
  });

  it('deve exibir o título correto na página inicial', () => {
    const titulo = element(by.css('h1')).getText();
    expect(titulo).toEqual('Quaddra');
  });

  it('deve preencher e enviar o formulário de contato com sucesso', ()
=> {
    const nomeInput = element(by.css('input[name="nome"]'));
    const emailInput = element(by.css('input[name="email"]'));
    const mensagemTextarea =
element(by.css('textarea[name="mensagem"]'));
    const enviarButton = element(by.css('button[type="submit"]'));

    nomeInput.sendKeys('Gabriel');
    emailInput.sendKeys('gabrielelias@gmail.com');
    mensagemTextarea.sendKeys('Esta é uma mensagem de teste.');
    enviarButton.click();
```

```
    const sucessoMsg = element(by.css('.mensagem-sucesso')).getText();
    expect(sucessoMsg).toEqual('Formulário enviado com sucesso!');
  });
});
```

Usamos o Protractor para navegar até a página inicial do aplicativo Angular antes de cada teste. Em seguida, usamos o Protractor para localizar elementos na página usando seletores CSS e realizar ações, como preencher campos de entrada e clicar em botões. Nos testes usamos o expect do Jasmine para verificar se os resultados são os esperados. No primeiro teste, verificamos se o título exibido na página inicial é "Quaddra". No segundo teste, preenchemos o formulário de contato com alguns dados de exemplo, clicamos no botão de envio e verificamos se uma mensagem de sucesso é exibida corretamente.

Atributos de Qualidade de Software

1- Usabilidade:Aplicativo foi desenvolvido com o foco na facilidade de manuseio do usuário, com passos fáceis e intuitivos.

2- Manutenibilidade: Aplicativo com clareza no código, fácil modificação e forma de evoluir.

3 - Testabilidade: Aplicativo tem a disponibilidade de ser testado a qualquer momento com ferramentas de teste e capacidade de reproduzir condições de teste.

4- Funcionalidade: Foi desenvolvido com foco em atender funcionalidades e recursos esperados,atendendo aos requisitos do usuário.