

**Escola Superior de Tecnologia e Gestão**  
Programação de Sistemas de Informação  
Comunicação e Tecnologias de Informação

# Right Price

Gestão de orçamentos

**Autores:**

- Hélder Alexandre Rosa Abrantes, n.º 2190785
- Marcos Santos Ferreira, n.º 2190779
- Rui Paulo Lopes Agostinho, n.º 2190783

**Destinatário:**

- Professor Carlos José da Rocha Ferreira



## Resumo

O objetivo deste trabalho é consolidar conhecimentos na área de Plataformas de Sistemas de Informação, aplicando os conceitos adquiridos na unidade de formação. Este trabalho corresponde a uma das componentes da UC de Projeto em PSI (Servidor Web) e consiste no desenvolvimento do back-office, front-office, RBAC e testes de software desta componente.

**Palavras-Chave:** role; instalador; fornecedor; testes;

## Índice

<b>ÍNDICE DE FIGURAS.....</b>	<b>3</b>
<b>INTRODUÇÃO .....</b>	<b>4</b>
<b>OBJETIVOS.....</b>	<b>5</b>
<b>REQUISITOS FUNCIONAIS .....</b>	<b>5</b>
<b>CASOS DE USO .....</b>	<b>6</b>
<b>USER STORIES .....</b>	<b>7</b>
<b>ROLES E PERMISSÕES.....</b>	<b>10</b>
<b>PLANO DE TESTES .....</b>	<b>11</b>
TESTES UNITÁRIOS .....	11
TESTES FUNCIONAIS.....	14
<b>CONCLUSÃO .....</b>	<b>19</b>
<b>REFERENCIAS BIBLIOGRÁFICAS .....</b>	<b>20</b>

## Índice de Figuras

Figura 1 - Casos de uso .....	6
Figura 2 - Exemplo Sistema de Permissões .....	10
Figura 3 - ClienteTest - Validações de inserção.....	11
Figura 4 - OrcamentoTest (OrcamentoSave) - Validações de inserção .....	11
Figura 5 - OrcamentoTest (OrcamentoTestOwner) - Valida o Dono do orçamento....	12
Figura 6 - FriendTest(AddFriend) - Validar campos vazios ou repetidos ou incorreto	13
Figura 7 - SignupCest.....	14
Figura 8 - LoginCest.....	15
Figura 9 - ContactCest.....	16
Figura 10 - ProdutosCest.....	17
Figura 11 - FriendCest.....	18

## Introdução

Neste projeto pretendesse criar uma plataforma de gestão em que se centralize o conjunto de dois tipos de atividades já existentes no mercado da construção civil. Atualmente o mercado da construção civil funciona da seguinte forma: um cliente (Terceiro, não um utilizador da aplicação) solicita um orçamento ou cotação a um instalador (utilizador da aplicação) que por sua vez, este utilizador recorre a fornecedores (utilizadores da aplicação) para saber o valor de determinado equipamento.

Este processo via email ou conversa telefónica seria demorado pelo que teria de haver trocas entre entidades para que se chegasse a um acordo entre os dois para o preço do equipamento. Pelo que a nossa equipa chegou a conclusão que este processo podia ser melhorado com uma aplicação de gestão centralizada.

Por isso, no espírito de facilitar o processo aos demais e de dar a conhecer aos utilizadores da aplicação o estado das suas atividades, recorremos a Plataforma Yii2 para solucionar este problema.

Criamos uma página de acesso para estes dois tipos de utilizadores (Instalador e Fornecedor) no frontend da aplicação e deixamos o controlo de acesso pelos administradores no Backend.

O instalador tem acesso aos produtos dos fornecedores que pretender, e pode fazer fichas de cliente, e orçamentos com os produtos disponibilizados pelos fornecedores, na página principal encontrará o estado da sua conta.

O fornecedor no seu espaço pessoal encontrará uma página que o deixa adicionar os seus produtos e preços, noutra página poderá visualizar os instaladores que o adicionaram para comprar dos seus produtos.

## Objetivos

- Agilizar o processo de criação de orçamentos e obras.
- Registrar instaladores e seus clientes.
- Registrar fornecedores e produtos.
- Aceder as informações do orçamento
- Gestão pormenorizada de orçamentos.
- Adicionar imagens a utilizadores e produtos.

## Requisitos Funcionais

- CRUD Utilizadores, Produtos, Cliente, Orçamentos, Categorias.
- O login deve validar o tipo de utilizador e apresentar o backoffice respetivo.
  - a. No Frontend não deve de permitir o acesso de administradores.
  - b. No Backend não deve de haver acesso de instaladores e fornecedores.
- Cada utilizador do tipo instalador, pode:
  - a. Inserir os seus clientes, adicionar orçamentos para esses clientes.
  - b. Fazer orçamentos com produtos disponibilizados pelos fornecedores que permitiu.
  - c. Inserir a margem de lucro que pretende ao orçamento.
  - d. Inserir produtos aos orçamentos de vários fornecedores.
  - e. Ver todos os fornecedores.
  - f. Estatística: N° de clientes, N° de orçamentos, total valor Orçamentado, detalhes de cada cliente.
- Cada utilizador do tipo fornecedor, pode:
  - a. Inserir e eliminar os produtos.
  - b. Visualizar os instaladores que o adicionaram.

- c. Estatísticas: número de produtos disponíveis, numero de produtos usados em encomendas, valor total em encomendas e os detalhes de produtos usados em encomendas.
- O administrador, pode:
  - a. Validar o registo (bloquear, ativar) dos utilizadores (Instalador, Fornecedor);
  - b. Visualizar o número de instaladores, número de Fornecedores, numero de clientes por categoria.

## Casos de Uso

Representação em UML:

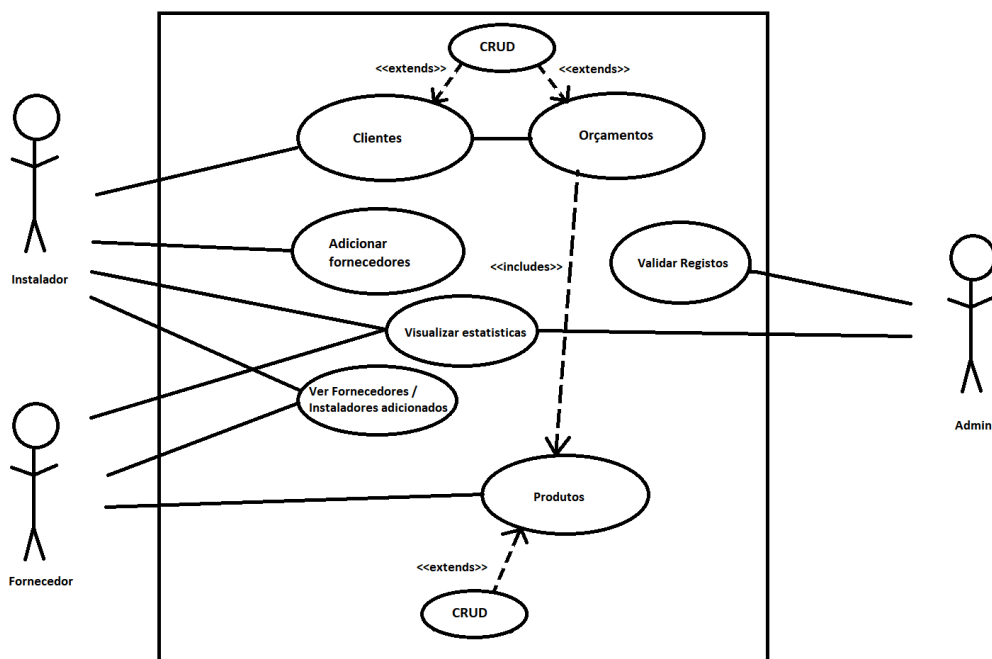


Figura 1 - Casos de uso



## User Stories

- User Story 1: Registrar um novo utilizador
  - Como: utilizador
  - Quero: fazer o meu registo e definir a minha Role e categoria
  - Para: poder fazer o login no front-end pessoal da aplicação.
  - Critérios de Aceitação:
    - Aceder a página de registo
  
- User Story 2: Fazer o login
  - Como: utilizador
  - Quero: fazer o login para aceder ao meu espaço pessoal
  - Critérios de Aceitação:
    - Aceder a página de login
    - Registo validado pelo admin
  
- User Story 3: Clientes
  - Como: Instalador
  - Quero: Adicionar clientes.
  - Para: Adicionar Orçamentos ao respetivo cliente
  - Critérios de Aceitação:
    - Role: Instalador;
    - Login efetuado no frontend;
    - Aceder a página de clientes
  
- User Story 4: Orçamentos
  - Como: Instalador
  - Quero: Adicionar Orçamentos aos Clientes
  - Para: Adicionar produtos ao orçamento e calcular o total do valor dos produtos adicionados.
  - Critérios de Aceitação:
    - Role: Instalador;

- Login efetuado no frontend;
  - Aceder a página de um cliente e clicar no botão adicionar Orçamento;
- 
- User Story 5: Adicionar Produtos aos Orçamentos
    - Como: Instalador
    - Quero: Adicionar Produtos aos Orçamentos
    - Para: poder saber o total do valor do orçamento.
    - Critérios de Aceitação:
      - Role: Instalador;
      - Login efetuado no frontend;
      - Aceder a página de um orçamento;
      - Ter adicionado Fornecedores;
- 
- User Story 6: Adicionar Fornecedores
    - Como: Instalador
    - Quero: Adicionar Fornecedores
    - Para: Ter acesso aos seus produtos nos orçamentos
    - Critérios de Aceitação:
      - Role: Instalador;
      - Login efetuado no frontend;
- 
- User Story 7: Visualizar as Estatísticas da conta
    - Como: Instalador, fornecedor.
    - Quero: Ter uma área pessoal
    - Para: poder visualizar os valores atuais da conta.
    - Critérios de Aceitação:
      - Role: Instalador, Fornecedor;
      - Login efetuado no frontend;

- User Story 8: Adicionar Produtos
  - Como: Fornecedor
  - Quero: adicionar Produtos e respetivos características
  - Para: que os instaladores possam adicionar aos seus orçamentos
  - Critérios de Aceitação:
    - Role: Fornecedor;
    - Login efetuado no frontend;
  
- User Story 9: Visualizar quem adicionei ou quem me adicionou
  - Como: Fornecedor, instalador
  - Quero: ver com quem estou relacionado de momento.
  - Critérios de Aceitação:
    - Role: Fornecedor, instalador;
  
- User Story 10: Validar o Registo.
  - Como: Admin
  - Quero: visualizar os utilizadores com e sem permissão de acesso à aplicação.
  - Para: poder controlar os utilizadores.
  - Critérios de Aceitação:
    - Role: admin;
    - Login efetuado no backoffice

## Roles e Permissões

Neste projeto consideramos as 3 roles (admin, fornecedor e instalador) no entanto não usamos qualquer tipo de permissão.

Optamos por não usar permissões porque no sentido da nossa aplicação os controladores são respetivos a uma só role, pelo que (como mostra a figura abaixo) optamos por permitir o acesso aos controladores a uma só role.

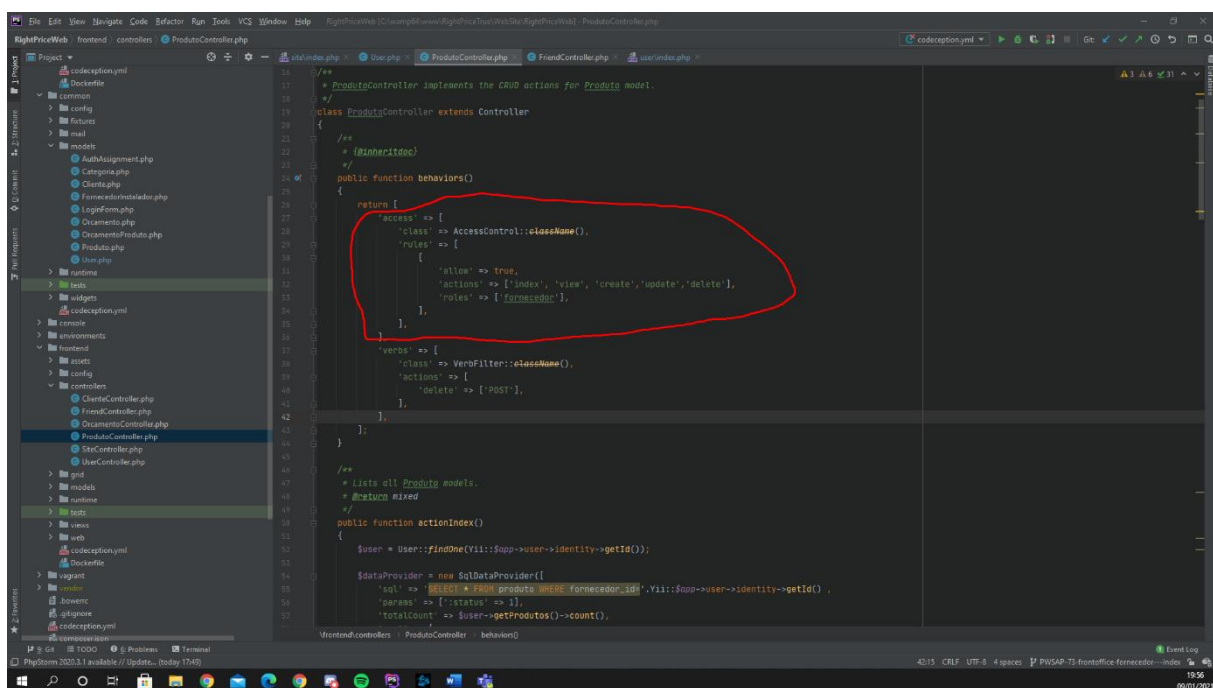


Figura 2 - Exemplo Sistema de Permissões

## Plano de testes

### Testes Unitários

- FrontEnd - ClienteTest - Validações de inserção

Valida se ao inserir na base de dados o modelo contem o email correto, nif correto, e se o nome tem o tamanho apropriado.

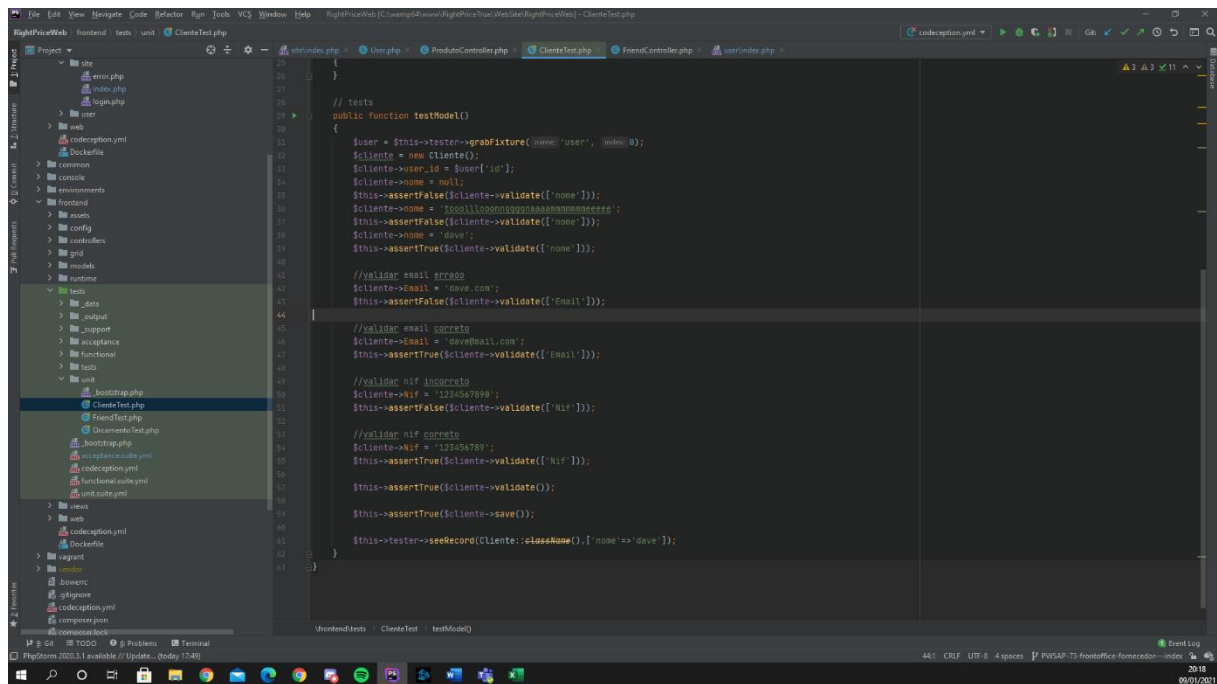


Figura 3 - ClienteTest - Validações de inserção

- FrontEnd - OrcamentoTest (OrcamentoSave) - Validações de inserção

Valida se ao inserir na base de dados o modelo contem o id do cliente



Figura 4 - OrcamentoTest (OrcamentoSave) - Validações de inserção

- FrontEnd - OrcamentoTest (OrcamentoTestOwner) - Valida o Dono do orçamento

Esta função testa a função `getOwner` do modelo `Orcamento` com o objectivo de devolver o dono do orçamento.

```

58
59 public function testOrcamentoTestOwner()
60 {
61     // esta função testa que o dono deste orçamento é o utilizador 2
62     // sabendo que o utilizador 2 criou o cliente 2
63     $orcamento = new Orcamento();
64     $orcamento->cliente_id = 2;
65
66     $this->assertTrue($orcamento->save());
67     $this->assertEquals( expected: '2', $orcamento->getOwner());
68 }
  
```

Figura 5 - OrcamentoTest (OrcamentoTestOwner) - Valida o Dono do orçamento

- FrontEnd - OrcamentoTest (OrcamentoTotal) - Valida o total do orçamento

Esta função testa o total do orçamento aplicando as margens do orçamento.

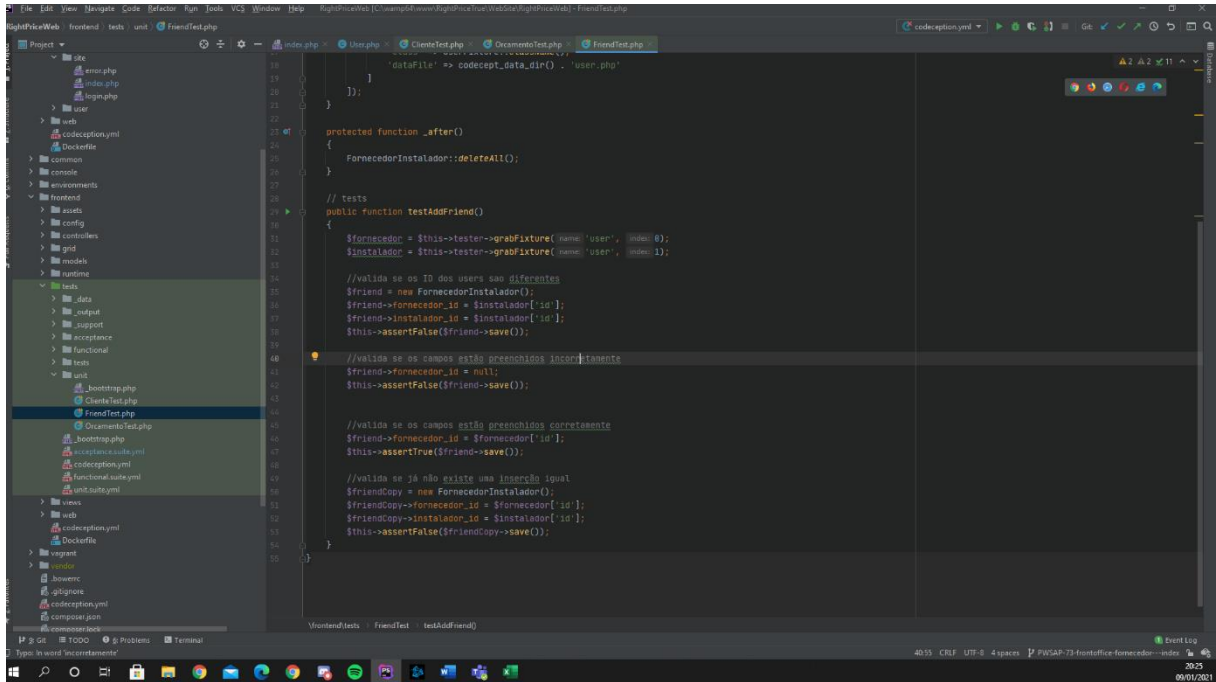
Começa por validar a inserção de dois produtos ao orçamento e de seguida calcula o total esperado.

```

57 $this->assertEquals( expected: '2', $orcamento->getOwner());
58 }
59
60 public function testOrcamentoTotal()
61 {
62     // esta função testa o total do orçamento aplicando as margens do orçamento
63     $orcamento = new Orcamento();
64     $orcamento->cliente_id = 2;
65     $orcamento->margem = 1;
66     $orcamento->nome = 'orçadia';
67     $this->assertTrue($orcamento->save());
68     $this->tester->seeRecord(Orcamento::className(), ['nome' => 'orçadia']);
69
70     //Adicionar 1 produto
71     $orcProd = new OrcamentoProduto();
72
73     $orcProd->orcamento_id = $orcamento['id'];
74     $orcProd->produto_id = 1;
75     $orcProd->quantidade = 1;
76     $this->assertTrue($orcProd->save());
77     $this->tester->seeRecord(OrcamentoProduto::className(), ['orcamento_id' => $orcamento['id'], 'produto_id' => '1']);
78
79     $this->assertEquals( expected: '18.1', $orcamento->getTotal());
80
81     $orcProd->orcamento_id = $orcamento['id'];
82     $orcProd->produto_id = 2;
83     $orcProd->quantidade = 2;
84     $this->assertTrue($orcProd->save());
85     $this->tester->seeRecord(OrcamentoProduto::className(), ['orcamento_id' => $orcamento['id'], 'produto_id' => '2']);
86
87     // $this->tester->seeInDatabase('orcamento_produto', ['orcamento_id' => $orcamento['id'], 'produto_id' => '2']);
88     $this->assertEquals( expected: '29.921000000000002', $orcamento->getTotal());
89 }
90
  
```

- FrontEnd - FriendTest(AddFriend) - Validar campos vazios ou repetidos ou incorreto

Esta Função testa o modelo Fornecedor Instalador para validar se os id dos users são diferentes, se estão preenchidos, se estão preenchidos corretamente e se já não se encontra inserido um registo igual.



```

18         'dataFile' => codecept_data_dir() . 'user.php'
19     );
20 }
21
22
23 protected function _after()
24 {
25     FornecedorInstalador::deleteAll();
26 }
27
28 // tests
29 public function testAddFriend()
30 {
31     $fornecedor = $this->tester->grabFixture('name', 'user', 'id', 0);
32     $instalador = $this->tester->grabFixture('name', 'user', 'id', 1);
33
34     //valida se os ID dos users são diferentes
35     $friend = new FornecedorInstalador();
36     $friend->fornecedor_id = $instalador['id'];
37     $friend->instalador_id = $instalador['id'];
38     $this->assertFalse($friend->save());
39
40     //valida se os campos estão preenchidos incorretamente
41     $friend->fornecedor_id = null;
42     $this->assertFalse($friend->save());
43
44     //valida se os campos estão preenchidos corretamente
45     $friend->fornecedor_id = $fornecedor['id'];
46     $this->assertTrue($friend->save());
47
48     //valida se já não existe uma inserção igual
49     $friendCopy = new FornecedorInstalador();
50     $friendCopy->fornecedor_id = $fornecedor['id'];
51     $friendCopy->instalador_id = $instalador['id'];
52     $this->assertFalse($friendCopy->save());
53 }
54
55

```

Figura 6 - FriendTest(AddFriend) - Validar campos vazios ou repetidos ou incorreto

## Testes Funcionais

- Frontend - SignupCest - Registar um User

Este teste valida os campos necessários para o registo de um utilizador.

```
<?php

namespace frontend\tests\functional;

use common\models\User;
use frontend\tests\FunctionalTester;

class SignupCest
{
    protected $formId = 'form-signup';

    public function _before(FunctionalTester $I)
    {
        $I->amOnRoute('route:site/signup');
    }

    public function signupWithEmptyFields(FunctionalTester $I)
    {
        $I->see('text:Signup', 'selector:h1');
        $I->see('text:Please fill out the following fields to signup:');
        $I->submitForm($this->formId, []);
        $I->see('text:Username cannot be blank.');
```

```
        $I->see('text:Email cannot be blank.');
```

```
        $I->see('text>Password cannot be blank.');
```

```
    }

    public function signupWithWrongEmail(FunctionalTester $I)
    {
        $I->submitForm(
            $this->formId, [
                'SignupForm[username]' => 'tester',
                'SignupForm[email]' => 'ttttt',
                'SignupForm[password]' => 'tester_password',
            ]
        );
        $I->dontSee('text:Username cannot be blank.');
```

```
        $I->dontSee('text>Password cannot be blank.');
```

```
        $I->see('text:Email is not a valid email address.');
```

```
    }

    public function signupSuccessfully(FunctionalTester $I)
    {
        $I->fillField(['name' => 'SignupForm[username]', 'value' => 'tester']);
        $I->fillField(['name' => 'SignupForm[name]', 'value' => 'tester hello']);
        $I->fillField(['name' => 'SignupForm[email]', 'value' => 'tester.email@example.com']);
        $I->fillField(['name' => 'SignupForm[password]', 'value' => 'tester_password']);

        $I->selectOption(['name' => 'SignupForm[categoria_id]', 'option' => 'Canalização']);
        $I->seeOptionIsSelected('selector:SignupForm[categoria_id]', 'optionText:Canalização');
```

```
        $I->selectOption(['name' => 'SignupForm[role]', 'option' => 'fornecedor']);
        $I->seeOptionIsSelected('selector:SignupForm[role]', 'optionText:Fornecedor');
```

```
        $I->click('link:signup-button');
```

```
        $I->seeRecord(User::className(), [
            'username' => 'tester',
            'email' => 'tester.email@example.com',
            'status' => \common\models\User::STATUS_INACTIVE
        ]);

        $I->seeEmailIsSent();
        $I->see('text:Thank you for registration. Please check your inbox for verification email.');
```

```
    }
}
```

Figura 7 - SignupCest



- Frontend - LoginCest - Fazer Login

Este teste confirma que o sistema de login trabalha corretamente.

```
<?php

namespace frontend\tests\Functional;

use frontend\tests\FunctionalTester;
use common\fixtures\UserFixture;

class LoginCest
{
    /**
     * Load fixtures before db transaction begin
     * Called in _before()
     * @see \Codeception\Module\Yii2::_before()
     * @see \Codeception\Module\Yii2::loadFixtures()
     * @return array
     */
    public function _fixtures()
    {
        return [
            'user' => [
                'class' => UserFixture::className(),
                'dataFile' => codecept_data_dir() . 'login_data.php',
            ],
        ];
    }

    public function _before(FunctionalTester $I)
    {
        $I->amOnRoute( route: 'site/login');
    }

    protected function formParams($login, $password)
    {
        return [
            'LoginForm[username]' => $login,
            'LoginForm[password]' => $password,
        ];
    }

    public function checkEmpty(FunctionalTester $I)
    {
        $I->submitForm( selector: '#login-form', $this->formParams( login: '', password: ''));
        $I->see( text: 'Username cannot be blank. ');
        $I->see( text: 'Password cannot be blank. ');
    }

    public function checkWrongPassword(FunctionalTester $I)
    {
        $I->submitForm( selector: '#login-form', $this->formParams( login: 'admin', password: 'wrong'));
        $I->see( text: 'Incorrect username or password. ');
    }

    public function checkInactiveAccount(FunctionalTester $I)
    {
        $I->submitForm( selector: '#login-form', $this->formParams( login: 'test.test', password: 'Test1234'));
        $I->see( text: 'Incorrect username or password. ');
    }

    public function checkValidLogin(FunctionalTester $I)
    {
        $I->submitForm( selector: '#login-form', [
            'LoginForm[username]' => 'grau',
            'LoginForm[password]' => 'password_0',
        ]);
        $I->see( text: 'Logout' );
        $I->see( text: 'grau' );
        $I->dontSeeLink( text: 'Login');
        $I->dontSeeLink( text: 'Signup');
    }
}
```

Figura 8 - LoginCest

- Frontend - ContactCest - Enviar um formulário de contacto

```
<?php

namespace frontend\tests\functional;

use frontend\tests\FunctionalTester;
use common\fixtures\UserFixture;

class LoginCest
{
    /**
     * Load fixtures before db transaction begin
     * Called in _before()
     * @see \Codeception\Module\Yii2::_before()
     * @see \Codeception\Module\Yii2::loadFixtures()
     * @return array
     */
    public function _fixtures()
    {
        return [
            'user' => [
                'class' => UserFixture::className(),
                'dataFile' => codecept_data_dir() . 'login_data.php',
            ],
        ];
    }

    public function _before(FunctionalTester $I)
    {
        $I->amOnRoute('route: site/login');
    }

    protected function formParams($login, $password)
    {
        return [
            'LoginForm[username]' => $login,
            'LoginForm[password]' => $password,
        ];
    }

    public function checkEmpty(FunctionalTester $I)
    {
        $I->submitForm('selector: #login-form', $this->formParams('login:', 'password:'));
        $I->see('text: Username cannot be blank.');
```

Figura 9 - ContactCest

- Frontend – Produtos

```
<?php

namespace frontend\tests\functional;

use frontend\tests\FunctionalTester;
use common\fixtures\UserFixture;

class LoginCest
{
    /**
     * Load fixtures before db transaction begin
     * Called in _before()
     * @see \Codeception\Module\Yii2::_before()
     * @see \Codeception\Module\Yii2::loadFixtures()
     * @return array
     */
    public function _fixtures()
    {
        return [
            'user' => [
                'class' => UserFixture::className(),
                'dataFile' => codecept_data_dir() . 'login_data.php',
            ],
        ];
    }

    public function _before(FunctionalTester $I)
    {
        $I->amOnRoute('route: site/login');
    }

    protected function formParams($login, $password)
    {
        return [
            'LoginForm[username]' => $login,
            'LoginForm[password]' => $password,
        ];
    }

    public function checkEmpty(FunctionalTester $I)
    {
        $I->submitForm('selector: #login-form', $this->formParams('login: ', 'password: '));
        $I->see('text: Username cannot be blank.');
```

Figura 10 - ProdutosCest

- Frontend - FriendController

```
<?php namespace frontend\tests\functional;
use common\fixtures\UserFixture;
use frontend\tests\FunctionalTester;

class FriendCest
{
    // APENAS A VISÃO DO INSTALADOR

    public function _fixtures()
    {
        return [
            'user' => [
                'class' => UserFixture::className(),
                'dataFile' => codecept_data_dir() . 'login_data.php',
            ],
        ];
    }

    public function _before(FunctionalTester $I)
    {
        $I->amOnRoute( route: 'site/login');
        $I->submitForm( selector: '#login-form', [
            'LoginForm[username]'=>'erau',
            'LoginForm[password]'=>'password_0',
        ]);
        $I->see( text: 'erau' );
        $I->amOnRoute( route: 'user/view?id=1');
        $I->see( text: 'sfriesen@jenkins.info' );
    }

    // tests
    public function tryAddFriend(FunctionalTester $I)
    {
        //para adicionar um fornecedor
        $I->amOnRoute( route: 'friend/index');
        $I->see( text: 'Test');
        $I->click( link: 'Adicionar', context: '#Test');
        $I->click( link: 'As suas conexoes');
        $I->amOnRoute( route: 'friend/view');
        $I->see( text: 'Test');
    }

    public function tryRemoveFriend(FunctionalTester $I)
    {
        //para remover um fornecedor
        $this->tryAddFriend($I);
        $I->amOnRoute( route: 'friend/view');
        $I->see( text: 'Test');
        $I->click( link: 'Remover', context: '#Test');
        $I->click( link: 'Adicionar Fornecedores');
        $I->see( text: 'Test');
    }
}
```

Figura 11 - FriendCest

## Conclusão

Com a conclusão deste Projeto, pretendemos dar a conhecer aos nossos utilizadores uma forma mais prática e rápida para a gestão dos seus negócios. Tivemos a oportunidade de ver o quanto fácil é o registo e a gestão de clientes por parte do instalador e em poucos cliques demos acesso a vários produtos de diversos fornecedores.

Na ótica do fornecedor, demos oportunidade de disponibilizar os seus produtos a diversos instaladores e oferecemos um conjunto de informação detalhada dos produtos que os instaladores estão utilizando nos seus orçamentos.

A nível de controlo administrativo, oferecemos ao administrador uma ferramenta que permite controlar o acesso à aplicação a distância dum clique.

Para nós desenvolvedores tivemos a oportunidade de trabalhar com uma framework php avançada o que reduziu significativamente o tempo de desenvolvimento e deu-nos um conjunto de conhecimentos que podem ser aplicados futuramente.

Também tivemos a oportunidade de fazer testes a nossa aplicação com a framework codeception o que nos levou a dar uma assistência a nível de controlo de erros.

## Referencias Bibliográficas

*Codeception - PHP Testing framework*. (Dezembro de 2020). Obtido de Codeception:  
<https://codeception.com/>

*The Definitive Guide to Yii2*. (Dezembro de 2020). Obtido de Yii PHP Framework:  
<https://www.yiiframework.com/>