

# **Documentação**

# **SIG – Sistema Integrado de**

# **Gestão**

# Ficha Técnica

---

## Equipe Responsável pela Elaboração

Bruno Alves da Silva - Analista de Sistemas/Negócios

Lorraine Silva Oliveira – Analista de Testes

Victor Hugo Lopes dos Santos Costa - Analista de Sistemas/Negócios

Willian Barreto Lope - Database Administrator

---

## Público Alvo

Este manual destina-se a equipe de desenvolvimento do modulo financeiro do sistema de ERP SIG – Sistema Integrado de Gestão.

---

Versão 2.0 - São Vicente , novembro de 2017.

---

Dúvidas, críticas e sugestões devem ser encaminhadas por escrito para o seguinte endereço eletrônico:

Bruno Alves da Silva – contato@alvesbruno.com

Willian Barreto Lopes – willianblopes@gmail.com

Victor Hugo Lopes dos Santos Costa – victor.hlscoستا@gmail.com

Lorraine Silva Oliveira – lorraineosilva@gmail.com

Recomendamos que o assunto seja identificado com o título desta obra. Alertamos ainda para a importância de se identificar o endereço e o nome completos do remetente para que seja possível o envio de respostas.

---

Windows e Microsof Word são marcas registradas da Microsoft Corporation

---

# Sumário

<b>INTRODUÇÃO .....</b>	<b>7</b>
<b>Visão geral deste documento.....</b>	<b>7</b>
<b>Convenções, termos e abreviações .....</b>	<b>8</b>
1. Identificação dos Requisitos .....	8
2. Prioridades dos Requisitos .....	8
<b>ESCOPO GERAL DO SISTEMA .....</b>	<b>9</b>
<b>Abrangência e sistemas relacionados .....</b>	<b>9</b>
<b>Descrição dos usuários .....</b>	<b>12</b>
1. CLIENTE .....	12
2. COMPRADOR.....	12
3. VENDEDOR .....	12
4. FINANCEIRO .....	12
5. GESTÃO .....	12
6. TÉCNICO .....	12
<b>DIAGRAMAS DE CASO DE USO .....</b>	<b>13</b>
<b>COMPRAS .....</b>	<b>13</b>
<b>COMERCIAL .....</b>	<b>14</b>
<b>FINANCEIRO .....</b>	<b>15</b>
<b>REQUISITOS FUNCIONAIS .....</b>	<b>16</b>
<b>COMPRAS .....</b>	<b>16</b>
[RF001] Solicitação de compras .....	16
[RF002] Ambiente de cotação de produtos .....	16
[RF003] Análise das propostas de compras .....	17
[RF004] Recebimento das propostas.....	17
[RF005] Cadastro de produtos.....	18
[RF006] Emissão de ordem de pagamento .....	19
[RF007] Faturamento .....	19
[RF008] Atualização de estoque .....	20
<b>VENDAS .....</b>	<b>20</b>
[RF009] Solicitação de orçamento .....	20
[RF010] Geração de orçamento.....	21
[RF011] Aprovação do orçamento .....	21
[RF012] Emissão da ordem de cobrança.....	22
[RF013] Geração de remessa para entrega .....	22
<b>TÉCNICO.....</b>	<b>23</b>

[RF014] Prestação de serviço .....	23
<b>GESTÃO .....</b>	<b>24</b>
[RF015] Serviços prestados .....	24
<b>FINANCEIRO .....</b>	<b>24</b>
[RF016] Contas .....	24
[RF017] Gestão de movimentações administrativas .....	25
[RF018] Emissão de relatórios .....	25
<b>REQUISITOS NÃO FUNCIONAIS .....</b>	<b>26</b>
<b>USABILIDADE .....</b>	<b>26</b>
[NF001] Responsividade .....	26
[NF002] Ajuda .....	26
<b>CONFIABILIDADE .....</b>	<b>26</b>
[NF003] Redundancia .....	26
[NF004] Severidade de falhas .....	26
<b>DESEMPENHO .....</b>	<b>26</b>
[NF005] Transferência mínima .....	26
[NF006] Processos complexos .....	26
<b>SEGURANÇA .....</b>	<b>26</b>
[NF007] Autenticação .....	27
[NF008] Criptografia .....	27
<b>DISTRIBUIÇÃO .....</b>	<b>27</b>
[NF009] Controle de acesso ao sistema .....	27
<b>PADRÕES .....</b>	<b>27</b>
[NF010] Validação W3C .....	27
<b>HARDWARE E SOFTWARE .....</b>	<b>27</b>
[NF011] navegadores .....	27
[NF012] Memória e espaço físico .....	27
[NF013] Sistemas operacionais .....	28
<b>ARQUITETURA DO SISTEMA .....</b>	<b>29</b>
<b>MVC .....</b>	<b>29</b>
<b>LARAVEL .....</b>	<b>31</b>
<b>MYSQL .....</b>	<b>31</b>
<b>FERRAMENTAS DE APOIO .....</b>	<b>31</b>
<b>BANCO DE DADOS .....</b>	<b>32</b>
<b>MODELO CONCEITUAL .....</b>	<b>32</b>

MODELO RELACIONAL .....	33
ANÁLISE DE RISCO .....	34
CODIFICAÇÃO E LINGUAGEM.....	36
INTERFACE GRÁFICA .....	37
CARACTERÍSTICAS HUMANAS .....	37
CARACTERÍSTICAS DE UMA BOA TELA.....	37
TELA DE LISTAGEM.....	38
TELA DE CADASTRO .....	38
TELA DE DETALHES .....	39
TELAS EM PRODUÇÃO.....	39
CÓDIGO FONTE.....	49
VIEW - Home .....	49
VIEW - Departamento .....	54
VIEW - Usuários .....	57
VIEW - Logon .....	61
VIEW - Produtos.....	66
VIEW - Contas .....	70
VIEW - Movimentações .....	76
Controller - Departamento .....	82
Controller - Logon .....	84
Controller - Usuários .....	85
Controller - Produtos.....	87
Controller - Contas .....	90
Controller - Movimentações .....	92
Model - Departamento .....	95
Model - Usuários .....	96
Model - Produtos .....	96
Model - Contas .....	97

Model – Movimentações .....	97
<b>PLANO DE TESTE .....</b>	<b>99</b>
REQUISITOS A SEREM TESTADOS .....	99
ESTRATÉGIAS E FERRAMENTAS DE TESTE .....	125
EQUIPE E INFRAESTRUTURA .....	125
CRONOGRAMA DE ATIVIDADES .....	126
<b>CONSIDERAÇÕES FINAIS .....</b>	<b>128</b>
<b>REFERÊNCIA BIBLIOGRÁFICA .....</b>	<b>129</b>

# Introdução

Este documento técnico especifica todos os requisitos **Sistema de Integrado de Gestão – SIG**, fornecendo aos desenvolvedores e leitores as informações necessárias para seu desenvolvimento, assim como para a realização dos testes e homologação do sistema.

## Visão geral deste documento

Esta introdução fornece as informações necessárias para fazer um bom uso deste documento, explicitando seus objetivos e as convenções que foram adotadas no texto, além de conter uma lista de referências para outros documentos relacionados. As demais seções apresentam a especificação do Sistema de Integrado de Gestão – SIG e estão organizadas como descrito abaixo.

- **Seção 2** – Descrição geral do sistema: apresenta uma visão geral do sistema, caracterizando qual é o seu escopo e descrevendo seus usuários.
- **Seção 3** – Diagramas de caso de uso: nessa seção são especificado os casos de usos de compra de produtos, comercialização de produtos e serviços e do setor financeiro.
- **Seção 4** – Requisitos funcionais: especifica todos os requisitos funcionais do sistema, descrevendo os fluxos de eventos, prioridades, atores, entradas e saídas de cada caso de uso a ser implementado.
- **Seção 5** – Requisitos não funcionais: especifica todos os requisitos não funcionais do sistema, divididos em requisitos de usabilidade, confiabilidade, desempenho, segurança, distribuição, adequação a padrões e requisitos de hardware e software.
- **Seção 6** – Arquitetura do sistema: nessa seção especifica o modelo de arquitetura utilizado na construção do sistema, além de demonstrar as ferramentas de apoio como Laravel, Git e Github, além de demonstrar o processo de desenvolvimento que todos os envolvidos do projeto deverão utilizar.
- **Seção 7** – Banco de dados: nessa seção estão o modelo conceitual, modelo relacional e script sql do banco de dados.
- **Seção 8** – Codificação e linguagem: nessa seção é apresentada as linguagem de programação utilizada para o desenvolvimento, além das linguagens de apoio, como por exemplo html, css e etc.

- **Seção 9** – Interface gráfica: especifica os modelos de telas que precisam ser utilizadas como modelo em todo o projeto, além de demonstrar as características humanas x boas práticas de criação de tela.

## Convenções, termos e abreviações

A correta interpretação deste documento exige o conhecimento de algumas convenções e termos específicos, que são descritos a seguir.

### 1. Identificação dos Requisitos

Por convenção, a referência a requisitos é feita através do nome da subseção onde eles estão descritos, seguido do identificador do requisito, de acordo com o esquema abaixo:

[nome da subseção.identificador do requisito]

Por exemplo, o requisito [Recuperação de dados.RF016] está descrito em uma subseção chamada “Recuperação de dados”, em um bloco identificado pelo número [RF016]. Já o requisito não funcional [Confiabilidade.NF008] está descrito na seção de requisitos não funcionais de Confiabilidade, em um bloco identificado por [NF008].

### 2. Prioridades dos Requisitos

Para estabelecer a prioridade dos requisitos foram adotadas as denominações “essencial”, “importante” e “desejável”.

- **Essencial** é o requisito sem o qual o sistema não entra em funcionamento. Requisitos essenciais são requisitos imprescindíveis, que têm que ser implementados impreterivelmente.
- **Importante** é o requisito sem o qual o sistema entra em funcionamento, mas de forma não satisfatória. Requisitos importantes devem ser implementados, mas, se não forem, o sistema poderá ser implantado e usado mesmo assim.
- **Desejável** é o requisito que não compromete as funcionalidades básicas do sistema, isto é, o sistema pode funcionar de forma satisfatória sem ele. Requisitos desejáveis são requisitos que podem ser deixados para versões posteriores do sistema, caso não haja tempo hábil para implementá-los na versão que está sendo especificada.



## Escopo geral do sistema

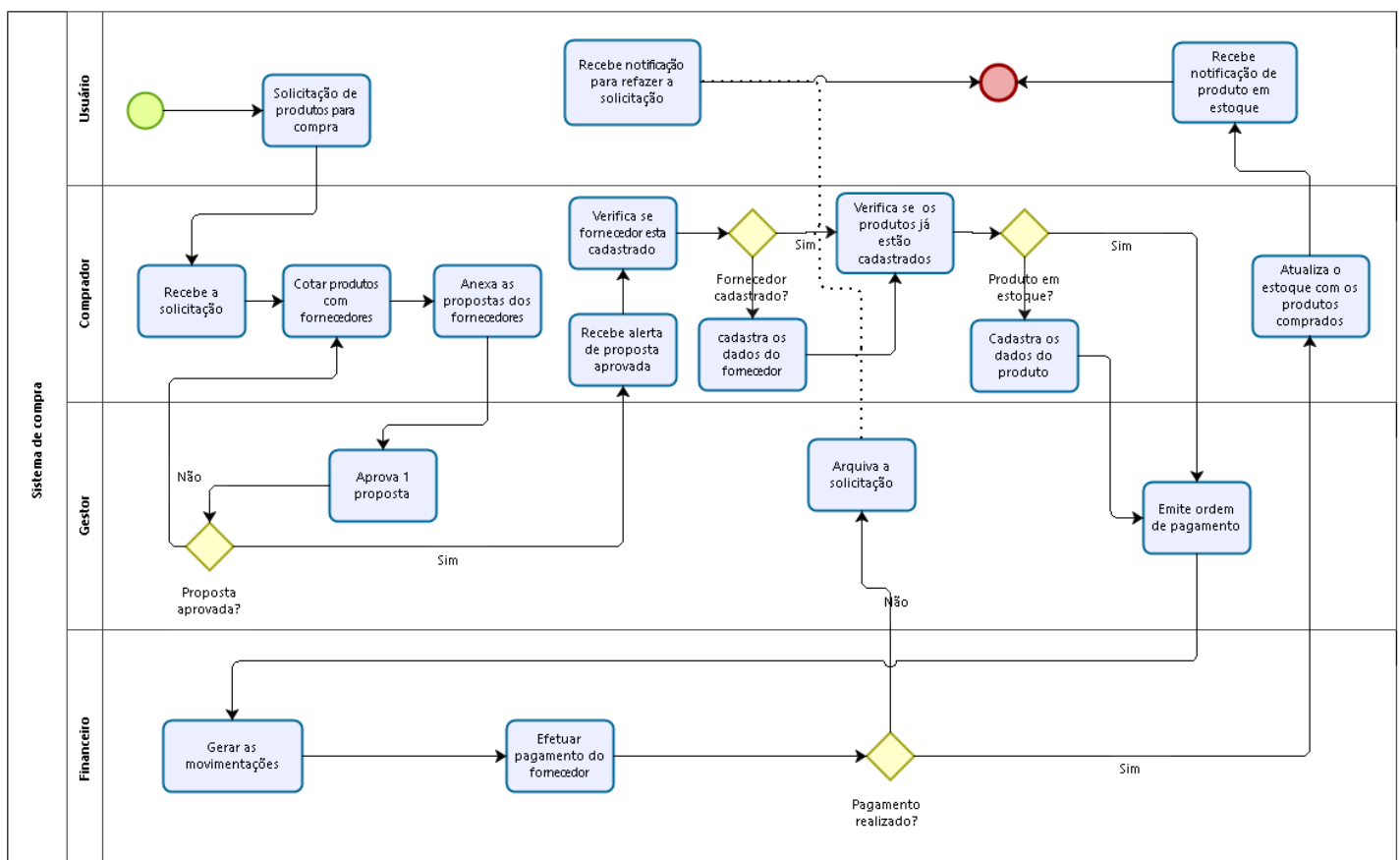
Com o crescimento do mercado de venda de produtos e prestação de serviço em geral, a procura por um sistema online de apoio as tarefas administrativas, financeiras, compra e venda de produtos se tornou o carro chefe do mercado tecnológico.

Pensando no cenário atual, será desenvolvido o **Sistema Integrado de Gestão – SIG** um ERP - Enterprise Resource Planning (Planejamento dos Recursos da Empresa) quem tem como objetivo auxiliar e automatizar os fluxos de compra e venda da empresa, mapeando desde a cotação de novos produtos até a sua venda para o mercado, tudo isso de maneira online e de fácil acesso a qualquer dispositivo mobile e desktop.

### Abrangência e sistemas relacionados

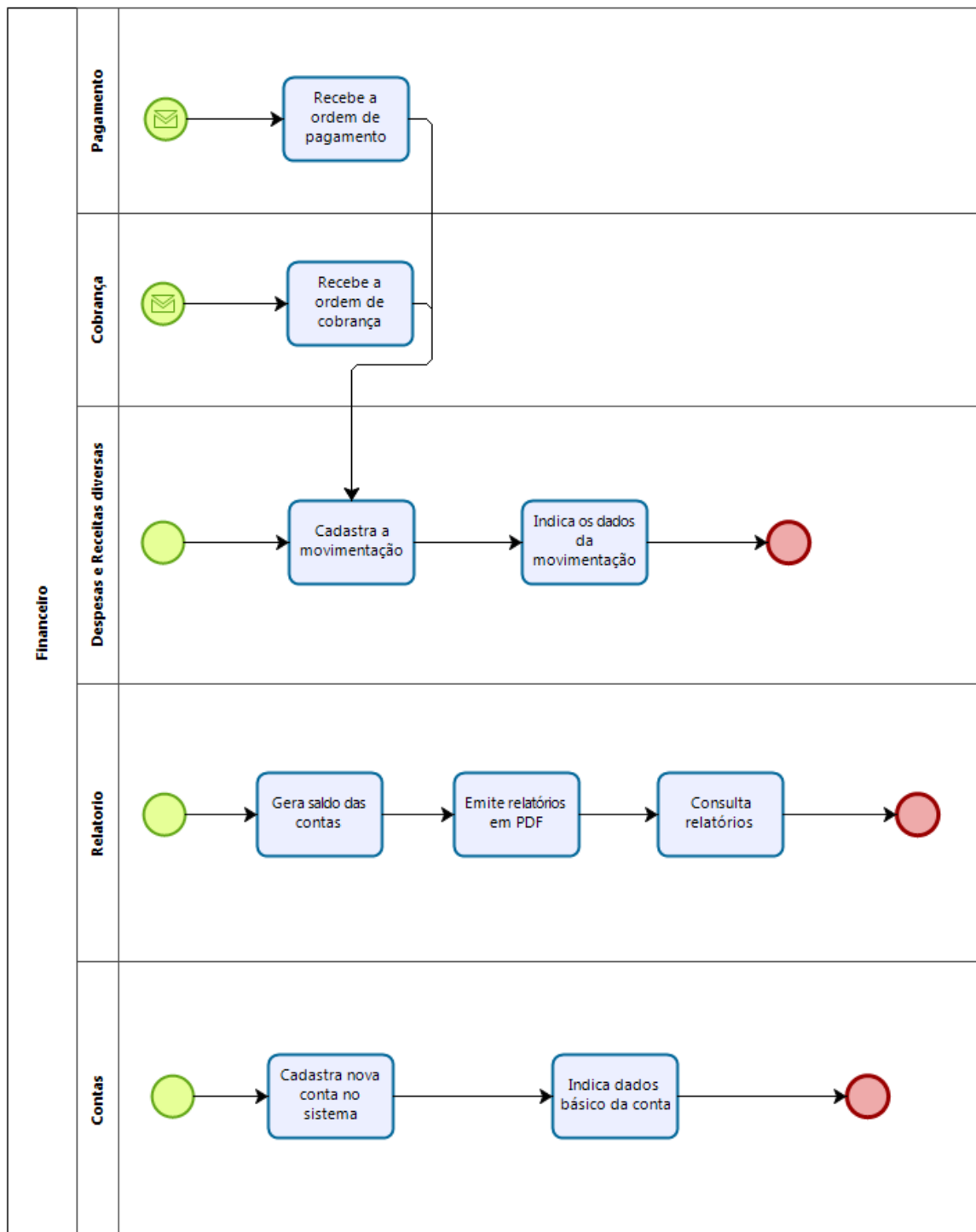
O SIG funcionará apoiado em processos pré definidos, onde cada usuário terá suas rotinas automatizadas e implementadas em atividades sequências, o mesmo possuirá 03 (três) processos principais:

**Compras:** Processo responsável pela cotação de novos produtos para empresa, geração de propostas, cadastro e atualização dos dados de fornecedores, cadastro e atualização dos produtos no estoque, emissão e controle das ordens de pagamento e geração de movimentações (abertura de novo contrato e movimentação de compra).





**Financeiro:** Processo responsável pelo faturamento de todas as transações de compra e vendas realizadas na empresa, e também controle dos saldos financeiros das contas da empresa.



Mesmo sendo apresentado separadamente, o fluxo financeiro possui presença marcante dentro dos os outros dois processos, pois é nele que são realizadas as liberações e movimentações de compra e venda.

## Descrição dos usuários

O sistema deverá possuir 6 (seis) tipos de usuários, onde cada possuirá suas atividades e processos interligados entre si, abrangendo os 3 (três) processos principais apresentados anterior, os usuários serão:

### 1. **CLIENTE**

Usuário responsável pela solicitação de orçamentos de compra de produtos.

### 2. **COMPRADOR**

Usuário responsável pela cotação de novos produtos, cadastro e atualização de dados de fornecedores e cadastro e atualização de produtos no estoque.

### 3. **VENDEDOR**

Usuário responsável pela elaboração e resposta dos pedidos de orçamento, geração da ordens de cobrança e emissão da remessa de entrega e prestação de serviço.

### 4. **FINANCEIRO**

Usuário responsável pela emissão das movimentações financeiras de compra de produtos, gerenciamento das ordens de pagamento de fornecedores, emissão e gerenciamento das cobrança de vendas de produtos e gerenciamento dos saldos das contas da empresa.

### 5. **GESTÃO**

Usuário responsável pela análise e liberação das solicitações de cotação de produtos, emissão da ordem de pagamento de fornecedores e visualização dos saldos das contas da empresa.

### 6. **TÉCNICO**

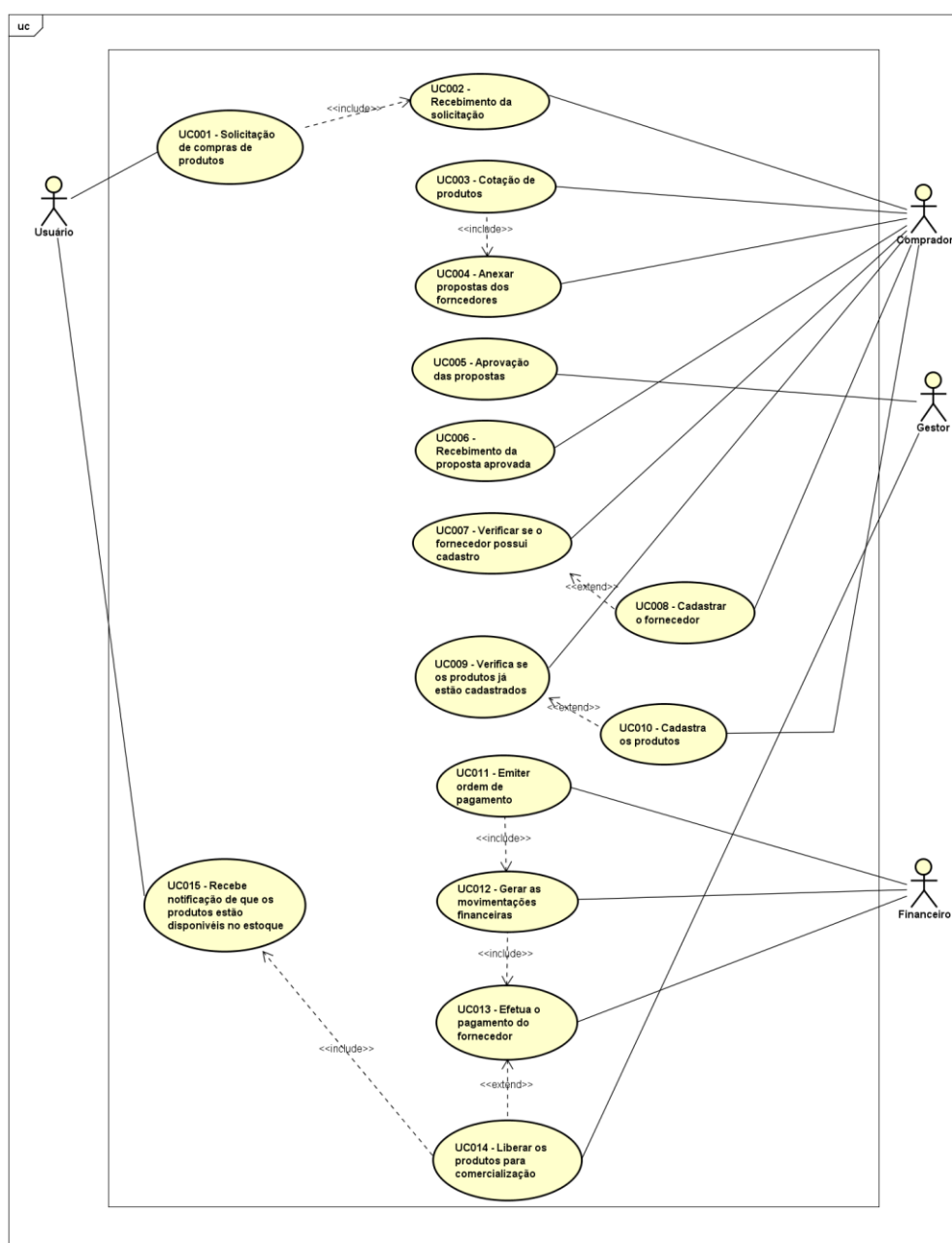
Usuário responsável pela prestação de serviços contratados pelo cliente.

# Diagramas de Caso de Uso

Após analisarmos os três processos principais para o funcionamento de uma empresa de venda de produtos e prestação de serviço, foi gerado três diagrama de caso de uso principais, que descriminam as ações necessárias para funcionamento, e quais são os autores de cada ação.

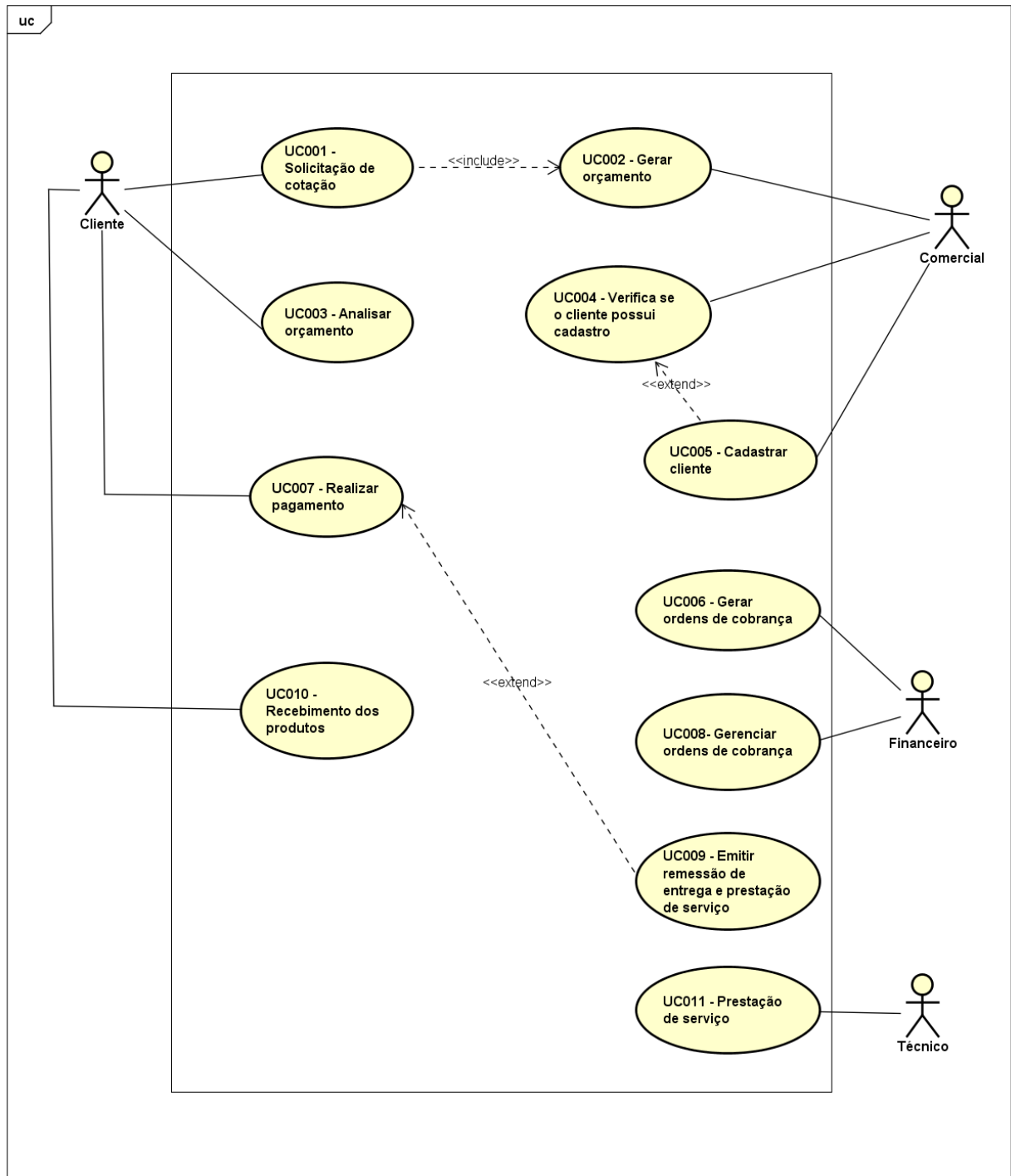
## COMPRAS

Diagrama de caso de uso responsável pela descrição de todos os requisitos que o processo de compras precisa ter para ser realizado, e também quais são os autores responsáveis pela interações das ações.



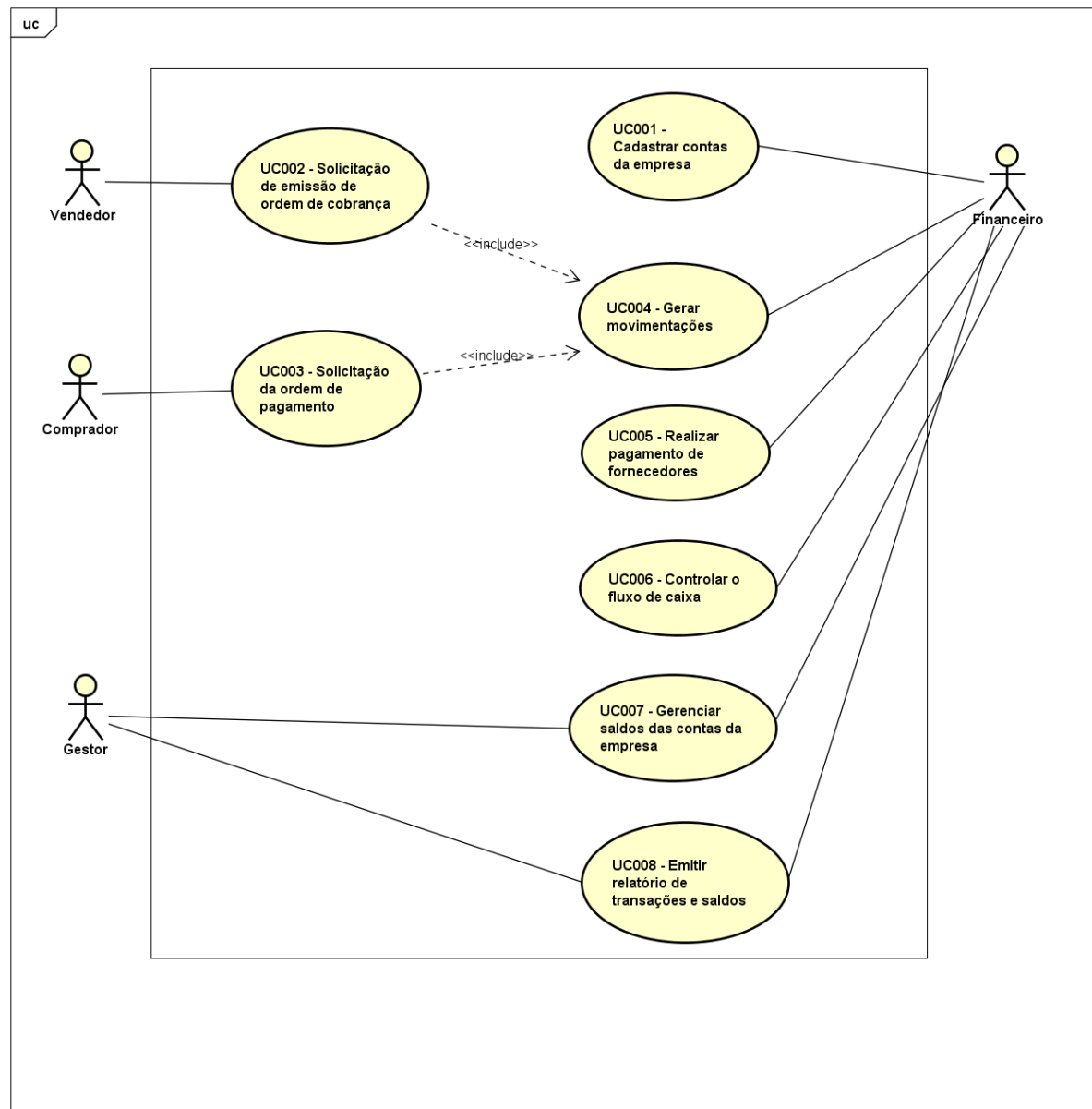
## COMERCIAL

Diagrama de caso de uso responsável pela descrição de todos os requisitos que o processo comercial precisa ter para ser realizado, e também quais são os autores responsáveis pela interações das ações.



## FINANCEIRO

Diagrama de caso de uso responsável pela descrição de todos os requisitos que o processo financeiro precisa ter para ser realizado, e também quais são os autores responsáveis pela interações das ações.



# Requisitos funcionais

## COMPRAS

Nessa seção estão todos os requisitos funcionais encontrados no processo de compra de produtos para comercialização.

### [RF001] Solicitação de compras

O sistema deverá ter um ambiente para solicitação de cotação de compras de produtos para empresa, onde o usuário irá dizer quais produtos precisam ser comprados. Além, de poder visualizar em qual passo o seu pedido está parado em com quem ele está, ele também deverá ter permissão de visualizar todas as informações geradas a partir do seu pedido de cotação, como por exemplo: propostas coletadas, proposta aprovada e propostas rejeitadas, data dos status, nota fiscal do fornecedor, produtos cotados e etc.

**Ator:** Usuário.

**Prioridade:**    ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O usuário deverá estar logado no sistema e ter permissão para solicitação de cotação de produtos.

**Saídas e pós condições:** Após o formulário de cotação ser preenchido, os dados deverão ser armazenados no sistema, e o pedido de cotação deverá ficar com o status de “Aguardando cotação”. Após realizar essas atividades o sistema deverá disponibilizar a cotação gerada para o responsável pela procura de fornecedores.

### Fluxo de eventos principal

- 1- O usuário loga no sistema;
- 2- O usuário abre uma nova cotação de compras de produtos;
- 3- O usuário preenche o formulário de cotação, com o nome do produto e descrição do mesmo e
- 4- Usuário confirma os dados do formulário preenchido e cadastra o mesmo no sistema.

### [RF002] Ambiente de cotação de produtos

O sistema deverá ter um ambiente para que o comprador possa receber e visualizar todos os pedidos de cotação de produtos solicitados, e posteriormente possa anexar as propostas dos fornecedores e enviar para o gestor aprovar.

**Ator:** Comprador.

**Prioridade:**    ☐ Essencial                      ☒ Importante                      ☐ Desejável

**Entradas e pré condições:** O comprador deverá estar logado no sistema e ter permissão para visualizar as cotações e anexar as propostas dos fornecedores.

**Saídas e pós condições:** Após o comprador anexar todas as propostas enviadas pelos fornecedores, a mesma deverá ser enviada para o gestor analisar e aprovar uma proposta, e também o status dessa cotação deverá ser alterado para “Aguardando aprovação”.



### Fluxo de eventos principal

- 1- O comprador loga no sistema;
- 2- O comprador acessa o ambiente de cotações pedentes para realização;
- 3- O comprador solicita as propostas para os fornecedores;
- 4- O comprador anexa as propostas coletadas dos fornecedores na cotação e
- 5- O comprador envia a cotação com as propostas anexadas para o gestor analisar.

### [RF003] Análise das propostas de compras

O sistema deverá ter um ambiente para que o gestor possa visualizar todas as propostas anexadas no pedido de cotação realizado pelo usuário, nesse ambiente ele deverá possuir a permissão de selecionar qual propostas ele irá aprovar.

**Ator:** Gestor.

**Prioridade:**     ☐ Essencial                    ☒ Importante                    ☐ Desejável

**Entradas e pré condições:** O gestor deverá estar logado no sistema e ter permissão para visualizar as propostas dos fornecedores anexadas na cotações de compras de produtos.

**Saídas e pós condições:** Após o gestor analisar as propostas anexadas no pedido de cotação de produtos, ele deverá ter a opção de selecionar uma proposta e envia-lá para o comprador dar entrada no cadastro de fornecedor e de produtos. Ele também deverá ter a opção de rejeitar todas as propostas e envia-lás novamente para o “ambiente de cotação de produtos”.

### Fluxo de eventos principal

- 1- O gestor loga no sistema;
- 2- O gestor acessa o ambiente de análise das propostas de cotação;
- 3- O gestor analisa as propostas anexadas no pedido de cotação de produtos;
- 4- O gestor aprova uma das propostas anexadas, e envia a mesma para o comprador dar sequencia no cadastro de fornecedor e de produtos. Nesse momento ele também altera o status do pedido de cotação para “Cotação liberada”.
- 5- Ou o gestor reprova todas as cotações e envia novamente o pedido de cotação de produtos para o “ambiente de cotação de produtos”, com o status de “Propostas reprovadas”.

### [RF004] Recebimento das propostas

O sistema deverá ter um ambiente para o recebimento da proposta aprovada e cadastro dos dados do fornecedor que irá fornecer os produtos solicitados na cotação.

**Ator:** Comprador.

**Prioridade:**     ☒ Essencial                    ☐ Importante                    ☐ Desejável

**Entradas e pré condições:** O comprador deverá estar logado no sistema e ter permissão para visualizar as propostas aprovadas e também permissão para dar entrada no cadastro de fornecedores.

**Saídas e pós condições:** Após o comprador receber a proposta aprovada do pedido de cotação de produtos, ele irá pesquisar se o fornecedor já está cadastrado no sistema e caso ele esteja só será realizado a atualização dos dados cadastrais, ou se não será realizado o cadastro do novo fornecedor no sistema, após preencher todos os dados ou confirma-lós o pedido de cotação juntamente com a proposta aprovada serão enviados para o ambiente de entrada de produtos no estoque.

### Fluxo de eventos principal

- 1- O comprador loga no sistema;
- 2- O comprador acessa o ambiente de recebimento de propostas;
- 3- O comprador verifica se o fornecedor possui cadastro ativo no sistema, caso ele possua cadastro só será realizado a atualização e confirmação dos dados cadastrais;
- 4- Caso o fornecedor não possua um cadastro ativo no sistema, será preenchidos os seguintes dados no sistema: Nome, razão social, nome fantasia, cpf ou cnpj, inscrição estadual, endereço, bairro, cidade, estado, tipo de pessoa (física ou jurídica) e situação cadastral e
- 5- Após os dados serem preenchidos ou atualizados pelo comprador, o pedido de cotação será enviado para o ambiente de entrada de produtos no estoque com o status do pedido de cotação em “Cadastro dos produtos no estoque”.

### [RF005] Cadastro de produtos

O sistema deverá ter um ambiente para o cadastro de novos produtos e atualização de estoque dos produtos existentes já no sistema.

**Ator:** Comprador.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O comprador deverá estar logado no sistema e ter permissão para visualizar os produtos já cadastrados no sistema, e também permissão para cadastrar novos produtos.

**Saídas e pós condições:** Após o comprador cadastrar ou atualizar o estoque dos produtos, os produtos deverão ficar aguardando a liberação do gestor para que eles possam estar disponíveis para comercialização.

### Fluxo de eventos principal

- 1- O comprador loga no sistema;
- 2- O comprador acessa o ambiente de cadastro de produtos;
- 3- O comprador verifica se o produto já está cadastrado no sistema, caso ele esteja só será realizada atualização da quantidade em estoque do produto,
- 4- Caso o produto não esteja cadastrado no sistema, o comprador irá realizar o cadastro do mesmo, preenchendo os seguintes dados: Nome, descrição, valor, foto, quantidade mínima no estoque, quantidade máxima no estoque e quantidade atual em estoque.
- 5- Após os dados serem preenchidos ou atualizados pelo comprador, o pedido de cotação será enviado para o ambiente de “emissão de ordem de pagamento” com o status do pedido de cotação em “Emissão de ordem de pagamento”.

## [RF006] Emissão de ordem de pagamento

O sistema deverá ter um ambiente para a emissão de ordem pagamento para as propostas aprovadas dos pedidos de cotação de produtos.

**Ator:** Gestor.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O gestor deverá estar logado no sistema e ter permissão para visualizar as cotações que estão aguardando a emissão da ordem de pagamento.

**Saídas e pós condições:** Após o gestor emitir a ordem de pagamento do fornecedor, a cotação deverá ser enviada para o “ambiente financeiro” para que seja realizada as movimentações de faturamento da cotação, ao realizar essa ação a cotação deverá ter seu status alterado para “Em faturamento”.

### Fluxo de eventos principal

- 1- O gestor loga no sistema;
- 2- O gestor acessa o ambiente de emissão de ordem de pagamento;
- 3- O gestor verifica se foi realizado corretamente o cadastro do fornecedor e dos produtos no sistema;
- 4- Após verificar todas as informações, o gestor libera o pedido de cotação para o ambiente de faturamento.

## [RF007] Faturamento

O sistema deverá ter um ambiente de faturamento para gerar as movimentações de pagamento do fornecedor e abertura de contrato de compra de mercadorias.

**Ator:** Financeiro.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O financeiro deverá estar logado no sistema e ter permissão realizar as movimentações de pagamento do fornecedor e abertura de contrato.

**Saídas e pós condições:** Após o financeiro dar o pagamento como aprovado, o pedido de cotação deverá ser enviado para o “ambiente de atualização de estoque” para que os produtos fiquem disponíveis para comercialização. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “Aguardando liberação do produto no estoque”.

### Fluxo de eventos principal

- 1- O financeiro loga no sistema;
- 2- O financeiro acessa o ambiente de faturamento;
- 3- O financeiro seleciona um pedido de cotação;
- 4- O financeiro analisa o pedido de cotação aprovado pelo gestor e abre o contrato de compra de produtos;
- 5- Após o contrato ser aberto, o financeiro emite a movimentação de pagamento no sistemas.

- 6- O financeiro fica aguardando a confirmação de recebimento do pagamento com o fornecedor e
- 7- Após o fornecedor confirmar o recebimento do pagamento, e anexada a nota fiscal do fornecedor no contrato, e o pedido de cotação e enviado para o comprador liberar os produtos no estoque.

### [RF008] Atualização de estoque

O sistema deverá ter um ambiente onde o comprador possa visualizar as cotações que o pagamento já foi recebido pelo fornecedor, e que estão aguardando a entrega dos produtos pelo fornecedor.

**Ator:** Comprador.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O comprador deverá estar logado no sistema e ter permissão para realizar a atualização de produtos no estoque do sistema.

**Saídas e pós condições:** Após os produtos serem entregues pelo fornecedor e dada entrada dos mesmos no sistema, deverá ser emitida uma notificação para o usuário que os produtos solicitados já estão disponíveis no sistema. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “cotação realizada”.

### Fluxo de eventos principal

- 1- O comprador loga no sistema;
- 2- O comprador acessa o ambiente de atualização de estoque;
- 3 - O comprador seleciona o pedido de cotação que será dado como entregue os produtos e
- 4 – O comprador libera os produtos para comercialização no sistema.

## VENDAS

Nessa seção estão todos os requisitos funcionais encontrados no processo de vendas de produtos ou prestação de serviços.

### [RF009] Solicitação de orçamento

O sistema deverá ter um ambiente para solicitação de orçamento de produtos, onde o cliente irá dizer quais produtos ele irá orçar.

**Ator:** Cliente

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O cliente deverá estar logado no sistema para que possa solicitar um orçamento.

**Saídas e pós condições:** Após o formulário de orçamento estiver totalmente preenchido, os dados deverão ser armazenados no sistema, onde um vendedor irá “Gerar o Orçamento” preenchendo com as informações relevantes ao cliente. Feito isso o cliente será notificado para avaliar se o orçamento está de acordo onde ele tomará a decisão de

comprar ou não os produtos, ao realizar essa ação o status do pedido de orçamento deverá ser alterado para “Aguardando geração do orçamento”.

### Fluxo de eventos principal

- 1- O cliente loga no sistema;
- 2- O cliente abre um novo orçamento de produtos ou serviços;
- 3- O cliente preenche o formulário de orçamento, selecionando os produtos e serviços desejados e
- 4- O cliente confirma os dados do formulário preenchido e cadastra o mesmo no sistema.

### [RF010] Geração de orçamento

O sistema deverá conter um ambiente onde o(s) vendedor(es) verão todos os orçamentos a serem respondidos.

**Ator:** Vendedor

**Prioridade:**    ☐ Essencial                    ☒ Importante                    ☐ Desejável

**Entradas e pré condições:** O vendedor deverá estar logado no sistema para que possa ver e preencher os orçamentos em aberto.

**Saídas e pós condições:** Após preencher os orçamentos o vendedor deverá enviar o orçamento para o cliente avaliar o mesmo, ao realizar essa ação o status do pedido de orçamento deverá ser alterado para “Aguardando aprovação do cliente”.

### Fluxo de eventos principal

- 1- O vendedor loga no sistema;
- 2- O vendedor escolher um orçamento para responder;
- 3- O vendedor preenche o orçamento, selecionando os produtos e serviços que o cliente necessita.
- 4- O vendedor confirma os dados preenchidos e cadastra o mesmo no sistema.

### [RF011] Aprovação do orçamento

O sistema deverá ter um ambiente onde o vendedor verifica os orçamentos com status “Aprovado pelo cliente “, para cadastrar o cliente no sistema.

**Ator:** Vendedor

**Prioridade:**    ☒ Essencial                    ☐ Importante                    ☐ Desejável

**Entradas e pré condições:** O vendedor deverá estar logado no sistema para que ele possa ver os orçamentos aguardando geração de pedido. O pedido só poderá ser gerado se o orçamento estiver com status “Aprovado pelo cliente”.

**Saídas e pós condições:** Após receber o orçamento aprovado pelo cliente, o vendedor irá pesquisar se o cliente já está cadastrado no sistema e caso ele esteja só será realizado a atualização dos dados cadastrais, ou se não será realizado o cadastro do novo cliente no

sistema, após preencher todos os dados ou confirma-lós o orçamento será enviado para o ambiente de geração de ordem de cobrança.

### Fluxo de eventos principal

- 1- O vendedor loga no sistema;
- 2- O vendedor escolher um orçamento com status “Aprovado pelo cliente”;
- 3- O vendedor gera o pedido, confirmando os produtos solicitados e valores e
- 4- O vendedor envia o orçamento para geração de cobrança.

### [RF012] Emissão da ordem de cobrança

O sistema deverá ter um ambiente de faturamento para gerar as movimentações de cobranças do cliente e abertura de contrato de venda de mercadorias ou prestação de serviços.

**Ator:** Financeiro.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O financeiro deverá estar logado no sistema e ter permissão para realizar as movimentações de cobrança dos clientes e abertura de contrato.

**Saídas e pós condições:** Após o financeiro dar o pagamento como aprovado, o pedido juntamente com o contrato serão enviados para o ambiente de emissão de remessão de entrega de produtos ou prestação de serviço. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “Aguardando liberação da remessão de entrega ou prestação de serviço”.

### Fluxo de eventos principal

- 1- O financeiro loga no sistema;
- 2- O financeiro acessa o ambiente de emissão da ordem de cobrança;
- 3- O financeiro seleciona um pedido aprovado pelo cliente;
- 4- O financeiro gera o contrato de venda de produtos ou prestação de serviço;
- 5- O financeiro gera a ordem de cobrança;
- 6- O financeiro aguarda o pagamento ser realizado e
- 7- O financeiro libera o pedido juntamente com o contrato para o ambiente de remessa de entrega e prestação de serviço.

### [RF013] Geração de remessa para entrega

O sistema deverá ter um ambiente onde o vendedor possa ver os pedidos com o status “Aguardando liberação da remessão de entrega ou prestação de serviço”, onde ele irá poder gerar a remessa de entrega ou prestação de serviço.

**Ator:** Vendedor

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O vendedor deverá estar logado no sistema para que possa ver os pedidos com status “Aguardando liberação da remessão de entrega ou prestação de serviço”.

**Saídas e pós condições:** Após emitir a remessa de entrega de produtos, deverá ser emitida uma notificação para cliente, informando que os produtos já foram separados e estão em trânsito para entrega. Caso no pedido contenha prestação de serviços, deverá ser enviada o pedido juntamente com o contrato para o técnico analisar e prestar o serviço para o cliente, caso seja necessário a prestação de serviço, o status do pedido deverá ser alterado para “Pedido em análise pela equipe técnica”.

### Fluxo de eventos principal

- 1- O vendedor loga no sistema;
- 2- O vendedor escolher um pedido com status “Aguardando liberação da remessa de entrega ou prestação de serviço”;
- 3- O vendedor confirma os dados de entrega e
- 4 - O vendedor separa os produtos no estoque;
- 5 - O vendedor emite a remessa de entrega de produtos para o cliente;
- 6 - Caso no pedido contenha prestação de serviço, o mesmo é enviado juntamente com o contrato para o setor técnico para que seja realizada a prestação de serviço para o cliente.
- 7 – Caso contrário o pedido deverá ser dado como concluído.

## TÉCNICO

Nessa seção estão todos os requisitos funcionais encontrados no processo de técnico.

### [RF014] Prestação de serviço

O sistema deverá ter um ambiente onde o técnico possa ver os pedidos com o status “Aguardando prestação de serviço”.

**Ator:** Técnico

**Prioridade:** ☐ Essencial ☒ Importante ☐ Desejável

**Entradas e pré condições:** O técnico deverá estar logado no sistema para que possa ver os pedidos com status “Aguardando prestação de serviço”.

**Saídas e pós condições:** Após o técnico prestar o serviço solicitado pelo cliente, o pedido deverá ser encerrado como concluído.

### Fluxo de eventos principal

- 1- O técnico loga no sistema;
- 2- O técnico escolher um pedido com status “Aguardando prestação de serviço”;
- 3- O técnico confirma os dados do local de prestação de serviço;
- 4 - O técnico agenda uma data;
- 5 - O técnico presta o serviço;
- 6 - O técnico dá baixa no pedido de prestação de serviço e
- 7 - O pedido é encerrado como concluído.

## GESTÃO

Nessa seção estão todos os requisitos funcionais encontrados no processo de gestão.

### [RF015] Serviços prestados

O sistema deverá ter um ambiente onde o gestor possa cadastrar e editar os serviços prestados pela empresa.

**Ator:** Gestor.

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O gestor deverá estar logado no sistema.

**Saídas e pós condições:** Após o gestor cadastrar ou editar os serviço no sistema, o mesmo já poderá ser solicitados pelos clientes.

### Fluxo de eventos principal

- 1- O gestor loga no sistema;
- 2- O gestor acessa o ambiente de cadastro de serviços.
- 3 – O gestor edita ou cadastra um novo serviço no sistema;
- 4 – O gestor salva as alteração ou cadastro no sistema e
- 5 – O serviço fica disponível para comercialização.

## FINANCEIRO

Nessa seção estão todos os requisitos funcionais do processo financeiro.

### [RF016] Contas

O sistema deverá ter um ambiente onde o financeiro possa cadastrar, editar e gerenciar todas as contas da empresa.

**Ator:** Financeiro

**Prioridade:**     ☒ Essencial                      ☐ Importante                      ☐ Desejável

**Entradas e pré condições:** O financeiro deverá estar logado no sistema e ter permissão no ambiente de contas.

**Saídas e pós condições:** Após o cadastro ou edição ad conta ser realizada, a mesma já poderá ser utilizada na criação de movimentações.

### Fluxo de eventos principal

- 1- O financeiro loga no sistema;
- 2- O financeiro seleciona uma conta ou cadastra uma nova;
- 3- O financeiro faz as alterações desejaveis e ou cadastra as informações e grava tudo no sistema.
- 4 - A conta fica disponível para utilização.



### [RF017] Gestão de movimentações administrativas

O sistema deverá ter um ambiente onde o financeiro possa cadastrar e gerenciar todas as movimentações financeiras administrativas da empresa.

**Ator:** Financeiro

**Prioridade:**    ☐ Essencial                    ☒ Importante                    ☐ Desejável

**Entradas e pré condições:** O financeiro deverá estar logado no sistema para que possa cadastrar e gerenciar todas as movimentações administrativas da empresa.

**Saídas e pós condições:** Após o financeiro cadastrar novas movimentações ou editar antigas movimentações ele deverá gravar no sistema.

#### Fluxo de eventos principal

- 1- O financeiro loga no sistema;
- 2- O financeiro seleciona uma movimentação ou cadastra uma nova;
- 3- O financeiro faz as alterações desejáveis ou cadastra as informações e
- 4 - A movimentação é salva no sistema.

### [RF018] Emissão de relatórios

O sistema deverá ter um ambiente onde o financeiro ou gestor possa emitir relatórios configuráveis de estoque, contratos, movimentações administrativas/compra/venda, contratos, serviços prestados e contas da empresa.

**Atores:** Financeiro e Gestor.

**Prioridade:**    ☒ Essencial                    ☐ Importante                    ☐ Desejável

**Entradas e pré condições:** O financeiro ou gestor deverá estar logado no sistema para que possa ser gerado os relatórios, esses relatórios só poderão ser emitidos para pessoas com permissão nesses ambientes.

**Saídas e pós condições:** Após selecionar os parâmetros de pesquisa, o relatório deverá ser emitido com opção de geração em PDF.

#### Fluxo de eventos principal

- 1- O financeiro ou gestor loga no sistema;
- 2- O financeiro ou gestor seleciona vai até o ambiente de emissão de relatórios;
- 3- O financeiro ou gestor seleciona os parâmetros desejados para geração do relatório.
- 4 - O relatório é gerado, com opção de download, caso o financeiro ou gestor desejem.

# Requisitos não funcionais

## USABILIDADE

Esta seção descreve os requisitos não funcionais associados à facilidade de uso da interface com o usuário, material de treinamento e documentação do sistema.

### [NF001] Responsividade

O sistema deverá ser responsivo para atingir o maior público possível, ou entregar o mínimo de informações necessárias.

### [NF002] Ajuda

Em cada tela do sistema ou por módulos, o sistema deverá contemplar a opção de consulta de ajuda.

## CONFIABILIDADE

Esta seção descreve os requisitos não funcionais associados à frequência, severidade de falhas do sistema e habilidade de recuperação das mesmas, bem como à correção do sistema.

### [NF003] Redundância

Todo o sistema trabalhará com dupla redundância de banco de dados, para evitar que o sistema fique um número expressivo de tempo fora do ar.

### [NF004] Severidade de falhas

Todo o sistema deverá tratar e notificar a todos os responsáveis as falhas que porventura poderão acontecer, para que possam ser tratadas em uma janela de tempo mínimo.

## DESEMPENHO

Esta seção descreve os requisitos não funcionais associados à eficiência, uso de recursos e tempo de resposta do sistema.

### [NF005] Transferência mínima

O sistema deverá compactar todos os arquivos que serão transferidos e minimizar o número de transferência em cima do protocolo http.

### [NF006] Processos complexos

O sistema deverá indicar visualmente todos os processos onde poderão ocorrer lentidão devido à complexidade do processo que está sendo rodado..

## SEGURANÇA

Esta seção descreve os requisitos não funcionais associados à integridade, privacidade e autenticidade dos dados do sistema.

**[NF007] Autenticação**

O sistema deverá consultar a todo o momento se o usuário está autenticado com as credenciais corretas, quando não, deverá solicitar que ele digite as informações para possa acessar dados internos.

**[NF008] Criptografia**

O sistema deverá criptografar todo os dados a respeito dos usuários com criptografia de 256bits

**DISTRIBUIÇÃO**

Esta seção descreve os requisitos não funcionais associados à distribuição da versão executável do sistema.

**[NF009] Controle de acesso ao sistema**

O sistema deverá sempre permitir o acesso de todos os usuários em uma janela de tempo de 20 horas indicada pelo cliente, as 4 horas restantes o sistema ficará inoperante para eventuais manutenções.

**PADRÕES**

Esta seção descreve os requisitos não funcionais associados a padrões ou normas que devem ser seguidos pelo sistema ou pelo seu processo de desenvolvimento.

**[NF010] Validação W3C**

O sistema deverá passar pela validação da ferramenta disponibilizada pela w3c para que todo o código exposto ao cliente esteja em conformidade e possa ser aceito pela maioria dos navegadores.

**HARDWARE E SOFTWARE**

Esta seção descreve os requisitos não funcionais associados ao hardware e software usados para desenvolver ou para executar o sistema.

**[NF011] navegadores**

- Google Chrome Desktop;
- Chrome Mobile;
- Firefox Desktop;
- Firefox Mobile;
- Internet Explore 7;
- Navegador Edge;
- Ópera Desktop e
- Ópera Mobile.

**[NF012] Memória e espaço físico**

- Mínimo de 1GB de RAM e 350MB de Disco Rígido.

**[NF013] Sistemas operacionais****Windows**

- Windows 7, Windows 8, Windows 8.1, Windows 10 ou versão posterior
- Processador Intel Pentium 4 ou posterior compatível com SSE2

**Mac**

- OS X Mavericks 10.9 ou versão posterior

**Linux**

- Debian 8+, openSUSE 13.3+, Fedora Linux 24+ ou Ubuntu 14.04+ de 64 bits
- Processador Intel Pentium 4 ou posterior compatível com SSE2

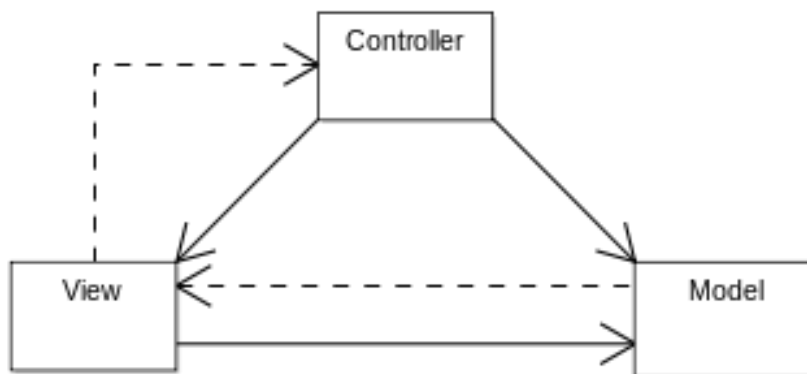
## Arquitetura do sistema

Tendo em vista que o sistema proposto se trata de um ERP, a sua arquitetura deverá ser separada em módulo, onde cada possui a sua lógica de arquivos separadas pelo padrão MVC. Podemos dizer que o sistema possui os seguintes módulos: chamado, cliente, conta, contrato, departamento, fornecedor, movimentação, produto, serviço e usuário. Além dos módulos do sistema, o mesmo possui bibliotecas de apoio, como por exemplo: conexão, routers e etc.

Abaixo uma breve descrição das tecnologias utilizadas e também das ferramentas de apoio.

### MVC

O MVC que em português significa modelo-visão-controlador, é um design pattern (padrão de arquitetura de software) que possibilita a divisão de um projeto em camadas, onde cada uma só executa o que lhe é definido e mais nada, a estrutura proposta por esse conceito possui três camadas, que são a Model, View e Controller.



Um diagrama simples exemplificando a relação entre Model, View e Controller.

**Fonte:** [https://www.oficinadanet.com.br/artigo/1687/mvc\\_-\\_o\\_padrao\\_de\\_arquitetura\\_de\\_software](https://www.oficinadanet.com.br/artigo/1687/mvc_-_o_padrao_de_arquitetura_de_software)

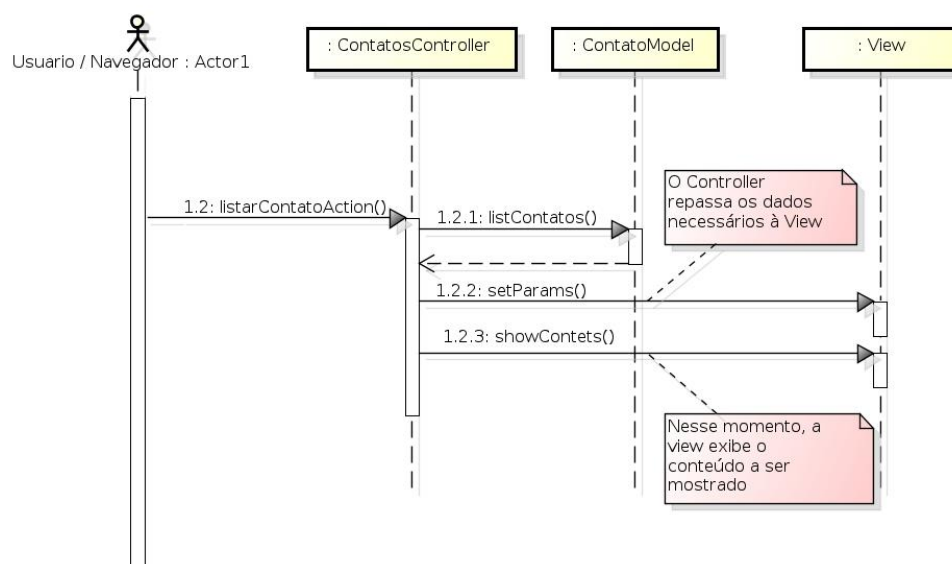
A utilização desse conceito possibilita a separação das camadas do projeto, fazendo com que a “camada de visualização do usuário” não tenha acesso direto a “camada de manipulação de dados”, deixando o reconhecimento, tratamento e envio a cargo da “camada de negócio”. Além do benefício de separação de camadas, o MVC possibilita a existência de várias interfaces onde que uma mudança na lógica de cadastro por exemplo não influencia na mudança de cada interface para adequação dessa nova rotina. Vale lembrar que, esse conceito de separação de estrutura, trabalha baseado em classes,

heranças e polimorfismo. Para saber a fundo como o MVC funciona e preciso saber sobre TOO (Teoria de Orientação a Objetos).

**CAMADA MODEL:** Na estrutura MVC o Model representa os dados da aplicação e as regras que governam o acesso e a modificação dos dados, além do controle dos dados, essa camada tem como objetivo manter o estado persistente dos dados do negócio e fornecer a camada de controller a capacidade de acessar as funcionalidades da aplicação encapsuladas pela própria model.

**CAMADA VIEW:** Já view dentro da estrutura MVC é camada onde o usuário tem acesso, e o seu objetivo é renderizar o conteúdo solicitado através de requisições enviadas para a camada de controller e devolvidas através da camada model.

**CAMADA CONTROLLER:** A camada de controller é responsável pela definição do comportamento da aplicação, e nela que é interpretada as ações do usuário e as mapeia para chamadas da camada model. Podemos pensar como ações do usuário cliques em botões ou seleções de menus. Com base na ação do usuário e no resultado do processamento da camada model, o controller seleciona uma view a ser exibida como parte da resposta a solicitação do usuário. Normalmente existe um controller para cada conjunto de funcionalidades relacionadas ao sistema.



*Diagrama de Sequência MVC*

**Fonte:** <http://www.digitaldev.com.br/2013/01/18/entendendo-o-mvc-model-view-controller/>

## LARAVEL

Laravel é um Framework PHP utilizado para o desenvolvimento web, que utiliza a arquitetura MVC e tem como principal característica ajudar a desenvolver aplicações seguras e performáticas de forma rápida, com código limpo e simples, já que ele incentiva o uso de boas práticas de programação e utiliza o padrão PSR-2 como guia para estilo de escrita do código.

## MYSQL

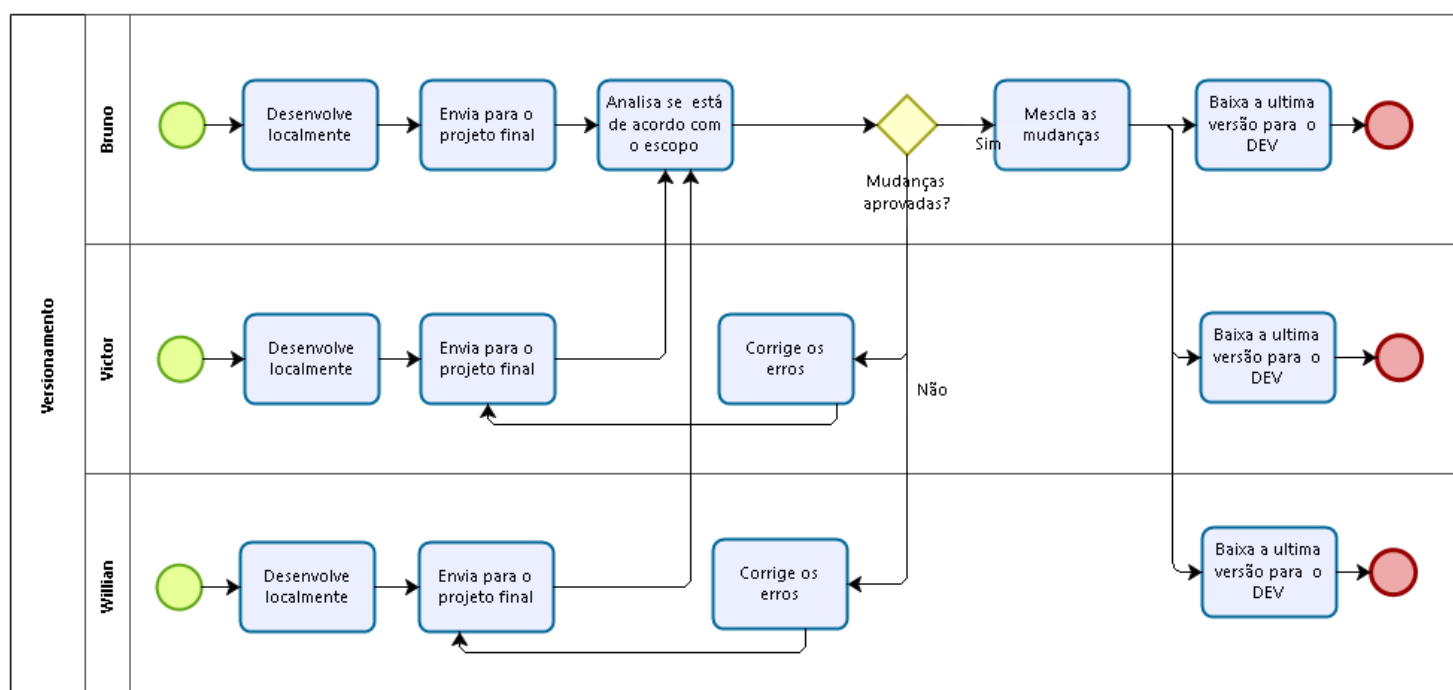
O MySQL é um SGBD (sistema de gerenciamento de banco de dados), que utiliza a linguagem SQL como interface, atualmente um dos bancos de dados mais populares, com mais de 10 milhões de instalações pelo mundo.

## FERRAMENTAS DE APOIO

Pensando no controle de versionamento do projeto, será utilizado duas ferramentas de apoio o **Git** e o **Github**.

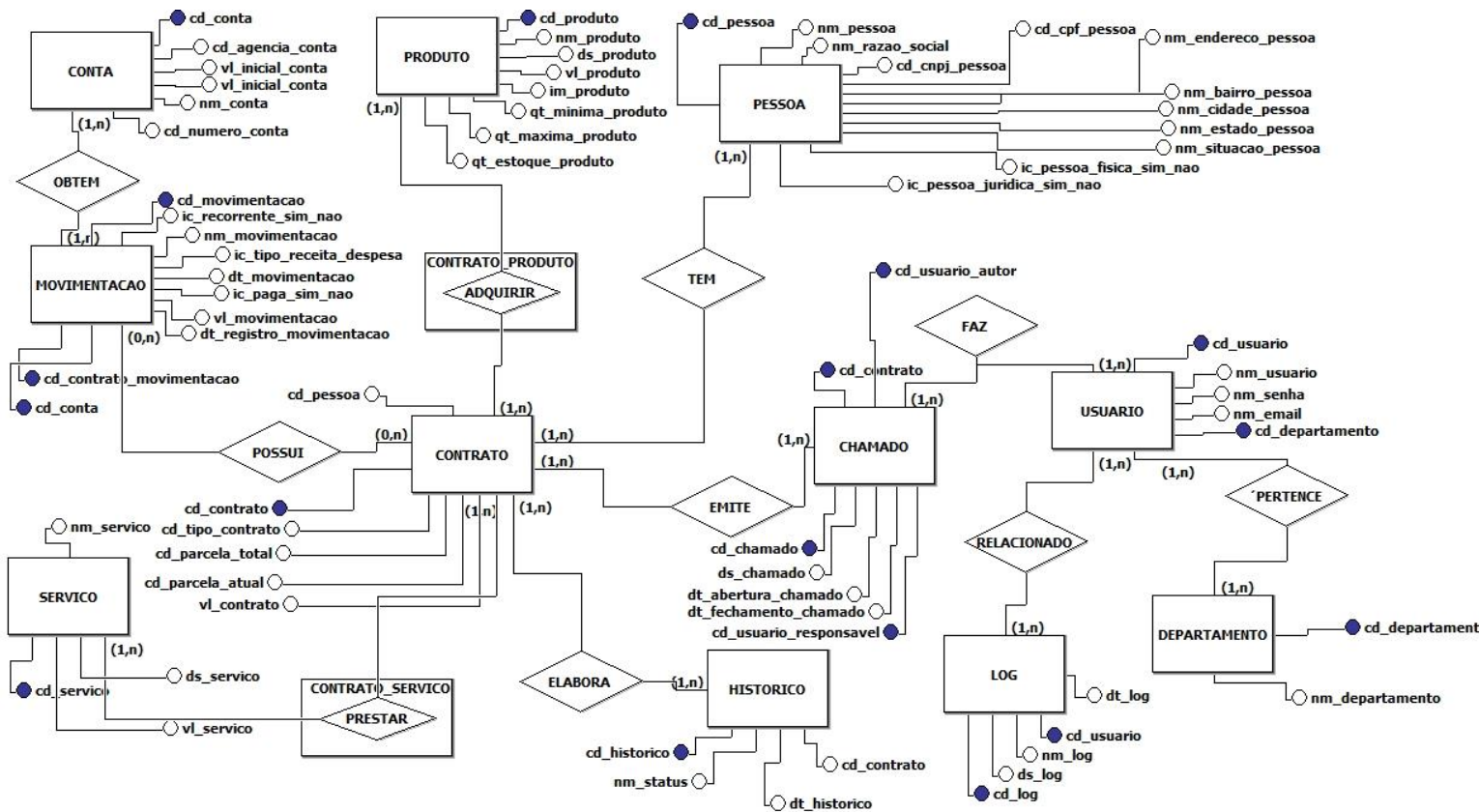
- **Git:** É um sistema de controle de versão de arquivos, onde através deles podemos desenvolver projetos na qual diversas pessoas podem contribuir simultaneamente no mesmo, editando e criando novos arquivos e permitindo que os mesmos possam existir sem o risco de suas alterações serem sobrescritas.
- **Github:** Serviço web gratuito utilizado para hospedar a versão do projeto que você estará trabalhando, através dele qualquer pessoa do projeto poderá acompanhar a evolução do seu trabalho.

O fluxo de trabalho proposto será o seguinte:



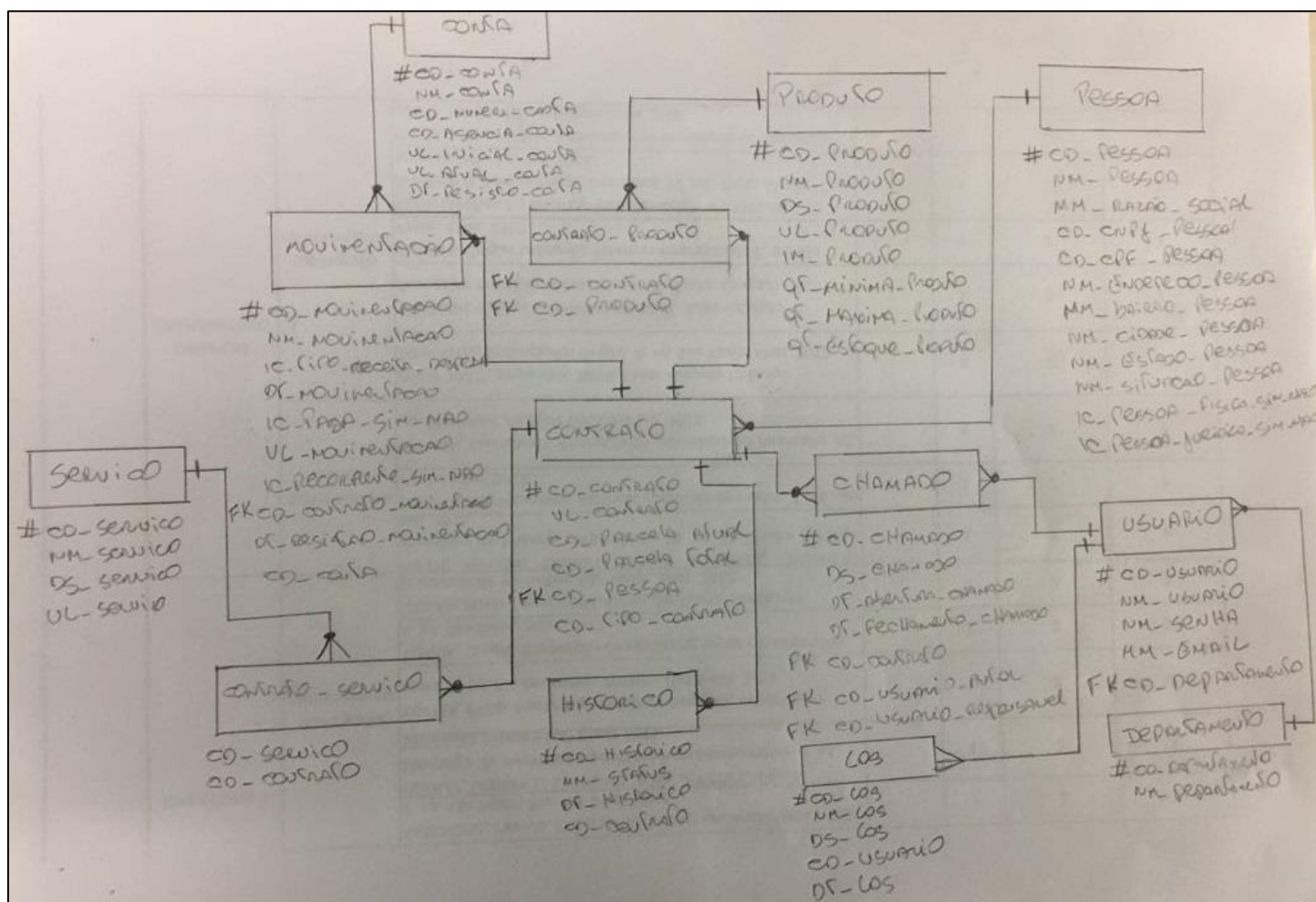
# Banco de dados

Após analisarmos o escopo do projeto, diagrama de caso de uso e os requisitos do





# MODELO RELACIONAL



## Análise de risco

O risco é inerente a atividade humana, sempre o acompanhando durante o passar do tempo. Cada atividade realizada acolhe um tipo de risco, sendo necessário assim dimensionar a solução ou alternativa que possa evitar a ocorrência do risco.

A definição de risco estabelecida pela norma ISO 31000 é “efeito da incerteza nos objetivos”. Em tudo o que fazemos causamos efeitos colaterais, a essa causa chamamos efeito “ação e reação”, quando estamos criando uma análise de risco, devemos antecipar todos os riscos que poderemos passar.

Bancos de Dados desprotegidos, usuário com acesso a informações privilegiadas sem permissão, computadores desprotegidos de supervisão, senhas fáceis de serem descobertas são exemplos de riscos que um sistema informatizado pode ocorrer.

No sistema SIG, usamos uma escala de 3 níveis para descrever cada risco, são elas “Alto risco”, “Médio risco”, “Baixo Risco”.

- Alto Risco, são rupturas ou vazamento de informações que podem prejudicar financeiramente a organização que utiliza o software. Por exemplo acesso por terceiro a dados de contas bancárias ou a senha dos profissionais envolvidos, perda dos dados.
- Médio risco, são erros e informações que podem levar o usuário a não conseguir utilizar o software ou até utiliza-lo de forma indevida, por exemplo, permitir deletar informações que não são do usuário. Cálculos indevidos, páginas com erros e links sem páginas.
- Baixo Risco, ações que impeçam o usuário de utilizar o sistema, mas que não gerem dados indevidos ou que atrasem a utilização do sistema pelo usuário. Exemplo usuário não consegue se autenticar, botões sem ações, impossibilidade de acesso a determinada página mesmo com permissão.

Nível	Risco	Alternativa
Baixo	Usuário esqueceu a senha	Cria uma opção para alterar a senha de acordo com algumas opções

Baixo	Senhas fáceis de lembrar	Criar mecanismo de complexidade de senhas
Baixo	Sem acesso ao sistema devido a quedas de rede	Criar um sistema off-line com sincronização automático
Baixo	Acesso por outras pessoas do sistema a dados de outras pessoas	Criar sistema de níveis e permissões
Baixo	Falta de conhecimento do utilizador	Criar treinamentos online e presenciais para os utilizadores
Baixo	Hardware não compatíveis com as exigências mínimas	Indicar ao responsáveis pela infraestrutura qual o equipamento mais adequado para ser utilizado e/ou quais serviços devem ser adquiridos
Médio	Link para páginas que não existem no sistema	Guardar todas as páginas que voltarem com erro 404 e informa-las ao administrador
Médio	Cálculo de Valores calculados de forma errada	Criar testes padronizados para cálculos e implementar opções de refazer cálculos e estorno
Médio	Páginas com erros de processamento	Criar log de páginas com erro 500 e informa-las ao administrador
Médio	Possibilidade de deletar informações que não sejam de propriedade do usuário	Criar sistema de voltar dados e guardar log de ações por usuários
Alto	Perca de dados	Mecanismo de backups diários com dois banco de dados distintos em sincronia
Alto	Acesso de dados por terceiros	Criptografia de dados com acesso apenas pelo sistema
Alto	Total perca dos dados	Criar backup fisicamente em mídia.
Alto	Servidores com falhas	Possibilidade de utilização de mais de um servidor
Alto	Queda de energia para alimentação dos servidores	Criação de um servidor na nuvem espelho para atender os clientes externos e uso de nobreaks

## Codificação e linguagem

Considerando atender um dos principais requisitos proposto no escopo do projeto, foi escolhido as linguagens abaixo, pois são as principais técnicas utilizadas para desenvolvimento de sistemas web, além de serem de fácil manipulação, multi plataforma, possuem suporte e possibilitam a criação de aplicações modulares.

- **HTML:** É uma linguagem de marcação utilizada na construção de páginas na web, os documentos html só são interpretados por navegadores, a tecnologia é fruto da junção entre os padrões HyTime e SGML.
- **CSS:** Mecanismo utilizado para adicionar estilo (cores, fontes, espaçamento etc) a um documento web, linguagem utilizada para criar os estilos que poderão ser utilizados em qualquer parte do sistema, através de uma classe pré definida no arquivo css.
- **JAVASCRIPT:** Linguagem de programação client-side, utilizada para controlar o html e o css para manipular comportamentos na página.
- **PHP:** Linguagem Server-Side interpretada livremente (não possui custo comercial para utilização), usada originalmente apenas para o desenvolvimento de aplicações presentes e atuantes no lado do servidor, capazes de gerar conteúdo dinâmico na world wide web, uma das principais linguagens que trabalha em conjunto com o html para criação do esqueleto das páginas.

### **Vantagens:**

- Integração com principais banco de dados;
- Velocidade e robustez;
- Multi-plataforma e
- Código-fonte aberto.
- **SQL:** Structured Query Language, ou Linguagem de Consulta Estruturada, o sql é a linguagem de pesquisa declarativa padrão para banco de dados relacional (base de dados relacional).

# Interface gráfica

Pensando em atender as melhores práticas para o desenvolvimento de uma interface amigável para qualquer tipo usuário, foi desenvolvido três tipos de padrões de telas, baseado nas “características humanas” e “boas práticas”, descritas abaixo:

## CARACTERÍSTICAS HUMANAS

- **Percepção:** uma boa tela deve facilmente ser percebida e reconhecida em suas funções e localização de suas opções;
- **Memória:** a memorização das opções e alternativas mais frequentemente usadas se dá mais facilmente quando uma tela está bem organizada e com disposição de itens de forma clara e peculiar;
- **Aprendizado:** uma boa tela tem a capacidade de fornecer um aprendizado aos seus usuários, sem que estes tenham que realizar algum curso ou treinamento para adquirir proficiência em seu uso;
- **Habilidade:** para usar uma tela não deve ser exigida nenhuma habilidade especial do usuário para compreendê-la ou utilizá-la e
- **Diferenças individuais:** deve-se considerar que os usuários apresentam diferenças individuais no que se refere a preferências, que vão desde cor, tamanho da letra, tipo de fonte, tipo de recurso de seleção, etc

## CARACTERÍSTICAS DE UMA BOA TELA

- Aparência limpa e ordenada;
- Indicação óbvia dos dados que estão sendo mostrados e o que deve ser feito com eles;
- Informação onde se espera que ela esteja;
- Indicação clara do que se relaciona com o que (cabeçalhos, instruções, opções, etc.);
- Vocabulário simples e explícito;
- Modo simples de encontrar o que está no sistema e como obtê-lo e
- Indicação clara de quando uma ação poderia realizar mudanças permanentes nos dados ou na operação do sistema.

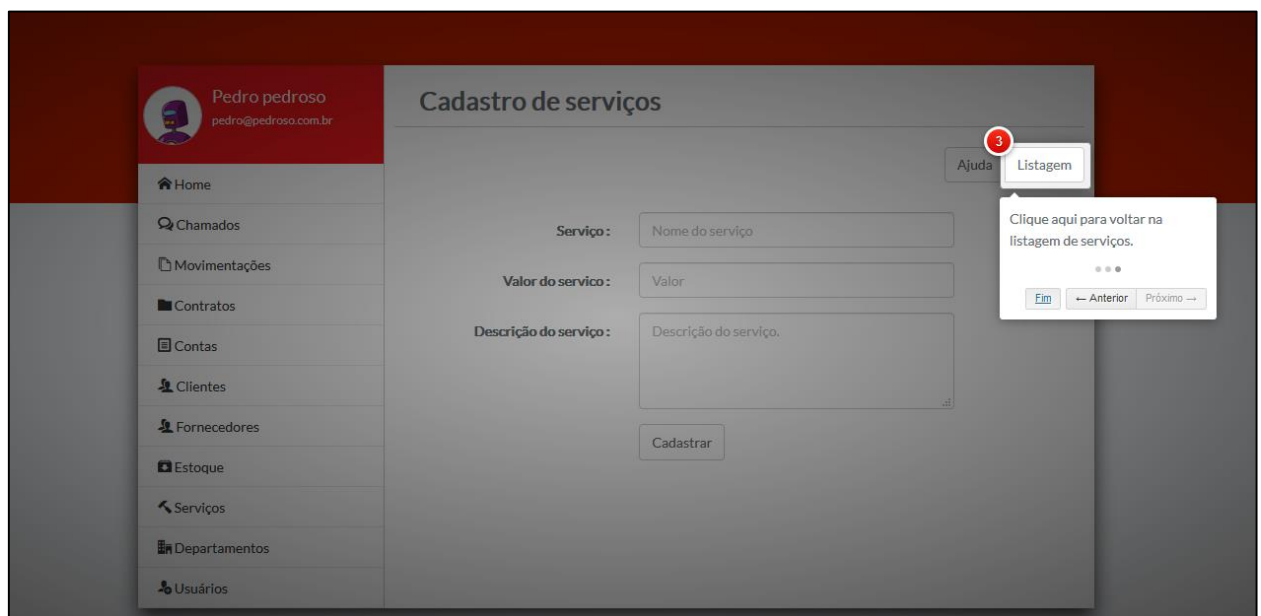
## TELA DE LISTAGEM

Tela padrão de listagem de informações, com ajuda interativa a todos os recursos existentes na mesma.



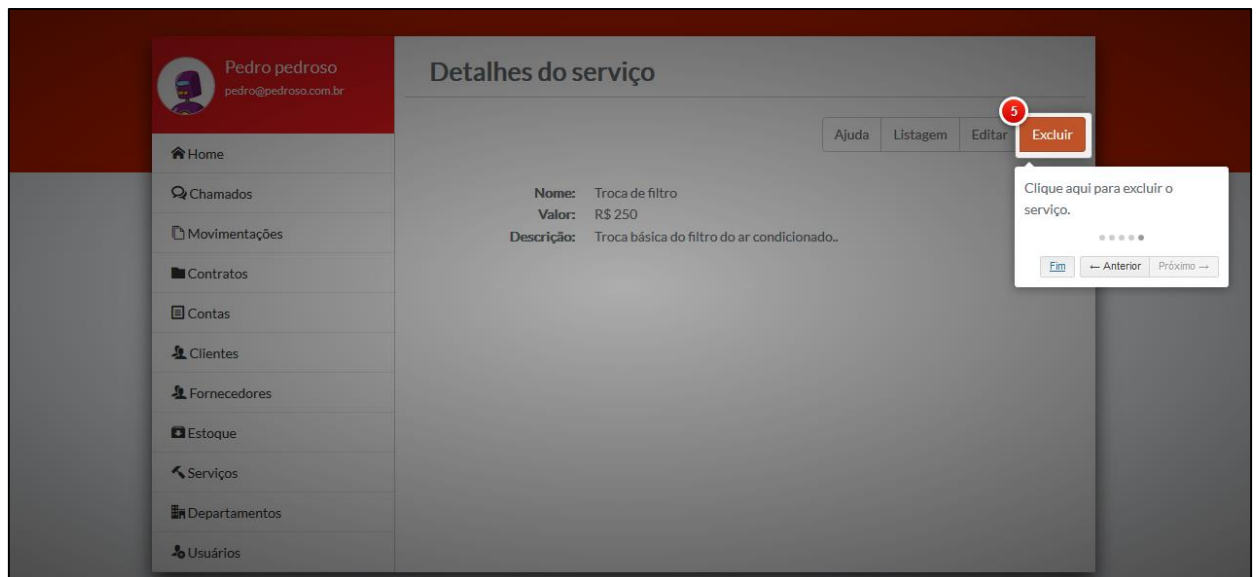
## TELA DE CADASTRO

Tela padrão de cadastro de dados, com ajuda interativa a todos os recursos existentes na mesma.



## TELA DE DETALHES

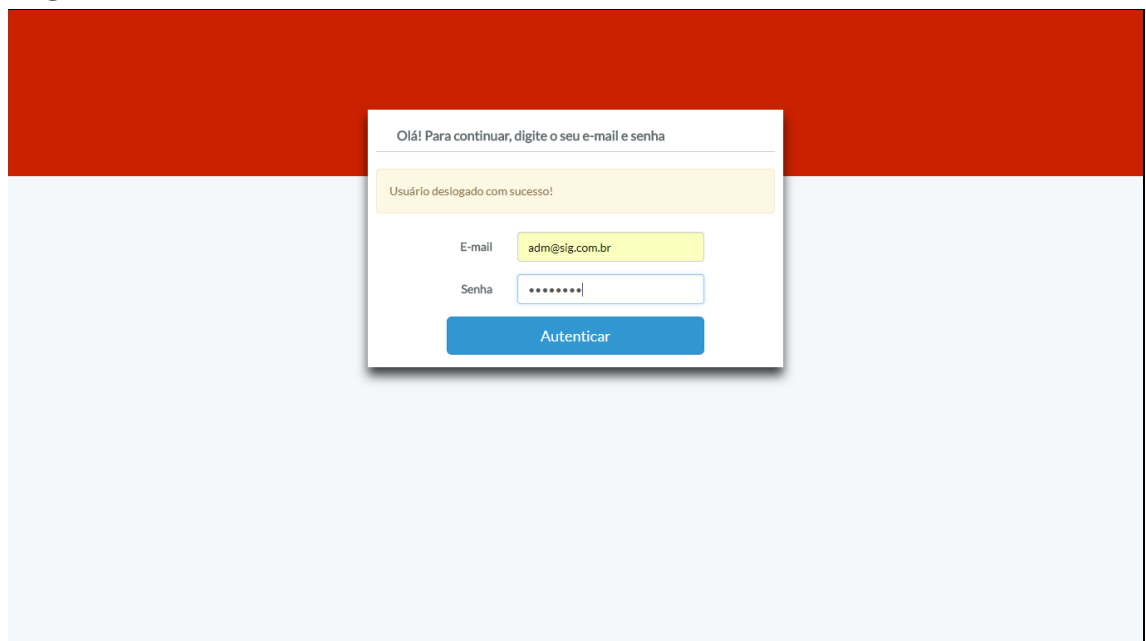
Tela padrão de detalhes de informações, com ajuda interativa a todos os recursos existentes na mesma.



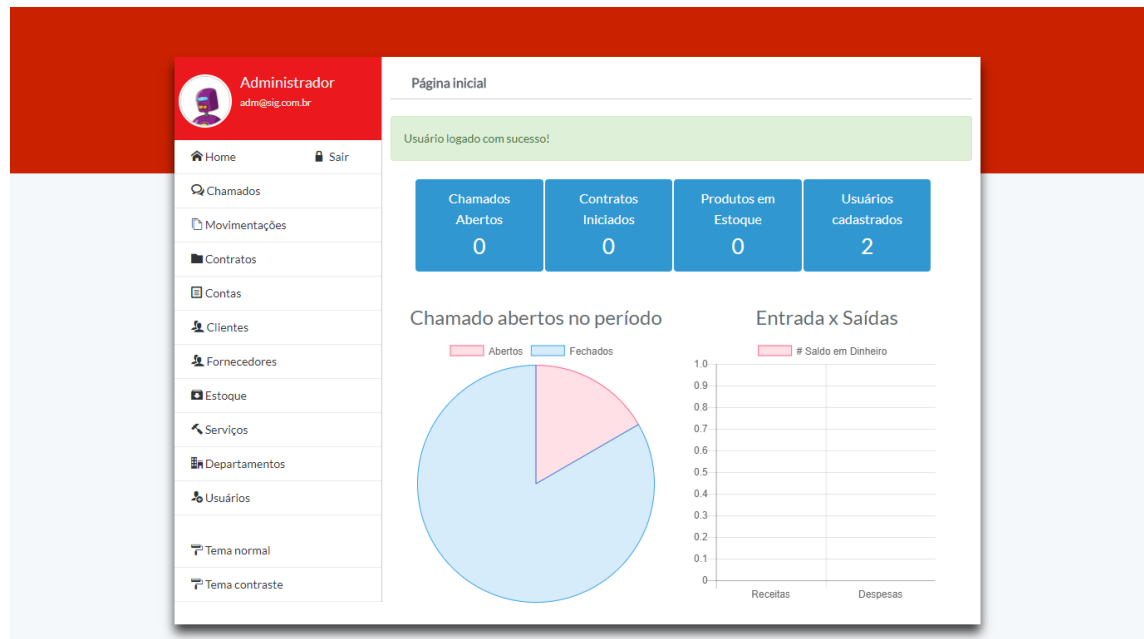
## TELAS EM PRODUÇÃO

Abaixo algumas telas que já foram finalizadas e já liberadas para a etapa de teste.

### Login



## Home padrão



## Home com acessibilidade





## Departamentos

Administrador  
adm@sig.com.br

Home Salir

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

Tema normal

Tema contraste

Listagem de departamentos

Ajuda PDF Atualizar Novo Registro

#	Nome	Ação
1	Administrativo	Ver

Administrador  
adm@sig.com.br

Home Salir

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

Tema normal

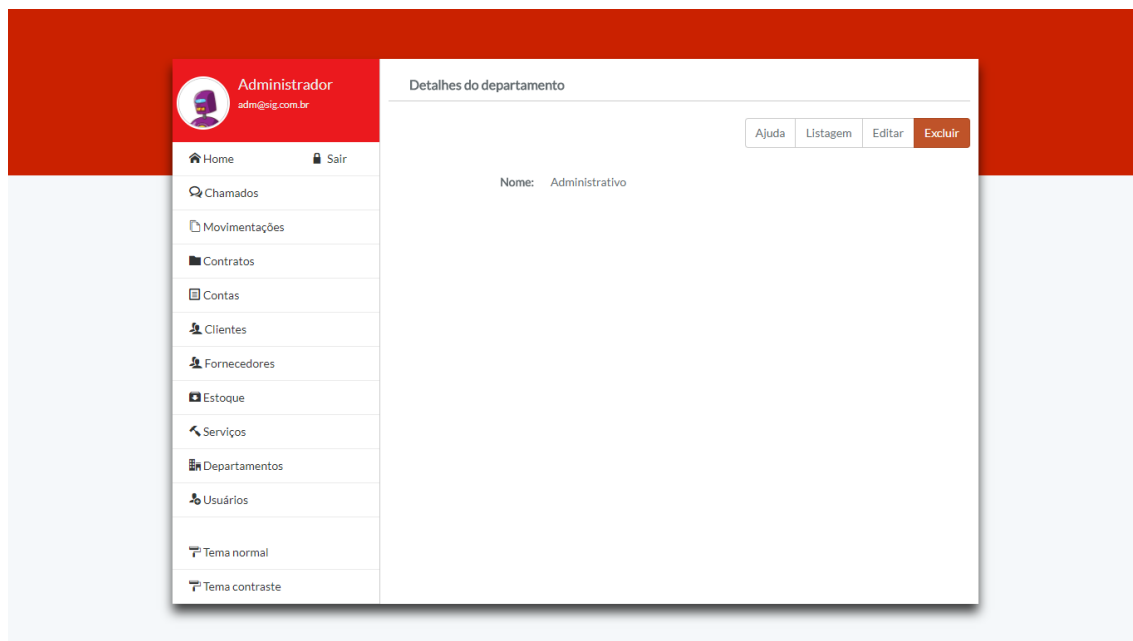
Tema contraste

Cadastro de departamento

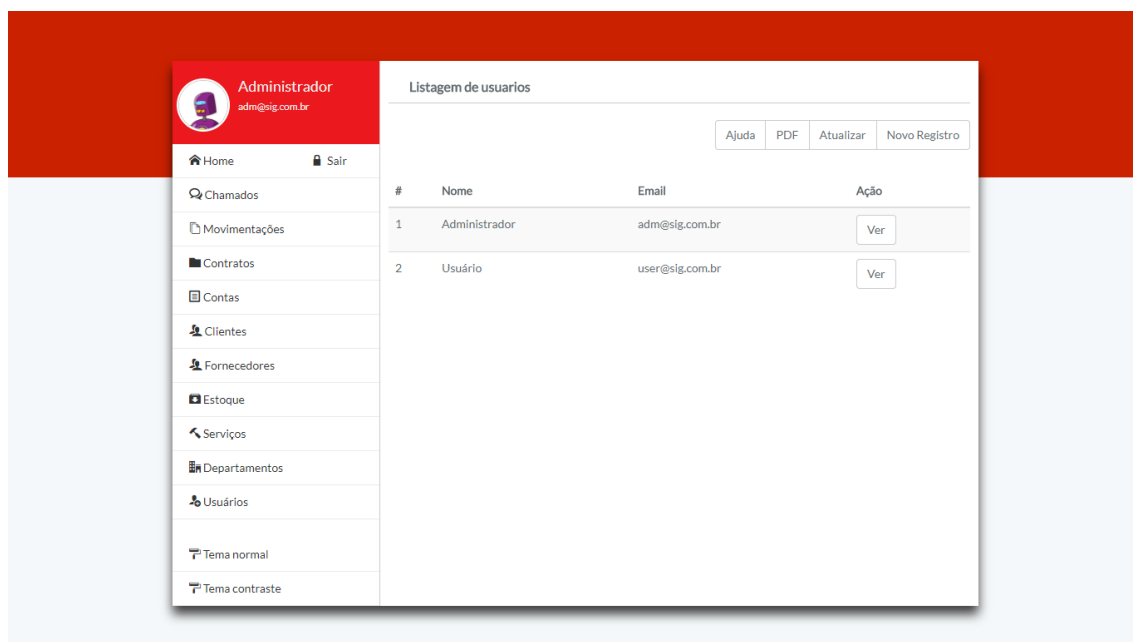
Ajuda Listagem


Nome: Nome da empresa

Cadastrar



## Gereciamento de usuários



**Administrador**  
adm@sig.com.br

[Home](#) [Sair](#)

[Chamados](#)

[Movimentações](#)

[Contratos](#)

[Contas](#)

[Clientes](#)

[Fornecedores](#)

[Estoque](#)

[Serviços](#)

[Departamentos](#)

[Usuários](#)

[Tema normal](#)

[Tema contraste](#)

### Cadastro de usuarios

[Ajuda](#) [Listagem](#)


Nome:

Email:

Senha:

Departamento:

127.0.0.1/usuario/

**Administrador**  
adm@sig.com.br

[Home](#) [Sair](#)

[Chamados](#)

[Movimentações](#)

[Contratos](#)

[Contas](#)

[Clientes](#)

[Fornecedores](#)

[Estoque](#)

[Serviços](#)

[Departamentos](#)

[Usuários](#)

[Tema normal](#)

[Tema contraste](#)

### Detalhes do usuario

[Ajuda](#) [Listagem](#) [Editar](#) [Excluir](#)

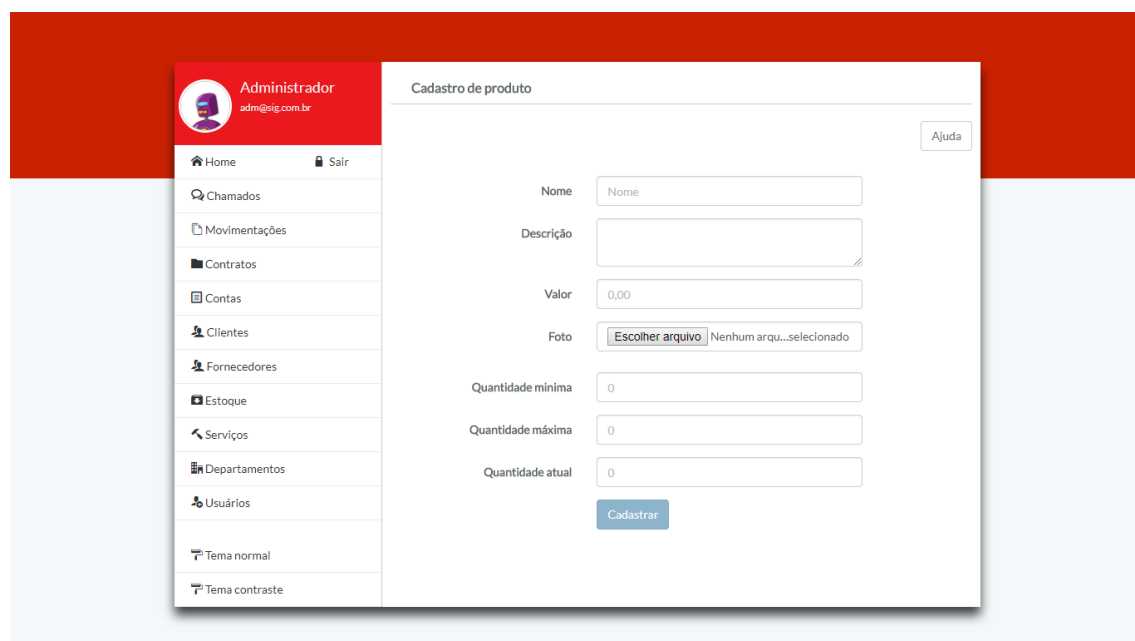
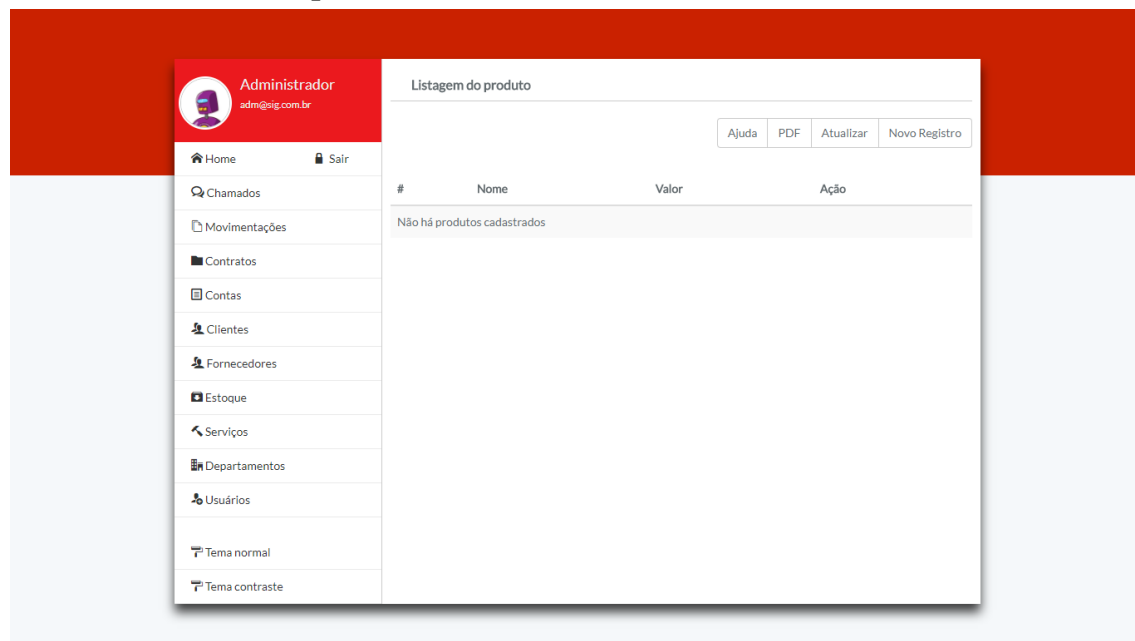
Nome: Administrador

Email: adm@sig.com.br

Senha: senhaadm

Codigo do Departamento: 1

## Gereciamento de estoque



**Administrador**  
adm@sig.com.br

Home Sair

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

Tema normal

Tema contraste

**Detalhes do produto**

Ajuda Listagem Editar Excluir

Nome: Condensadora Split  
Descrição: condensadora split carrier  
Valor: 1.000,00  
Foto:

Qtd mínima: 10  
Qtd máxima: 15  
Qtd estoque: 10

## Gerenciamento de contas

**Administrador**  
adm@sig.com.br

Home Sair

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

Tema normal

Tema contraste

**Listagem de contas**

Ajuda PDF Atualizar Novo Registro

#	Nome	Tipo	Saldo atual	Ação
1	Caixa	caixa	0	Ver

**Administrador**  
adm@sig.com.br

Home Sair

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

Tema normal

Tema contraste

### Cadastro de contas

Ajuda Listagem

Nome:

Agência:

Conta:

Valor Inicial:

Tipo de conta:

- ☐ Poupança
- ☐ Corrente
- ☐ Cartão
- ☐ Investimento
- ☐ Caixa
- ☐ Outras

Cadastrar

## Movimentações

**Administrador**  
adm@sig.com.br

Home Sair

Chamados

Movimentações

Contratos

Contas

Clientes

Fornecedores

Estoque

Serviços

Departamentos

Usuários

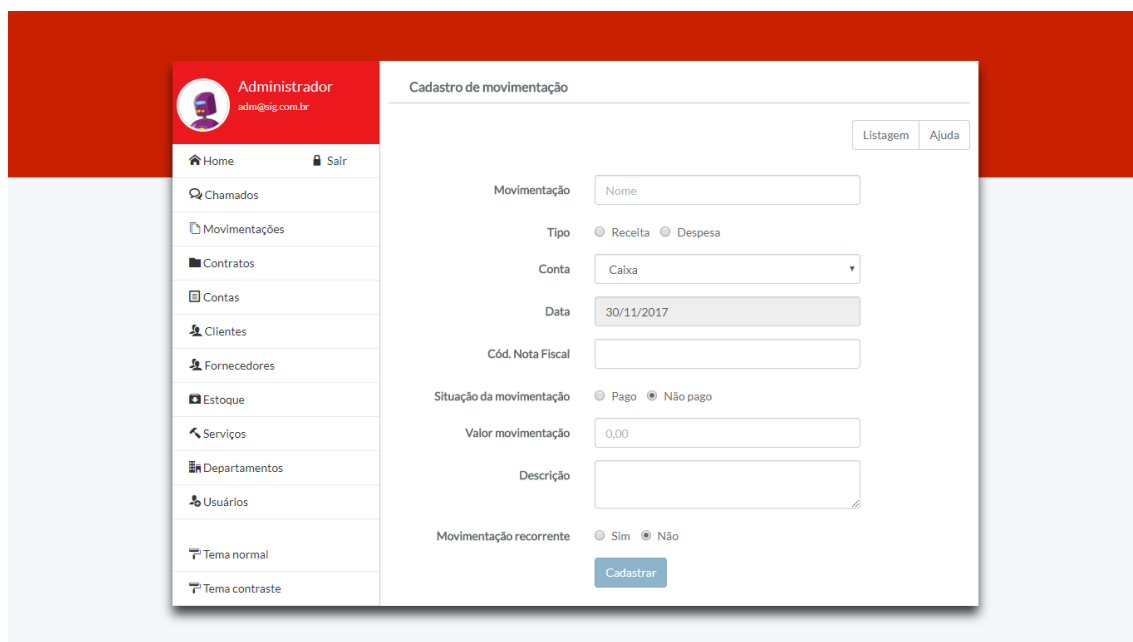
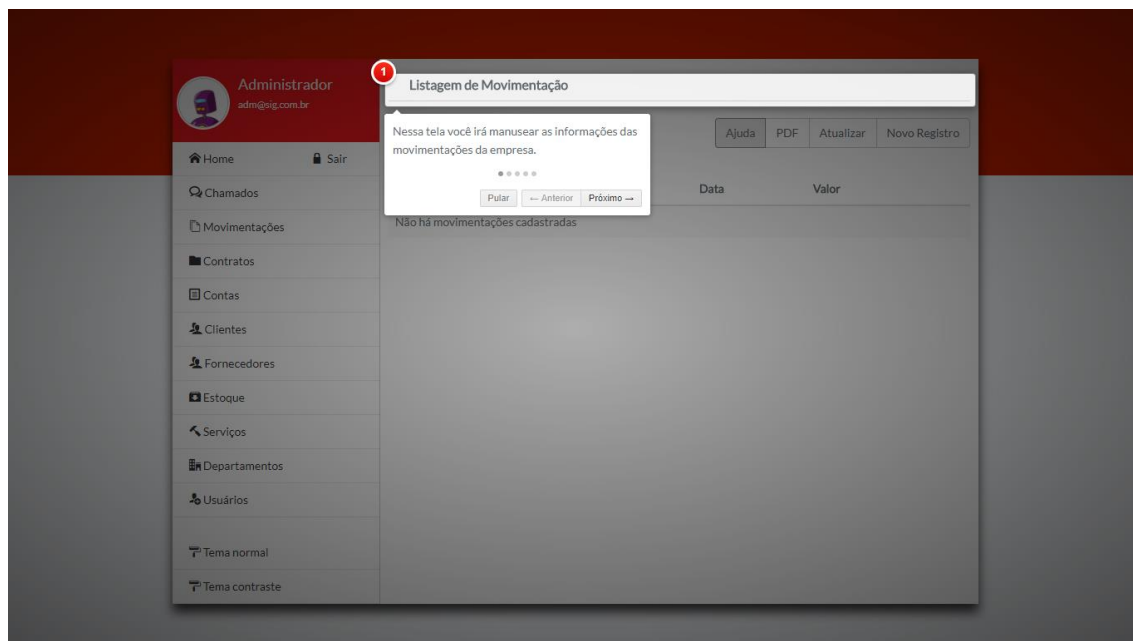
Tema normal

Tema contraste

### Listagem de Movimentação

Ajuda PDF Atualizar Novo Registro

#	Tipo	Nome	Data	Valor
Não há movimentações cadastradas				



The screenshot displays a web application interface. On the left is a sidebar menu with a red header containing the user profile 'Administrador' and email 'adm@eig.com.br'. The menu items are: Home, Sair, Chamados, Movimentações, Contratos, Contas, Clientes, Fornecedores, Estoque, Serviços, Departamentos, Usuários, Tema normal, and Tema contraste. The main content area is titled 'Detalhes da movimentacao' and features a table of transaction details. At the top right of this area are buttons for 'Ajuda', 'Listagem', 'Editar', and 'Excluir'.

Nome:	troca pc
Tipo:	Débito
Conta:	Caixa
Data:	30/11/2017
Cód. Nota Fiscal:	10
Situação:	Movimentação paga
Valor:	1.000,00
Descrição:	troca pc financeiro
Recorrente:	Movimentação não recorrente



## Código fonte

Nessa seção será apresentada as codificações geradas na produção do sistema separadas pela estrutura presente no MVC.

### VIEW - Home

```
HOME
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você irá manusear as informações dos usuarios.'>
        <h1>Página inicial</h1>
    </div>

    @if(Session::has('message'))
        <div class="alert alert-success">
            {{ Session::get('message') }}
        </div>
    @endif

    <div class="container-fluid">

        <div class="col-md-12 col-xs-12">
            <ul class="nav nav-pills component-counters" role="tablist">
                <li role="presentation" class="active"><a href="#">Chamados
                Abertos<span>{{ $countChamado }}</span></a></li>
                <li role="presentation" class="active"><a href="#">Contratos
                Iniciados<span>{{ $countContratos }}</span></a></li>
                <li role="presentation" class="active"><a href="#">Produtos em
                Estoque<span>{{ $countProdutos }}</span></a></li>
                <li role="presentation" class="active"><a href="#">Usuários
                cadastrados<span>{{ $countUsuarios }}</span></a></li>
            </ul>
        </div>

        <div class="row component-graph">
            <div class="col-md-6 col-xs-12">
                <h3 class="text-center">Chamado abertos no período</h3>
                <canvas id="chamadosAbertos" height="300" data-fechados="500" data-
                abertos="100"></canvas>
            </div>

            <div class="col-md-6 col-xs-12">
                <h3 class="text-center">Entrada x Saídas</h3>
                <canvas id="conflitoSaldo" height="300" data-receitas="{{ $sumReceitas }}" data-
                despesas="{{ $sumDespesas }}"></canvas>
            </div>
        </div>
    </div>
@stop
```

## LIST

```

@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você poderá criar, editar e excluir os registros de
produtos cadastrados no sistema.'>
        <h1>Listagem do produto</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/produto/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
            <a href="#/produto/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório
da página atual em PDF.'>PDF</a>
            <a href="/produto" class="btn btn-default" data-intro='Clique aqui para atualizar a
tela.'>Atualizar</a>
            <a href="/produto/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
        </div>
    </div>

    @if(Session::has('message'))
    <div class="alert alert-success">
        {{ Session::get('message') }}
    </div>
    @endif

    <table class="table table-striped table-hovered table-condensed" data-intro='Aqui você confere todos
os registros já cadastrados.'>
        <thead>
            <tr>
                <th>#</th>
                <th>Nome</th>
                <th>Valor</th>
                <th>Ação</th>
            </tr>
        </thead>
        <tbody>
            @forelse($listaProdutos as $produto)
                <tr>
                    <td>{{ $produto->cd_produto }}</td>
                    <td>{{ $produto->nm_produto }}</td>
                    <td>{{ dinheiro($produto->vl_produto) }}</td>
                    <td><a href="/produto/{{ $produto->cd_produto }}" type="button" class="btn btn-
default">Ver</a></td>
                </tr>
            @empty
                <tr>
                    <td colspan="6" class="aling-center">
                        Não há produtos cadastrados
                    </td>
                </tr>
            @endforelse
        </tbody>
    </table>
@stop

```

## SHOW

```

@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Detalhes do produto</h1>
    </div>

    <form class="form-horizontal" action="/produto/{{ $produto->cd_produto }}" method="POST">
        {{ csrf_field() }}
        {{ method_field('DELETE') }}
        <div class="component-barra-menu">
            <div class="btn-group pull-right" role="group">
                <a href="/produto/help" class="btn btn-default btn-help">Ajuda</a>
                <a href="/produto/" class="btn btn-default">Listagem</a>
                <a href="/produto/{{ $produto->cd_produto }}/edit" class="btn btn-default">Editar</a>
                <button type="submit" class="btn btn-danger">Excluir</button>
            </div>
        </div>
    </form>

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">

                <dl class="dl-horizontal">
                    <dt>Nome:</dt>
                    <dd>{{ $produto->nm_produto }}</dd>
                    <dt>Descrição:</dt>
                    <dd>{{ $produto->ds_produto }}</dd>
                    <dt>Valor:</dt>
                    <dd>{{ dinheiro($produto->vl_produto) }}</dd>
                    <dt>Foto:</dt>
                    <dd></dd>
                    <dt>Qtd minima:</dt>
                    <dd>{{ $produto->qt_minima_produto }}</dd>
                    <dt>Qtd máxima:</dt>
                    <dd>{{ $produto->qt_maxima_produto }}</dd>
                    <dt>Qtd estoque:</dt>
                    <dd>{{ $produto->qt_estoque_produto }}</dd>
                </dl>

            </div>
        </div>
    </div>
@stop

```

## EDIT

```

@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você irá editar um usuario já cadastrado no sistema.'>
        <h1>Edição do usuario</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma ajuda igual a essa na página.'>Ajuda</a>
        </div>
    </div>

```

```

@if(Session::has('message'))
    <div class="alert alert-success">
        {{ Session::get('message') }}
    </div>
@endif

@if (count($errors) > 0)
    <div class="alert alert-danger">
        <ul>
            @foreach ($errors->all() as $error)
                <li>{{ $error }}</li>
            @endforeach
        </ul>
    </div>
@endif
<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">
            <form class="form-horizontal" action="/usuario/{{ $usuario->cd_usuario }}" method="POST">
                <div class="form-group">
                    <label for="nm_usuario" class="col-md-4 control-label">Nome :</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="nm_usuario" name="nm_usuario"
placeholder="Nome do usuario" value="{{ $usuario->nm_usuario }}" required/>
                    </div>
                </div>
                <div class="form-group">
                    <label for="nm_senha" class="col-md-4 control-label">Senha :</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="nm_senha" name="nm_senha"
placeholder="Senha do usuario" value="{{ $usuario->nm_senha }}" required/>
                    </div>
                </div>
                <div class="form-group">
                    <label for="nm_email" class="col-md-4 control-label">Email :</label>
                    <div class="col-md-6">
                        <input type="email" class="form-control" id="nm_email" name="nm_email"
placeholder="Email do usuario" value="{{ $usuario->nm_email }}" required/>
                    </div>
                </div>
                <div class="form-group">
                    <label for="cd_departamento" class="col-md-4 control-label">Departamento</label>
                    <div class="col-md-6">
                        {{ Form::select('cd_departamento', $departamentos, NULL,['class' => 'form-control']) }}
                    </div>
                </div>
                <div class="form-group">
                    <div class="col-md-offset-4 col-md-6">
                        <button type="submit" class="btn btn-info">Editar</button>
                        {{ csrf_field() }}
                        {{ method_field('PUT') }}
                    </div>
                </div>
            </form>
        </div>
    </div>
</div>
@stopLIST
@extends('layout/public')

```

```

@section('content')
<div class="component-title" data-intro='Nessa tela você irá manusear as informações dos usuarios.'>
  <h1>Listagem de usuarios</h1>
</div>

<div class="component-barra-menu">
  <div class="btn-group pull-right" role="group">
    <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
    <a href="#/usuario/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório
da página atual em PDF.'>PDF</a>
    <a href="/usuario" class="btn btn-default" data-intro='Clique aqui para atualizar a
página.'>Atualizar</a>
    <a href="/usuario/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
  </div>
</div>

<table class="table table-striped table-hovered table-condensed" data-intro='Aqui você confere todos
os registros já cadastrados.'>
  <thead>
    <tr>
      <th>#</th>
      <th>Nome</th>
      <th>Email</th>
      <th>Ação</th>
    </tr>
  </thead>
  <tbody>
    @forelse($listaUsuario as $usuario)
      <tr>
        <td>{{ $usuario->cd_usuario }}</td>
        <td>{{ $usuario->nm_usuario }}</td>
        <td>{{ $usuario->nm_email }}</td>
        <td><a href="/usuario/{{ $usuario->cd_usuario }}" type="button" class="btn btn-
default">Ver</a></td>
      </tr>
    @empty
      <tr>
        <td colspan="6" class="aling-center">
          Não há usuarios cadastrados
        </td>
      </tr>
    @endforelse
  </tbody>
</table>
@stop

```

## SHOW

```

@extends('layout/public')
@section('content')
  <div class="component-title" data-intro='Nessa tela você poderá editar e excluir os registros do
usuario.'>
    <h1>Detalhes do usuario</h1>
  </div>

  <form class="form-horizontal" action="/usuario/{{ $usuario->cd_usuario }}" method="POST">
    {{ csrf_field() }}
    {{ method_field('DELETE') }}
  </div>
  <div class="component-barra-menu">

```

```

    <div class="btn-group pull-right" role="group">
      <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
      <a href="/usuario/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem de
usuarios.'>Listagem</a>
      <a href="/usuario/{{ $usuario->cd_usuario }}/edit" class="btn btn-default" data-intro='Clique aqui
para editar os dados do usuario.'>Editar</a>
      <button type="submit" class="btn btn-danger" data-intro='Clique aqui para excluir o
usuario.'>Excluir</button>
    </div>
  </div>
</form>

<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">

      <dl class="dl-horizontal">
        <dt>Nome:</dt>
        <dd>{{ $usuario->nm_usuario }}</dd>
      </dl>
      <dl class="dl-horizontal">
        <dt>Email:</dt>
        <dd>{{ $usuario->nm_email }}</dd>
      </dl>
      <dl class="dl-horizontal">
        <dt>Senha:</dt>
        <dd>{{ $usuario->nm_senha }}</dd>
      </dl>
      <dl class="dl-horizontal">
        <dt>Codigo do Departamento:</dt>
        <dd>{{ $usuario->cd_departamento }}</dd>
      </dl>
    </div>
  </div>
</div>
@stop

```

## VIEW - Departamento

```

CREATE
@extends('layout/public')
@section('content')
  <div class="component-title" data-intro='Nessa tela você irá cadastrar um novo departamento no
sistema.'>
    <h1>Cadastro de departamento</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
      <a href="#/departamento/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter
uma ajuda igual a essa na página.'>Ajuda</a>
      <a href="/departamento/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem
de departamentos.'>Listagem</a>
    </div>
  </div>
  @if (count($errors) > 0)
    <div class="alert alert-danger">
      <ul>

```

```

        @foreach ($errors->all() as $error)
            <li>{{ $error }}</li>
        @endforeach
    </ul>
</div>
@endif
<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">
            <form class="form-horizontal" action="/departamento" method="POST">
                <div class="form-group">
                    <label for="nm_departamento" class="col-md-4 control-label">Nome :</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="nm_departamento" name="nm_departamento"
placeholder="Nome da empresa" required/>
                    </div>
                </div>
                <div class="form-group">
                    <div class="col-md-offset-4 col-md-6">
                        <button type="submit" class="btn btn-default">Cadastrar</button>
                        {{ csrf_field() }}
                        {{ method_field('POST') }}
                    </div>
                </div>
            </form>
        </div>
    </div>
</div>
@stop
EDIT
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você irá editar o departamento já cadastrado no
sistema.'>
        <h1>Edição do departamento</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/departamento/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter
uma ajuda igual a essa na página.'>Ajuda</a>
            <a href="/departamento/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem
de departamentos.'>Listagem</a>
        </div>
    </div>

    @if (count($errors) > 0)
        <div class="alert alert-danger">
            <ul>
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif
<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">

```

```

<form class="form-horizontal" action="/departamento/{{ $departamento->cd_departamento }}"
method="POST">
    <div class="form-group">
        <label for="nm_departamento" class="col-md-4 control-label">Nome :</label>
        <div class="col-md-6">
            <input type="text" class="form-control" id="nm_departamento" name="nm_departamento"
placeholder="Nome do departamento" value="{{ $departamento->nm_departamento }}" required/>
        </div>
    </div>
    <div class="form-group">
        <div class="col-md-offset-4 col-md-6">
            <button type="submit" class="btn btn-info">Editar</button>
            {{ csrf_field() }}
            {{ method_field('PUT') }}
        </div>
    </div>
</form>
</div>
</div>
</div>
@stop
LIST
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro="Nessa tela você irá manusear as informações dos
departamentos da empresa.">
        <h1>Listagem de departamentos</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/departamento/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter
uma ajuda igual a essa na página.'>Ajuda</a>
            <a href="#/departamento/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um
relatório da página atual em PDF.'>PDF</a>
            <a href="/departamento" class="btn btn-default" data-intro='Clique aqui para atualizar a
página.'>Atualizar</a>
            <a href="/departamento/create" class="btn btn-default" data-intro='Clique aqui para adicionar um
novo registro.'>Novo Registro</a>
        </div>
    </div>

    @if(Session::has('message'))
        <div class="alert alert-success">
            {{ Session::get('message') }}
        </div>
    @endif

    <table class="table table-striped table-hovered table-condensed" data-intro='Aqui você confere todos
os registros já cadastrados.'>
        <thead>
            <tr>
                <th>#</th>
                <th>Nome</th>
                <th>Ação</th>
            </tr>
        </thead>
        <tbody>
            @forelse($listaDepartamentos as $departamento)
                <tr>

```



```

        <td>{{ $departamento->cd_departamento }}</td>
        <td>{{ $departamento->nm_departamento }}</td>
        <td><a href="/departamento/{{ $departamento->cd_departamento }}" type="button" class="btn
btn-default">Ver</a></td>
    </tr>
    @empty
    <tr>
        <td colspan="6" class="aling-center">
            Não há departamentos cadastrados.
        </td>
    </tr>
    @endforelse
</tbody>
</table>
@stop
SHOW
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você poderá editar ou excluir o departamento.'>
        <h1>Detalhes do departamento</h1>
    </div>

    <form class="form-horizontal" action="/departamento/{{ $departamento->cd_departamento }}"
method="POST">
        {{ csrf_field() }}
        {{ method_field('DELETE') }}
        <div class="component-barra-menu">
            <div class="btn-group pull-right" role="group">
                <a href="#/departamento/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter
uma ajuda igual a essa na página.'>Ajuda</a>
                <a href="/departamento/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem
de departamentos.'>Listagem</a>
                <a href="/departamento/{{ $departamento->cd_departamento }}/edit" class="btn btn-default" data-
intro='Clique aqui para editar os dados do departamento.'>Editar</a>
                <button type="submit" class="btn btn-danger" data-intro='Clique aqui para excluir o
departamento.'>Excluir</button>
            </div>
        </div>
    </form>

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">

                <dl class="dl-horizontal">
                    <dt>Nome:</dt>
                    <dd>{{ $departamento->nm_departamento }}</dd>
                </dl>
            </div>
        </div>
    </div>
@stop

```

## VIEW - Usuários

```

CREATE
@extends('layout/public')
@section('content')
    <div class="component-title">

```

```

    <h1>Cadastro de produto</h1>
</div>

<div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
        <a href="/produto/help" class="btn btn-default">Ajuda</a>
    </div>
</div>

@if (count($errors) > 0)
<div class="alert alert-danger">
    <ul>
        @foreach ($errors->all() as $error)
            <li>{{ $error }}</li>
        @endforeach
    </ul>
</div>
@endif

<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">
            <form class="form-horizontal" action="/produto" method="POST">
                <div class="form-group">
                    <label for="nm_produto" class="col-md-4 control-label">Nome</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="nm_produto" name="nm_produto"
placeholder="Nome" />
                    </div>
                </div>
                <div class="form-group">
                    <label for="ds_produto" class="col-md-4 control-label">Descrição</label>
                    <div class="col-md-6">
                        <textarea class="form-control" id="ds_produto" name="ds_produto"></textarea>
                    </div>
                </div>
                <div class="form-group">
                    <label for="vl_produto" class="col-md-4 control-label">Valor</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="vl_produto" name="vl_produto"
placeholder="R$ 0,00" data-dinheiro="true" required />
                    </div>
                </div>
                <div class="form-group">
                    <label for="im_produto" class="col-md-4 control-label">Foto</label>
                    <div class="col-md-6">
                        <input type="file" data-loadimg=".component-preloader" data-srcimg="#im_produto"
class="form-control" placeholder="Foto" />
                        <input type="hidden" id="im_produto" name="im_produto" value="" />
                        <div class="component-preloader"></div>
                    </div>
                </div>
                <div class="form-group">
                    <label for="qt_minima_produto" class="col-md-4 control-label">Quantidade minima</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="qt_minima_produto"
name="qt_minima_produto" placeholder="0" required />
                    </div>
                </div>
                <div class="form-group">

```

```

        <label for="qt_maxima_produto" class="col-md-4 control-label">Quantidade máxima</label>
        <div class="col-md-6">
            <input type="text" class="form-control" id="qt_maxima_produto"
name="qt_maxima_produto" placeholder="0" required />
        </div>
    </div>
    <div class="form-group">
        <label for="qt_estoque_produto" class="col-md-4 control-label">Quantidade atual</label>
        <div class="col-md-6">
            <input type="text" class="form-control" id="qt_estoque_produto"
name="qt_estoque_produto" placeholder="0" required />
        </div>
    </div>
    <div class="form-group">
        <div class="col-md-offset-4 col-md-6">
            <button type="submit" class="btn btn-info">Cadastrar</button>
            {{ csrf_field() }}
            {{ method_field('POST') }}
        </div>
    </div>
</form>
</div>
</div>
</div>
@stopEDIT
@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Cadastro de produto</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="/produto/help" class="btn btn-default">Ajuda</a>
        </div>
    </div>

    @if (count($errors) > 0)
    <div class="alert alert-danger">
        <ul>
            @foreach ($errors->all() as $error)
                <li>{{ $error }}</li>
            @endforeach
        </ul>
    </div>
    @endif

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">
                <form class="form-horizontal" action="/produto/{{ $produto->cd_produto }}" method="POST">
                    <div class="form-group">
                        <label for="nm_produto" class="col-md-4 control-label">Nome</label>
                        <div class="col-md-6">
                            <input type="text" class="form-control" id="nm_produto" name="nm_produto"
placeholder="Nome" value="{{ $produto->nm_produto }}" />
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="ds_produto" class="col-md-4 control-label">Descrição</label>

```

```

        <div class="col-md-6">
            <textarea class="form-control" id="ds_produto" name="ds_produto">{{ $produto-
>ds_produto }}</textarea>
        </div>
    </div>
    <div class="form-group">
        <label for="im_produto" class="col-md-4 control-label">Foto</label>
        <div class="col-md-6">
            <input type="file" data-loading=".component-preloader" data-srcimg="#im_produto"
class="form-control" placeholder="Foto" />
            <input type="hidden" id="im_produto" name="im_produto" value="{{ $produto-
>im_produto }}" />
            <div class="component-preloader"></div>
        </div>
    </div>
    <div class="form-group">
        <div class="col-md-offset-4 col-md-6">
            <button type="submit" class="btn btn-info">Editar</button>
            {{ csrf_field() }}
            {{ method_field('PUT') }}
        </div>
    </div>
</form>
</div>
</div>
</div>
@stopLIST
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você poderá criar, editar e excluir os registros de
produtos cadastrados no sistema.'>
        <h1>Listagem do produto</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/produto/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
            <a href="#/produto/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório
da página atual em PDF.'>PDF</a>
            <a href="/produto" class="btn btn-default" data-intro='Clique aqui para atualizar a
tela.'>Atualizar</a>
            <a href="/produto/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
        </div>
    </div>

    @if(Session::has('message'))
    <div class="alert alert-success">
        {{ Session::get('message') }}
    </div>
    @endif

    <table class="table table-striped table-hovered table-condesend" data-intro='Aqui você confere todos
os registros já cadastrados.'>
        <thead>
            <tr>
                <th>#</th>
                <th>Nome</th>

```

```

        <th>Valor</th>
        <th>Ação</th>
    </tr>
</thead>
<tbody>
    @forelse($listaProdutos as $produto)
    <tr>
        <td>{{ $produto->cd_produto }}</td>
        <td>{{ $produto->nm_produto }}</td>
        <td>{{ dinheiro($produto->vl_produto) }}</td>
        <td><a href="/produto/{{ $produto->cd_produto }}" type="button" class="btn btn-
default">Ver</a></td>
    </tr>
    @empty
    <tr>
        <td colspan="6" class="aling-center">
            Não há produtos cadastrados
        </div>
    </tr>
    @endforelse
</tbody>
</table>
@stop

```

## VIEW - Logon

### LOGON

```

<!DOCTYPE html>
<html lang="{{ config('app.locale') }}">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <meta name="viewport" content="width=device-width, initial-scale=1, maximum-scale=1, user-
scalable=no">

        <title>Autenticação</title>

        <!-- Fonts -->
        <link href="https://fonts.googleapis.com/css?family=Raleway:100,600" rel="stylesheet"
type="text/css">
        <link href="/css/app.css?{{ time() }}" rel="stylesheet" type="text/css">
    </head>
    <body>

        <div class="layout-logon">

            @if (count($errors) > 0)
            <div class="alert alert-danger">
                <ul>
                    @foreach ($errors->all() as $error)
                        <li>{{ $error }}</li>
                    @endforeach
                </ul>
            </div>
            @endif

            <div class="layout-conteudo">
                <div class="component-title">
                    <h1>Olá! Para continuar, digite o seu e-mail e senha</h1>
                </div>
            </div>
        </div>
    </body>
</html>

```

```

    @if(Session::has('message'))
        <div class="alert alert-warning">
            {{ Session::get('message') }}
        </div>
    @endif

    <form class="form-horizontal" action="/" method="POST">
        <div class="form-group">
            <label for="nm_email" class="col-md-4 control-label">E-mail</label>
            <div class="col-md-6">
                <input type="text" class="form-control" name="nm_email" placeholder="E-mail"
required />
            </div>
        </div>
        <div class="form-group">
            <label for="nm_senha" class="col-md-4 control-label">Senha</label>
            <div class="col-md-6">
                <input type="password" class="form-control" name="nm_senha" placeholder="Senha"
required />
            </div>
        </div>
        <div class="form-group">
            <div class="col-md-offset-2 col-md-8">
                <button type="submit" class="btn btn-primary btn-lg btn-block">Autenticar</button>
                {{ csrf_field() }}
                {{ method_field('POST') }}
            </div>
        </div>
    </form>

</div>

</div>

<script src="/js/app.js?{{ time() }}" type="text/javascript"></script>
<script src="/js/intro.js?{{ time() }}" type="text/javascript"></script>
</body>
</html>
SHOW
@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Detalhes do produto</h1>
    </div>

    <form class="form-horizontal" action="/produto/{{ $produto->cd_produto }}" method="POST">
        {{ csrf_field() }}
        {{ method_field('DELETE') }}
        <div class="component-barra-menu">
            <div class="btn-group pull-right" role="group">
                <a href="/produto/help" class="btn btn-default btn-help">Ajuda</a>
                <a href="/produto/" class="btn btn-default">Listagem</a>
                <a href="/produto/{{ $produto->cd_produto }}/edit" class="btn btn-default">Editar</a>
                <button type="submit" class="btn btn-danger">Excluir</button>
            </div>
        </div>
    </form>

```

```

<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">

      <dl class="dl-horizontal">
        <dt>Nome:</dt>
        <dd>{{ $produto->nm_produto }}</dd>
        <dt>Descrição:</dt>
        <dd>{{ $produto->ds_produto }}</dd>
        <dt>Valor:</dt>
        <dd>{{ dinheiro($produto->vl_produto) }}</dd>
        <dt>Foto:</dt>
        <dd></dd>
        <dt>Qtd minima:</dt>
        <dd>{{ $produto->qt_minima_produto }}</dd>
        <dt>Qtd máxima:</dt>
        <dd>{{ $produto->qt_maxima_produto }}</dd>
        <dt>Qtd estoque:</dt>
        <dd>{{ $produto->qt_estoque_produto }}</dd>
      </dl>

    </div>
  </div>
</div>
@stop
EDIT
@extends('layout/public')
@section('content')
  <div class="component-title" data-intro="Nessa tela você irá editar um usuario já cadastrado no sistema.">
    <h1>Edição do usuario</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
      <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma ajuda igual a essa na página.'>Ajuda</a>
    </div>
  </div>

  @if(Session::has('message'))
    <div class="alert alert-success">
      {{ Session::get('message') }}
    </div>
  @endif

  @if (count($errors) > 0)
    <div class="alert alert-danger">
      <ul>
        @foreach ($errors->all() as $error)
          <li>{{ $error }}</li>
        @endforeach
      </ul>
    </div>
  @endif
<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <form class="form-horizontal" action="/usuario/{{ $usuario->cd_usuario }}" method="POST">

```

```

<div class="form-group">
  <label for="nm_usuario" class="col-md-4 control-label">Nome :</label>
  <div class="col-md-6">
    <input type="text" class="form-control" id="nm_usuario" name="nm_usuario"
placeholder="Nome do usuario" value="{{ $usuario->nm_usuario }}" required/>
  </div>
</div>

  <div class="form-group">
    <label for="nm_senha" class="col-md-4 control-label">Senha :</label>
    <div class="col-md-6">
      <input type="text" class="form-control" id="nm_senha" name="nm_senha"
placeholder="Senha do usuario" value="{{ $usuario->nm_senha }}" required/>
    </div>
  </div>

  <div class="form-group">
    <label for="nm_email" class="col-md-4 control-label">Email :</label>
    <div class="col-md-6">
      <input type="email" class="form-control" id="nm_email" name="nm_email"
placeholder="Email do usuario" value="{{ $usuario->nm_email }}" required/>
    </div>
  </div>

  <div class="form-group">
    <label for="cd_departamento" class="col-md-4 control-label">Departamento</label>
    <div class="col-md-6">
      {{ Form::select('cd_departamento', $departamentos, NULL,['class' => 'form-control']) }}
    </div>
  </div>

  <div class="form-group">
    <div class="col-md-offset-4 col-md-6">
      <button type="submit" class="btn btn-info">Editar</button>
      {{ csrf_field() }}
      {{ method_field('PUT') }}
    </div>
  </div>
</form>
</div>
</div>
</div>
@stopLIST
@extends('layout/public')
@section('content')
  <div class="component-title" data-intro="Nessa tela você irá manusear as informações dos usuarios.">
    <h1>Listagem de usuarios</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
      <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
      <a href="#/usuario/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório
da página atual em PDF.'>PDF</a>
      <a href="/usuario" class="btn btn-default" data-intro='Clique aqui para atualizar a
página.'>Atualizar</a>
      <a href="/usuario/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
    </div>
  </div>

  <table class="table table-striped table-hovered table-condesend" data-intro='Aqui você confere todos
os registros já cadastrados.'>

```



```

<thead>
  <tr>
    <th>#</th>
    <th>Nome</th>
    <th>Email</th>
    <th>Ação</th>
  </tr>
</thead>
<tbody>
  @forelse($listaUsuario as $usuario)
    <tr>
      <td>{{ $usuario->cd_usuario }}</td>
      <td>{{ $usuario->nm_usuario }}</td>
      <td>{{ $usuario->nm_email }}</td>
      <td><a href="/usuario/{{ $usuario->cd_usuario }}" type="button" class="btn btn-
default">Ver</a></td>
    </tr>
  @empty
    <tr>
      <td colspan="6" class="aling-center">
        Não há usuarios cadastrados
      </td>
    </tr>
  @endforelse
</tbody>
</table>
@stop
SHOW
@extends('layout/public')
@section('content')
  <div class="component-title" data-intro='Nessa tela você poderá editar e excluir os registros do
usuario.'>
    <h1>Detalhes do usuario</h1>
  </div>

  <form class="form-horizontal" action="/usuario/{{ $usuario->cd_usuario }}" method="POST">
    {{ csrf_field() }}
    {{ method_field('DELETE') }}
    <div class="component-barra-menu">
      <div class="btn-group pull-right" role="group">
        <a href="#/usuario/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
        <a href="/usuario/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem de
usuarios.'>Listagem</a>
        <a href="/usuario/{{ $usuario->cd_usuario }}/edit" class="btn btn-default" data-intro='Clique aqui
para editar os dados do usuario.'>Editar</a>
        <button type="submit" class="btn btn-danger" data-intro='Clique aqui para excluir o
usuario.'>Excluir</button>
      </div>
    </div>
  </form>

  <div class="container-fluid">
    <div class="row">
      <div class="col-md-12">

        <dl class="dl-horizontal">
          <dt>Nome:</dt>
          <dd>{{ $usuario->nm_usuario }}</dd>
        </dl>

```

```

<dl class="dl-horizontal">
  <dt>Email:</dt>
  <dd>{{ $usuario->nm_email }}</dd>
</dl>
<dl class="dl-horizontal">
  <dt>Senha:</dt>
  <dd>{{ $usuario->nm_senha }}</dd>
</dl>
<dl class="dl-horizontal">
  <dt>Codigo do Departamento:</dt>
  <dd>{{ $usuario->cd_departamento }}</dd>
</dl>
</div>
</div>
</div>
@stop

```

## VIEW - Produtos

```

CREATE
@extends('layout/public')
@section('content')
  <div class="component-title">
    <h1>Cadastro de produto</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
      <a href="/produto/help" class="btn btn-default">Ajuda</a>
    </div>
  </div>

  @if (count($errors) > 0)
  <div class="alert alert-danger">
    <ul>
      @foreach ($errors->all() as $error)
        <li>{{ $error }}</li>
      @endforeach
    </ul>
  </div>
  @endif

  <div class="container-fluid">
    <div class="row">
      <div class="col-md-12">
        <form class="form-horizontal" action="/produto" method="POST">
          <div class="form-group">
            <label for="nm_produto" class="col-md-4 control-label">Nome</label>
            <div class="col-md-6">
              <input type="text" class="form-control" id="nm_produto" name="nm_produto"
placeholder="Nome" />
            </div>
          </div>
          <div class="form-group">
            <label for="ds_produto" class="col-md-4 control-label">Descrição</label>
            <div class="col-md-6">
              <textarea class="form-control" id="ds_produto" name="ds_produto"></textarea>
            </div>
          </div>
        </form>
      </div>
    </div>
  </div>

```

```

<div class="form-group">
  <label for="vl_produto" class="col-md-4 control-label">Valor</label>
  <div class="col-md-6">
    <input type="text" class="form-control" id="vl_produto" name="vl_produto"
placeholder="R$ 0,00" data-dinheiro="true" required />
  </div>
</div>
<div class="form-group">
  <label for="im_produto" class="col-md-4 control-label">Foto</label>
  <div class="col-md-6">
    <input type="file" data-loading=".component-preloader" data-srcimg="#im_produto"
class="form-control" placeholder="Foto" />
    <input type="hidden" id="im_produto" name="im_produto" value="" />
    <div class="component-preloader"></div>
  </div>
</div>
<div class="form-group">
  <label for="qt_minima_produto" class="col-md-4 control-label">Quantidade minima</label>
  <div class="col-md-6">
    <input type="text" class="form-control" id="qt_minima_produto"
name="qt_minima_produto" placeholder="0" required />
  </div>
</div>
<div class="form-group">
  <label for="qt_maxima_produto" class="col-md-4 control-label">Quantidade máxima</label>
  <div class="col-md-6">
    <input type="text" class="form-control" id="qt_maxima_produto"
name="qt_maxima_produto" placeholder="0" required />
  </div>
</div>
<div class="form-group">
  <label for="qt_estoque_produto" class="col-md-4 control-label">Quantidade atual</label>
  <div class="col-md-6">
    <input type="text" class="form-control" id="qt_estoque_produto"
name="qt_estoque_produto" placeholder="0" required />
  </div>
</div>
<div class="form-group">
  <div class="col-md-offset-4 col-md-6">
    <button type="submit" class="btn btn-info">Cadastrar</button>
    {{ csrf_field() }}
    {{ method_field('POST') }}
  </div>
</div>
</form>
</div>
</div>
</div>
@stop
EDIT
@extends('layout/public')
@section('content')
  <div class="component-title">
    <h1>Cadastro de produto</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group">
      <a href="/produto/help" class="btn btn-default">Ajuda</a>
    </div>

```

```

</div>

@if (count($errors) > 0)
<div class="alert alert-danger">
    <ul>
        @foreach ($errors->all() as $error)
            <li>{{ $error }}</li>
        @endforeach
    </ul>
</div>
@endif

<div class="container-fluid">
    <div class="row">
        <div class="col-md-12">
            <form class="form-horizontal" action="/produto/{{ $produto->cd_produto }}" method="POST">
                <div class="form-group">
                    <label for="nm_produto" class="col-md-4 control-label">Nome</label>
                    <div class="col-md-6">
                        <input type="text" class="form-control" id="nm_produto" name="nm_produto"
placeholder="Nome" value="{{ $produto->nm_produto }}" />
                    </div>
                </div>
                <div class="form-group">
                    <label for="ds_produto" class="col-md-4 control-label">Descrição</label>
                    <div class="col-md-6">
                        <textarea class="form-control" id="ds_produto" name="ds_produto">{{ $produto-
>ds_produto }}</textarea>
                    </div>
                </div>
                <div class="form-group">
                    <label for="im_produto" class="col-md-4 control-label">Foto</label>
                    <div class="col-md-6">
                        <input type="file" data-loadimg=".component-preloader" data-srcimg="#im_produto"
class="form-control" placeholder="Foto" />
                        <input type="hidden" id="im_produto" name="im_produto" value="{{ $produto-
>im_produto }}" />
                        <div class="component-preloader"></div>
                    </div>
                </div>
                <div class="form-group">
                    <div class="col-md-offset-4 col-md-6">
                        <button type="submit" class="btn btn-info">Editar</button>
                        {{ csrf_field() }}
                        {{ method_field('PUT') }}
                    </div>
                </div>
            </form>
        </div>
    </div>
</div>

@stop
LIST
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro="Nessa tela você poderá criar, editar e excluir os registros de
produtos cadastrados no sistema.">
        <h1>Listagem do produto</h1>
    </div>

```

```

<div class="component-barra-menu">
  <div class="btn-group pull-right" role="group">
    <a href="#/produto/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma
ajuda igual a essa na página.'>Ajuda</a>
    <a href="#/produto/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório
da página atual em PDF.'>PDF</a>
    <a href="/produto" class="btn btn-default" data-intro='Clique aqui para atualizar a
tela.'>Atualizar</a>
    <a href="/produto/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
  </div>
</div>

@if(Session::has('message'))
<div class="alert alert-success">
  {{ Session::get('message') }}
</div>
@endif

<table class="table table-striped table-hovered table-condensed" data-intro='Aqui você confere todos
os registros já cadastrados.'>
  <thead>
    <tr>
      <th>#</th>
      <th>Nome</th>
      <th>Valor</th>
      <th>Ação</th>
    </tr>
  </thead>
  <tbody>
    @forelse($listaProdutos as $produto)
      <tr>
        <td>{{ $produto->cd_produto }}</td>
        <td>{{ $produto->nm_produto }}</td>
        <td>{{ dinheiro($produto->vl_produto) }}</td>
        <td><a href="/produto/{{ $produto->cd_produto }}" type="button" class="btn btn-
default">Ver</a></td>
      </tr>
    @empty
      <tr>
        <td colspan="6" class="aling-center">
          Não há produtos cadastrados
        </td>
      </tr>
    @endforelse
  </tbody>
</table>

@stop
SHOW
@extends('layout/public')
@section('content')
  <div class="component-title">
    <h1>Detalhes do produto</h1>
  </div>

  <form class="form-horizontal" action="/produto/{{ $produto->cd_produto }}" method="POST">
    {{ csrf_field() }}
    {{ method_field('DELETE') }}
  </div>
  <div class="component-barra-menu">

```

```

<div class="btn-group pull-right" role="group">
  <a href="/produto/help" class="btn btn-default btn-help">Ajuda</a>
  <a href="/produto/" class="btn btn-default">Listagem</a>
  <a href="/produto/{ {$produto->cd_produto} }/edit" class="btn btn-default">Editar</a>
  <button type="submit" class="btn btn-danger">Excluir</button>
</div>
</div>
</form>

<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">

      <dl class="dl-horizontal">
        <dt>Nome:</dt>
        <dd>{ {$produto->nm_produto} }</dd>
        <dt>Descrição:</dt>
        <dd>{ {$produto->ds_produto} }</dd>
        <dt>Valor:</dt>
        <dd>{ {$dinheiro($produto->vl_produto)} }</dd>
        <dt>Foto:</dt>
        <dd></dd>
        <dt>Qtd mínima:</dt>
        <dd>{ {$produto->qt_minima_produto} }</dd>
        <dt>Qtd máxima:</dt>
        <dd>{ {$produto->qt_maxima_produto} }</dd>
        <dt>Qtd estoque:</dt>
        <dd>{ {$produto->qt_estoque_produto} }</dd>
      </dl>

    </div>
  </div>
</div>
@stop

```

## VIEW - Contas

```

CREATE
@extends('layout/public')
@section('content')
  <div class="component-title" data-intro="Nessa tela você irá cadastrar uma nova conta no sistema.">
    <h1>Cadastro de contas</h1>
  </div>

  <div class="component-barra-menu">
    <div class="btn-group pull-right" role="group" >
      <a href="#/conta/help" class="btn btn-default btn-help" data-intro="Clique aqui para ter uma ajuda igual a essa na página.">Ajuda</a>
      <a href="/conta/" class="btn btn-default" data-intro="Clique aqui para voltar na listagem de contas.">Listagem</a>
    </div>
  </div>

  @if (count($errors) > 0)
    <div class="alert alert-danger">
      <ul>
        @foreach ($errors->all() as $error)
          <li>{ { $error } }</li>
        @endforeach
      </ul>
    </div>
  @endif

```

```

</div>
@endif

<div class="container-fluid">
  <div class="row">
    <div class="col-md-12">
      <form class="form-horizontal" action="/conta" method="POST">
        <div class="form-group">
          <label for="nm_conta" class="col-md-4 control-label">Nome :</label>
          <div class="col-md-6">
            <input type="text" class="form-control" id="nm_conta" name="nm_conta"
placeholder="Nome da conta" required/>
          </div>
        </div>
        <div class="form-group">
          <label for="cd_agencia_conta" class="col-md-4 control-label">Agência :</label>
          <div class="col-md-6">
            <input type="number" class="form-control" id="cd_agencia_conta"
name="cd_agencia_conta" placeholder="Nº da agência" required/>
          </div>
        </div>
        <div class="form-group">
          <label for="cd_numero_conta" class="col-md-4 control-label">Conta :</label>
          <div class="col-md-6">
            <input type="number" class="form-control" id="cd_numero_conta"
name="cd_numero_conta" placeholder="Nº da conta" required/>
          </div>
        </div>
        <div class="form-group">
          <label for="vl_inicial_conta" class="col-md-4 control-label">Valor Inicial:</label>
          <div class="col-md-6">
            <input type="text" class="form-control" id="vl_inicial_conta" name="vl_inicial_conta"
placeholder="Valor inicial R$" required/>
          </div>
        </div>
        <div class="form-group">
          <label for="ic_tipo_conta" class="col-md-4 control-label">Tipo de conta :</label>
          <div class="col-md-6">
            <div class="radio">
              <label>
                <input type="radio" id="nm_tipo_conta0" name="nm_tipo_conta"
value="poupanca"> Poupança
              </label>
            </div>
            <div class="radio">
              <label>
                <input type="radio" id="nm_tipo_conta1" name="nm_tipo_conta"
value="corrente"> Corrente
              </label>
            </div>
            <div class="radio">
              <label>
                <input type="radio" id="nm_tipo_conta2" name="nm_tipo_conta"
value="cartao"> Cartão
              </label>
            </div>
            <div class="radio">
              <label>
                <input type="radio" id="nm_tipo_conta3" name="nm_tipo_conta"
value="investimento"> Investimento
              </label>
            </div>
          </div>
        </div>
      </form>
    </div>
  </div>
</div>

```

```

        </label>
    </div>
    <div class="radio">
        <label>
            <input type="radio" id="nm_tipo_conta4" name="nm_tipo_conta" value="caixa">
Caixa
        </label>
    </div>
    <div class="radio">
        <label>
            <input type="radio" id="nm_tipo_conta5" name="nm_tipo_conta"
value="outras"> Outras
        </label>
    </div>
</div>
<div class="form-group">
    <div class="col-md-offset-4 col-md-6">
        <button type="submit" class="btn btn-default">Cadastrar</button>
        {{ csrf_field() }}
        {{ method_field('POST') }}
    </div>
</div>
</form>
</div>
</div>
</div>
@stop
EDIT
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você irá editar a conta já cadastrada no sistema.'>
        <h1>Edição da conta</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/conta/help" class="btn btn-default btn-help">Ajuda</a>
            <a href="/conta/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem de
contas.'>Listagem</a>
        </div>
    </div>
    @if (count($errors) > 0)
        <div class="alert alert-danger">
            <ul>
                @foreach ($errors->all() as $error)
                    <li>{{ $error }}</li>
                @endforeach
            </ul>
        </div>
    @endif

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">
                <form class="form-horizontal" action="/conta/{{ $conta->cd_conta }}" method="POST">
                    <div class="form-group">
                        <label for="nm_conta" class="col-md-4 control-label">Nome :</label>
                        <div class="col-md-6">

```



```

        <input type="text" class="form-control" id="nm_conta" name="nm_conta"
placeholder="Nome da conta" value="{{ $conta->nm_conta }}" required/>
    </div>
</div>
<div class="form-group">
    <label for="cd_agencia_conta" class="col-md-4 control-label">Agência :</label>
    <div class="col-md-6">
        <input type="number" class="form-control" id="cd_agencia_conta"
name="cd_agencia_conta" placeholder="Nº da agência" value="{{ $conta->cd_agencia_conta }}"
required/>
    </div>
</div>
<div class="form-group">
    <label for="cd_numero_conta" class="col-md-4 control-label">Conta :</label>
    <div class="col-md-6">
        <input type="number" class="form-control" id="cd_numero_conta"
name="cd_numero_conta" placeholder="Nº da conta" value="{{ $conta->cd_numero_conta }}" required/>
    </div>
</div>
<div class="form-group">
    <label for="vl_inicial_conta" class="col-md-4 control-label">Valor Inicial:</label>
    <div class="col-md-6">
        <input type="text" class="form-control" id="vl_inicial_conta" name="vl_inicial_conta"
placeholder="Valor inicial R$" value="{{ $conta->vl_inicial_conta }}" required/>
    </div>
</div>
<div class="form-group">
    <label for="ic_tipo_conta" class="col-md-4 control-label">Tipo de conta :</label>
    <div class="col-md-6">
        <div class="radio">
            <label>
                <input type="radio" id="nm_tipo_conta0" name="nm_tipo_conta"
value="poupanca" {{ $conta->nm_tipo_conta=="poupanca"?checked:"" }}> Poupança
            </label>
        </div>
        <div class="radio">
            <label>
                <input type="radio" id="nm_tipo_conta1" name="nm_tipo_conta"
value="corrente" {{ $conta->nm_tipo_conta=="corrente"?checked:"" }}> Corrente
            </label>
        </div>
        <div class="radio">
            <label>
                <input type="radio" id="nm_tipo_conta2" name="nm_tipo_conta" value="cartao"
{{ $conta->nm_tipo_conta=="cartao"?checked:"" }}> Cartão
            </label>
        </div>
        <div class="radio">
            <label>
                <input type="radio" id="nm_tipo_conta3" name="nm_tipo_conta"
value="investimento" {{ $conta->nm_tipo_conta=="investimento"?checked:"" }}> Investimento
            </label>
        </div>
        <div class="radio">
            <label>
                <input type="radio" id="nm_tipo_conta4" name="nm_tipo_conta" value="caixa"
{{ $conta->nm_tipo_conta=="caixa"?checked:"" }}> Caixa
            </label>
        </div>
        <div class="radio">

```

```

        <label>
            <input type="radio" id="nm_tipo_conta5" name="nm_tipo_conta" value="outras"
{{ $conta->nm_tipo_conta=="outras"? "checked":"" }}> Outras
        </label>
    </div>
</div>
</div>
<div class="form-group">
    <div class="col-md-offset-4 col-md-6">
        <button type="submit" class="btn btn-info">Editar</button>
        {{ csrf_field() }}
        {{ method_field('PUT') }}
    </div>
</div>
</form>
</div>
</div>
</div>
@stop

```

## LIST

```

@extends('layout/public')
@section('content')
    <div class="component-title" data-intro="Nessa tela você irá manusear as informações das contas da
empresa.">
        <h1>Listagem de contas</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/conta/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma ajuda
igual a essa na página.'>Ajuda</a>
            <a href="#/conta/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um relatório da
página atual em PDF.'>PDF</a>
            <a href="/conta" class="btn btn-default" data-intro='Clique aqui para atualizar a
tela.'>Atualizar</a>
            <a href="/conta/create" class="btn btn-default" data-intro='Clique aqui para adicionar um novo
registro.'>Novo Registro</a>
        </div>
    </div>

    @if(Session::has('message'))
        <div class="alert alert-success">
            {{ Session::get('message') }}
        </div>
    @endif

    <table class="table table-striped table-hovered table-condensed" data-intro='Aqui você confere todos
os registros já cadastrados.'>
        <thead>
            <tr>
                <th>#</th>
                <th>Nome</th>
                <th>Tipo</th>
                <th>Saldo atual</th>
                <th>Ação</th>
            </tr>
        </thead>
        <tbody>
            @forelse($listaContas as $conta)

```

```

        <tr>
            <td>{{ $conta->cd_conta }}</td>
            <td>{{ $conta->nm_conta }}</td>
            <td>{{ $conta->nm_tipo_conta }}</td>
            <td>{{ $conta->vl_atual_conta }}</td>
            <td><a href="/conta/{{ $conta->cd_conta }}" type="button" class="btn btn-
default">Ver</a></td>
        </tr>
    @empty
        <tr>
            <td colspan="5" class="aling-center">
                Não há contas cadastradas
            </td>
        </tr>
    @endforelse
</tbody>
</table>
@stop
SHOW
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você poderá editar ou excluir a conta.'>
        <h1>Detalhes da conta</h1>
    </div>

    <form class="form-horizontal" action="/conta/{{ $conta->cd_conta }}" method="POST">
        {{ csrf_field() }}
        {{ method_field('DELETE') }}
        <div class="component-barra-menu">
            <div class="btn-group pull-right" role="group">
                <a href="#/conta/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter uma ajuda
igual a essa na página.'>Ajuda</a>
                <a href="/conta/" class="btn btn-default" data-intro='Clique aqui para voltar na listagem de
contas.'>Listagem</a>
                <a href="/conta/{{ $conta->cd_conta }}/edit" class="btn btn-default" data-intro='Clique aqui para
editar os dados da conta.'>Editar</a>
                <button type="submit" class="btn btn-danger" data-intro='Clique aqui para excluir a
conta.'>Excluir</button>
            </div>
        </div>
    </form>

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">

                <dl class="dl-horizontal">
                    <dt>Nome:</dt>
                    <dd>{{ $conta->nm_conta }}</dd>
                    <dt>Agência:</dt>
                    <dd>{{ $conta->cd_agencia_conta }}</dd>
                    <dt>Conta:</dt>
                    <dd>{{ $conta->cd_numero_conta }}</dd>
                    <dt>Valor Atual:</dt>
                    <dd>{{ $conta->vl_atual_conta }}</dd>
                </dl>

            </div>
        </div>
    </div>@stop

```

## VIEW - Movimentações

```

CREATE
@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Cadastro de movimentação</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="/movimentacao" class="btn btn-default">Listagem</a>
            <a href="/movimentacao/help" class="btn btn-default">Ajuda</a>
        </div>
    </div>

    @if (count($errors) > 0)
    <div class="alert alert-danger">
        <ul>
            @foreach ($errors->all() as $error)
                <li>{{ $error }}</li>
            @endforeach
        </ul>
    </div>
    @endif

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">
                <form class="form-horizontal" action="/movimentacao" method="POST">
                    <div class="form-group">
                        <label for="nm_movimentacao" class="col-md-4 control-label">Movimentação</label>
                        <div class="col-md-6">
                            <input type="text" class="form-control" name="nm_movimentacao" placeholder="Nome"
required />
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="ic_tipo_movimentacao" class="col-md-4 control-label">Tipo</label>
                        <div class="col-md-6">
                            <label class="radio-inline">
                                <input type="radio" name="ic_tipo_movimentacao" value="credito" required /> Receita
                            </label>
                            <label class="radio-inline">
                                <input type="radio" name="ic_tipo_movimentacao" value="debito" required /> Despesa
                            </label>
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="cd_conta" class="col-md-4 control-label">Conta</label>
                        <div class="col-md-6">
                            {{ Form::select('cd_conta', $contas, NULL, ['class' => 'form-control',
required=>'required']) }}
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="dt_movimentacao" class="col-md-4 control-label">Data</label>
                        <div class="col-md-6">
                            <input type="text" class="form-control" name="dt_movimentacao"
value="{{ date('d/m/Y') }}" required data-calendar="true" />
                        </div>
                    </div>
                </form>
            </div>
        </div>
    </div>

```

```

    </div>
</div>
<div class="form-group">
  <label for="cd_nf_movimentacao" class="col-md-4 control-label">Cód. Nota Fiscal</label>
  <div class="col-md-6">
    <input type="text" class="form-control" name="cd_nf_movimentacao" value="" />
  </div>
</div>
<div class="form-group">
  <label for="ic_pago_sim_nao" class="col-md-4 control-label">Situação da
movimentação</label>
  <div class="col-md-6">
    <label class="radio-inline">
      <input type="radio" name="ic_pago_sim_nao" value="sim" required /> Pago
    </label>
    <label class="radio-inline">
      <input type="radio" name="ic_pago_sim_nao" value="nao" required checked /> Não
    </label>
  </div>
</div>
<div class="form-group">
  <label for="vl_movimentacao" class="col-md-4 control-label">Valor movimentação</label>
  <div class="col-md-6">
    <input type="text" class="form-control" name="vl_movimentacao" placeholder="0,00"
required data-dinheiro="true" />
  </div>
</div>
<div class="form-group">
  <label for="ds_movimentacao" class="col-md-4 control-label">Descrição</label>
  <div class="col-md-6">
    <textarea class="form-control" name="ds_movimentacao"></textarea>
  </div>
</div>
<div class="form-group">
  <label for="ic_recorrente_sim_nao" class="col-md-4 control-label">Movimentação
recorrente</label>
  <div class="col-md-6">
    <label class="radio-inline">
      <input type="radio" name="ic_recorrente_sim_nao" value="sim" required /> Sim
    </label>
    <label class="radio-inline">
      <input type="radio" name="ic_recorrente_sim_nao" value="nao" required checked />
    </label>
  </div>
</div>
<div class="form-group">
  <div class="col-md-offset-4 col-md-6">
    <button type="submit" class="btn btn-info">Cadastrar</button>
    {{ csrf_field() }}
    {{ method_field('POST') }}
  </div>
</div>
</form>
</div>
</div>
</div>
@stop
EDIT

```

```

@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Cadastro de movimentacao</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="/movimentacao/{{ $movimentacao->cd_movimentacao }}" class="btn btn-
default">Voltar</a>
            <a href="/movimentacao/help" class="btn btn-default">Ajuda</a>
        </div>
    </div>

    @if (count($errors) > 0)
    <div class="alert alert-danger">
        <ul>
            @foreach ($errors->all() as $error)
                <li>{{ $error }}</li>
            @endforeach
        </ul>
    </div>
    @endif

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">
                <form class="form-horizontal" action="/movimentacao/{{ $movimentacao->cd_movimentacao }}"
method="POST">
                    <div class="form-group">
                        <label for="nm_movimentacao" class="col-md-4 control-label">Movimentação</label>
                        <div class="col-md-6">
                            <input type="text" class="form-control" name="nm_movimentacao" placeholder="Nome"
required value="{{ $movimentacao->nm_movimentacao }}" />
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="ic_tipo_movimentacao" class="col-md-4 control-label">Tipo</label>
                        <div class="col-md-6">
                            <label class="radio-inline">
                                <input type="radio" name="ic_tipo_movimentacao" value="credito" required
{{ checkSe('credito', $movimentacao->ic_tipo_movimentacao)}} /> Receita
                            </label>
                            <label class="radio-inline">
                                <input type="radio" name="ic_tipo_movimentacao" value="debito" required
{{ checkSe('debito', $movimentacao->ic_tipo_movimentacao)}} /> Despesa
                            </label>
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="cd_conta" class="col-md-4 control-label">Conta</label>
                        <div class="col-md-6">
                            {{ Form::select('cd_conta', $contas, $movimentacao->cd_conta, ['class' => 'form-control',
'required'=>'required']) }}
                        </div>
                    </div>
                    <div class="form-group">
                        <label for="dt_movimentacao" class="col-md-4 control-label">Data</label>
                        <div class="col-md-6">

```

```

        <input type="text" class="form-control" name="dt_movimentacao"
value="{{ $movimentacao->dt_movimentacao->format('d/m/Y') }}" required data-calendario="true" />
    </div>
</div>
<div class="form-group">
    <label for="cd_nf_movimentacao" class="col-md-4 control-label">Cód. Nota Fiscal</label>
    <div class="col-md-6">
        <input type="text" class="form-control" name="cd_nf_movimentacao"
value="{{ $movimentacao->cd_nf_movimentacao }}" />
    </div>
</div>
<div class="form-group">
    <label for="ic_pago_sim_nao" class="col-md-4 control-label">Situação da
movimentação</label>
    <div class="col-md-6">
        <label class="radio-inline">
            <input type="radio" name="ic_pago_sim_nao" value="sim" required {{ checkSe('sim',
$movimentacao->ic_pago_sim_nao)}} /> Pago
        </label>
        <label class="radio-inline">
            <input type="radio" name="ic_pago_sim_nao" value="nao" required {{ checkSe('nao',
$movimentacao->ic_pago_sim_nao)}} /> Não pago
        </label>
    </div>
</div>
<div class="form-group">
    <label for="vl_movimentacao" class="col-md-4 control-label">Valor movimentação</label>
    <div class="col-md-6">
        <input type="text" class="form-control" name="vl_movimentacao"
value="{{ dinheiro($movimentacao->vl_movimentacao) }}" required data-dinheiro="true" />
    </div>
</div>
<div class="form-group">
    <label for="ds_movimentacao" class="col-md-4 control-label">Descrição</label>
    <div class="col-md-6">
        <textarea class="form-control" name="ds_movimentacao">{{ $movimentacao-
>ds_movimentacao }}</textarea>
    </div>
</div>
<div class="form-group">
    <label for="ic_recorrente_sim_nao" class="col-md-4 control-label">Movimentação
recorrente</label>
    <div class="col-md-6">
        <label class="radio-inline">
            <input type="radio" name="ic_recorrente_sim_nao" value="sim" required
{{ checkSe('sim', $movimentacao->ic_recorrente_sim_nao)}} /> Sim
        </label>
        <label class="radio-inline">
            <input type="radio" name="ic_recorrente_sim_nao" value="nao" required
{{ checkSe('nao', $movimentacao->ic_recorrente_sim_nao)}} /> Não
        </label>
    </div>
</div>
<div class="form-group">
    <div class="col-md-offset-4 col-md-6">
        <button type="submit" class="btn btn-info">Editar</button>
        {{ csrf_field() }}
        {{ method_field('PUT') }}
    </div>
</div>

```

```

</form>

</div>
</div>
</div>
@stop
LIST
@extends('layout/public')
@section('content')
    <div class="component-title" data-intro='Nessa tela você irá manusear as informações das
    movimentações da empresa.'>
        <h1>Listagem de Movimentação</h1>
    </div>

    <div class="component-barra-menu">
        <div class="btn-group pull-right" role="group">
            <a href="#/movimentacao/help" class="btn btn-default btn-help" data-intro='Clique aqui para ter
            uma ajuda igual a essa na página.'>Ajuda</a>
            <a href="#/movimentacao/pdf" class="btn btn-default" data-intro='Clique aqui para baixar um
            relatório da página atual em PDF.'>PDF</a>
            <a href="/movimentacao" class="btn btn-default" data-intro='Clique aqui para atualizar a
            tela.'>Atualizar</a>
            <a href="/movimentacao/create" class="btn btn-default" data-intro='Clique aqui para adicionar
            um novo registro.'>Novo Registro</a>
        </div>
    </div>

    @if(Session::has('message'))
        <div class="alert alert-success">
            {{ Session::get('message') }}
        </div>
    @endif

    <table class="table table-striped table-hovered table-condensed">
        <thead>
            <tr>
                <th>#</th>
                <th>Tipo</th>
                <th>Nome</th>
                <th>Data</th>
                <th>Valor</th>
                <th></th>
            </tr>
        </thead>
        <tbody>
            @forelse($listaMovimentacoes as $movimentacao)
                <tr>
                    <td>{{ $movimentacao->cd_movimentacao }}</td>
                    <td>{{ $movimentacao->conta->nm_conta }}</td>
                    <td>{{ $movimentacao->nm_movimentacao }}</td>
                    <td>{{ $movimentacao->dt_movimentacao->format('d/m/Y') }}</td>
                    <td>{{ dinheiro($movimentacao->vl_movimentacao) }}</td>
                    <td><a href="/movimentacao/{{ $movimentacao->cd_movimentacao }}" type="button"
                    class="btn btn-default">Ver</a></td>
                </tr>
            @empty
                <tr>
                    <td colspan="6" class="aling-center">
                        Não há movimentações cadastradas
                    </td>
                </tr>
            @endforelse
        </tbody>
    </table>

```



```

        </div>
    </tr>
@endforelse
</tbody>
</table>
@stop
SHOW
@extends('layout/public')
@section('content')
    <div class="component-title">
        <h1>Detalhes da movimentacao</h1>
    </div>

    <form class="form-horizontal" action="/movimentacao/{{ $movimentacao->cd_movimentacao }}"
method="POST">
        {{ csrf_field() }}
        {{ method_field('DELETE') }}
        <div class="component-barra-menu">
            <div class="btn-group pull-right" role="group">
                <a href="/movimentacao/help" class="btn btn-default btn-help">Ajuda</a>
                <a href="/movimentacao/" class="btn btn-default">Listagem</a>
                <a href="/movimentacao/{{ $movimentacao->cd_movimentacao }}/edit" class="btn btn-
default">Editar</a>
                <button type="submit" class="btn btn-danger">Excluir</button>
            </div>
        </div>
    </form>

    <div class="container-fluid">
        <div class="row">
            <div class="col-md-12">

                <dl class="dl-horizontal">
                    <dt>Nome:</dt>
                    <dd>{{ $movimentacao->nm_movimentacao }}</dd>
                    <dt>Tipo:</dt>
                    <dd>
                        @if($movimentacao->ic_tipo_movimentacao=="credito")
                            Crédito
                        @else
                            Débito
                        @endif
                    </dd>
                    <dt>Conta:</dt>
                    <dd>{{ $movimentacao->conta->nm_conta }}</dd>
                    <dt>Data:</dt>
                    <dd>{{ $movimentacao->dt_movimentacao->format('d/m/Y') }}</dd>
                    <dt>Cód. Nota Fiscal:</dt>
                    <dd>{{ $movimentacao->cd_nf_movimentacao }}</dd>
                    <dt>Situação:</dt>
                    <dd>
                        @if($movimentacao->ic_pago_sim_nao=="sim")
                            Movimentação paga
                        @else
                            Movimentação não paga
                        @endif
                    </dd>
                    <dt>Valor:</dt>
                    <dd>{{ dinheiro($movimentacao->vl_movimentacao) }}</dd>
                    <dt>Descrição:</dt>

```

```

        <dd>{ {nl2br($movimentacao->ds_movimentacao)} }</dd>
        <dt>Recorrente:</dt>
        <dd>
            @if($movimentacao->ic_recorrente_sim_nao=="sim")
                Movimentação recorrente
            @else
                Movimentação não recorrente
            @endif
        </dd>
    </dl>

</div>
</div>
</div>
@stop

```

## Controller - Departamento

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class DepartamentoController extends Controller
{
    private $nameFolder = "departamento";

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $departamento = Departamento::all();

        //
        return view("{ $this->nameFolder }/list", ["listaDepartamentos"=>$departamento]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        //
        return view("{ $this->nameFolder }/create");
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {

```

```

// Valida
$this->validate($request, [
    'nm_departamento' => 'required'

]);

// Adiciona e salva
$departamento = new Departamento();
$departamento->nm_departamento = $request->nm_departamento;
$departamento->save();

// Redireciona
return redirect('departamento')->with('message', 'Departamento salvo com sucesso!');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
    $departamento = Departamento::find($id);

    return view("{ $this->nameFolder }/show", ["departamento"=>$departamento]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //
    $departamento = Departamento::find($id);

    return view("{ $this->nameFolder }/edit", ["departamento"=>$departamento]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // Valida
    $this->validate($request, [
        'nm_departamento' => 'required'

    ]);

    // Adiciona e salva
    $departamento = Departamento::find($id);

```

```

$departamento->nm_departamento = $request->nm_departamento;
$departamento->save();

// Redireciona
return redirect("departamento/$id")->with('message', 'Departamento salvo com sucesso!');
}

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    // Adiciona e salva
    $departamento = Departamento::find($id);
    $departamento->delete();

    // Redireciona
    return redirect('departamento')->with('message', 'Departamento excluido com sucesso!');
}
}

```

## Controller - Login

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class AutenticacaoController extends Controller
{
    /**
     * Mostra a página de login do sistema
     * @return \Illuminate\Http\Response
     */
    public function index(Request $request)
    {
        return view('sistema/logon');
    }

    /**
     * Valida se um usuário existe no banco de dados
     * e loga ele no sistema
     * @return \Illuminate\Http\Response
     */
    public function logon(Request $request)
    {
        $user = Usuario::where('nm_email', $request->nm_email)
            ->where('nm_senha', $request->nm_senha)
            ->first();

        // Tem um usuário cadastrado
        if ($user) {
            $request->session()->put('user_on', $user->cd_usuario);
            $request->session()->put('user_name', $user->nm_usuario);
            $request->session()->put('user_email', $user->nm_email);

```

```

        return redirect('/home')->with('message', 'Usuário logado com sucesso!');
    }

    return back()->with('message', 'Usuário ou senha inválidos');
}

/**
 * Desloga o usuário do sistema
 * @return \Illuminate\Http\Response
 */
public function logout(Request $request)
{
    $request->session()->put('user_on', 0);
    $request->session()->forget('user_on');

    return redirect('/?3')->with('message', 'Usuário deslogado com sucesso!');
}
}

```

## Controller - Usuários

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class UsuarioController extends Controller
{
    private $nameFolder = "usuario";

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        // Recupera todos os usuários
        $usuario = Usuario::all();

        // Página de listagem
        return view("{ $this->nameFolder }/list", ["listaUsuario"=>$usuario]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        // Lista de departamento
        $departamentos = array();
        foreach(Departamento::all() as $departamento){
            $departamentos[$departamento->cd_departamento] = $departamento->nm_departamento;
        }
        //
        return view("{ $this->nameFolder }/create", ['departamentos'=>$departamentos]);
    }
}

```

```

}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    // Valida
    $this->validate($request,
    [
        'nm_usuario' => 'required',
        'nm_email' => 'required',
        'nm_senha' => 'required',
        'cd_departamento' => 'required',

    ]);

    // Adiciona e salva
    $usuario = new Usuario();
    $usuario->nm_usuario = $request->nm_usuario;
    $usuario->nm_email = $request->nm_email;
    $usuario->nm_senha = $request->nm_senha;
    $usuario->cd_departamento = $request->cd_departamento;
    $usuario->save();

    // Redireciona
    return redirect('usuario')->with('message', 'Usuario salvo com sucesso!');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    // Recupera um registro
    $usuario = Usuario::find($id);

    // Formulário de visualização
    return view("{ $this->nameFolder }/show", ["usuario"=>$usuario]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //Usuario
    $usuario = Usuario::find($id);

    // Lista de departamento
    $departamentos = array();

```

```

foreach(Departamento::all() as $departamento){
    $departamentos[$departamento->cd_departamento] = $departamento->nm_departamento;
}

return view("{ $this->nameFolder}/edit", ["usuario"=>$usuario, 'departamentos'=>$departamentos]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // Valida
    $this->validate($request,
    [
        'nm_usuario' => 'required',
        'nm_email' => 'required',
        'nm_senha' => 'required',
        'cd_departamento' => 'required',
    ]);

    // Adiciona e salva
    $usuario = Usuario::find($id);
    $usuario->nm_usuario = $request->nm_usuario;
    $usuario->nm_email = $request->nm_email;
    $usuario->nm_senha = $request->nm_senha;
    $usuario->cd_departamento = $request->cd_departamento;
    $usuario->save();

    // Redireciona
    return redirect("usuario/$id")->with('message', 'Usuario editado com sucesso!');
}

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    // Adiciona e salva
    $usuario = Usuario::find($id);
    $usuario->delete();

    // Redireciona
    return redirect('usuario')->with('message', 'Usuario deletado com sucesso!');
}
}

```

## Controller - Produtos

<?php

```

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ProdutoController extends Controller
{
    private $nameFolder = "produto";

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $produtos = Produto::all();

        //
        return view("{ $this->nameFolder}/list", ["listaProdutos"=>$produtos]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        //
        return view("{ $this->nameFolder}/create");
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        // Valida
        $this->validate($request, [
            'nm_produto' => 'required',
            'ds_produto' => 'required',
            'vl_produto' => 'required',
            'im_produto' => 'required',
            'qt_minima_produto' => 'required',
            'qt_maxima_produto' => 'required',
            'qt_estoque_produto' => 'required',
        ]);

        // Adiciona e salva
        $produto = new Produto();
        $produto->nm_produto = $request->nm_produto;
        $produto->ds_produto = $request->ds_produto;
        $produto->vl_produto = money2float($request->vl_produto);
        $produto->im_produto = $request->im_produto;
        $produto->qt_minima_produto = $request->qt_minima_produto;
        $produto->qt_maxima_produto = $request->qt_maxima_produto;
    }
}

```



```

$produto->qt_estoque_produto = $request->qt_estoque_produto;
$produto->save();

// Redireciona
return redirect('produto')->with('message', 'Produto salvo com sucesso!');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
    $produto = Produto::find($id);

    return view("{ $this->nameFolder}/show", ["produto"=>$produto]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //
    $produto = Produto::find($id);

    return view("{ $this->nameFolder}/edit", ["produto"=>$produto]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // Valida
    $this->validate($request, [
        'nm_produto' => 'required',
        'ds_produto' => 'required',
        'vl_produto' => 'required',
        'im_produto' => 'required',
        'qt_minima_produto' => 'required',
        'qt_maxima_produto' => 'required',
        'qt_estoque_produto' => 'required',
    ]);

    // Adiciona e salva
    $produto = Produto::find($id);
    $produto->nm_produto = $request->nm_produto;
    $produto->ds_produto = $request->ds_produto;
    $produto->vl_produto = money2float($request->vl_produto);

```

```

$produto->im_produto = $request->im_produto;
$produto->qt_minima_produto = $request->qt_minima_produto;
$produto->qt_maxima_produto = $request->qt_maxima_produto;
$produto->qt_estoque_produto = $request->qt_estoque_produto;
$produto->save();

// Redireciona
return redirect("produto/$id")->with('message', 'Produto salvo com sucesso!');
}

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    // Adiciona e salva
    $produto = Produto::find($id);
    $produto->delete();

    // Redireciona
    return redirect('produto')->with('message', 'Produto salvo com sucesso!');
}
}

```

## Controller - Contas

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class ContaController extends Controller
{
    private $nameFolder = "conta";

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $contas = Conta::all();

        //
        return view("{ $this->nameFolder}/list", ["listaContas"=>$contas]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()

```

```

{
    //
    return view("{ $this->nameFolder}/create");
}

/**
 * Store a newly created resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @return \Illuminate\Http\Response
 */
public function store(Request $request)
{
    // Valida
    $this->validate($request, [
        'nm_conta' => 'required',
        'cd_agencia_conta' => 'required',
        'cd_numero_conta' => 'required',
        'vl_inicial_conta' => 'required',
        'nm_tipo_conta' => 'required',
    ]);

    // Adiciona e salva
    $contas = new Conta();
    $contas->nm_conta = $request->nm_conta;
    $contas->cd_agencia_conta = $request->cd_agencia_conta;
    $contas->cd_numero_conta = $request->cd_numero_conta;
    $contas->vl_inicial_conta = $request->vl_inicial_conta;
    $contas->vl_atual_conta = $request->vl_inicial_conta;
    $contas->nm_tipo_conta = $request->nm_tipo_conta;
    $contas->dt_registro_conta = date('Y-m-d H:i:s');
    $contas->save();

    // Redireciona
    return redirect('conta')->with('message', 'Conta criada com sucesso!');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
    $conta = Conta::find($id);

    return view("{ $this->nameFolder}/show", ["conta"=>$conta]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    //

```

```

        $conta = Conta::find($id);

        return view("{ $this->nameFolder}/edit", ["conta"=>$conta]);
    }

    /**
     * Update the specified resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @param int $id
     * @return \Illuminate\Http\Response
     */
    public function update(Request $request, $id)
    {
        // Valida
        $this->validate($request, [
            'nm_conta' => 'required',
            'cd_agencia_conta' => 'required',
            'cd_numero_conta' => 'required',
            'vl_inicial_conta' => 'required',
        ]);

        // Adiciona e salva
        $contas = Conta::find($id);
        $contas->nm_conta = $request->nm_conta;
        $contas->cd_agencia_conta = $request->cd_agencia_conta;
        $contas->cd_numero_conta = $request->cd_numero_conta;
        $contas->vl_inicial_conta = $request->vl_inicial_conta;
        $contas->save();

        // Redireciona
        return redirect("conta/$id")->with('message', 'Conta salva com sucesso!');
    }

    /**
     * Remove the specified resource from storage.
     *
     * @param int $id
     * @return \Illuminate\Http\Response
     */
    public function destroy($id)
    {
        // Adiciona e salva
        $conta = Conta::find($id);
        $conta->delete();

        // Redireciona
        return redirect('conta')->with('message', 'Conta deletada com sucesso!');
    }
}

```

## Controller - Movimentações

```
<?php
```

```
namespace App\Http\Controllers;
```

```
use Illuminate\Http\Request;
```

```

class MovimentacaoController extends Controller
{
    private $nameFolder = "movimentacao";

    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $movimentacao = Movimentacao::all();

        //
        return view("{ $this->nameFolder}/list", ["listaMovimentacoes"=>$movimentacao]);
    }

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        // Lista de contas
        $contas = array();
        foreach(Conta::all() as $conta){
            $contas[$conta->cd_conta] = $conta->nm_conta;
        }

        //
        return view("{ $this->nameFolder}/create", ['contas'=>$contas]);
    }

    /**
     * Store a newly created resource in storage.
     *
     * @param \Illuminate\Http\Request $request
     * @return \Illuminate\Http\Response
     */
    public function store(Request $request)
    {
        // Valida
        $this->validate($request, [
            'nm_movimentacao' => 'required',
            'ic_tipo_movimentacao' => 'required',
            'vl_movimentacao' => 'required',
        ]);

        // Adiciona e salva
        $movimentacao = new Movimentacao();
        $movimentacao->nm_movimentacao = $request->nm_movimentacao;
        $movimentacao->ic_tipo_movimentacao = $request->ic_tipo_movimentacao;
        $movimentacao->cd_conta = $request->cd_conta;
        $movimentacao->dt_movimentacao = $request->dt_movimentacao;
        $movimentacao->cd_nf_movimentacao = $request->cd_nf_movimentacao;
        $movimentacao->ic_pago_sim_nao = $request->ic_pago_sim_nao;
        $movimentacao->vl_movimentacao = money2float($request->vl_movimentacao);
        $movimentacao->ds_movimentacao = $request->ds_movimentacao;
    }
}

```

```

$movimentacao->ic_recorrente_sim_nao = $request->ic_recorrente_sim_nao;
$movimentacao->dt_registro_movimentacao = date('Y-m-d H:i:s');
$movimentacao->cd_conta = $request->cd_conta;
$movimentacao->save();

// Redireciona
return redirect('movimentacao')->with('message', 'Movimentacao salva com sucesso!');
}

/**
 * Display the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function show($id)
{
    //
    $movimentacao = Movimentacao::find($id);

    return view("{ $this->nameFolder}/show", ["movimentacao"=>$movimentacao]);
}

/**
 * Show the form for editing the specified resource.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function edit($id)
{
    // Movimentações
    $movimentacao = Movimentacao::find($id);

    // Lista de contas
    $contas = array();
    foreach(Conta::all() as $conta){
        $contas[$conta->cd_conta] = $conta->nm_conta;
    }

    return view("{ $this->nameFolder}/edit", ["movimentacao"=>$movimentacao, 'contas'=>$contas]);
}

/**
 * Update the specified resource in storage.
 *
 * @param \Illuminate\Http\Request $request
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function update(Request $request, $id)
{
    // Valida
    $this->validate($request, [
        'nm_movimentacao' => 'required',
        'ic_tipo_movimentacao' => 'required',
        'vl_movimentacao' => 'required',
    ]);

    // Adiciona e salva

```

```

$movimentacao = Movimentacao::find($id);
$movimentacao->nm_movimentacao = $request->nm_movimentacao;
$movimentacao->ic_tipo_movimentacao = $request->ic_tipo_movimentacao;
$movimentacao->cd_conta = $request->cd_conta;
$movimentacao->dt_movimentacao = $request->dt_movimentacao;
$movimentacao->cd_nf_movimentacao = $request->cd_nf_movimentacao;
$movimentacao->ic_pago_sim_nao = $request->ic_pago_sim_nao;
$movimentacao->vl_movimentacao = money2float($request->vl_movimentacao);
$movimentacao->ds_movimentacao = $request->ds_movimentacao;
$movimentacao->ic_recorrente_sim_nao = $request->ic_recorrente_sim_nao;
$movimentacao->cd_conta = $request->cd_conta;
$movimentacao->save();

// Redireciona
return redirect("movimentacao/$id")->with('message', 'Movimentacao editado com sucesso!');
}

/**
 * Remove the specified resource from storage.
 *
 * @param int $id
 * @return \Illuminate\Http\Response
 */
public function destroy($id)
{
    // Adiciona e salva
    $movimentacao = Movimentacao::find($id);
    $movimentacao->delete();

    // Redireciona
    return redirect('movimentacao')->with('message', 'Movimentacao deletada com sucesso!');
}
}

```

## Model - Departamento

```

<?php
namespace App\Http\Controllers;

use Illuminate\Database\Eloquent\Model;

class Departamento extends Model
{
    // Nome da tabela
    protected $table = 'departamento';

    // Chave primário
    protected $primaryKey = 'cd_departamento';

    // Removendo os campos de tempo
    public $timestamps = false;

    /**
     * The attributes that should be mutated to dates.
     *
     * @var array
     */
    protected $dates = [

```

```

        'created_at',
        'updated_at',
        'deleted_at'
    ];

```

## Model - Usuários

```

<?php
namespace App\Http\Controllers;

use Illuminate\Database\Eloquent\Model;

class Usuario extends Model
{
    // Nome da tabela
    protected $table = 'usuario';

    // Chave primário
    protected $primaryKey = 'cd_usuario';

    // Removendo os campos de tempo
    public $timestamps = false;

    /**
     * Relacionamento hasOne
     * Departamento
     */
    public function departamento(){
        return $this->belongsTo('App\Http\Controllers\Departamento', 'cd_departamento',
        'cd_departamento');
    }

    /**
     * The attributes that should be mutated to dates.
     *
     * @var array
     */
    protected $dates = [
        'created_at',
        'updated_at',
        'deleted_at'
    ];
}

```

## Model - Produtos

```

<?php
namespace App\Http\Controllers;

use Illuminate\Database\Eloquent\Model;

class Produto extends Model
{
    // Nome da tabela
    protected $table = 'produto';

    // Chave primário

```



```

protected $primaryKey = 'cd_produto';

// Removendo os campos de tempo
public $timestamps = false;

/**
 * The attributes that should be mutated to dates.
 *
 * @var array
 */
protected $dates = [
    'created_at',
    'updated_at',
    'deleted_at'
];

```

## Model - Contas

```

<?php
namespace App\Http\Controllers;

use Illuminate\Database\Eloquent\Model;

class Conta extends Model
{
    // Nome da tabela
    protected $table = 'conta';

    // Chave primário
    protected $primaryKey = 'cd_conta';

    // Removendo os campos de tempo
    public $timestamps = false;

    /**
     * The attributes that should be mutated to dates.
     *
     * @var array
     */
    protected $dates = [
        'created_at',
        'updated_at',
        'deleted_at'
    ];
}

```

## Model – Movimentações

```

<?php
namespace App\Http\Controllers;

use Illuminate\Database\Eloquent\Model;

class Movimentacao extends Model
{
    // Nome da tabela
    protected $table = 'movimentacao';

    // Chave primário
}

```

```

protected $primaryKey = 'cd_movimentacao';

// Removendo os campos de tempo
public $timestamps = false;

/**
 * Relacionamento hasOne
 * Conta
 */
public function conta(){
    return $this->belongsTo('App\Http\Controllers\Conta', 'cd_conta', 'cd_conta');
}

/**
 * The attributes that should be mutated to dates.
 *
 * @var array
 */
protected $dates = [
    'dt_movimentacao',
    'dt_registro_movimentacao',
];
}

```

## Plano de teste

Com o crescimento do mercado de venda de produtos e prestação de serviço em geral, a procura por um sistema online de apoio as tarefas administrativas, financeiras, compra e venda de produtos se tornou o carro chefe do mercado tecnológico.

Pensando no cenário atual, será desenvolvido o **Sistema Integrado de Gestão – SIG** um *ERP - Enterprise Resource Planning* (Planejamento dos Recursos da Empresa) quem tem como objetivo auxiliar e automatizar os fluxo de compra e venda da empresa, mapeando deste a cotação de novos produtos até a sua venda para o mercado, tudo isso de maneira online e de fácil acesso a qualquer dispositivo mobile e desktop.

Considerando que o SIG é um sistema baseado totalmente em processos pré definidos, os testes deverão ser realizados sequencialmente, sempre partindo da caso de teste inicial até a atividade que será testada.

Para melhor entendimento dos casos de testes, segue abaixo uma lista de termos utilizados nos casos de testes:

- **Parte pública:** Website.
- **Parte administrativa:** Ambiente de trabalho dos colaboradores da empresa.
- **Orçamento:** Pedido.

### REQUISITOS A SEREM TESTADOS

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 001 – Cadastro de departamento</b>  |
| <b>Localização</b>   | Site > Menu > Departamentos   |
| <b>Pré-condições</b> | O usuário deverá estar logado no sistema, e ter permissão para cadastrar novos departamentos.   |
| <b>Procedimento</b>  | 1- Logar no sistema;<br>2- Após logar no sistema ir na opções de Departamento dentro de menu;<br>3- Clicar no botão “novo registro”<br>4- Preencher os campos nome do departamento; |

|                           |  |
|---------------------------|--|
|                           | 5- Clicar no botão cadastrar.  |
| <b>Resultado esperado</b> | Após realizar essa ação, o departamento deverá estar disponível na listagem de departamentos.          |
| <b>Dados de entrada</b>   | Nome.  |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>            | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 002 – Editar departamento</b>   |
| <b>Localização</b>        | Site > Menu > Departamentos   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para editar departamentos já cadastrados no sistema.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de departamento, dentro de menu;</li> <li>3- Escolher um departamento;</li> <li>4- Clicar em ver;</li> <li>5- Clicar no botão editar;</li> <li>6- Realizar a alteração desejada no conteúdo e</li> <li>7- Clicar no botão editar para finalizar a ação.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação os dados do departamento deverão ser alterados.   |
| <b>Dados de entrada</b>   | Nome.   |

|                   |  |
|-------------------|--|
| <b>Prioridade</b> | Alta.  |
| <b>Ambiente</b>   | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>    | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 003 – Cadastro de usuários</b>  |
| <b>Localização</b>        | Site > Menu > Usuários  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para cadastrar novos usuários.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de usuários dentro de menu;</li> <li>3- Clicar no botão “novo registro”</li> <li>4- Preencher os campos de nome, email, senha e departamento e</li> <li>5- Clicar no botão cadastrar.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação, o serviços deverá estar disponível na listagem de serviços.  |
| <b>Dados de entrada</b>   | Nome, email, senha e departamento.  |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.   |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 004 – Editar usuários</b>   |
| <b>Localização</b>        | Site > Menu > Usuários  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para editar usuários já cadastrados no sistema.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de usuários dentro de menu;</li> <li>3- Escolher um usuário;</li> <li>4- Clicar em ver;</li> <li>5- Clicar no botão editar;</li> <li>6- Realizar a alteração desejada no conteúdo e</li> <li>7- Clicar no botão editar para finalizar a ação.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação os dados do usuário deverão ser alterados.  |
| <b>Dados de entrada</b>   | Nome, email, senha e departamento.  |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |  |
|----------------------|--|
| <b>Caso de Teste</b> | <b>CT 005 – Cadastro de serviços</b>   |
| <b>Localização</b>   | Site > Menu > Serviços   |
| <b>Pré-condições</b> | O usuário deverá estar logado no sistema, e ter permissão para cadastrar novos serviços. |
| <b>Procedimento</b>  | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> </ol>                   |

|                           |   |
|---------------------------|---|
|                           | <p>2- Após logar no sistema ir na opções de serviço dentro de menu;</p> <p>3- Clicar no botão “novo registro”</p> <p>4- Preencher os campos de nome do serviços, valor e descrição e</p> <p>5- Clicar no botão cadastrar.</p> |
| <b>Resultado esperado</b> | Após realizar essa ação, o serviços deverá estar disponível na listagem de serviços.  |
| <b>Dados de entrada</b>   | Nome, valor e descrição.  |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |  |
|----------------------|--|
| <b>Caso de Teste</b> | <b>CT 006 – Editar serviço</b>   |
| <b>Localização</b>   | Site > Menu > Serviços   |
| <b>Pré-condições</b> | O usuário deverá estar logado no sistema, e ter permissão para editar serviços já cadastrados no sistema.  |
| <b>Procedimento</b>  | <p>1- Logar no sistema;</p> <p>2- Após logar no sistema ir na opções de serviço dentro de menu;</p> <p>3- Escolher um pedido;</p> <p>4- Clicar em ver;</p> <p>5- Clicar no botão editar;</p> <p>6- Realizar a alteração desejada no conteúdo e</p> <p>7- Clicar no botão editar para finalizar a ação.</p> |

|                           |  |
|---------------------------|--|
| <b>Resultado esperado</b> | Após realizar essa ação os dados do serviço deverão ser alterados.                                     |
| <b>Dados de entrada</b>   | Nome, valor e descrição.   |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>            | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 007 – Cadastro de contas</b>  |
| <b>Localização</b>        | Site > Menu > Contas  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para cadastrar novas contas.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de contas dentro de menu;</li> <li>3- Clicar no botão “novo registro”</li> <li>4- Preencher os campos nome, número da agência, número da conta, valor inicial e tipo da conta;</li> <li>5- Clicar no botão cadastrar.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação, a conta deverá estar disponível na listagem de contas.   |
| <b>Dados de entrada</b>   | Nome, número da agência, número da conta, valor inicial e tipo da conta.  |
| <b>Prioridade</b>         | Alta.   |



|                 |  |
|-----------------|--|
| <b>Ambiente</b> | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>  | Manual.  |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 008 – Editar contas</b>  |
| <b>Localização</b>        | Site > Menu > Contas   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para editar contas já cadastradas no sistema.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de contas dentro de menu;</li> <li>3- Escolher uma conta;</li> <li>4- Clicar em ver;</li> <li>5- Clicar no botão editar;</li> <li>6- Realizar a alteração desejada no conteúdo e</li> <li>7- Clicar no botão editar para finalizar a ação.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação os dados da conta deverão ser alterados.   |
| <b>Dados de entrada</b>   | Nome, número da agência, número da conta, valor inicial e tipo da conta.   |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94,<br>Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 009 – Cadastro de movimentações administrativas</b>   |
| <b>Localização</b>        | Site > Menu > Movimentações   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para cadastrar novas movimentações.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de movimentações dentro de menu;</li> <li>3- Clicar no botão “novo registro”</li> <li>4- Preencher os campos nome, tipo, conta, data da movimentação, código nota fiscal, situação da movimentação, valor da movimentação, descrição e tipo da movimentação.</li> <li>5- Clicar no botão cadastrar.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação, a movimentações deverá estar disponível na listagem de movimentações.  |
| <b>Dados de entrada</b>   | Nome, tipo, conta, data da movimentação, código nota fiscal, situação da movimentação, valor da movimentação, descrição e tipo da movimentação.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |                                      |
|----------------------|--------------------------------------|
| <b>Caso de Teste</b> | <b>CT 010 – Editar movimentações</b> |
| <b>Localização</b>   | Site > Menu > Movimentações          |

|                           |  |
|---------------------------|--|
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para editar movimentações já cadastradas no sistema.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de movimentações dentro de menu;</li> <li>3- Escolher uma movimentação;</li> <li>4- Clicar em ver;</li> <li>5- Clicar no botão editar;</li> <li>6- Realizar a alteração desejada no conteúdo e</li> <li>7- Clicar no botão editar para finalizar a ação.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação os dados da movimentação deverão ser alterados.  |
| <b>Dados de entrada</b>   | Nome, tipo, conta, data da movimentação, código nota fiscal, situação da movimentação, valor da movimentação, descrição e tipo da movimentação.  |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 011 – Login administrativo</b>  |
| <b>Localização</b>   | Site > Login administrativo   |
| <b>Pré-condições</b> | Estar na tela de login administrativo   |
| <b>Procedimento</b>  | <ol style="list-style-type: none"> <li>1) O usuário informa um usuário e senha.</li> <li>2) O usuário seleciona a opção “Autenticar”.</li> <li>3) O sistema verifica se os dados digitados existem e se o mesmo possui permissão para logar.</li> </ol> |

|                           |   |
|---------------------------|---|
| <b>Resultado esperado</b> | O usuário acessa a parte administrativa do sistema.   |
| <b>Dados de entrada</b>   | E-mail e senha.   |
| <b>Prioridade</b>         | Alta  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>            | Automatizado.   |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 012 – Solicitação de compras</b>   |
| <b>Localização</b>        | Ambiente administrativo > Menu > Cotação > Nova cotação de compras.  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para solicitação de cotação de produtos.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- O usuário loga no sistema;</li> <li>2- O usuário abre uma nova cotação de compras de produtos;</li> <li>3- O usuário preenche o formulário de cotação, com o nome do produto e descrição do mesmo e</li> <li>4- Usuário confirma os dados do formulário preenchido e cadastra o mesmo no sistema.</li> </ol> |
| <b>Resultado esperado</b> | Após o formulário de cotação ser preenchido, os dados deverão ser armazenados no sistema, e o pedido de cotação deverá ficar com o status de “Aguardando cotação”. Após realizar essas atividades o sistema deverá disponibilizar a cotação gerada para o responsável pela procura de fornecedores.  |
| <b>Dados de entrada</b>   | Nome do produto e descrição.   |
| <b>Prioridade</b>         | Alta   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |

|                |         |
|----------------|---------|
| <b>Técnica</b> | Manual. |
|----------------|---------|

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 013 – Ambiente de cotação de produtos</b>  |
| <b>Localização</b>        | Ambiente administrativo > Menu > Cotação de compras.   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para visualizar as cotações e anexar as propostas dos fornecedores.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Loga no sistema;</li> <li>2- Acessar o ambiente de cotações pedentes para realização;</li> <li>3- Solicitar as propostas para os fornecedores (de forma manual);</li> <li>4- Anexar as propostas em PDF coletadas dos fornecedores na cotação e</li> <li>5- Enviar a cotação para etapa de análise.</li> </ol> |
| <b>Resultado esperado</b> | Após anexar todas as propostas enviadas pelos fornecedores, a mesma deverá ser enviada para a etapa de “análise”, ao realizar essa ação o status dessa cotação deverá ser alterado para “Aguardando aprovação”.  |
| <b>Dados de entrada</b>   | Anexar arquivos em PDF's das propostas coletadas com os fornecedores.  |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                      |  |
|----------------------|--|
| <b>Caso de Teste</b> | <b>CT 014 – Análise das propostas de compras</b> |
|----------------------|--|

|                           |   |
|---------------------------|---|
| <b>Localização</b>        | Ambiente administrativo > Menu > Cotação de compras > Cotações em análise.  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para visualizar as propostas dos fornecedores anexadas nas cotações de compras de produtos.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente de análise das propostas de cotação;</li> <li>3- Visualizar as propostas anexadas no pedido de cotação de produtos;</li> <li>4- Aprovar uma das propostas anexadas.</li> <li>5- Reprovar as demais todas as propostas de uma cotação.</li> </ol>                           |
| <b>Resultado esperado</b> | <p>Nesse caso de teste são esperado dois resultados:</p> <p><b>Item 4 do procedimento:</b> A proposta é enviada para etapa de cadastro de produtos, alterando o status da cotação para “Cotação liberada”.</p> <p><b>Item 5 do procedimento:</b> A cotações é enviada novamente para o “ambiente de cotação de produtos”, com o status de “Propostas reprovadas”.</p> |
| <b>Dados de entrada</b>   | Nenhum nesse caso de teste.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 015 – Recebimento das propostas</b>                                 |
| <b>Localização</b>   | Ambiente administrativo > Menu > Recebimentos das propostas > Fornecedor. |

|                           |  |
|---------------------------|--|
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para visualizar as propostas aprovadas e também permissão para dar entrada no cadastro de fornecedores.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente de cadastro de fornecedores, dentro de recebimento das propostas;</li> <li>3- Verificar se o fornecedor possui cadastro ativo no sistema, caso ele possua cadastro só será realizado a atualização e confirmação dos dados cadastrais;</li> <li>4- Caso o fornecedor não possua um cadastro ativo no sistema, será preenchidos os seguintes dados no sistema: Nome, razão social, nome fantasia, cpf ou cnpj, inscrição estadual, endereço, bairro, cidade, estado, tipo de pessoa (física ou jurídica) e situação cadastral e</li> <li>5- Após os dados serem preenchidos ou atualizados pelo comprador, clique no botão avançar.</li> </ol> |
| <b>Resultado esperado</b> | O pedido de cotação será enviado para o ambiente de entrada de produtos no estoque com o status do pedido de cotação em “Cadastro dos produtos no estoque”.  |
| <b>Dados de entrada</b>   | Nome, razão social, nome fantasia, cpf ou cnpj, inscrição estadual, endereço, bairro, cidade, estado, tipo de pessoa (física ou jurídica) e situação cadastral.  |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                      |                                     |
|----------------------|-------------------------------------|
| <b>Caso de Teste</b> | <b>CT 016 – Entrada de produtos</b> |
| <b>Localização</b>   |                                     |

|                           |   |
|---------------------------|---|
|                           | Ambiente administrativo > Menu > Recebimentos das propostas > Entrada de produtos.  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para visualizar os produtos já cadastrados no sistema, e também permissão para cadastrar novos produtos.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente de entrada de produtos, dentro de recebimento das propostas;</li> <li>3- Verificar se o produto já está cadastro no sistema, caso ele esteja só será realizada atualização da quantidade em estoque do produto,</li> <li>4- Caso o produto não esteja cadastrado no sistema, realizar o cadastro do mesmo, preenchendo os seguintes dados: Nome, descrição, valor, foto, quantidade mínima no estoque, quantidade máxima no estoque e quantidade atual em estoque e</li> <li>5- Após os dados serem preenchidos ou atualizados pelo comprador, clique no botão finalizar recebimento das propostas.</li> </ol> |
| <b>Resultado esperado</b> | Após os dados serem preenchidos ou atualizados pelo comprador, o pedido de cotação será enviado para o ambiente de “emissão de ordem de pagamento” com o status do pedido de cotação em “Emissão de ordem de pagamento”.  |
| <b>Dados de entrada</b>   | Nome, descrição, valor, foto, quantidade mínima no estoque, quantidade máxima no estoque e quantidade atual em estoque.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 017 – Emissão de ordem de pagamento</b> |
|----------------------|---|



|                           |   |
|---------------------------|---|
| <b>Localização</b>        | Ambiente administrativo > Menu > Emissão de ordem de pagamento.   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para visualizar as cotações que estão aguardando a emissão da ordem de pagamento.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente emissão de ordem de pagamento;</li> <li>3- Clicar em mais detalhes de uma ordem pagamento;</li> <li>4- Verificar se foi realizado corretamente o cadastro do fornecedor e dos produtos no sistema e</li> <li>5- Após verificar todas as informações, clicar no botão “liberar faturamento”.</li> </ol> |
| <b>Resultado esperado</b> | Após emitir a ordem de pagamento do fornecedor, a cotação deverá ser enviada para o “ambiente financeiro” para que seja realizada as movimentações de faturamento da cotação, ao realizar essa ação a cotação deverá ter seu status alterado para “Em faturamento”.   |
| <b>Dados de entrada</b>   | Nesse caso de teste não será necessário nenhuma entrada de dados.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 018 – Faturamento</b>                   |
| <b>Localização</b>   | Ambiente administrativo > Menu > Faturamento. |

|                           |  |
|---------------------------|--|
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão realizar as movimentações de pagamento do fornecedor e abertura de contrato.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente emissão de faturamento;</li> <li>3- Clicar em mais detalhes de pedido de cotação;</li> <li>4- Verificar se os dados do fornecedor e dos produtos não foram alterados e</li> <li>5- Abrir um novo contrato de compra de produtos, e preencher o campo de quantidade de parcelas e valor total do contrato;</li> <li>6- Após abrir o novo contrato, clicar no botão “emite a movimentação de pagamento”;</li> <li>7- Anexar um arquivo em PDF na aba “nota fiscal do fornecedor” e</li> <li>8- Após anexar o arquivo em PDF, clique no botão “liberar produtos”.</li> </ol> |
| <b>Resultado esperado</b> | Após o usuário dar o pagamento como aprovado, o pedido de cotação deverá ser enviado para o “ambiente de atualização de estoque” para que os produtos fiquem disponíveis para comercialização. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “Aguardando liberação do produto no estoque”.  |
| <b>Dados de entrada</b>   | Nesse caso de teste não será necessário nenhuma entrada de dados.  |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 019 – Atualização de estoque</b>                  |
| <b>Localização</b>   | Ambiente administrativo > Menu > Liberação de produtos. |

|                           |   |
|---------------------------|---|
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para realizar a atualização de produtos no estoque do sistema.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente liberação de produtos;</li> <li>3- Clicar em mais detalhes de pedido de cotação;</li> <li>4- Verifica se os dados e ações realizadas no caso de testes anteriores estão corretos e</li> <li>5- Caso esteja tudo correto, clique no botão “libera os produtos para comercialização”.</li> </ol> |
| <b>Resultado esperado</b> | Deverá ser emitida uma notificação para o usuário que os produtos solicitados já estão disponíveis no sistema. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “cotação realizada”.  |
| <b>Dados de entrada</b>   | Parcelas e valor total do contrato  |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |   |
|----------------------|---|
| <b>Caso de Teste</b> | <b>CT 020 – Realizar cadastro como cliente</b>  |
| <b>Localização</b>   | Site > Menu > Solicite um orçamento.  |
| <b>Pré-condições</b> | Nenhuma.  |
| <b>Procedimento</b>  | <ol style="list-style-type: none"> <li>1- Acessar a parte pública do sistema;</li> <li>2- Ir até a opção solicite um orçamento;</li> <li>3- Preencher o cadastro de cliente e</li> <li>4- Clicar no botão cadastrar.</li> </ol> |

|                           |   |
|---------------------------|---|
| <b>Resultado esperado</b> | Após preencher os dados, o sistema deverá redirecionar o usuário para a tela de orçamento.  |
| <b>Dados de entrada</b>   | Tipo de cadastrado (Física ou Jurídica), nome, razão social, cnpj ou cpf, endereço, complemento, bairro, cidade, estado, telefone, email e senha. |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 021 – Login cliente</b>  |
| <b>Localização</b>        | Site > Menu > Solicite um orçamento.   |
| <b>Pré-condições</b>      | Possuir um cadastro como cliente.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1) Informa um usuário e senha.</li> <li>2) Selecionar a opção “Autenticar”.</li> <li>3) O sistema verifica se os dados digitados existem e se o mesmo possui permissão para logar.</li> </ol> |
| <b>Resultado esperado</b> | O usuário acessa a parte de solicitação de orçamento.  |
| <b>Dados de entrada</b>   | E-mail e senha.  |
| <b>Prioridade</b>         | Alta   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 022 – Solicitar orçamento</b>   |
| <b>Localização</b>        | Site > Menu > Solicite um orçamento > Logar > Novo orçamento.   |
| <b>Pré-condições</b>      | Estar logado na parte pública do sistema.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1) Clicar no botão “novo orçamento”.</li> <li>2) Selecionar os produtos disponíveis para compra e a quantidade desejada para compra;</li> <li>3) Selecionar os serviços disponíveis para aquisição e a quantidade desejada para compra;</li> <li>4) Colocar o endereço de entrega dos produtos;</li> <li>5) Colocar o endereço para prestação dos serviços contratados e</li> <li>6) Finalizar orçamento.</li> </ol> |
| <b>Resultado esperado</b> | Após preencher todas as informações o orçamento deverá ser cadastrado no sistema, e o mesmo deverá estar com o status “Aguardando geração do orçamento”.  |
| <b>Dados de entrada</b>   | Confirma os dados de endereço, como rua, número, bairro, cidade, estado, país e CEP, conforme cadastrado CT021.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

|                      |                                      |
|----------------------|--------------------------------------|
| <b>Caso de Teste</b> | <b>CT 023 – Geração de orçamento</b> |
| <b>Localização</b>   | Site > Menu > Orçamentos             |
| <b>Pré-condições</b> |                                      |

|                           |  |
|---------------------------|--|
|                           | O usuário deverá estar logado no sistema para que possa ver e preencher os orçamentos com o status em aberto.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de orçamentos dentro de menu;</li> <li>3- Escolher um orçamento com o “status em aberto”;</li> <li>4- Selecionar os produtos e serviços que o cliente está solicitando e</li> <li>5- Clicar no botão “Fechar orçamento”.</li> </ol> |
| <b>Resultado esperado</b> | Após preencher os orçamentos o mesmo deverá ser enviar para o cliente avaliar o mesmo, ao realizar essa ação o status do pedido de orçamento deverá ser alterado para “Aguardando aprovação do cliente”.   |
| <b>Dados de entrada</b>   | Quantidade de produtos e serviços.   |
| <b>Prioridade</b>         | Alta.  |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.  |
| <b>Técnica</b>            | Manual.  |

|                      |  |
|----------------------|--|
| <b>Caso de Teste</b> | <b>CT 024 – Negociação do orçamento</b>  |
| <b>Localização</b>   | Site > Menu > Solicite um orçamento > Logar > Listagem de orçamentos.  |
| <b>Pré-condições</b> | Estar logado na parte pública do sistema.  |
| <b>Procedimento</b>  | <ol style="list-style-type: none"> <li>1) Clicar na opção de “mais detalhes” do orçamento;</li> <li>2) Clicar no botão abrir negociação;</li> <li>3) Digitar a justificativa da negociação e</li> <li>4) Clicar no botão “enviar negociação”.</li> </ol> |

|                           |   |
|---------------------------|---|
| <b>Resultado esperado</b> | Cadastrar a justificativa no sistema.   |
| <b>Dados de entrada</b>   | Texto de justificativa da negociação.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>            | Manual.   |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 025 – Negociação do orçamento</b>   |
| <b>Localização</b>        | Site > Menu > Orçamentos  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para trabalhar nos orçamentos com status “Em negociação”.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de orçamentos dentro de menu;</li> <li>3- Escolher um orçamento com o “Em negociação”;</li> <li>4- Clicar na opção de “mais detalhes” do orçamento;</li> <li>5- Clicar no botão atender negociação;</li> <li>6- Digitar a resposta da negociação e</li> <li>7- Clicar no botão “enviar negociação”.</li> </ol> |
| <b>Resultado esperado</b> | Enviar para o cliente avaliar a resposta da negociação, ao realizar essa ação o status do pedido de orçamento deverá ser alterado para “Aguardando aprovação do cliente”.   |
| <b>Dados de entrada</b>   | Texto de justificativa da negociação.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |

|                |         |
|----------------|---------|
| <b>Técnica</b> | Manual. |
|----------------|---------|

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 026 – Aprovação do orçamento</b>  |
| <b>Localização</b>        | Site > Menu > Solicite um orçamento > Logar > Listagem de orçamentos.   |
| <b>Pré-condições</b>      | Estar logado na parte pública do sistema.   |
| <b>Procedimento</b>       | 1) Clicar na opção de “mais detalhes” do orçamento e<br>2) Clicar no botão “Aprovação do Orçamento”.                |
| <b>Resultado esperado</b> | Finalizar a etapa de negociação e alterar o status do orçamento para “Aguardando a geração da remessa de cobrança”. |
| <b>Dados de entrada</b>   | Nenhuma dado necessário nesse caso de teste.  |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.                 |
| <b>Técnica</b>            | Manual.   |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 027 – Cancelamento do orçamento</b>  |
| <b>Localização</b>        | Site > Menu > Solicite um orçamento > Logar > Listagem de orçamentos.                            |
| <b>Pré-condições</b>      | Estar logado na parte pública do sistema.  |
| <b>Procedimento</b>       | 1) Clicar na opção de “mais detalhes” do orçamento e<br>2) Clicar no botão “Cancelar Orçamento”. |
| <b>Resultado esperado</b> | Finalizar a etapa de negociação e alterar o status do orçamento para “Cancelado”.                |
| <b>Dados de entrada</b>   | Nenhuma dado necessário nesse caso de teste.   |



|                   |   |
|-------------------|---|
| <b>Prioridade</b> | Alta.   |
| <b>Ambiente</b>   | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>    | Manual.   |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 028 – Emissão da ordem de cobrança</b>   |
| <b>Localização</b>        | Site > Menu > Emissão da ordem de cobrança   |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema e ter permissão para realizar as movimentações de cobrança dos clientes e abertura de contrato.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Acessar o ambiente de emissão da ordem de cobrança, localizado no menu;</li> <li>3- Selecionar um pedido aprovado pelo cliente;</li> <li>4- Gerar o contrato de venda de produtos ou prestação de serviço;</li> <li>5- Gerar a ordem de cobrança;</li> <li>6- O Liberar o pedido juntamente com o contrato para o ambiente de remessa de entrega e prestação de serviço, clicando no botão “Gerar remessa”.</li> </ol> |
| <b>Resultado esperado</b> | Após o usuário dar o pagamento como aprovado, o pedido juntamente com o contrato serão enviados para o ambiente de emissão de remessa de entrega de produtos ou prestação de serviço. Ao realizar essa ação o status do pedido de cotação deverá ser alterado para “Aguardando liberação da remessa de entrega ou prestação de serviço”.   |
| <b>Dados de entrada</b>   | <p>Item 4 – Quantidade de parcelas.</p> <p>Item 5 – Número da nota fiscal, pagamento realizado (Sim/Não) e descrição da movimentação.</p>  |

|                   |   |
|-------------------|---|
| <b>Prioridade</b> | Alta.   |
| <b>Ambiente</b>   | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>    | Manual.   |

|                           |  |
|---------------------------|--|
| <b>Caso de Teste</b>      | <b>CT 029 – Geração de remessa para entrega</b>  |
| <b>Localização</b>        | Site > Menu > Geração de remessa para entrega  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema para que possa ver os pedidos com status “Aguardando liberação da remessa de entrega ou prestação de serviço”.  |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Escolher um pedido com status “Aguardando liberação da remessa de entrega ou prestação de serviço”;</li> <li>3- Clicar em mais detalhes do pedido;</li> <li>4- Confirmar os dados de entrega e</li> <li>5- Clicar no botão, “Emitir a remessa de entrega de produtos para o cliente”;</li> <li>6- Verificar se a quantidade de produtos foi alterada e</li> <li>7- Caso no pedido contenha prestação de serviço, clicar no botão “Liberar prestação de serviços”.</li> </ol> |
| <b>Resultado esperado</b> | Após emitir a remessa de entrega de produtos, deverá ser emitida uma notificação para cliente, informando que os produtos já foram separados e estão em transito para entrega. Caso no pedido contenha prestação de serviços, deverá ser enviada o pedido juntamente com o contrato para o técnico analisar e prestar o serviço para o cliente, caso seja necessário   |

|                         |  |
|-------------------------|--|
|                         | a prestação de serviço, o status do pedido deverá ser alterado para “Pedido em análise pela equipe técnica”. |
| <b>Dados de entrada</b> | Nenhum dado necessário nesse caso de teste.  |
| <b>Prioridade</b>       | Alta.  |
| <b>Ambiente</b>         | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.          |
| <b>Técnica</b>          | Manual.  |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 030 – Agendamento</b>   |
| <b>Localização</b>        | Site > Menu > Prestação de Serviço  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para possa ver os pedidos com status “Aguardando prestação de serviço”.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de prestação de serviço dentro de menu;</li> <li>3- Escolher um pedido com status “Aguardando prestação de serviço”;</li> <li>4- Clicar em “mais detalhes”;</li> <li>5- Cadastrar a “data de prestação de serviço” e</li> <li>6- Clicar no botão atualizar.</li> </ol> |
| <b>Resultado esperado</b> | Após realizar essa ação, o status do pedido deverá ser alterado para “Prestação de serviço agendada”.   |
| <b>Dados de entrada</b>   | Data da prestação de serviço.   |
| <b>Prioridade</b>         | Alta.   |

|                 |   |
|-----------------|---|
| <b>Ambiente</b> | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql. |
| <b>Técnica</b>  | Manual.   |

|                           |   |
|---------------------------|---|
| <b>Caso de Teste</b>      | <b>CT 031 – Encerramento</b>  |
| <b>Localização</b>        | Site > Menu > Prestação de Serviço  |
| <b>Pré-condições</b>      | O usuário deverá estar logado no sistema, e ter permissão para possa ver os pedidos com status “Prestação de serviço agendada”.   |
| <b>Procedimento</b>       | <ol style="list-style-type: none"> <li>1- Logar no sistema;</li> <li>2- Após logar no sistema ir na opções de prestação de serviço dentro de menu;</li> <li>3- Escolher um pedido com status “Prestação de serviço agendada”;</li> <li>4- Clicar em “mais detalhes” e</li> <li>5- Clicar no botão fechar pedido.</li> </ol> |
| <b>Resultado esperado</b> | Após fechar o serviço o pedido deverá ser encerrado, alterando o status para “concluído”.   |
| <b>Dados de entrada</b>   | Data da prestação de serviço.   |
| <b>Prioridade</b>         | Alta.   |
| <b>Ambiente</b>           | Windows 10, Google Chrome Versão 62.0.3202.94, Servidor de Aplicação Apache e Banco de dados Mysql.   |
| <b>Técnica</b>            | Manual.   |

## ESTRATÉGIAS E FERRAMENTAS DE TESTE

O tipo de teste realizado no SIG será o de “sistema” (teste funcional ou caixa-preta), teste esse que será realizado pela equipe de testadores (os testadores não fazem parte da equipe de desenvolvimento nesse tipo de teste) através da interface gráfica, essa abordagem de homologação tem como objetivo validar se o sistema está em conformidade com a especificação de requisitos pré estabelecidos no escopo do projeto. Esse tipo de teste é baseado em roteiros de teste (caso de testes, item abordado anteriormente) criados a partir da especificação.

Além do tipo de teste escolhido pelo gerente de teste, é importante reforçar que um caso de testes só poderá ser executado caso os anteriores já tenham sido validados, uma vez que o SIG é um sistema baseado totalmente e um processo contínuo, onde uma atividade (caso de teste) só pode ser realizada caso as anteriores tenham sido executadas de forma correta.

Como o sistema foi desenvolvido em formato web, será utilizado como ferramentas de testes os navegadores Google Chrome versão 62.0.3202.94 de 64 bits e Firefox Quantum versão 57.0 de 64 bits. Outra ferramenta de apoio utilizada nos testes será “ferramenta de captura” nativa do Windows, ferramenta essa que será utilizada para gravar os possíveis erros que poderão ocorrer durante a homologação do sistema.

## EQUIPE E INFRAESTRUTURA

Para que os testes sejam realizados de maneira correta e pelos responsáveis corretos, foi dividido abaixo os cargos, funções e profissionais que irão compor a equipe de homologação do sistema.

**Gerente de teste:** Victor Hugo Lopes dos Santos Costa

Função:

- Liderar a equipe de teste;
- Responsável por planejar e definir a estratégia de teste que será utilizada pelos testadores e
- Encarregado pela comunicação entre a equipe de teste e de desenvolvimento.

**Testadores:** Loraine e Willian Barreto Lopes.

Função:

- Executar os testes no sistema, baseado nos casos de testes e
- Descrever os erros encontrados de forma objetiva.

Os testes utilizaram como equipamentos dois notebooks com sistema operacional Windows 10, processador I5 com 4G de memória RAM e 500G de HD. Além da configuração dos equipamentos, os mesmos deverão estar com software XAMPP instalados para que seja utilizado o servidor apache do software para que o sistema possa funcionar, essa instalação é obrigatória uma vez que o sistema deverá ser homologado em ambiente de teste.

## CRONOGRAMA DE ATIVIDADES

| Status   | Cor |
|--|-----|
| Aguardando a realização do teste                   |     |
| Caso de teste em homologação.                      |     |
| Caso de teste homologado e liberado para produção. |     |
| Caso de teste com erro.                            |     |

| Cronograma de Plano de Teste - SIG                 |                 | Versão 1.0 |        |
|--|-----------------|------------|--------|
| Caso de teste                                      | Data realização | Técnica    | Status |
| CT 001 – Cadastro de departamento                  | Janeiro/2017    | Manual     |        |
| CT 002 – Editar departamento                       | Janeiro/2017    | Manual     |        |
| CT 003 – Cadastro de Usuários                      | Janeiro/2017    | Manual     |        |
| CT 004 – Editar usuários                           | Janeiro/2017    | Manual     |        |
| CT 005 – Cadastro de serviços                      | Janeiro/2017    | Manual     |        |
| CT 006 – Editar serviço                            | Janeiro/2017    | Manual     |        |
| CT 007 – Cadastro de contas                        | Janeiro/2017    | Manual     |        |
| CT 008 – Editar contas                             | Janeiro/2017    | Manual     |        |
| CT 009 – Cadastro de movimentações administrativas | Janeiro/2017    | Manual     |        |
| CT 010 – Editar movimentações                      | Janeiro/2017    | Manual     |        |

|   |              |        |  |
|---|--------------|--------|--|
| CT 011 – Login administrativo             | Janeiro/2017 | Manual |  |
| CT 012 – Solicitação de compras           | Janeiro/2017 | Manual |  |
| CT 013 – Ambiente de cotação de produtos  | Janeiro/2017 | Manual |  |
| CT 014 – Análise das propostas de compras | Janeiro/2017 | Manual |  |
| CT 015 – Recebimento das propostas        | Janeiro/2017 | Manual |  |
| CT 016 – Entrada de produtos              | Janeiro/2017 | Manual |  |
| CT 017 – Emissão de ordem de pagamento    | Janeiro/2017 | Manual |  |
| CT 018 – Faturamento                      | Janeiro/2017 | Manual |  |
| CT 019 – Atualização de estoque           | Janeiro/2017 | Manual |  |
| CT 020 – Realizar cadastro como cliente   | Janeiro/2017 | Manual |  |
| CT 021 – Login cliente                    | Janeiro/2017 | Manual |  |
| CT 022 – Solicitar orçamento              | Janeiro/2017 | Manual |  |
| CT 023 – Geração de orçamento             | Janeiro/2017 | Manual |  |
| CT 024 – Negociação do orçamento          | Janeiro/2017 | Manual |  |
| CT 025 – Negociação do orçamento          | Janeiro/2017 | Manual |  |
| CT 026 – Aprovação do orçamento           | Janeiro/2017 | Manual |  |
| CT 027 – Cancelamento do orçamento        | Janeiro/2017 | Manual |  |
| CT 028 – Emissão da ordem de cobrança     | Janeiro/2017 | Manual |  |
| CT 029 – Geração de remessa para entrega  | Janeiro/2017 | Manual |  |
| CT 030 – Agendamento                      | Janeiro/2017 | Manual |  |
| CT 031 – Encerramento                     | Janeiro/2017 | Manual |  |

| CT | Descrição dos erros encontrados |
|----|---------------------------------|
|    |                                 |
|    |                                 |
|    |                                 |
|    |                                 |
|    |                                 |

## Considerações finais

Podemos concluir ao termino da segunda versão da documentação técnica do sistema ERP SIG – Sistema Integrado de Gestão que Análise de Risco é o Pano de Teste da ferramenta já foram elaborados, o plano de teste e já está apto para utilização. Além da documentação técnica, podemos complementar que o processo de desenvolvimento da parte publicado do site e das ferramentas de processos já foi iniciado.



## Referência bibliográfica

**MVC** – Model-View-Controller, Disponível em: <http://protocoloti.blogspot.com.br/2012/12/mvc-model-view-controller.html>. Acessado em 14 de abril de 2017.

**MVC** - O padrão de arquitetura de software, Disponível em: [https://www.oficinadanet.com.br/artigo/1687/mvc\\_o\\_padrao\\_de\\_arquitetura\\_de\\_softwa](https://www.oficinadanet.com.br/artigo/1687/mvc_o_padrao_de_arquitetura_de_softwa) re. Acessado em 14 de abril de 2017.

**Entendendo o MVC** (Model-View-Controller), Disponível em: <http://www.digitaldev.com.br/2013/01/18/entendendo-o-mvc-model-view-controller/>. Acessado em 14 de abril de 2017.

**Introdução ao Laravel Framework PHP**, Disponível em: <http://www.devmedia.com.br/introducao-ao-laravel-framework-php/33173>. Acessado em 06 de maio de 2017.

**Tudo que você queria saber sobre Git e GitHub, mas tinha vergonha de perguntar**, Disponível em: <https://tableless.com.br/tudo-que-voce-queria-saber-sobre-git-e-github-mas-tinha-vergonha-de-perguntar/>. Acessado em 06 de maio de 2017.